

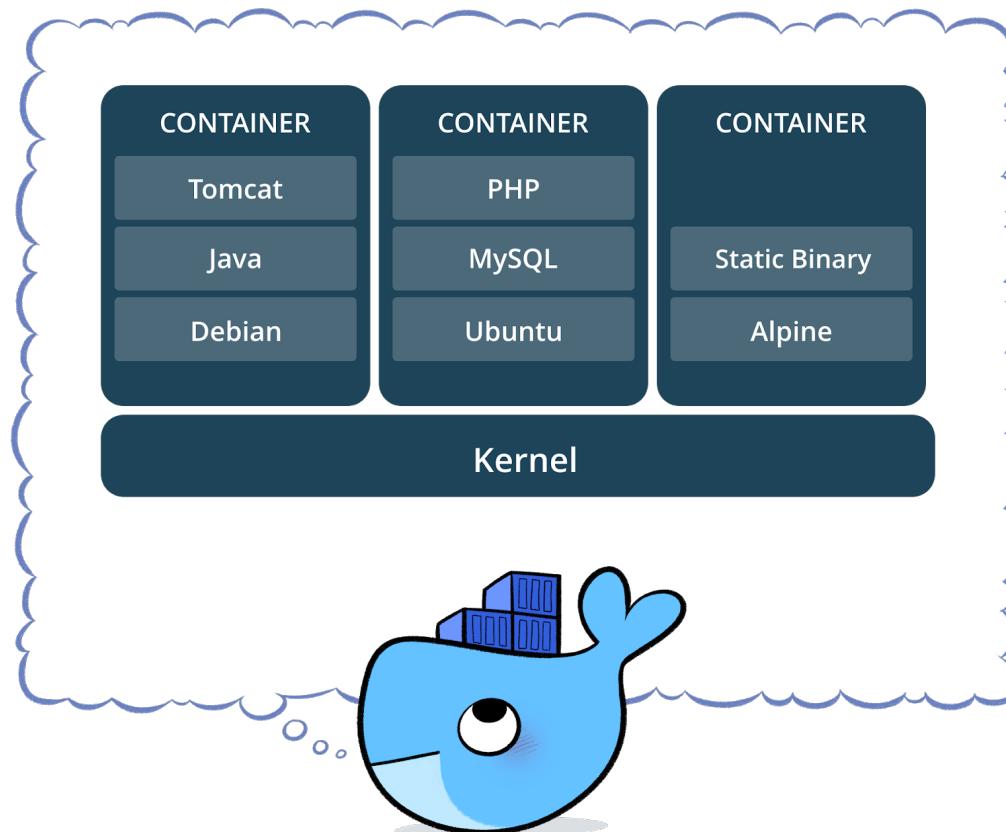
# NetPyNE Tutorial

- NEURON vs NEURON GUI vs NEtPyNE vs NetPyNE UI
- Intro to netpyne – motivation + structure
- Simple tutorial
  - 2 populations
  - 1 cell rule (soma+dend)
  - 1 synMech
  - Iclamp -> S
  - 1 conn rule – probability – S->M
  - Basic plots
- Biophys tutorial
  - Build on prev tutorial?
  - Import cell params from ModelDB – Hay 2011 and inhibitory cell
  - 2 layers (L23, L4), each with exc+inhib pop
  - 2 synMechs
  - NetStim -> all
  - Conn rules – yfrac-based weight; dist-dep inh conn
- Import model/tutorial – purkinje?

# NetPyNE GUI Tutorial: Installation

Instructions: <https://github.com/MetaCell/NetPyNE-UI/blob/master/README.md>

- 1) Install **Docker**: an application that allows you to run “containers” with everything you need already pre-installed.



# NetPyNE GUI Tutorial: Installation

Instructions: <https://github.com/MetaCell/NetPyNE-UI/blob/master/README.md>

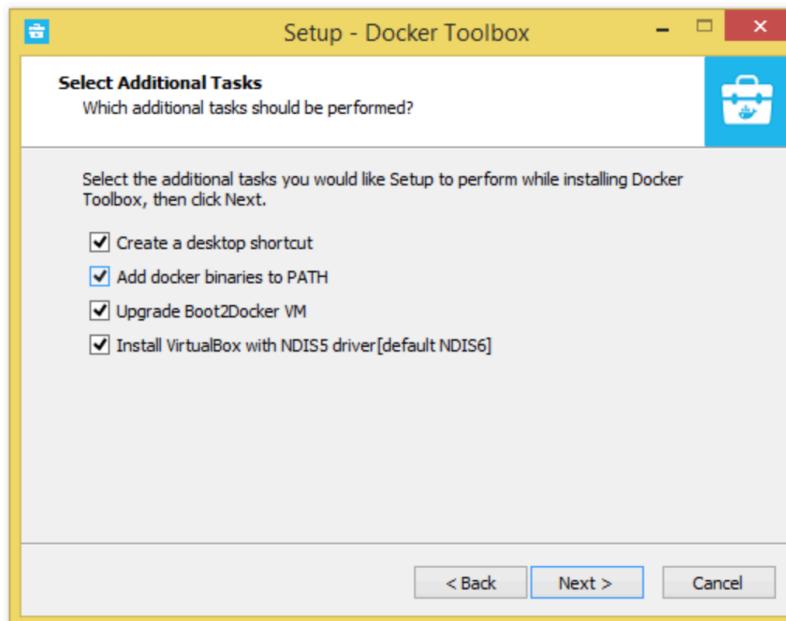
## 1) Install Docker (Windows)

### For Windows users

To install docker on Windows download docker-toolbox from the link below and follow the onscreen instructions. [Windows](#)

Few notes regarding the windows installation:

- During the step "Select Additional Tasks", select all the checkboxes as the image here.



# NetPyNE GUI Tutorial: Installation

**Instructions:** <https://github.com/MetaCell/NetPyNE-UI/blob/master/README.md>

## 1) Install Docker (Linux and Mac)

### For Linux users

To install docker on Linux follow the link below with the instructions. [Linux](#)

Few notes about the linux installation:

- Once the installation has been completed, to run docker from your user you need to add it to the docker group. To do this you can run the command below

```
sudo usermod -a -G docker $USERNAME
```

Once the command has been executed you need to open a new terminal or login in your environment again in order to make the change effective.

### For MacOS users

To install docker on Mac OS follow the link below with the instructions. [MacOS](#)

# NetPyNE GUI Tutorial: Installation

Instructions: <https://github.com/MetaCell/NetPyNE-UI/blob/master/README.md>

2) Get the NetPyNE GUI container:

**Install NEURON-UI using Docker from command line**

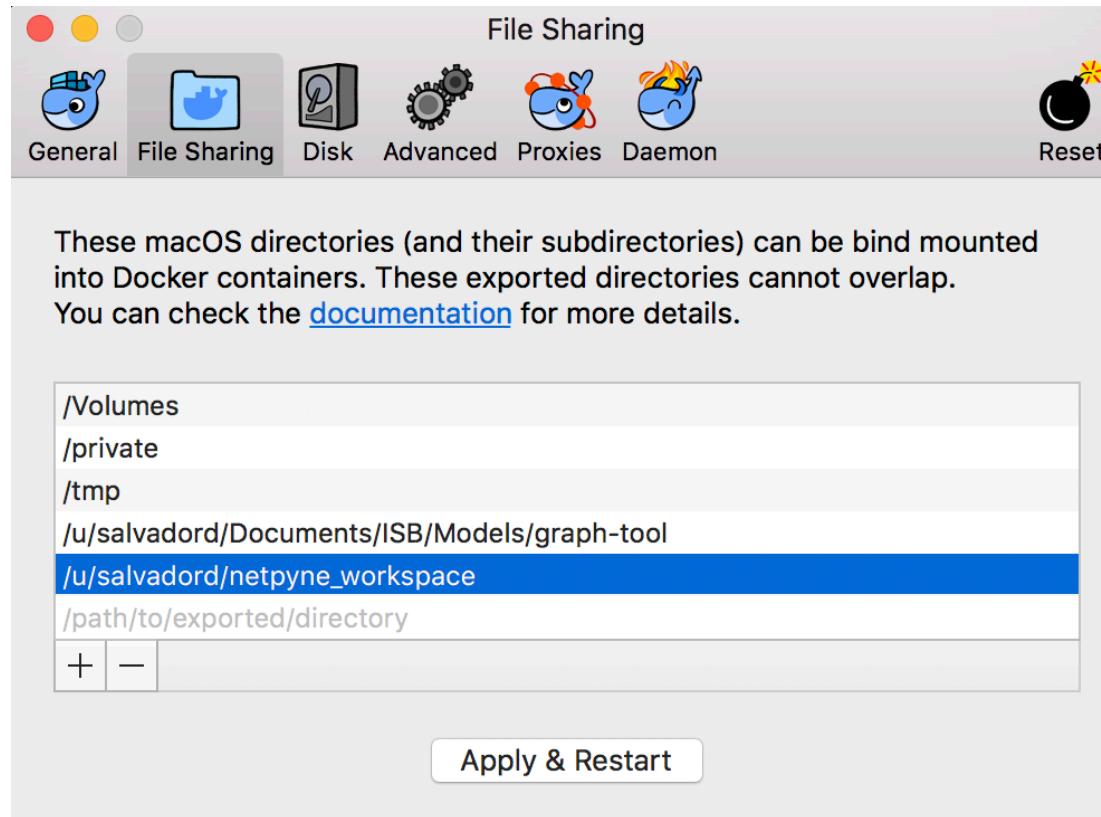
To pull the docker container:

```
docker pull metacell/netpyne-ui
```

# NetPyNE GUI Tutorial: Installation

Instructions: <https://github.com/MetaCell/NetPyNE-UI/blob/master/README.md>

3) Add a shared folder via Docker Settings -> Preferences -> File Sharing



# NetPyNE GUI Tutorial: Installation

Instructions: <https://github.com/MetaCell/NetPyNE-UI/blob/master/README.md>

## 4) Run the NetPyNE GUI container (Linux and Mac):

To run the docker container:

```
docker run -it -p 8888:8888 metacell/netpyne-ui
```

Or alternatively, if you want a local folder outside of the Docker container to host the NetPyNE-UI workspace (where you can import models from, export models to, inspect the log and the jupyter notebook) you can execute the following command:

```
docker run -it -v ~/folder_in_your_computer:/home/jovyan/netpyne_workspace -p 8888:8888 metacell/netpyne-u;
```

This will mount your local folder `folder_in_your_computer` as a volume inside the container and will be used to host the NetPyNE-UI workspace (`/home/jovyan/netpyne_workspace` inside the container).

Once you run your container you can open your browser and connect to <http://localhost:8888/gepetto>.

# NetPyNE GUI Tutorial: Installation

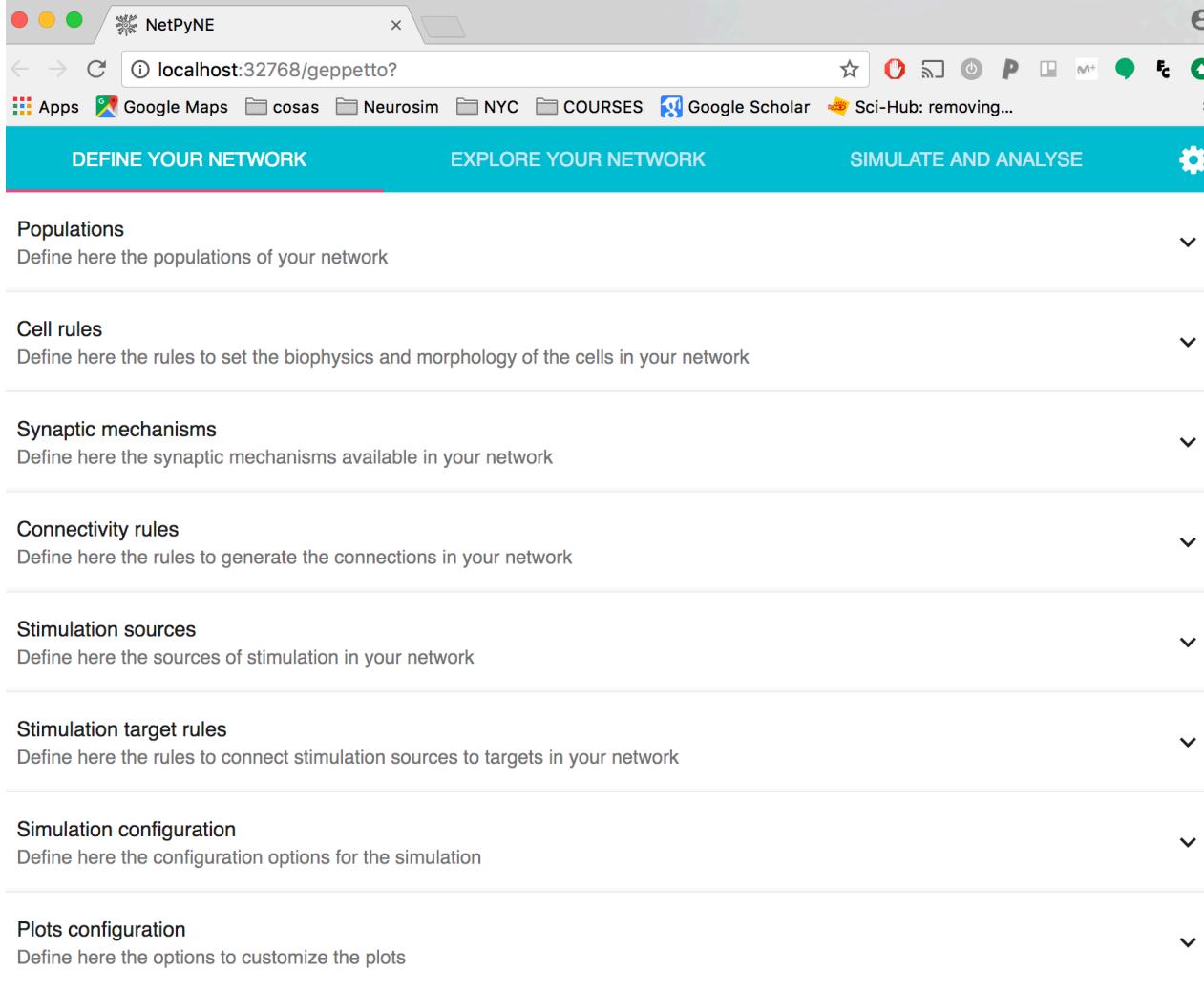
**Instructions:** <https://github.com/MetaCell/NetPyNE-UI/blob/master/README.md>

## 4) Run the NetPyNE GUI container (Windows):

One note for Windows users - once the docker image has been pulled and the NetPyNE-UI container created, you can reach your instance of NetPyNE-UI by default at the address 192.168.99.100:8888, if something has been changed in the settings this will be visible when the docker terminal will be open, as per image below.

# NetPyNE GUI Tutorial: Installation

Instructions: <https://github.com/MetaCell/NetPyNE-UI/blob/master/README.md>



# NetPyNE GUI Tutorial: Installation

Clone this Github repo with the workspace files:

[https://github.com/Neurosim-lab/netpyne\\_workspace.git](https://github.com/Neurosim-lab/netpyne_workspace.git)

The screenshot shows the GitHub repository page for 'Neurosim-lab / netpyne\_workspace'. The page includes the repository name, statistics (3 commits, 1 branch, 0 releases, 1 contributor), and a list of recent commits. The 'Clone or download' button is highlighted in green.

No description, website, or topics provided. [Edit](#)

Add topics

3 commits 1 branch 0 releases 1 contributor

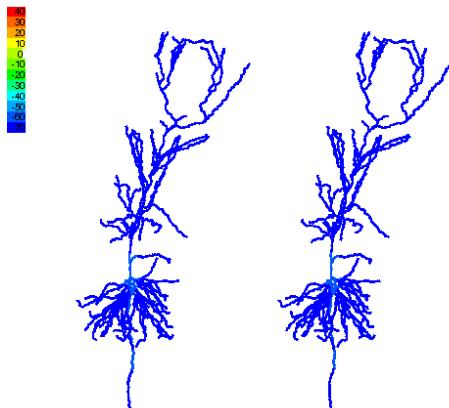
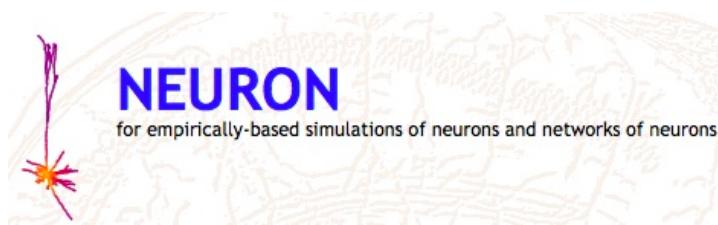
Branch: master ▾ New pull request Create new file Upload files Find file Clone or download ▾

salvadord	added readme	Latest commit 74306b8 4 minutes ago
cells	added cells and mod	5 minutes ago
mod	added cells and mod	5 minutes ago
README	addeded readme	4 minutes ago
gui_tut1.py	added tuts, cells and mod	5 minutes ago
gui_tut2.py	added tuts, cells and mod	5 minutes ago
gui_tut3.py	added tuts, cells and mod	5 minutes ago

# What is what...

NEURON

# What is what...



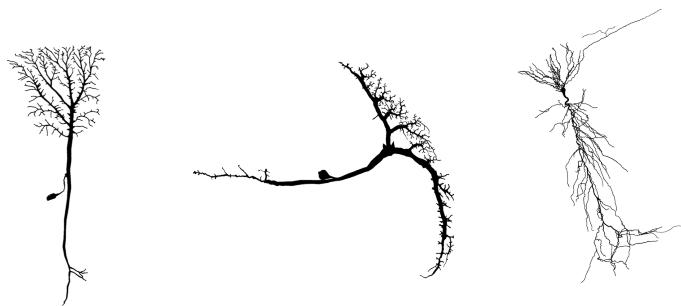
Biophysically-detailed simulation at multiple scales:

- molecular reaction and diffusion
- ionic channels and synapses
- detailed cell morphologies
- large-scale networks

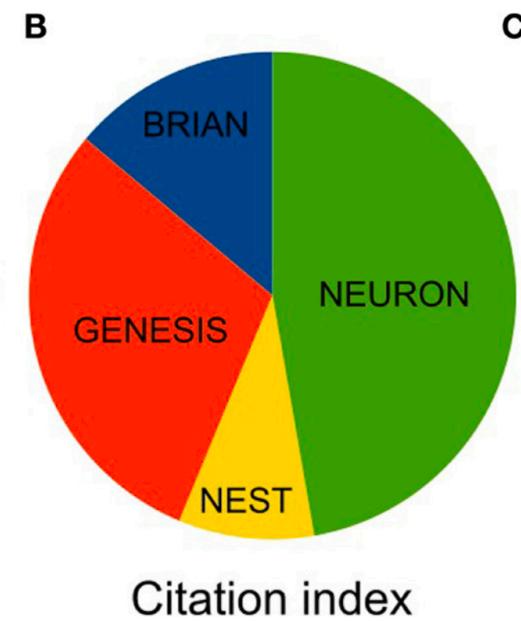
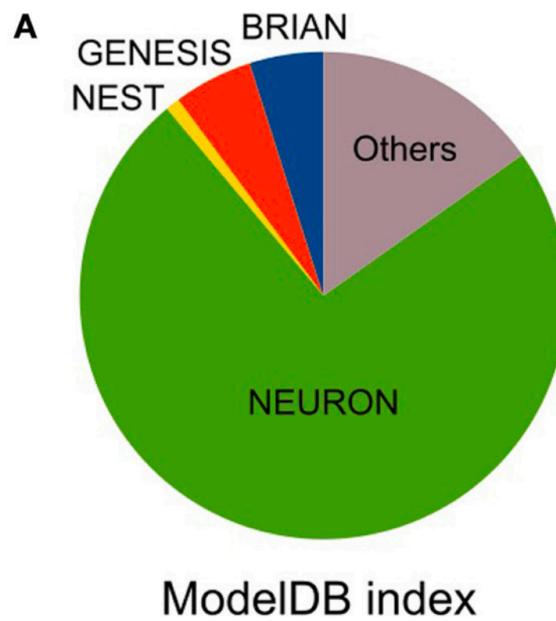
NEURON

# What is what...

- Widely used (new papers each year)
- Many reusable components: channels, cells, ...
- Comprehensive documentation and tutorials



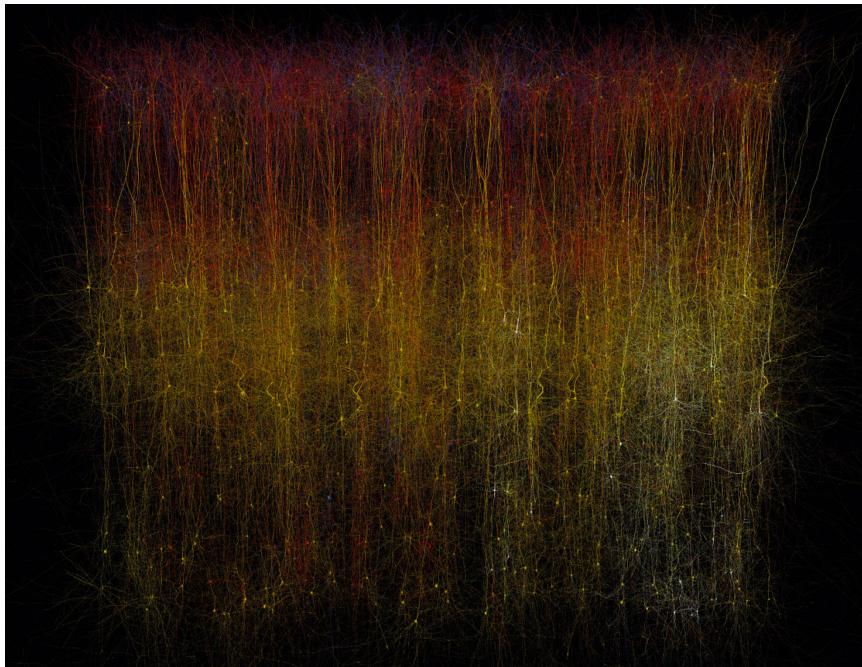
NEURON



C

# What is what...

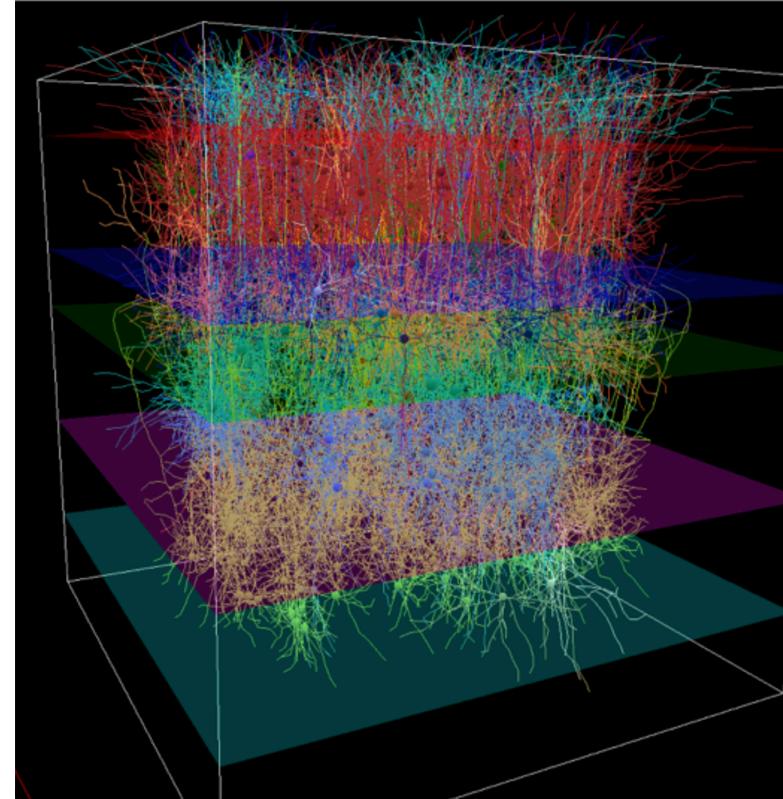
- Used in major simulation projects



**Human Brain Project**  
(Rat S1)

NEURON

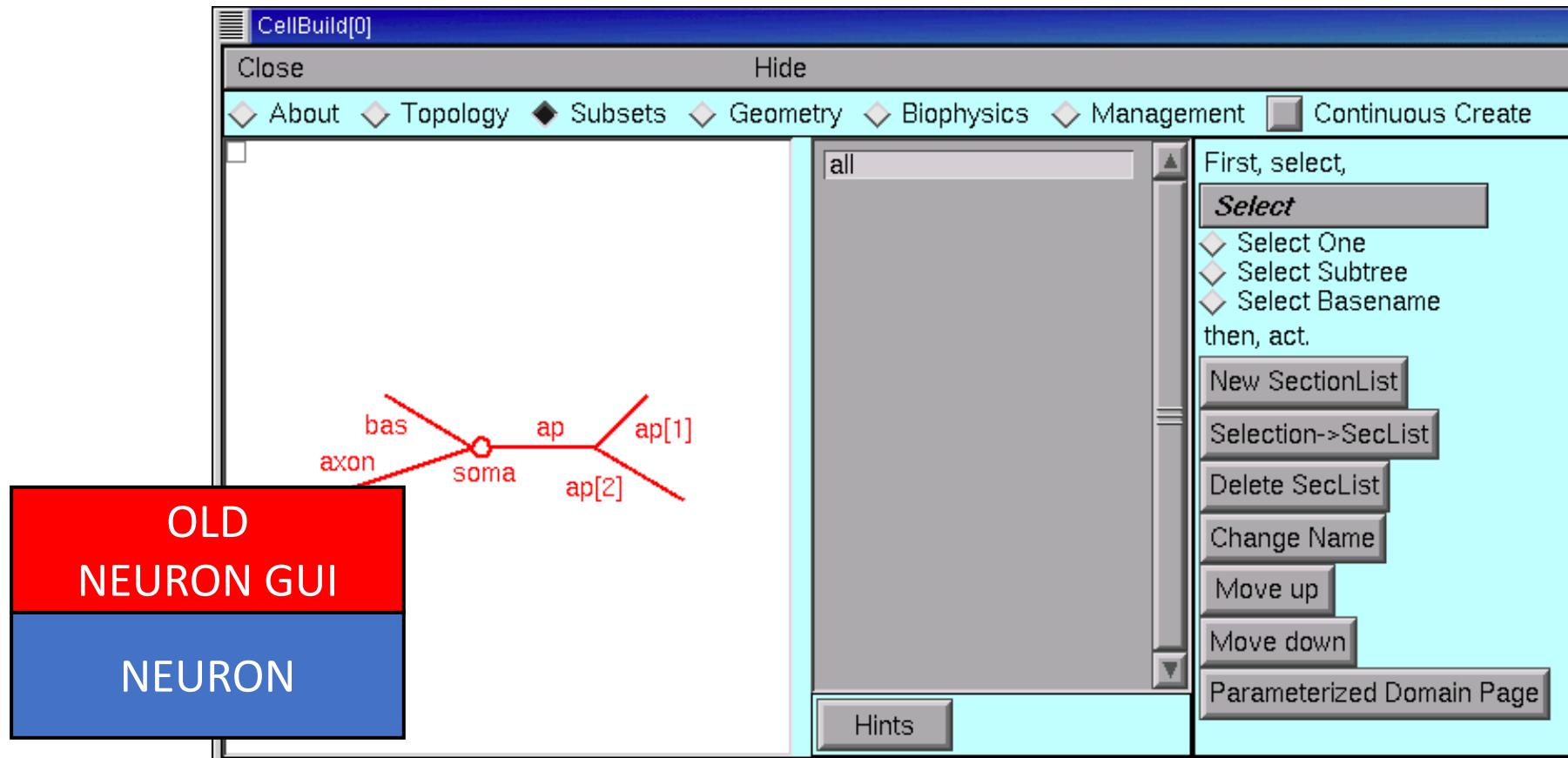
**Allen Brain Institute**  
(Mouse V1)



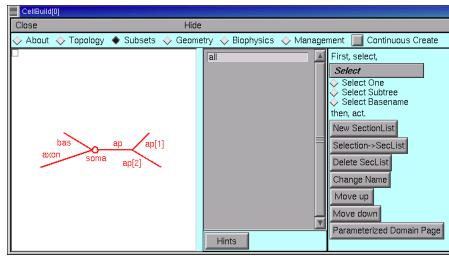
# What is what...



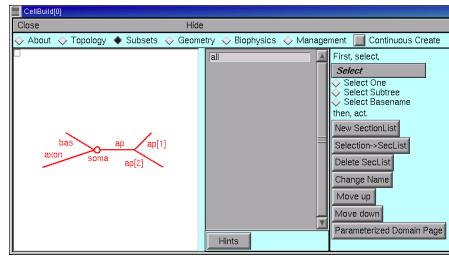
# What is what...



# What is what...



# What is what...

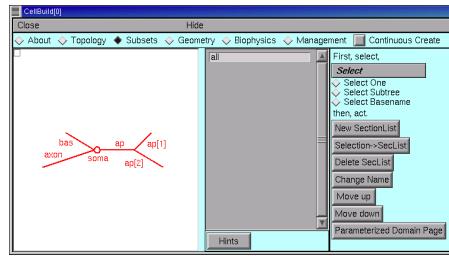


OLD  
NEURON GUI

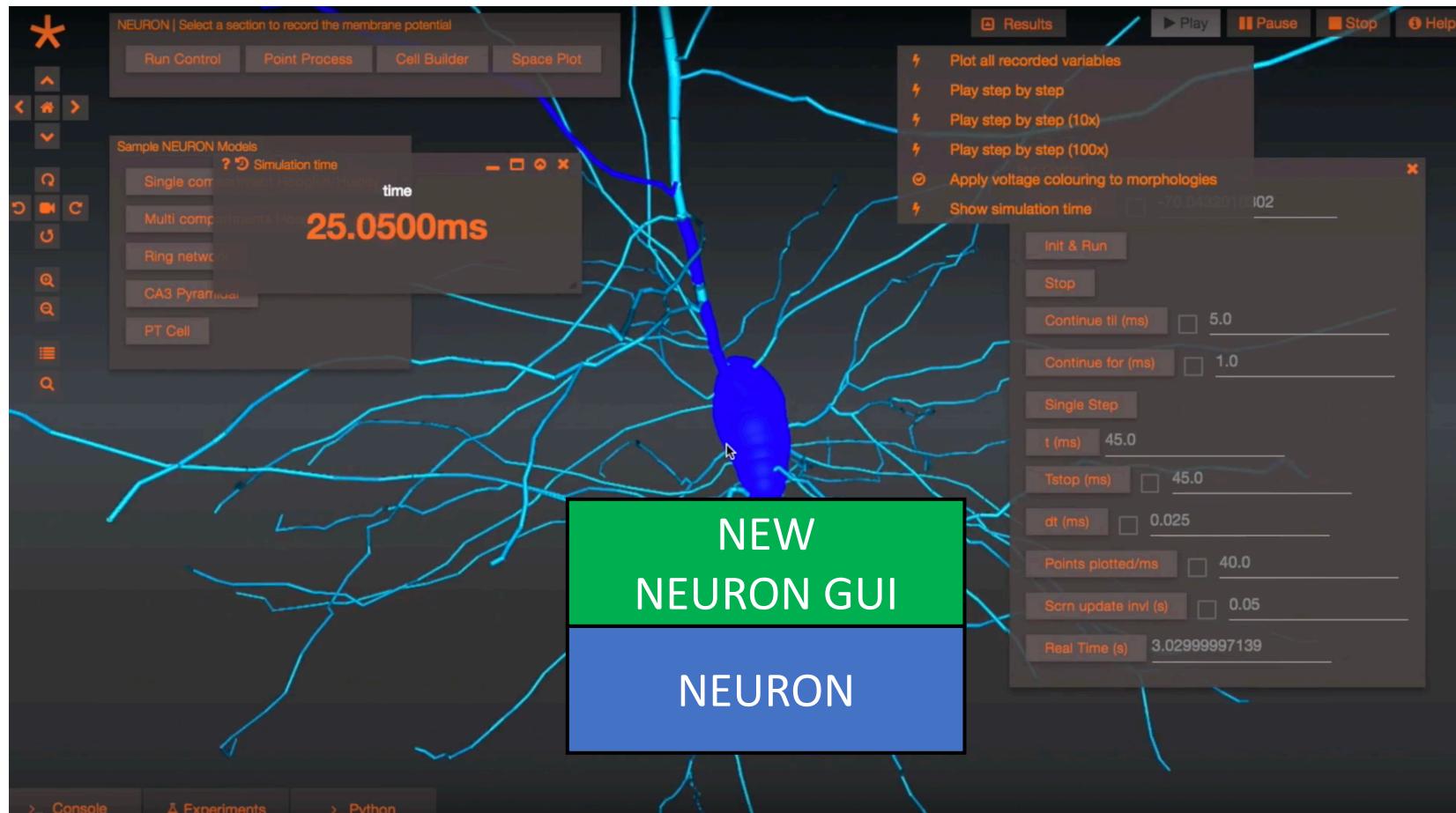
NEURON

NEURON

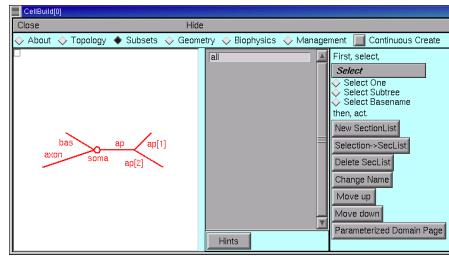
# What is what...



# What is what...



# What is what...



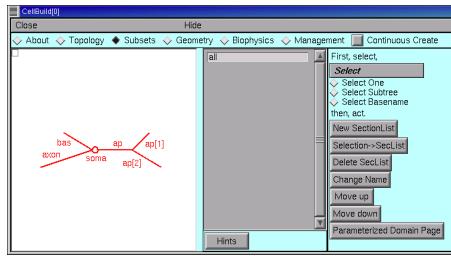
OLD  
NEURON GUI

NEURON

NEW  
NEURON GUI

NEURON

# What is what...



OLD  
NEURON GUI

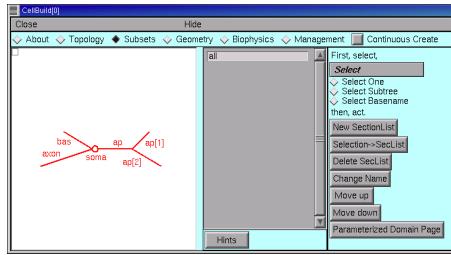
NEURON

NEW  
NEURON GUI

NEURON

NEURON

# What is what...



OLD  
NEURON GUI

NEURON

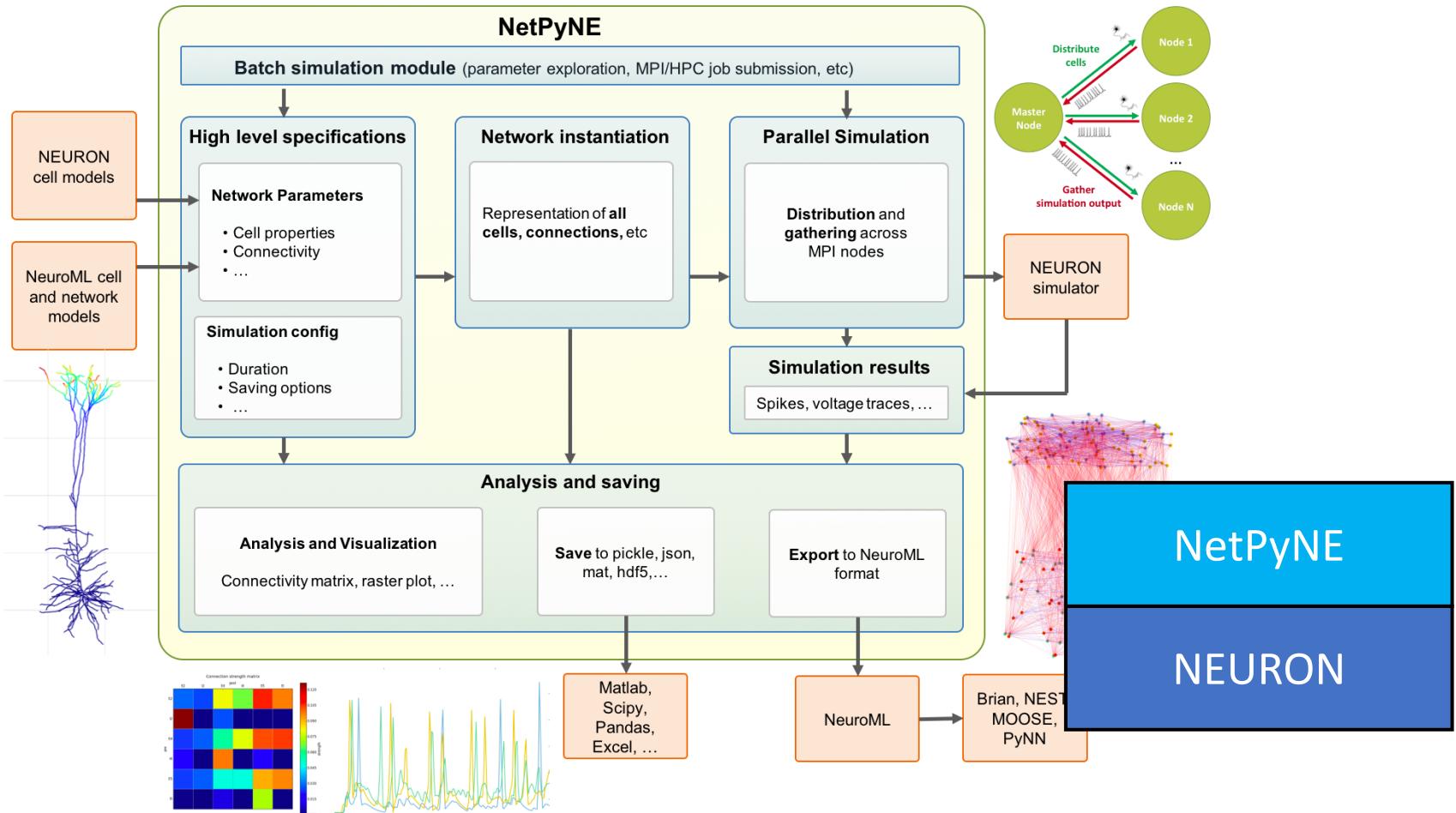
NEW  
NEURON GUI

NEURON

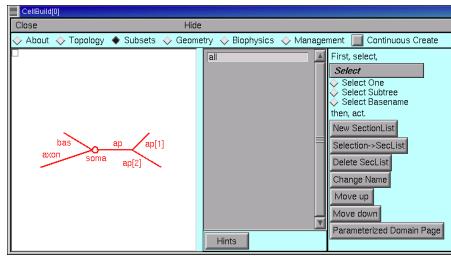
NetPyNE

NEURON

# What is what...



# What is what...



OLD  
NEURON GUI

NEURON

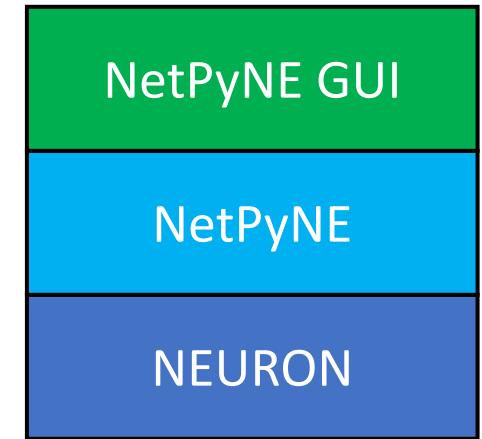
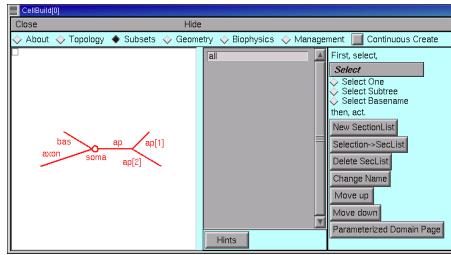
NEW  
NEURON GUI

NEURON

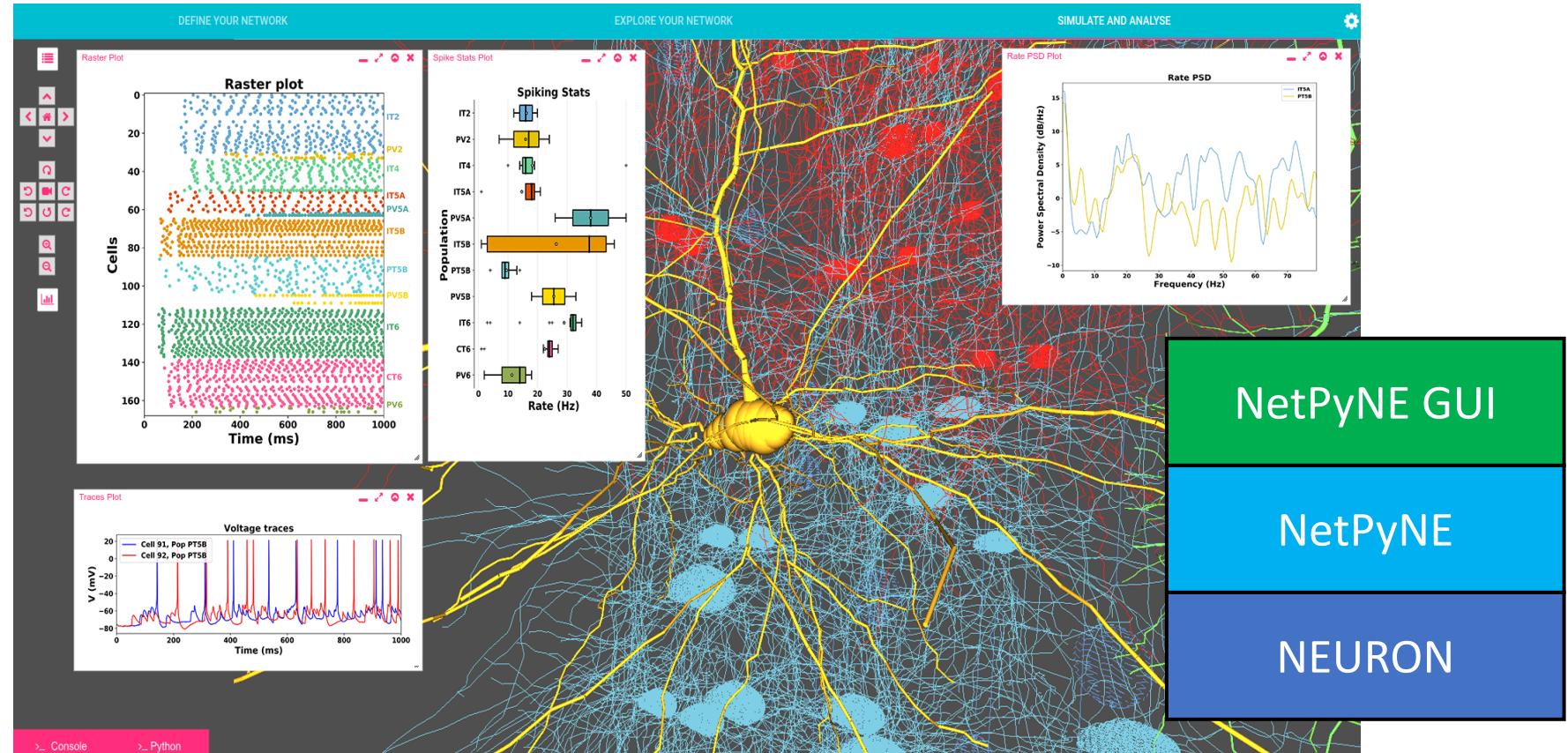
NetPyNE

NEURON

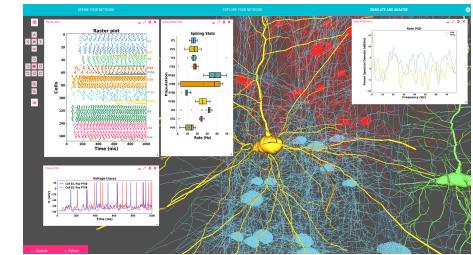
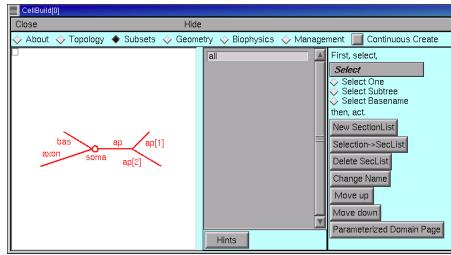
# What is what...



# What is what...



# What is what...



OLD  
NEURON GUI

NEURON

NEW  
NEURON GUI

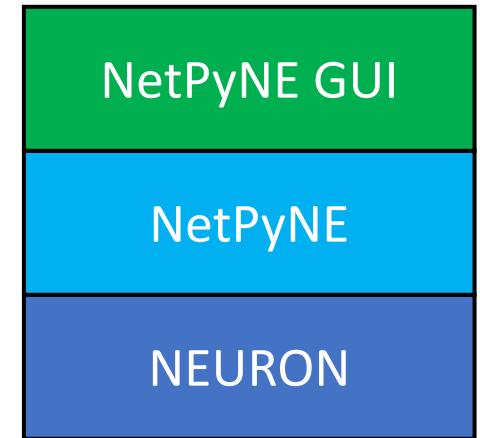
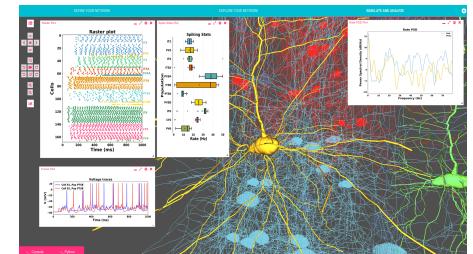
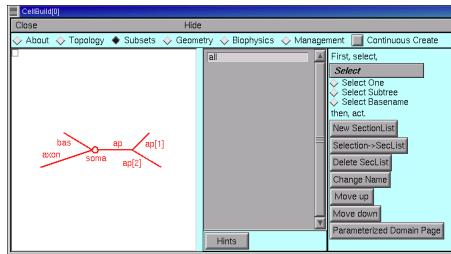
NEURON

NetPyNE GUI

NetPyNE

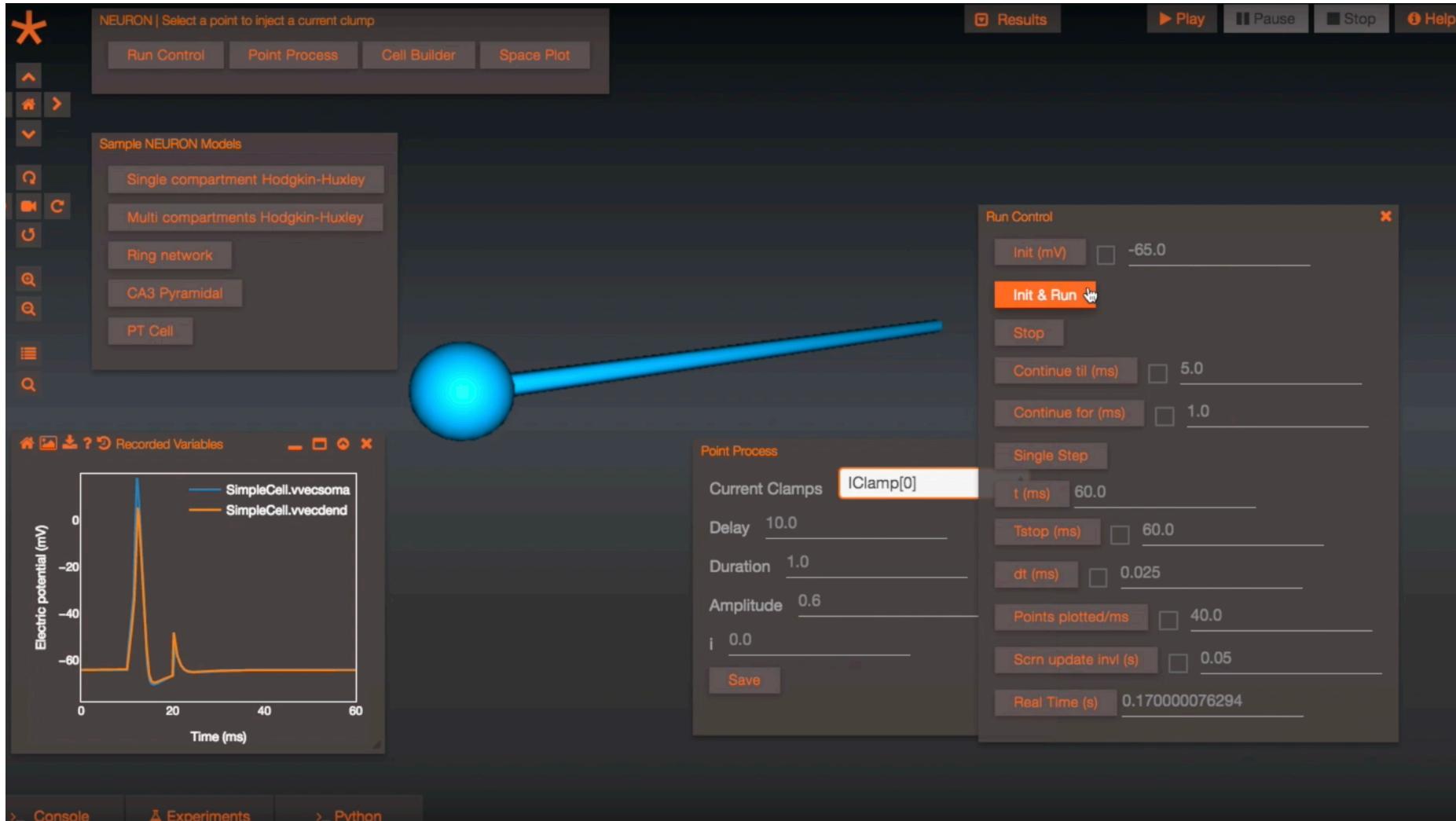
NEURON

# New NEURON GUI

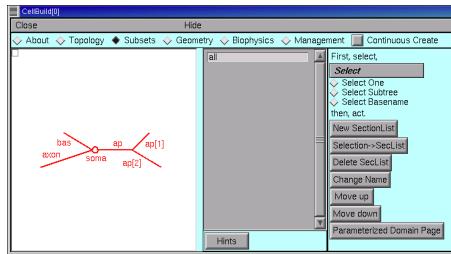


# New NEURON GUI

<https://www.youtube.com/watch?v=CjoA3lTa25I>



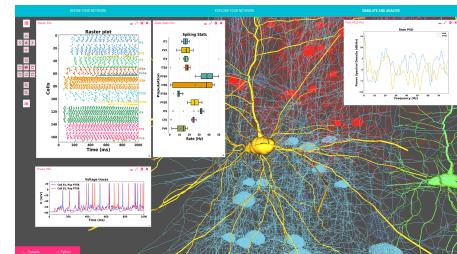
# NetPyNE



OLD  
NEURON GUI  
NEURON



NEW  
NEURON GUI  
NEURON

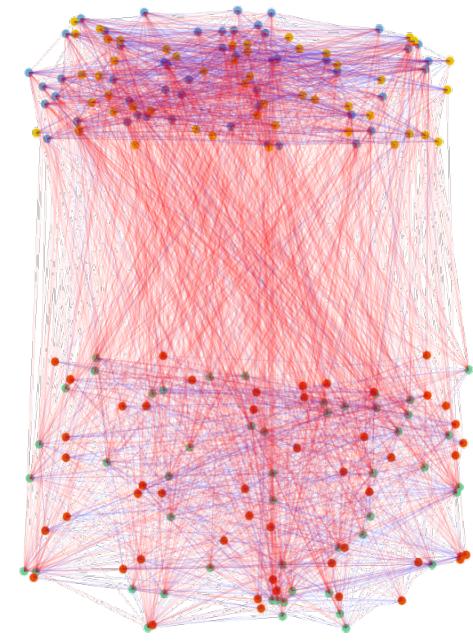
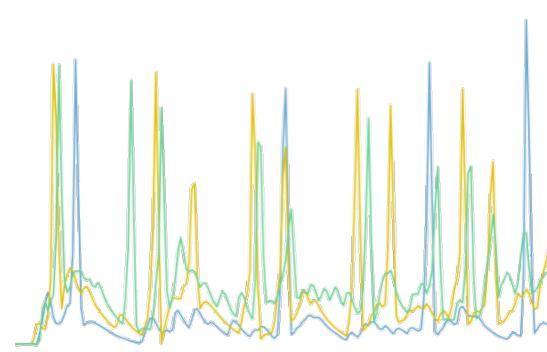
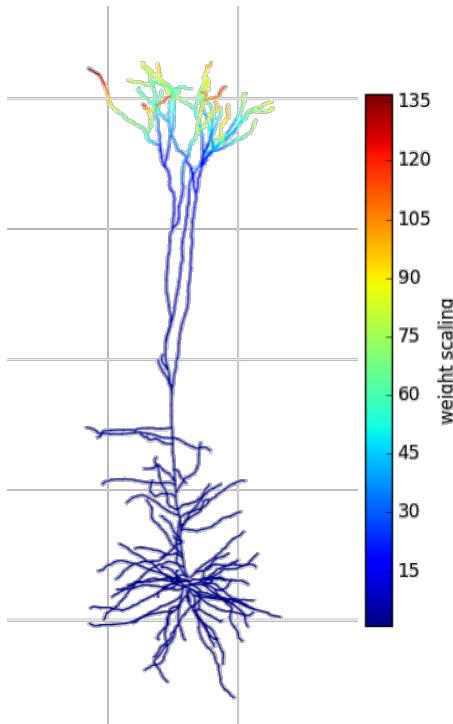


NetPyNE GUI  
NetPyNE  
NEURON



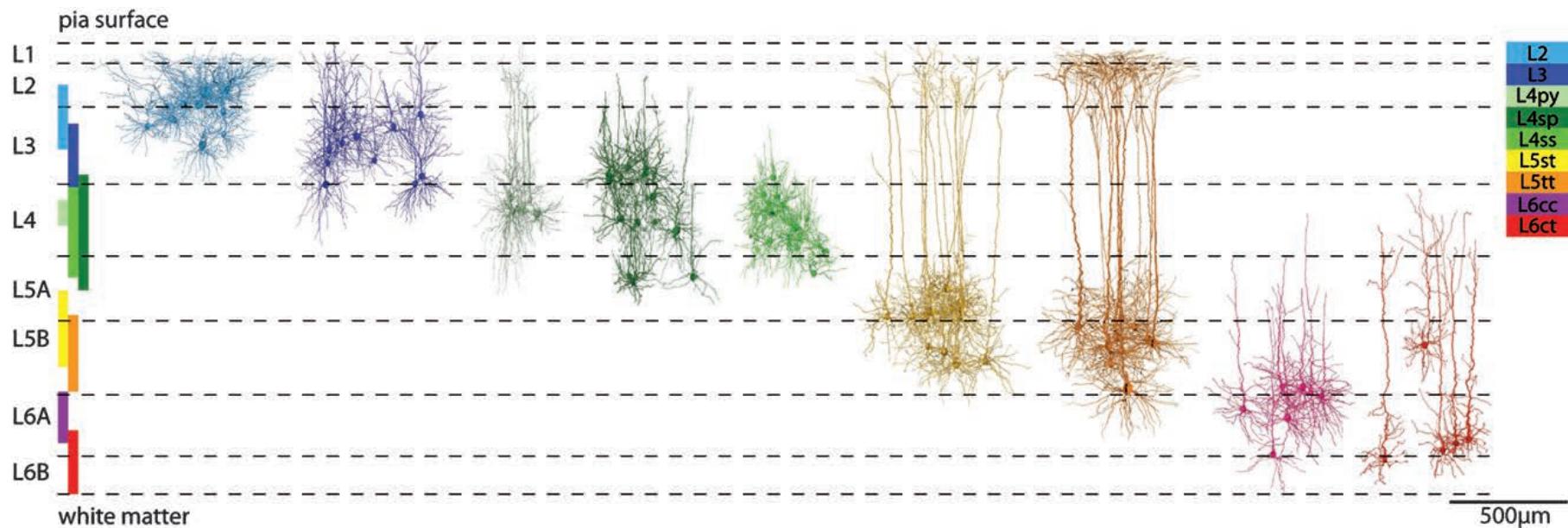
A Python package to facilitate the  
development, simulation and analysis of  
biological neuronal networks in NEURON

[www.netpyne.org](http://www.netpyne.org)



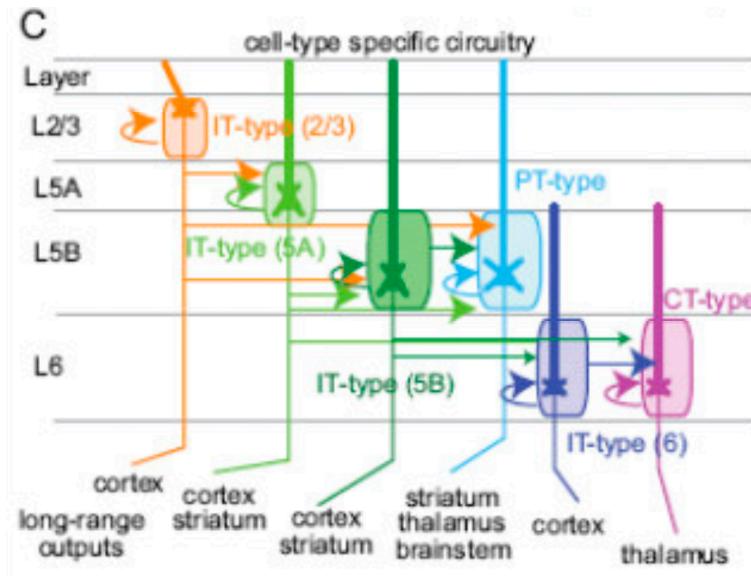
# NetPyNE: Motivation

- Facilitate incorporation of experimental data at multiple scales



# NetPyNE: Motivation

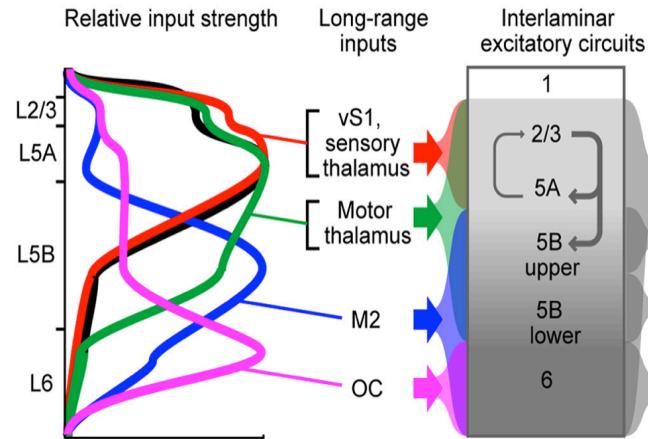
- Facilitate incorporation of experimental data at multiple scales



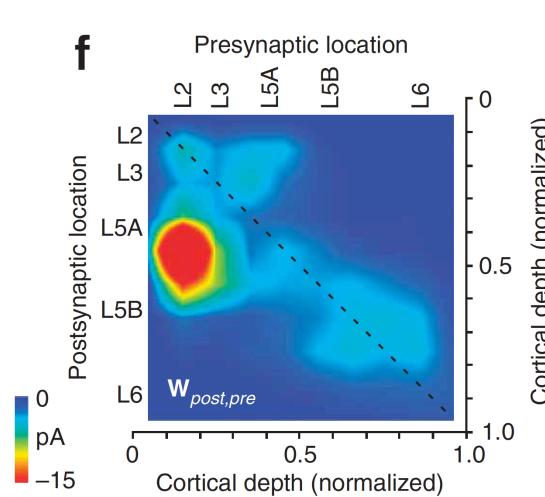
# NetPyNE: Motivation

- Facilitate incorporation of experimental data at multiple scales

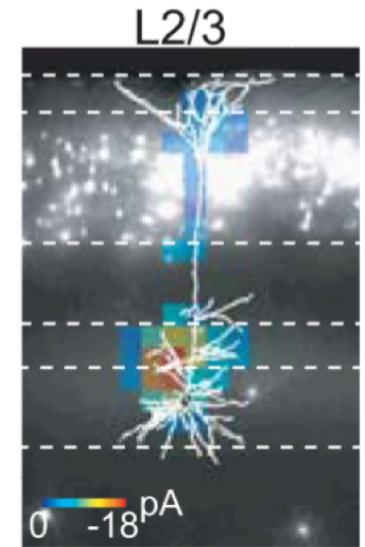
Long-range inputs



Local microcircuits



Dendritic inputs



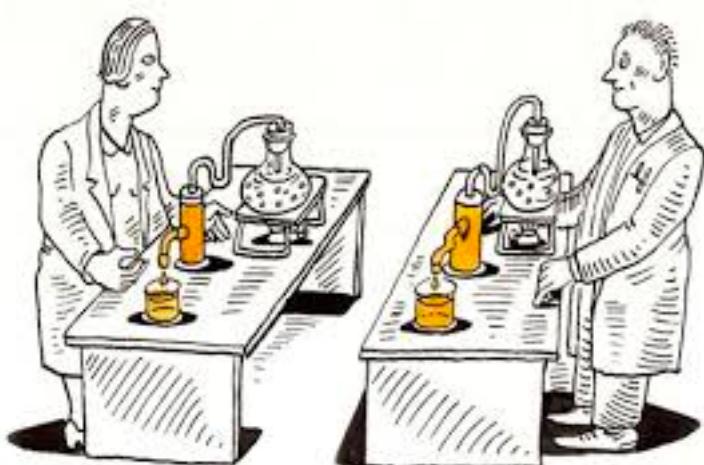
# NetPyNE: Motivation

- 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'
```

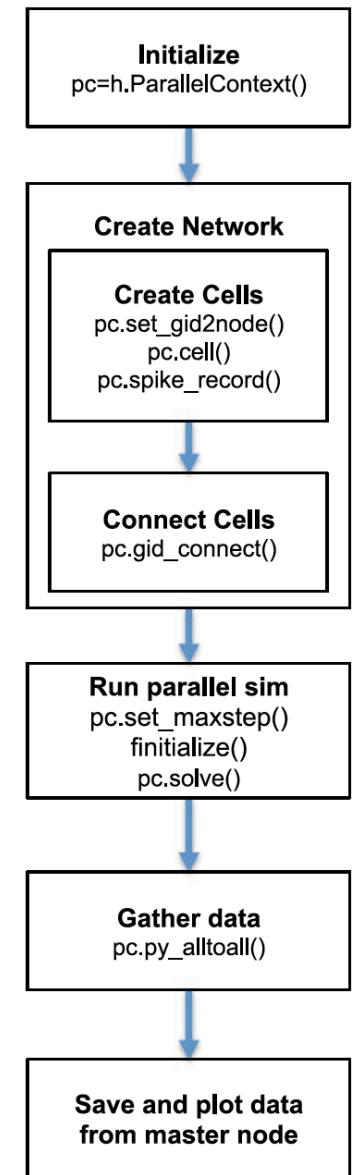
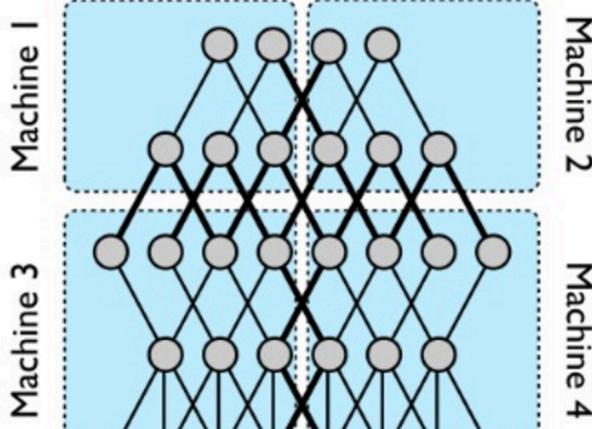


**Replicate:** get same thing to run again

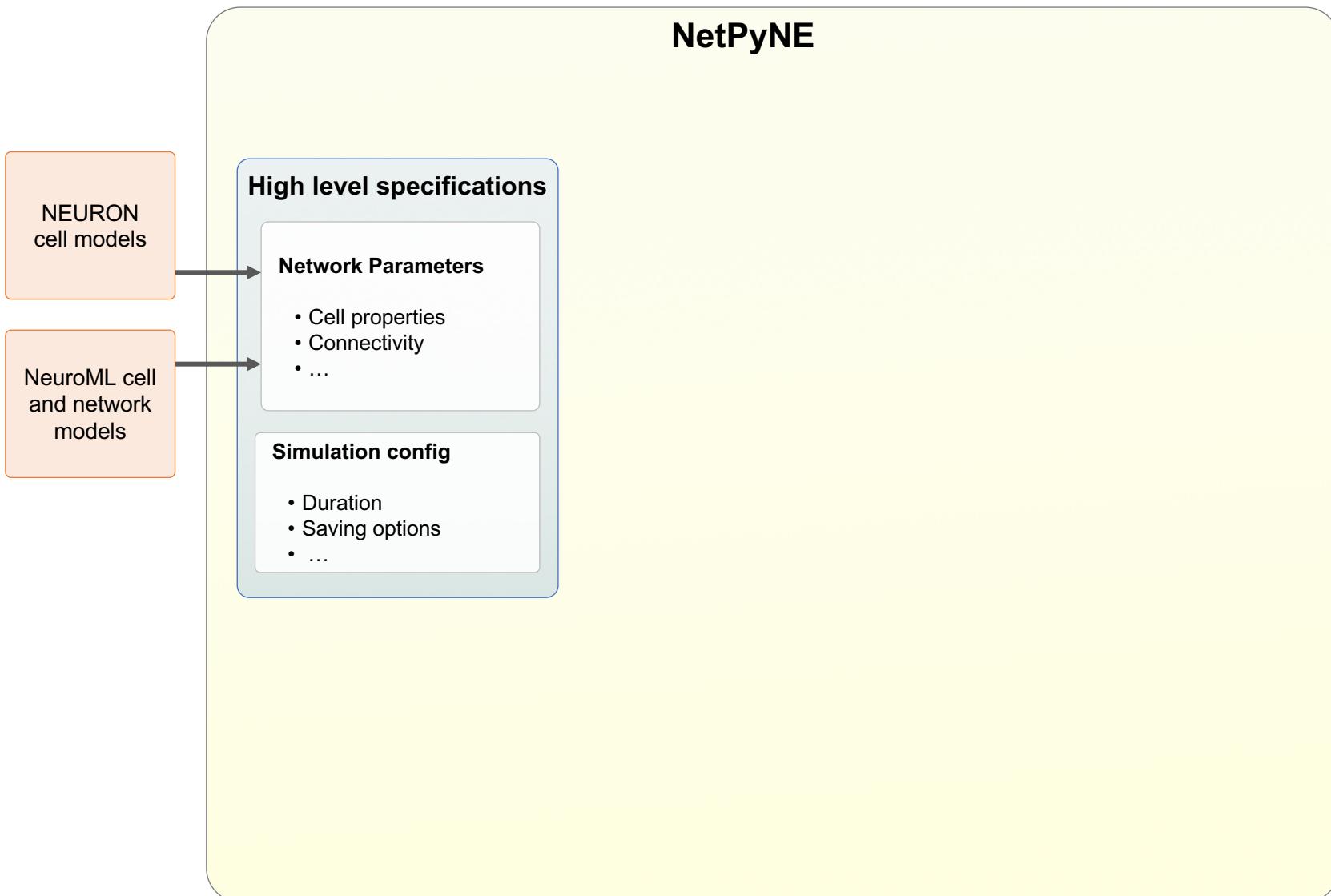
**Reproduce:** make it yourself

# NetPyNE: Motivation

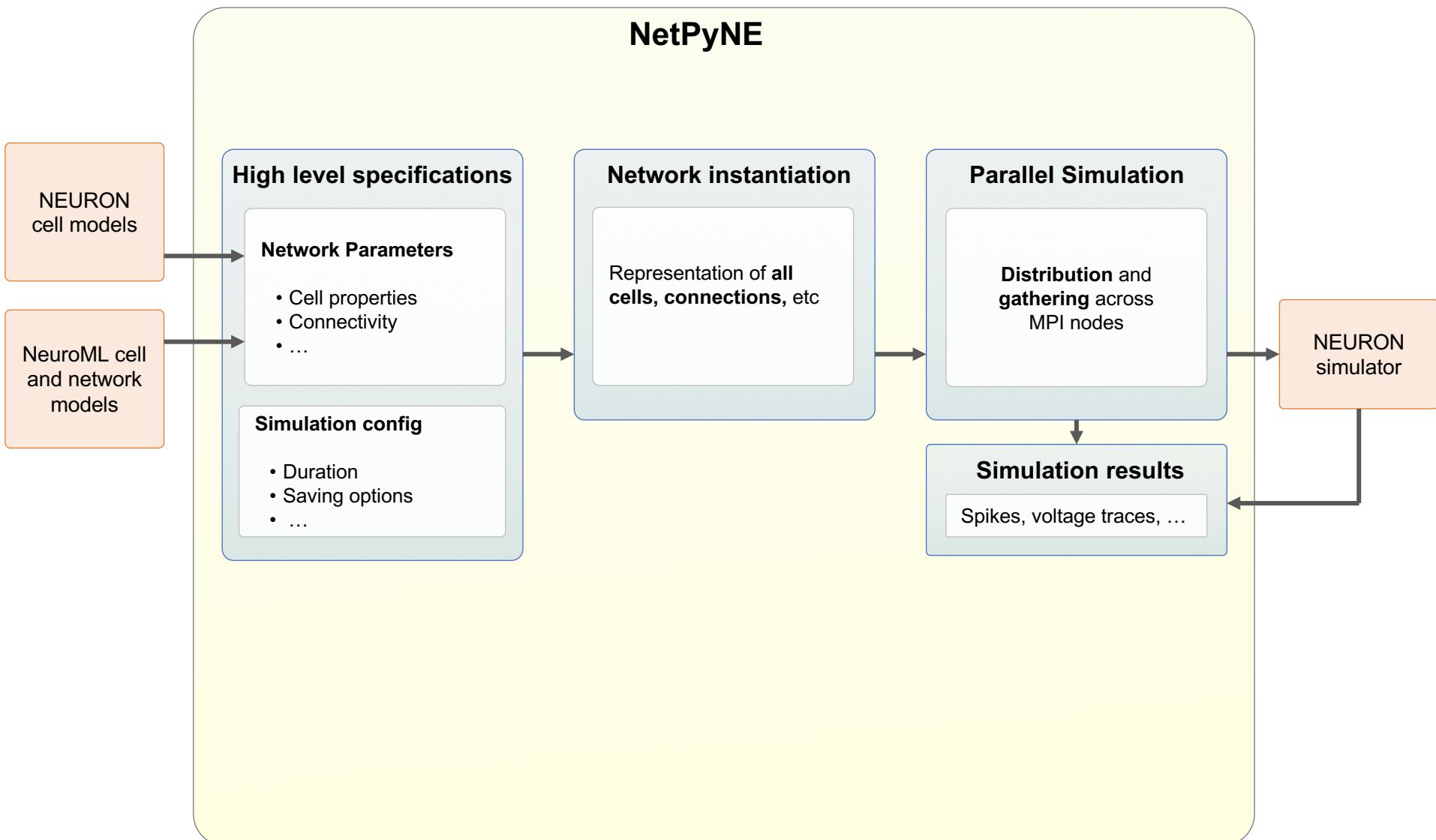
- Facilitate model parallelization (HPCs)
- Batch parameter exploration/optimization



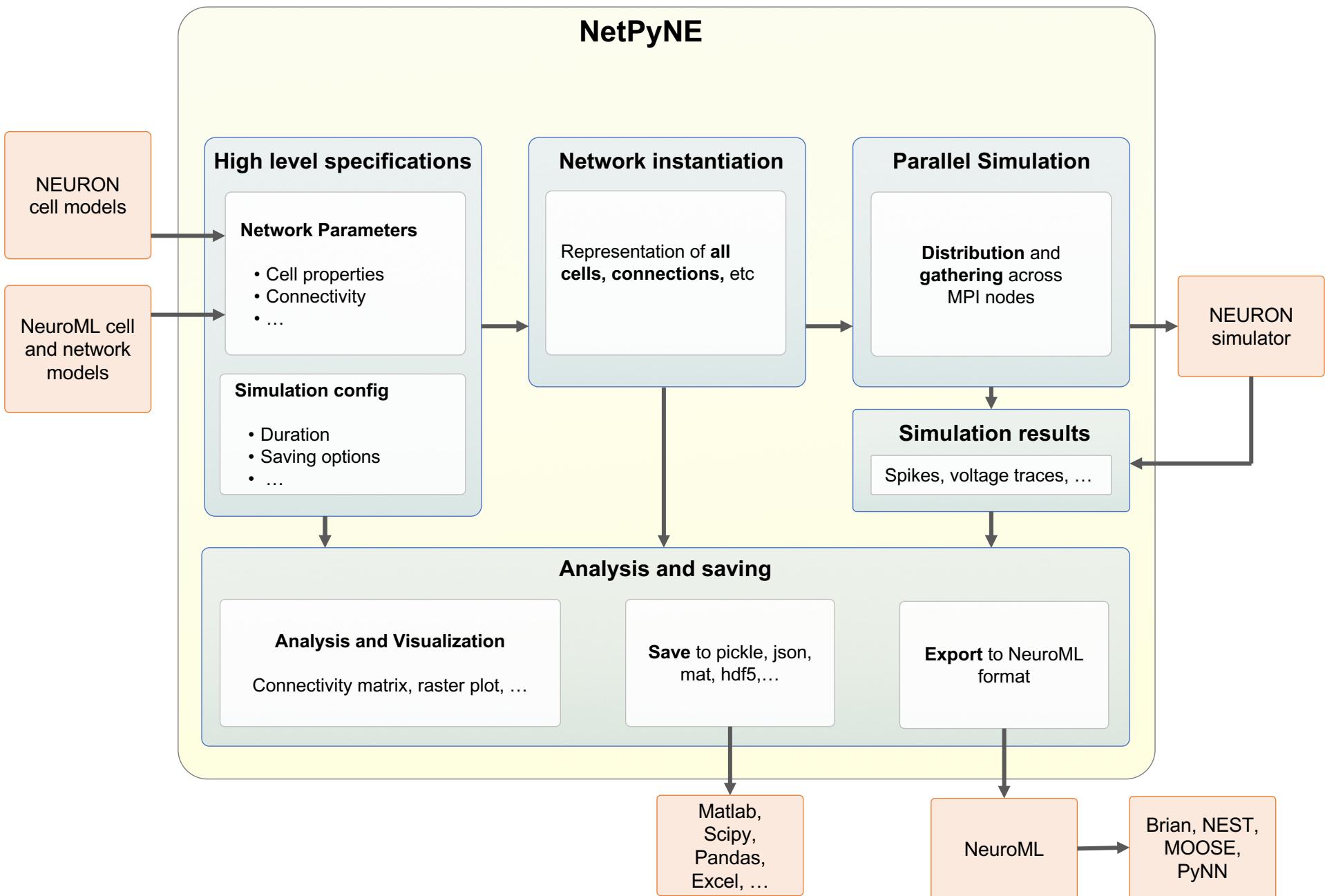
# NetPyNE



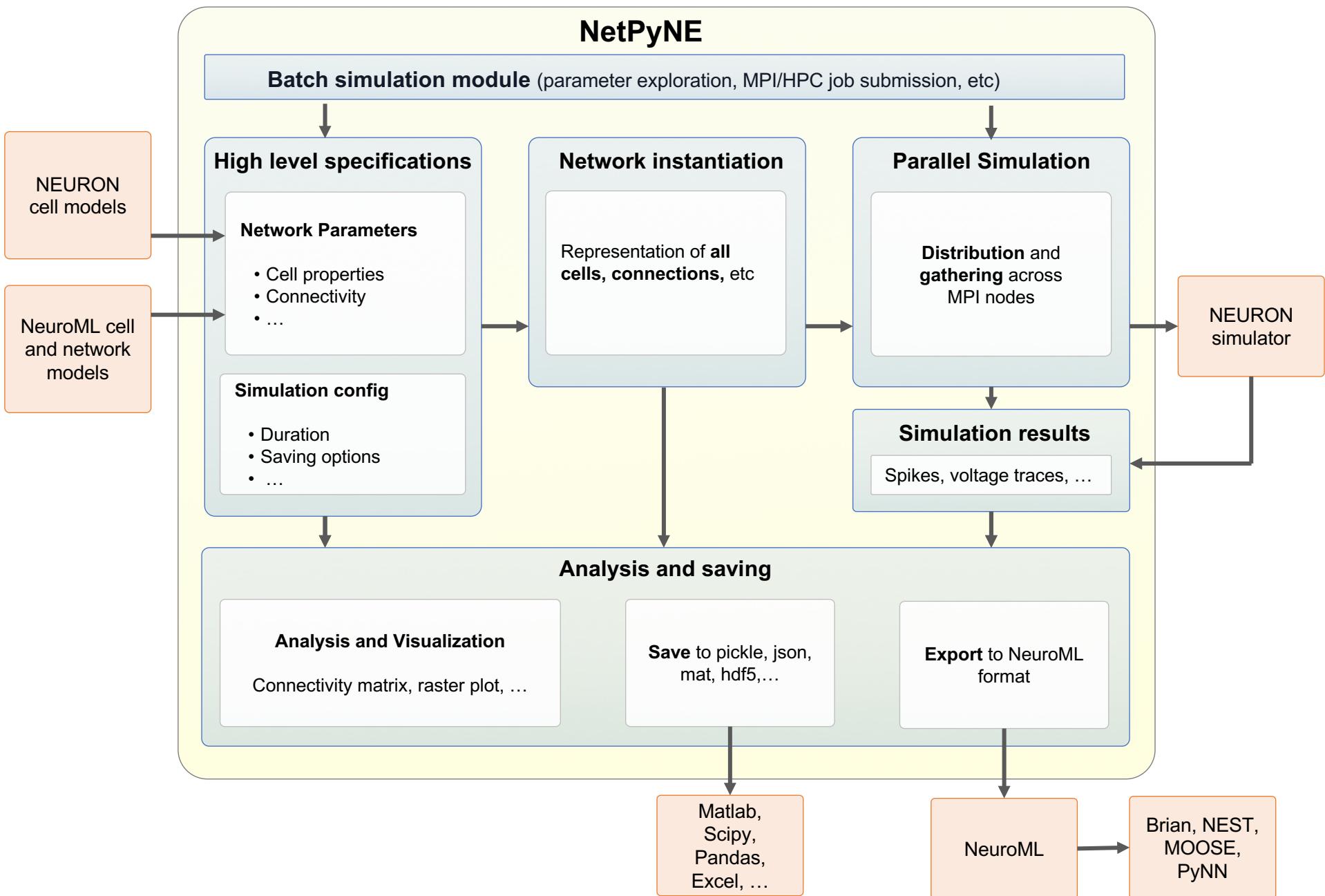
# NetPyNE



# NetPyNE

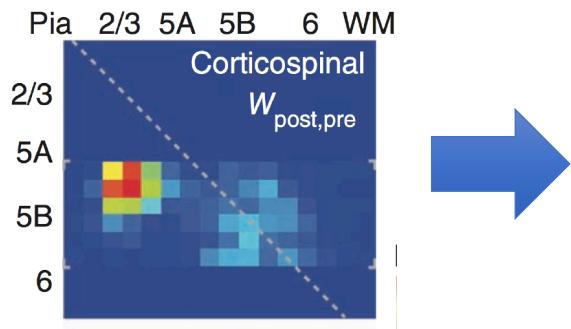


# NetPyNE



# NetPyNE: 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.



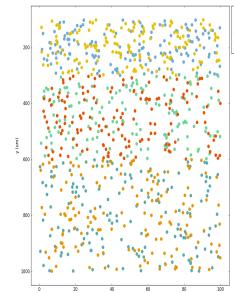
```
connParams['bin3->CSP'] = {  
    'preConds': {'y': [100, 150]},  
    'postConds': {'pop': 'CSP'},  
    'probability': 0.15,  
    'weight': 0.4,  
    'delay': 5,  
    'synMech': 'AMPA'}
```

NetPyNE facilitates  
building models  
based on  
experimental data

# NetPyNE: High level specifications

- User can define:

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



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



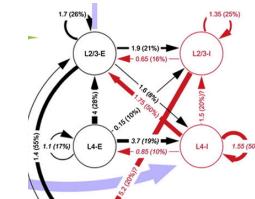
- **Synaptic mechanisms:** Time constants, reversal potential, implementation, ...



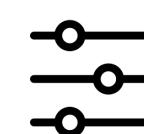
- **Stimulation:** Spike generators, current clamps, spatiotemporal properties, ...



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

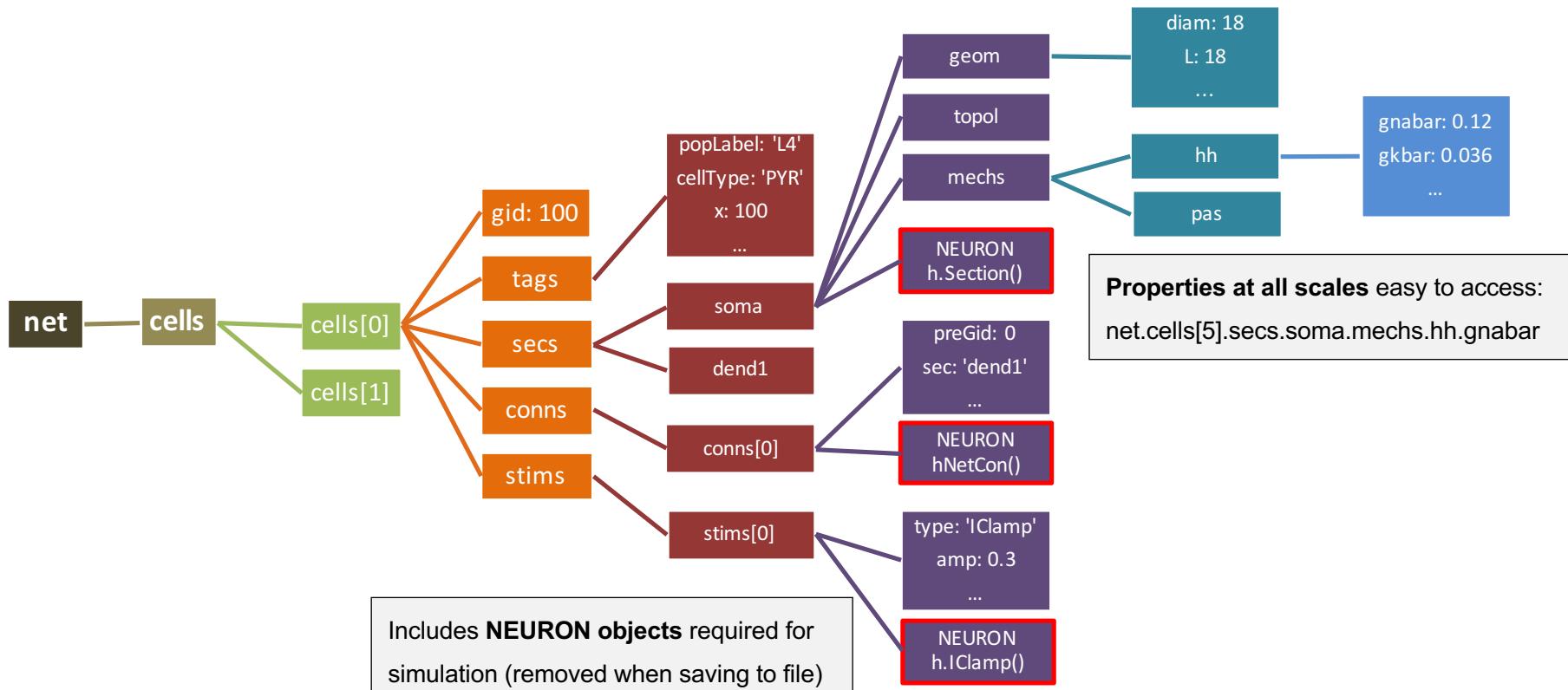
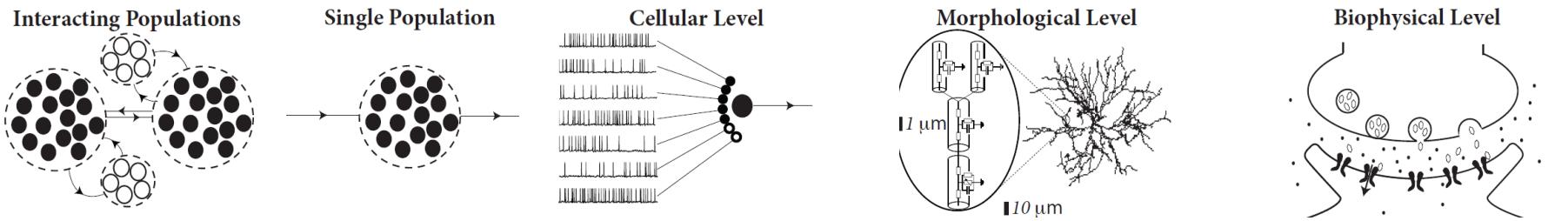


- **Simulation configuration:** duration, saving and analysis, graphical output, ...



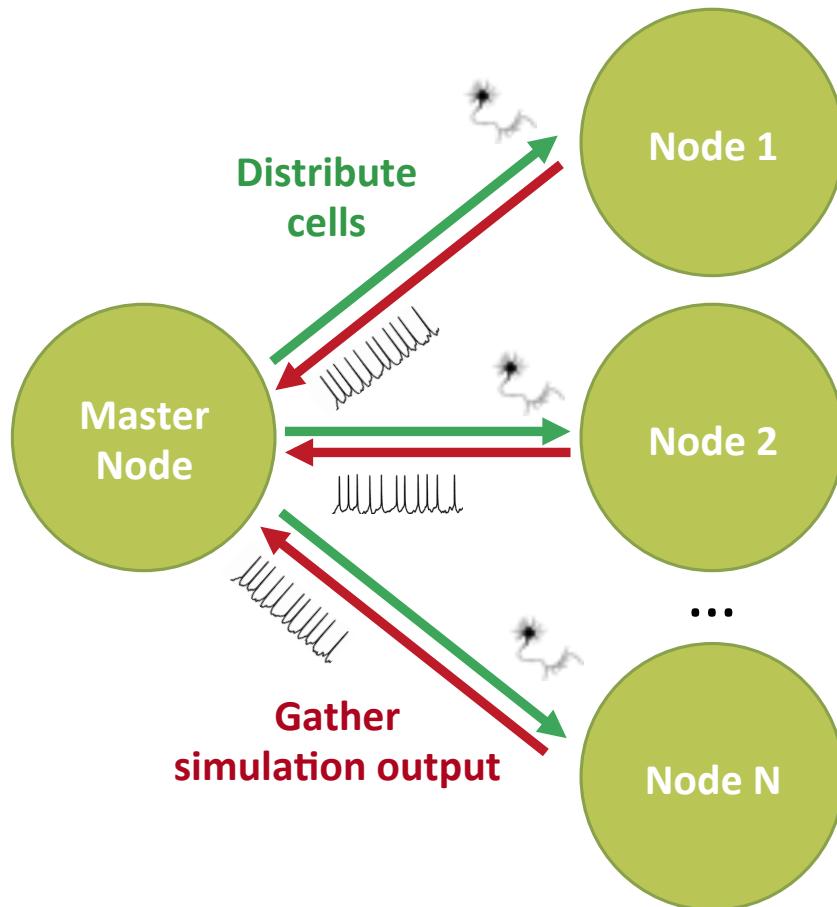
# NetPyNE: Network Instantiation

- Network is created as Python-based **standardized hierarchical data structure**.



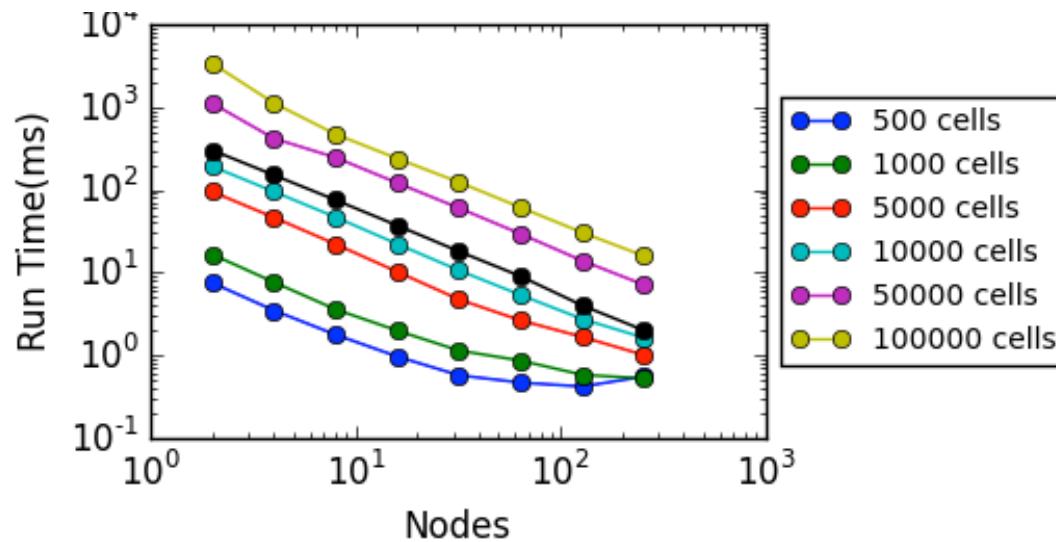
# NetPyNE: Parallel Simulation

- ❑ 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.



# NetPyNE: Parallel simulation

- NetPyNE available on **the Neuroscience Gateway (NSG)** supercomputing platform.



Simulation **run time** as a function of number of cells and number of nodes (*Neural Comput*, 2016).

Results obtained using **NetPyNE** on **NSG**.

# NetPyNE: Batch parallel simulations

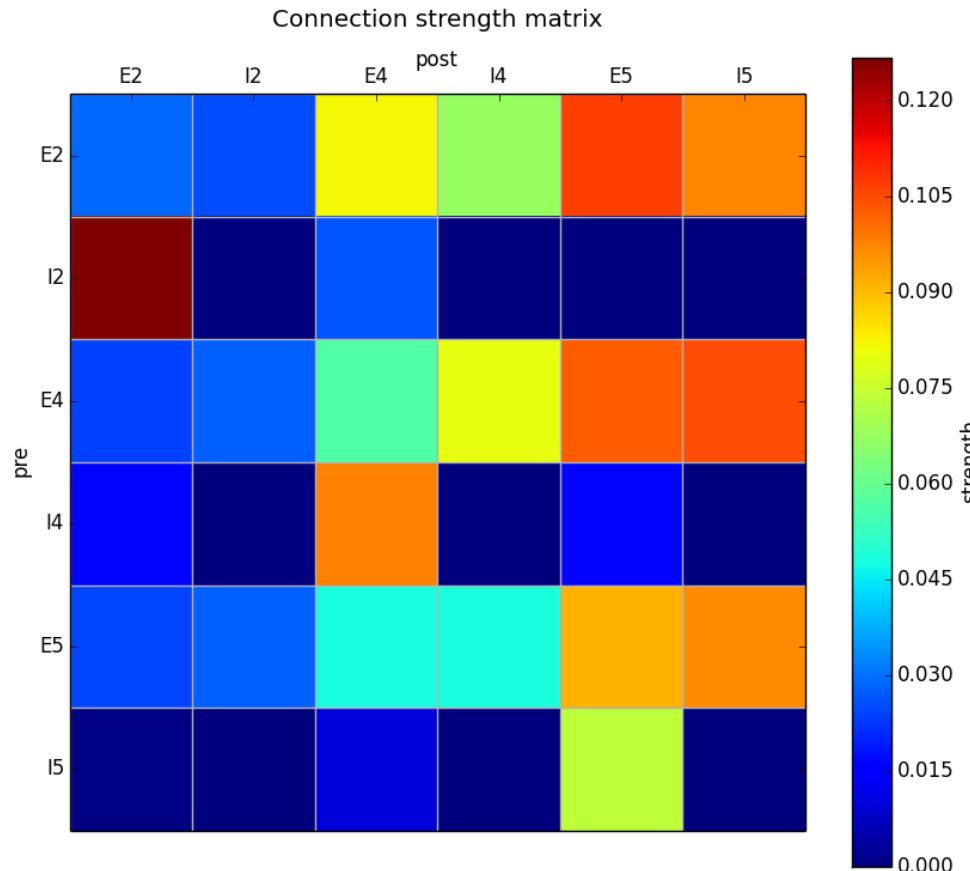
- **Easy specification** of parameters and range of values to explore in batch simulations.
- **Pre-defined, configurable** setups to automatically **submit jobs** in multicore machines (Bulletin board) or supercomputers (SLURM or PBS Torque)



**SDSC** SAN DIEGO  
SUPERCOMPUTER CENTER

# NetPyNE: Analysis

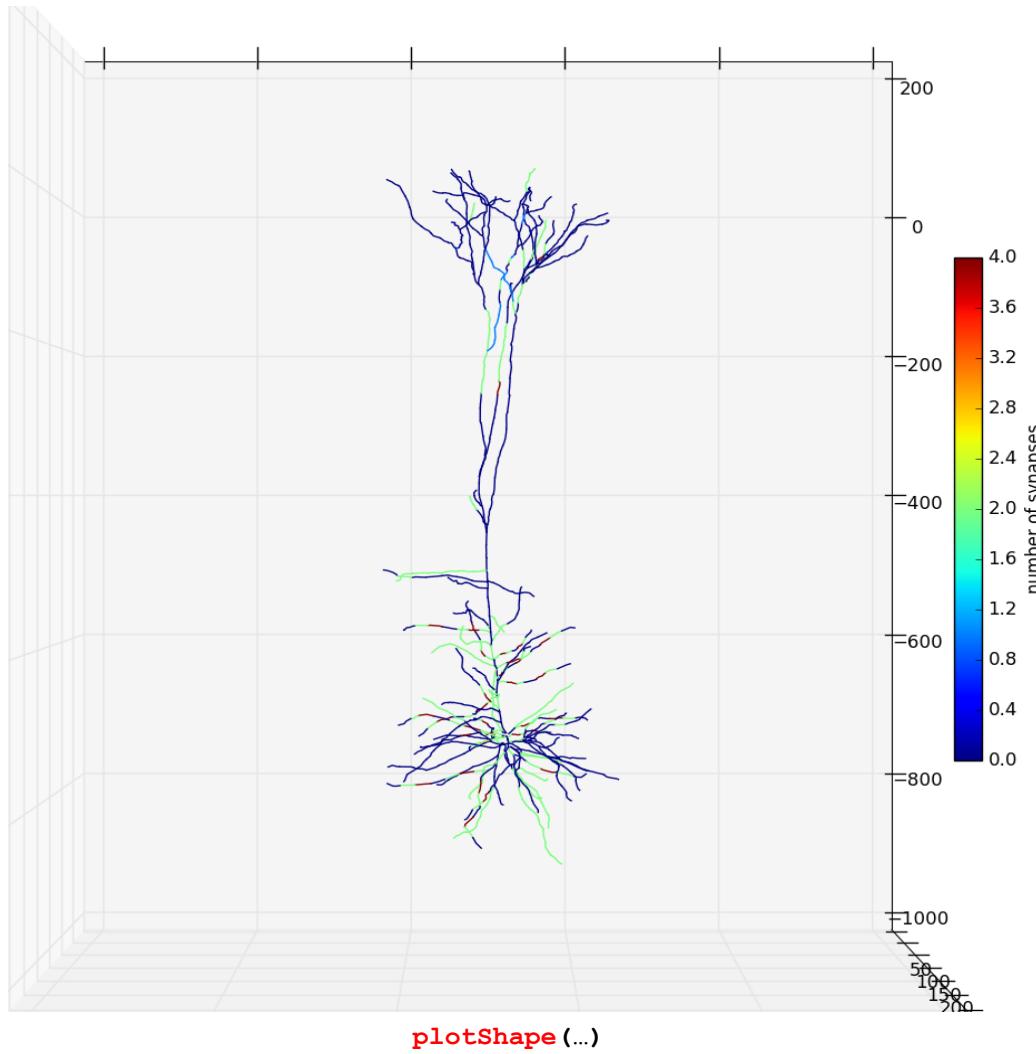
- **Connectivity matrix** at cell or population level (weights, num connections, probability,...)



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

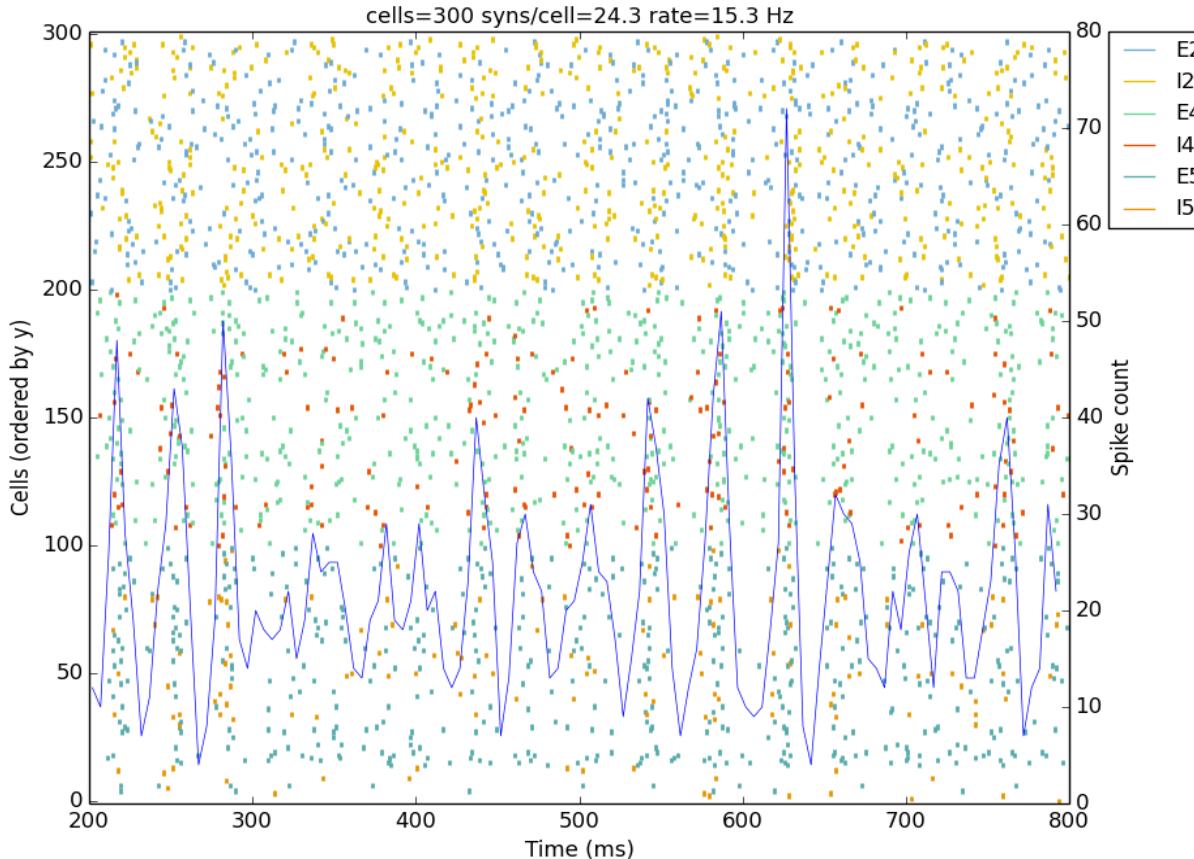
# NetPyNE: Analysis

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



# NetPyNE: Analysis

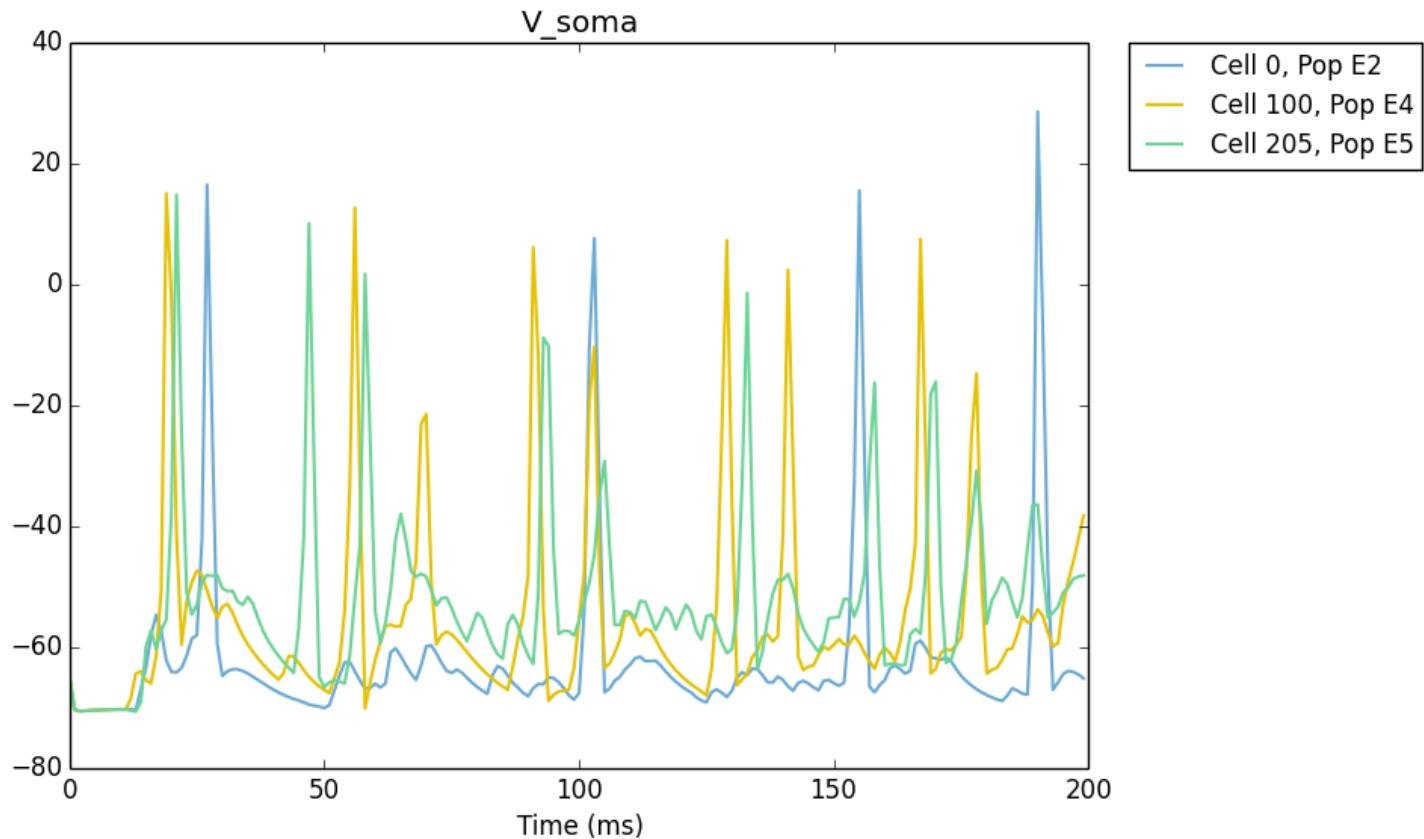
- Easy-to-use functions for **analysis and plotting** of network and simulation output
  - **Raster plot** of any subset of cells
  - **Spike histogram** of populations or subsets of cells



```
plotRaster(include=['allCells'], timeRange=[200,800], orderBy='y',
orderInverse=True, spikeHist='overlay', spikeHistBin=5)
```

# NetPyNE: Analysis

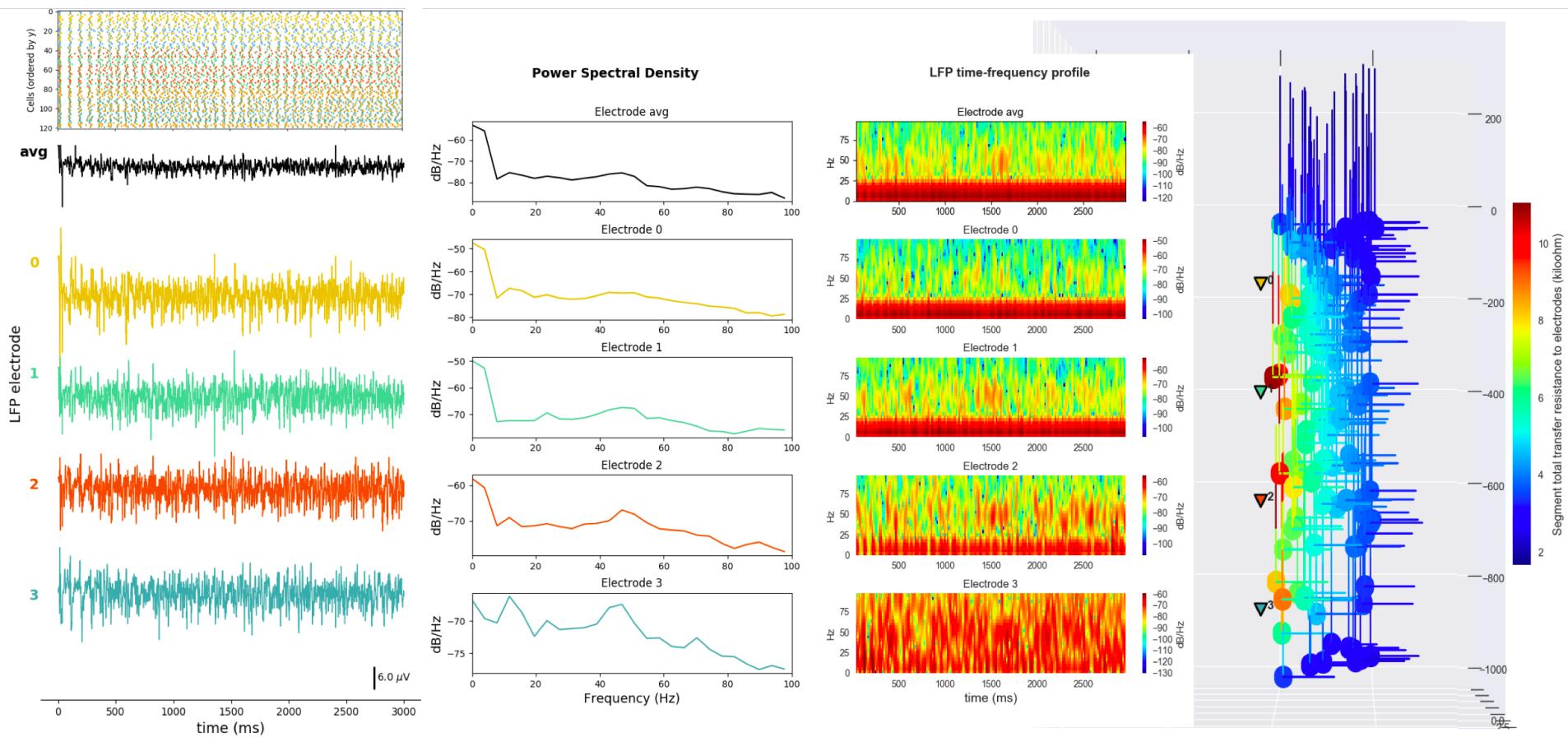
- Intrinsic cell variables (voltages, currents, conductance) **trace plots**



```
plotTraces(include=[('E2',0), ('E4',0), ('E5',5)],  
           timeRange=[0,200], overlay=True, oneFigPer='trace')
```

# NetPyNE: Analysis

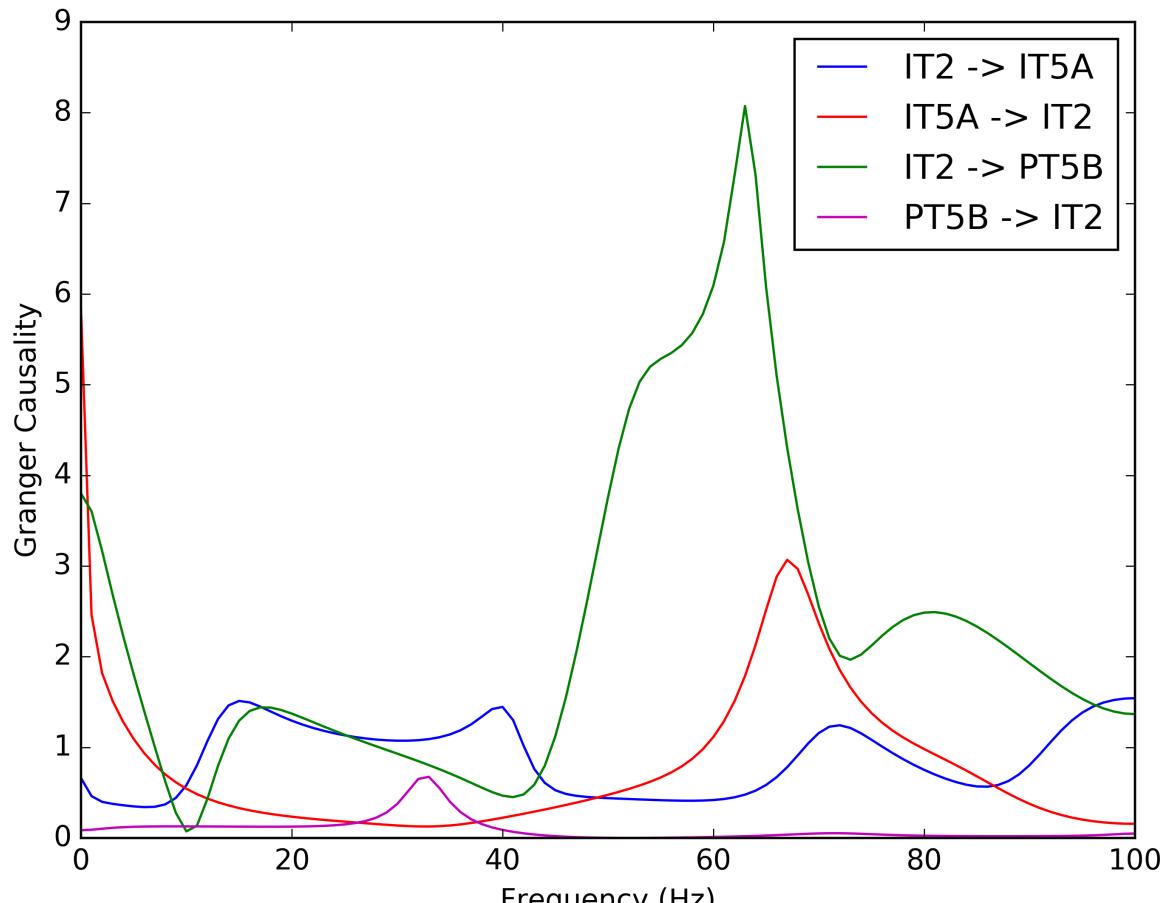
- LFP time-series, PSD, spectrogram and electrode locations



`plotLFP(...)`

# NetPyNE: Analysis

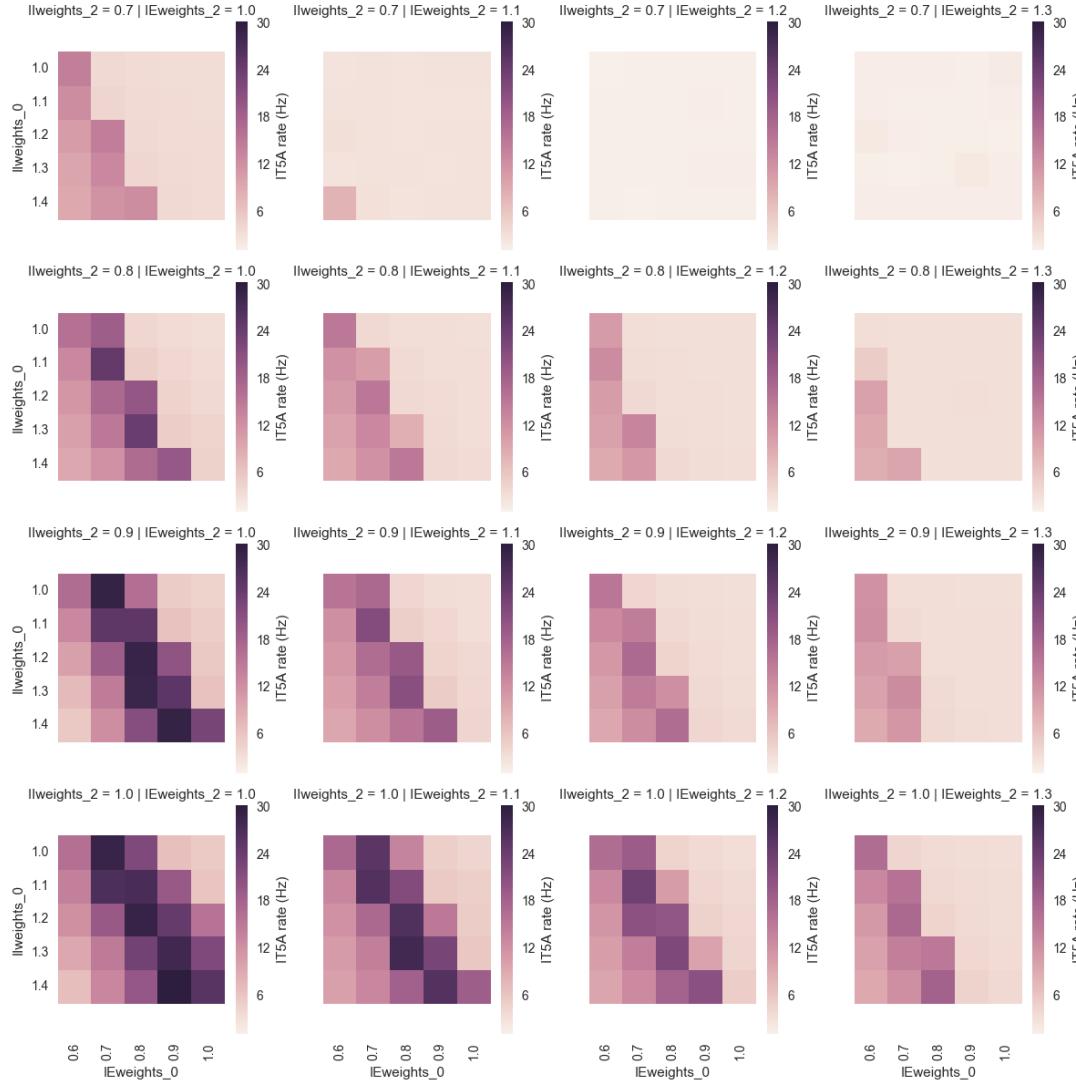
- Spectral **Granger causality**
- Normalized transfer entropy



`plotGranger(...)`

# NetPyNE: Analysis

- Analysis and visualization of multidimensional batch simulation results.



# NetPyNE: 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}



HDF

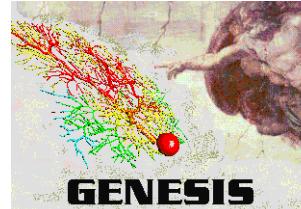
# NetPyNE: Data saving and exporting



Import/export to standard format



Import/export to other simulators



# NetPyNE: Documentation and Tutorials

[www.netpyne.org](http://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



The screenshot shows the NetPyNE forum page on the website [www.neuron.yale.edu](http://www.neuron.yale.edu). The page has a blue header with the site logo and navigation links for "Quick links", "FAQ", "Register", and "Login". Below the header, there's a breadcrumb trail: "Board index < Tools of interest to NEURON users < NetPyNE". The main content area is titled "NetPyNE" and shows a "Moderator: tom\_morse". It features two sections: "ANNOUNCEMENTS" and "TOPICS".

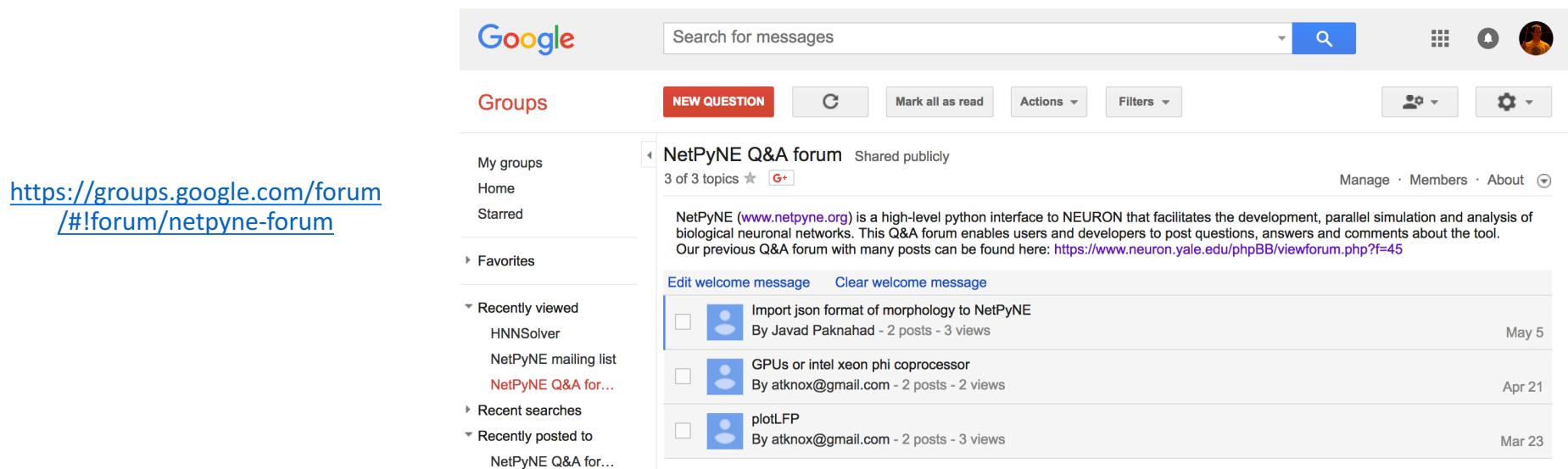
**ANNOUNCEMENTS**

	REPLIES	VIEWS	LAST POST
<a href="#">VERSION RELEASES</a> by salvadord » Fri Jun 09, 2017 10:41 pm	12	7554	by bremen Sat Apr 28, 2018 4:05 pm
<a href="#">Welcome to the NetPyNE Forum!</a> 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
<a href="#">Spike source and target sections</a> by salvadord » Mon Nov 27, 2017 12:03 pm	17	4342	by bremen Sat May 12, 2018 12:07 pm
<a href="#">Import json format of morphology to NetPyNE</a> by Javad » Fri May 04, 2018 3:02 pm	2	75	by ted Sun May 06, 2018 1:30 pm
<a href="#">Slow speed to save sim results</a> by bremen » Sat Apr 21, 2018 10:32 am	2	51	by bremen Sat Apr 28, 2018 3:15 pm
<a href="#">Field names are restricted to 31 characters</a> by bremen » Sat Mar 24, 2018 1:36 pm	2	55	by bremen Sun Mar 25, 2018 6:21 am
<a href="#">plotLFP</a> by atknox » Fri Mar 02, 2018 6:44 pm	1	72	by salvadord Wed Mar 21, 2018 6:20 pm
<a href="#">Mat file not saved properly in batch functions</a> by Vittorio » Thu Feb 15, 2018 10:58 am	1	91	by salvadord Thu Feb 15, 2018 11:30 am
<a href="#">Gap junction support - parallel simulation?</a> 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>



The screenshot shows the Google Groups forum for the "NetPyNE Q&A forum". The page has a header with the Google logo and a search bar. The main content area shows the forum's details and a list of recent posts.

**Groups**

[NEW QUESTION](#) [C](#) [Mark all as read](#) [Actions](#) [Filters](#) [Members](#) [About](#)

**NetPyNE Q&A forum** Shared publicly  
3 of 3 topics [G+](#)

NetPyNE ([www.netpyne.org](http://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](#)

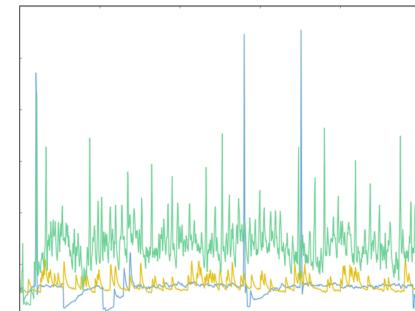
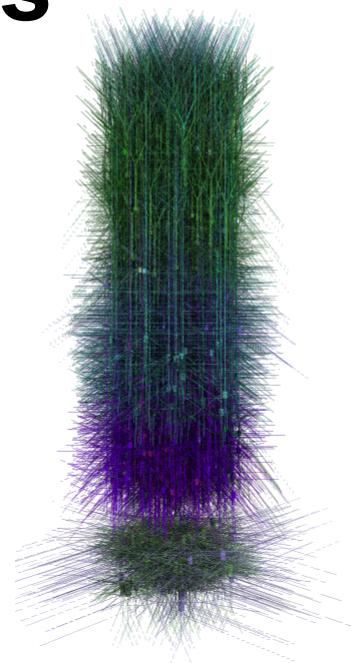
<input type="checkbox"/>  Import json format of morphology to NetPyNE By Javad Paknahad - 2 posts - 3 views	May 5
<input type="checkbox"/>  GPUs or intel xeon phi coprocessor By atknox@gmail.com - 2 posts - 2 views	Apr 21
<input type="checkbox"/>  plotLFP By atknox@gmail.com - 2 posts - 3 views	Mar 23

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

# NetPyNE: Existing models

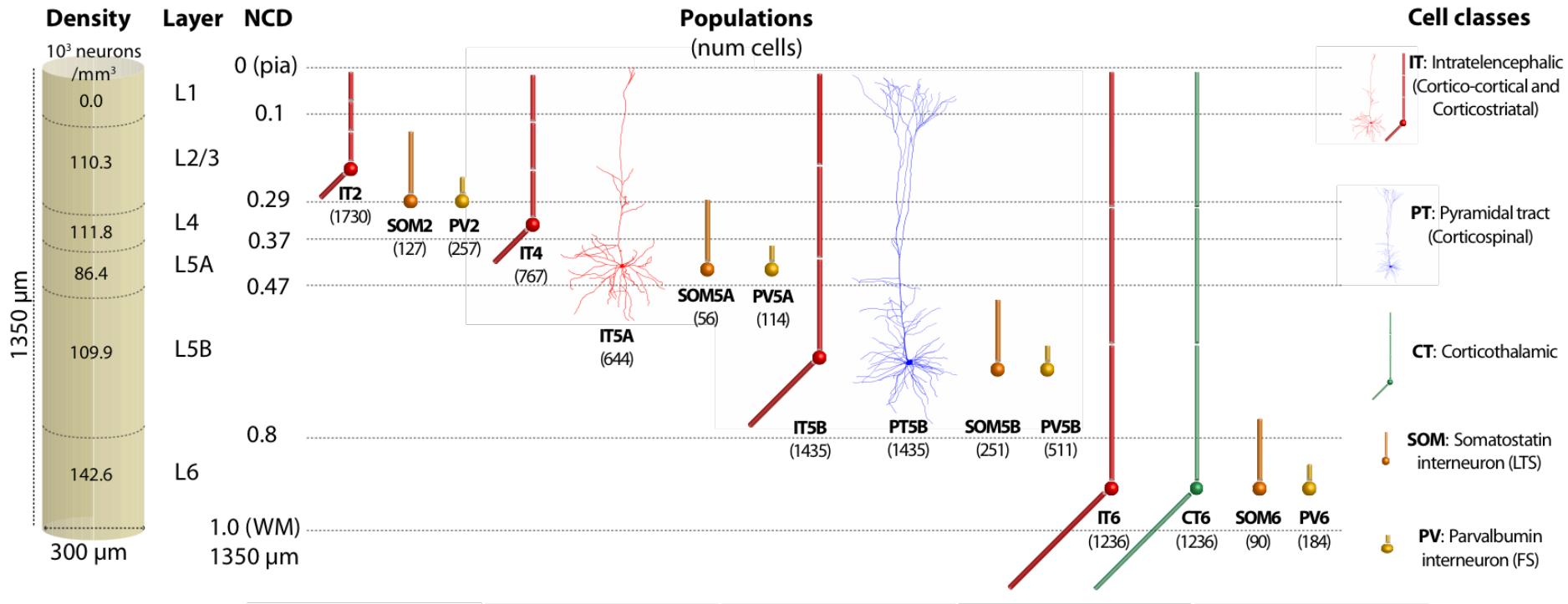
## □ Other models in progress:

- Traub **thalamocortical** network (Padraig Gleeson, UCL)
- Hippocampus **CA3** (Ben Tessler, SUNY DMC)
- **Ischemia** in cortical network (Alex Seidenstein, SUNY DMC)
- **STDP** in biophysically detailed networks (Anatoly Buchin, Allen Brain)
- **Basal Ganglia** network (Lucas, UCD)
- **LFP** oscillations (Christian Fink, Ohio Wesleyan)
- **Dendritic** computations (Birgit Kriener, Oslo)
- Thalamocortical **epilepsy** network (Andrew Knox, Cincinnati Hospital)
- **V1** network with Allen Brain cells (SUNY DMC)
- **Schizophrenia** in cortical network (Cristoph Metzner, Hertfordshire)
- **Spinal cord** circuits (Vittorio Caggiano, IBM Watson)
- Full list of 43 models: <https://drive.google.com/open?id=1bkWHakgZoEkYIkzrAS8sIKCvO5PSuUXLLRjNdN2pseY>



# NetPyNE: M1 microcircuits

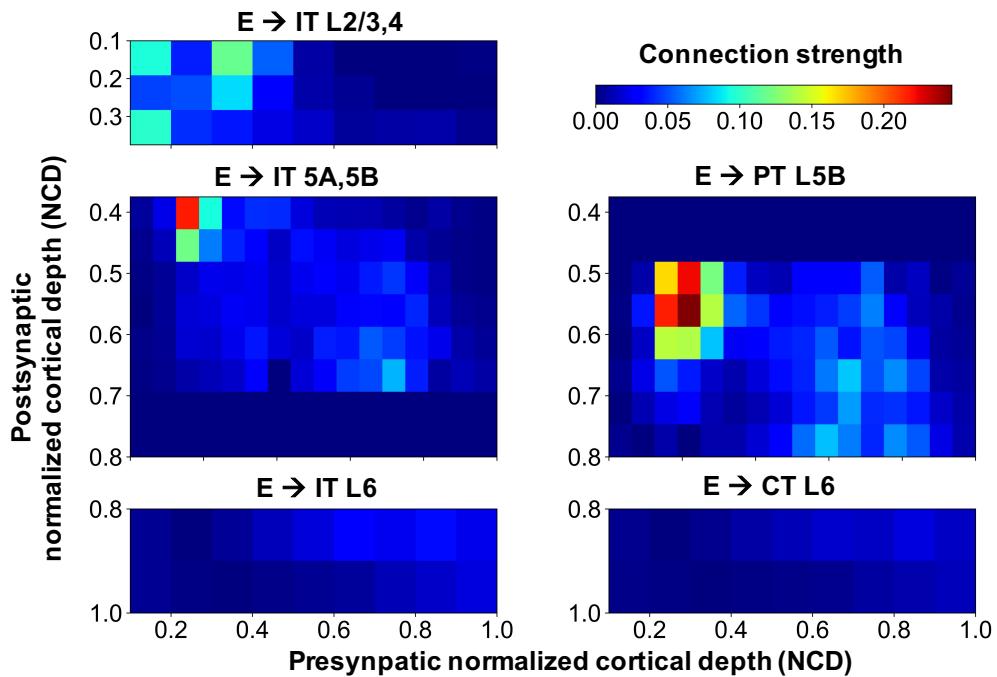
- Data-driven multiscale network model of **M1 microcircuits**



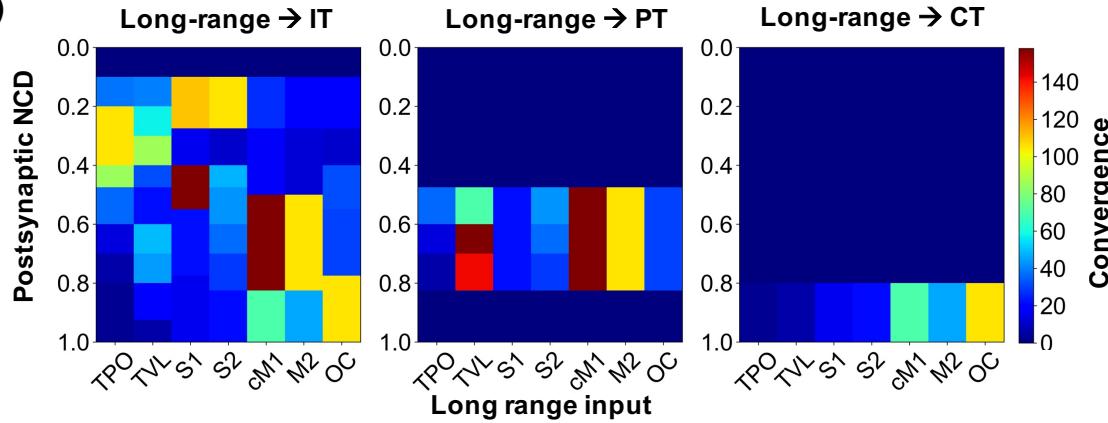
Mouse 6-layer M1 with **10,074 neurons** of 5 classes distributed in 15 populations;  
Full scale cylindric volume of **300 μm** (diameter) x **1350 μm** (cortical depth)

# NetPyNE: M1 microcircuits

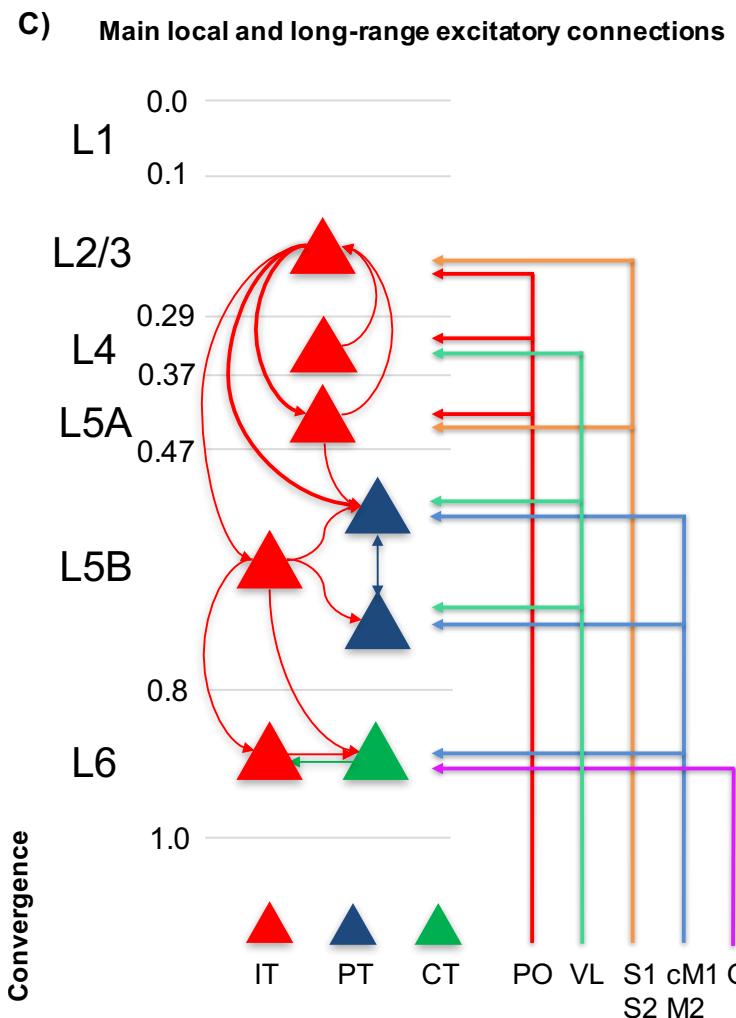
A)



B)



C)



# NetPyNE: Acknowledgments

## Contributors:

- Salvador Dura-Bernal (SUNY DMC)
- Ben Suter (Northwestern)
- Adrian Quintana (EyeSeeTea Ltd)
- Matteo Cantarelli (Metacell Ltd)
- Facundo Rodriguez (Instituto Balseiro)
- Padraig Gleeson (UCL)
- Joe Graham (SUNY DMC)
- Cliff Kerr (Sydney University)
- Robert McDougal (Yale)
- Michael Hines (Yale)
- Gordon MG Shepherd (Northwestern)
- William Lytton (SUNY DMC)

Lab website: [www.neurosimlab.org](http://www.neurosimlab.org)

NetPyNE Website: [www.netpyne.org](http://www.netpyne.org)

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

## Funding:

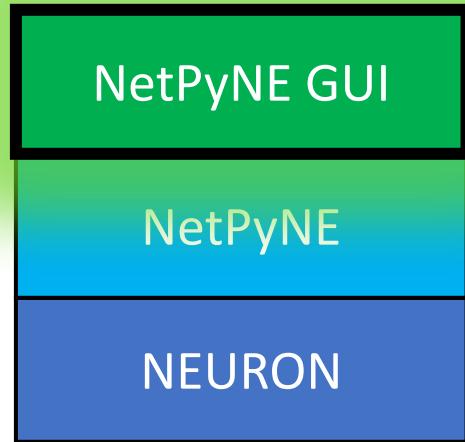
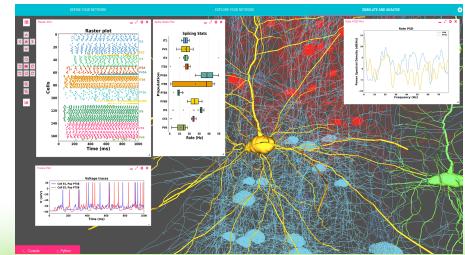
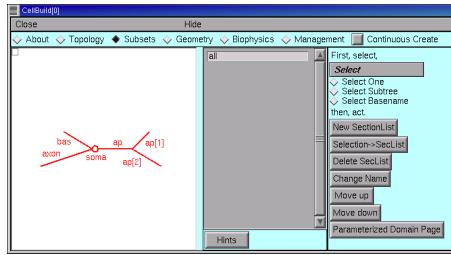
- NIH Grant U01EB017695
- NIH Grant R01EB022903
- NIH Grant R01MH086638
- NYS Grant DOH01-C32250GG-3450000



**D**SUNY  
**DOWNSTATE**  
Medical Center



# NetPyNE GUI Tutorial



# NetPyNE GUI Tutorial: Simple cell net

## 1) Open NetPyNE GUI on web browser

The screenshot shows the NetPyNE GUI interface with a teal header bar. The header contains three tabs: 'DEFINE YOUR NETWORK' (which is highlighted in red), 'EXPLORE YOUR NETWORK', and 'SIMULATE AND ANALYSE'. A gear icon is located in the top right corner of the header.

The main area consists of several sections, each with a title and a descriptive subtitle. Each section has a small downward arrow icon to its right, indicating it can be collapsed. The sections are:

- Populations**  
Define here the populations of your network
- Cell rules**  
Define here the rules to set the biophysics and morphology of the cells in your network
- Synaptic mechanisms**  
Define here the synaptic mechanisms available in your network
- Connectivity rules**  
Define here the rules to generate the connections in your network
- Stimulation sources**  
Define here the sources of stimulation in your network
- Stimulation target rules**  
Define here the rules to connect stimulation sources to targets in your network
- Simulation configuration**  
Define here the configuration options for the simulation
- Plots configuration**  
Define here the options to customize the plots

# NetPyNE GUI Tutorial: Simple cell net

2) Add a population 'E' of 20 'pyr' cells

## Populations

Define here the populations of your network



The name of your population

E

Cell type

pyr

Cell model

Number of cells

Number of cells

Number of cells

20

# NetPyNE GUI Tutorial: Simple cell net

3) Add a cell rule 'pyr\_rule' for 'pyr' cells

## Cell rules

Define here the rules to set the biophysics and morphology of the cells in your network

The name of the cell rule  
pyr\_rule

Conditions:

Cell type  
pyr

Cell model

Population

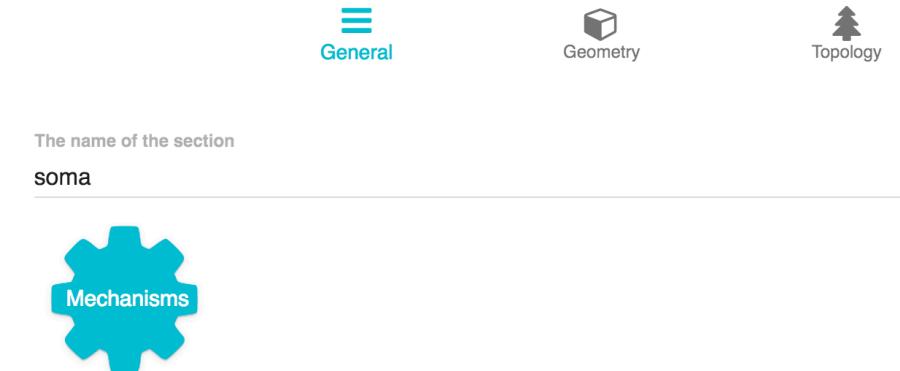
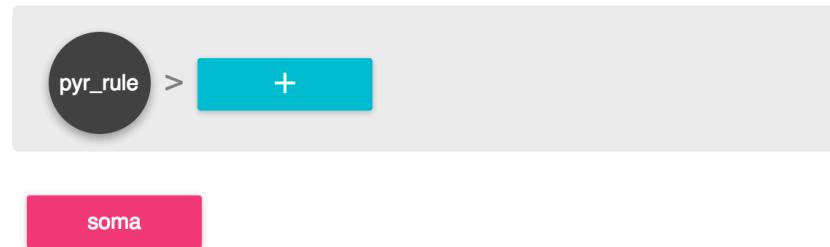
SECTIONS IMPORT TEMPLATE

# NetPyNE GUI Tutorial: Simple cell net

4) Add a ‘soma’ section in the ‘pyr\_rule’

## Cell rules

Define here the rules to set the biophysics and morphology of the cells in your network



The name of the section  
soma

Mechanisms

# NetPyNE GUI Tutorial: Simple cell net

5) Add the geometry of the ‘soma’ section in the ‘pyr\_rule’

## Cell rules

Define here the rules to set the biophysics and morphology of the cells in your network



General



Geometry



Topology

Diameter (um)

20

Length (um)

20

Axial resistance, Ra (ohm-cm)

100

Membrane capacitance, cm (uF/cm<sup>2</sup>)

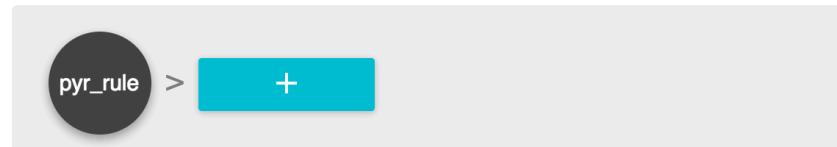
1

# NetPyNE GUI Tutorial: Simple cell net

6) Add a ‘dend’ section in the ‘pyr\_rule’

## Cell rules

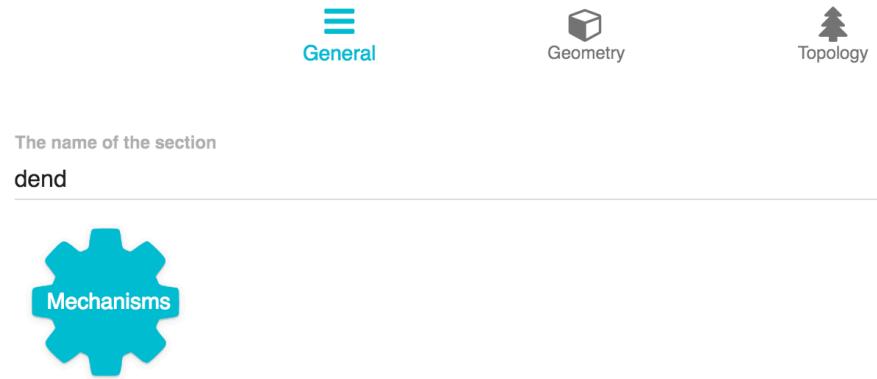
Define here the rules to set the biophysics and morphology of the cells in your network



The name of the section  
dend

Mechanisms

General      Geometry      Topology



# NetPyNE GUI Tutorial: Simple cell net

7) Add the geometry of the ‘dend’ section in the ‘pyr\_rule’

## Cell rules

Define here the rules to set the biophysics and morphology of the cells in your network

pyr\_rule > +

soma dend

General

Geometry

Topology

Diameter (um)  
5

Length (um)  
150

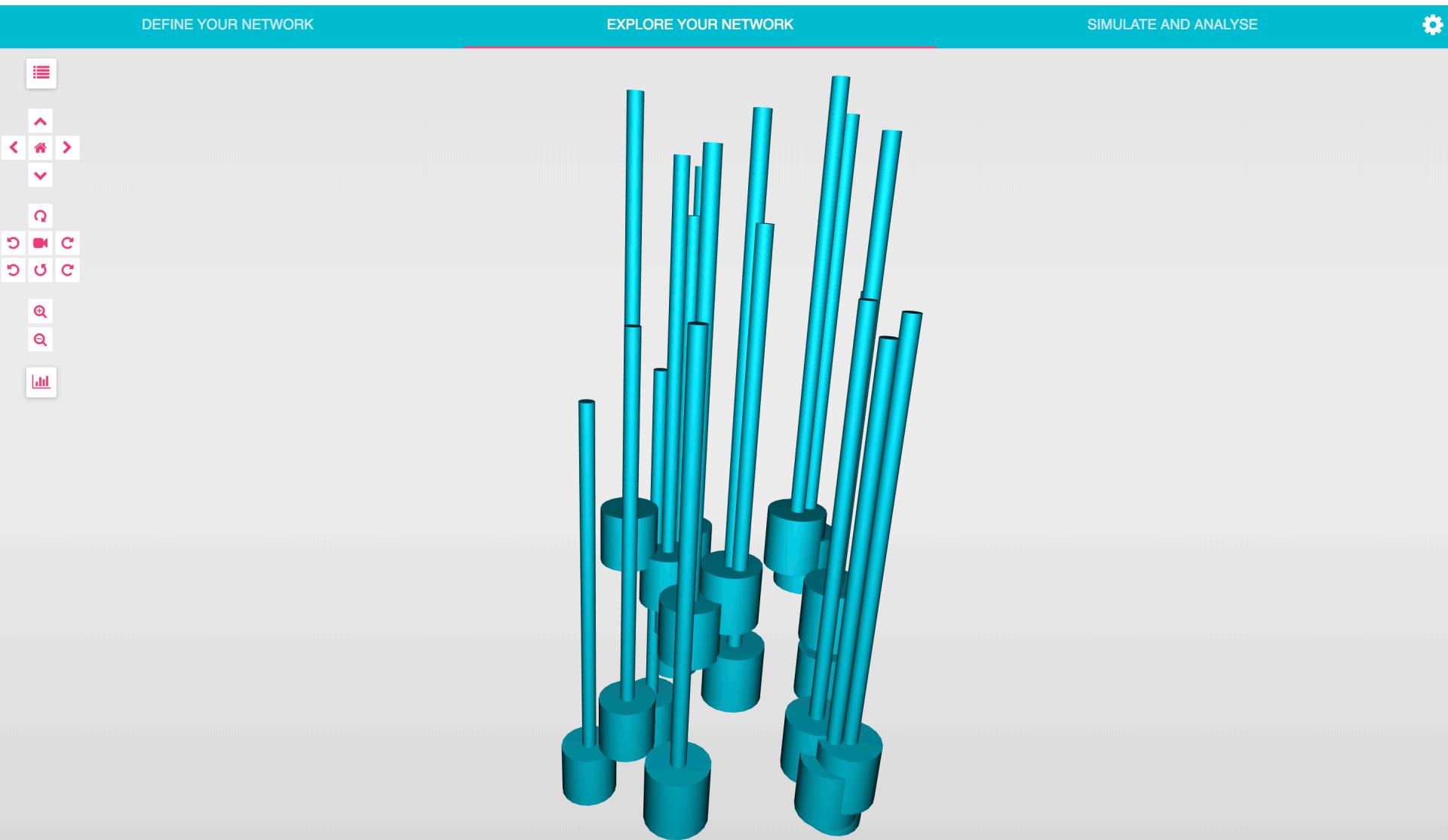
Axial resistance, Ra (ohm-cm)  
100

Membrane capacitance, cm (uF/cm<sup>2</sup>)  
1

Pt3d

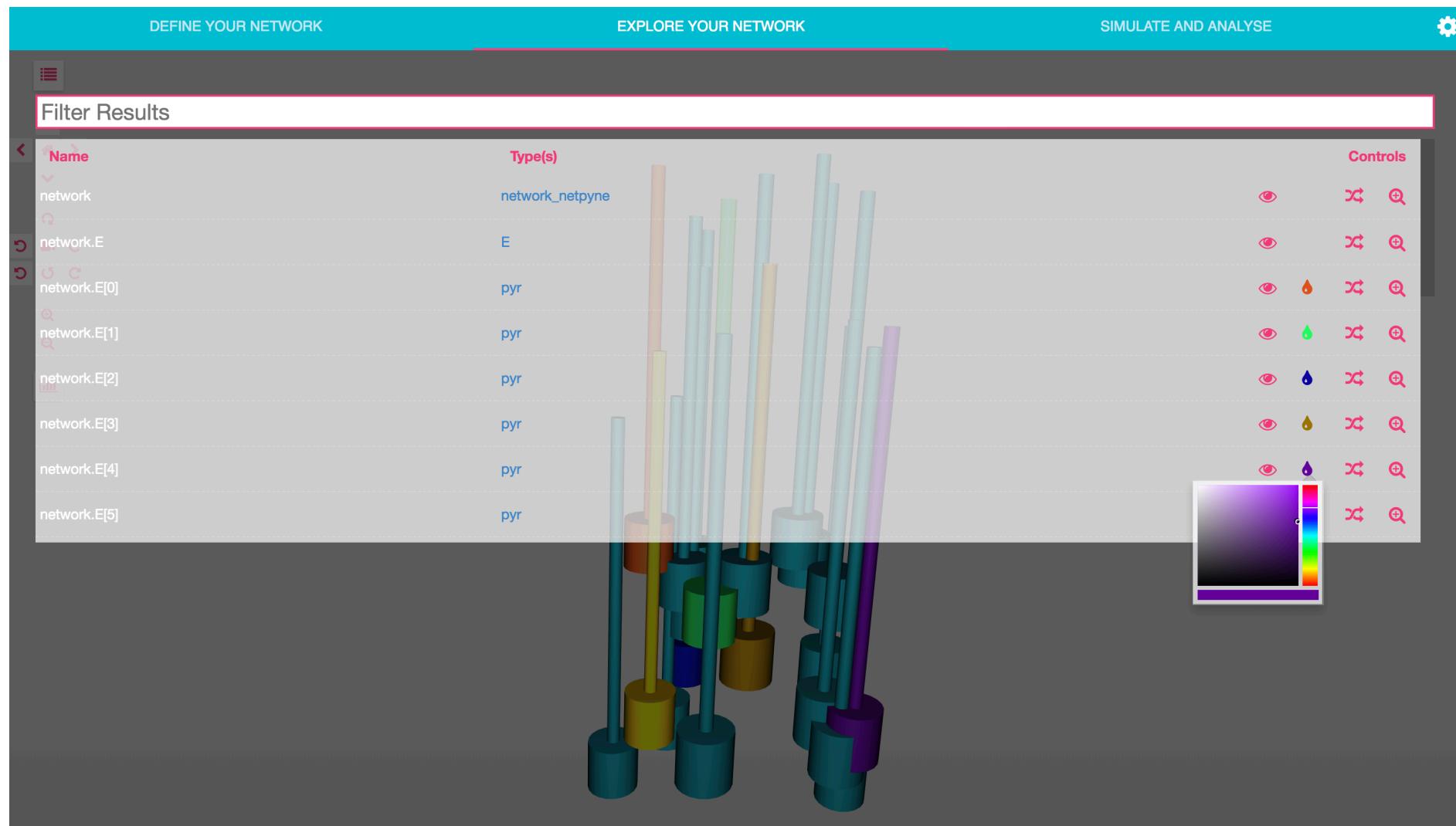
# NetPyNE GUI Tutorial: Simple cell net

8) Explore your network ... feel free to rotate, zoom and move around!



# NetPyNE GUI Tutorial: Simple cell net

9) Customize the color of your population or cells



# NetPyNE GUI Tutorial: Simple cell net

10) To add channels: go to ‘Define your network’ → ‘Cell rules’ → ‘pyr rule’ → ‘soma’ → ‘mechanisms’ → (+)

The screenshot shows the NetPyNE GUI interface. At the top, there are three tabs: 'DEFINE YOUR NETWORK' (highlighted in red), 'EXPLORE YOUR NETWORK', and 'SIMULATE AND ANALYSE'. Below the tabs, under 'Cell rules', there is a sequence of components: 'pyr\_rule' > 'soma' > '+'. A dropdown menu is open at the '+' sign, listing 'pas', 'hh', and 'fastpas'. The 'fastpas' option is highlighted with a blue border. Other sections visible include 'Populations' (with a note to define network populations), 'Synaptic mechanisms' (with a note to define available mechanisms), and 'Connectivity rules'.

DEFINE YOUR NETWORK EXPLORE YOUR NETWORK SIMULATE AND ANALYSE

Populations  
Define here the populations of your network

Cell rules  
Define here the rules to set the biophysics and morphology of the cells in your network

pyr\_rule > soma > +

pas  
hh  
fastpas

Synaptic mechanisms  
Define here the synaptic mechanisms available

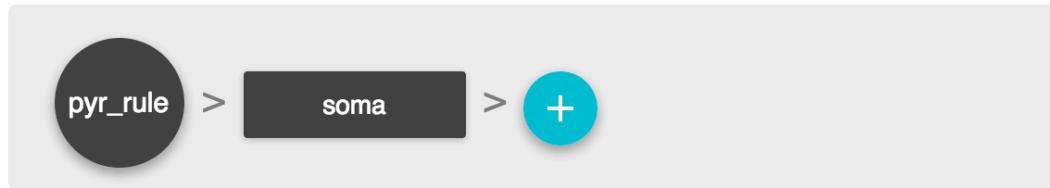
Connectivity rules

# NetPyNE GUI Tutorial: Simple cell net

11) Add the 'hh' (Hodgkin-Huxley) mechanism to the 'soma' with the following parameters:

## Cell rules

Define here the rules to set the biophysics and morphology of the cells in your network



### Mechanism

hh

gnabar

0.12

gkbar

0.036

gl

0.003

el

-70

# NetPyNE GUI Tutorial: Simple cell net

12) Add the ‘pas’ (passive) mechanism to the ‘dend’ with the following parameters:

## Cell rules

Define here the rules to set the biophysics and morphology of the cells in your network

The screenshot shows the NetPyNE GUI interface for defining cell rules. On the left, there is a flowchart-like editor with three nodes: a black circle labeled "pyr\_rule", a dark grey rectangle labeled "dend", and a teal circle with a plus sign. Arrows point from "pyr\_rule" to "dend" and from "dend" to the plus sign. To the right of this editor is a large pink gear icon with the word "pas" written on it. On the far right, there is a configuration panel for the "pas" mechanism. It includes a section header "Mechanism" followed by a dropdown menu set to "pas". Below this are input fields for parameters "g" (set to 0.0004) and "e" (set to -70).

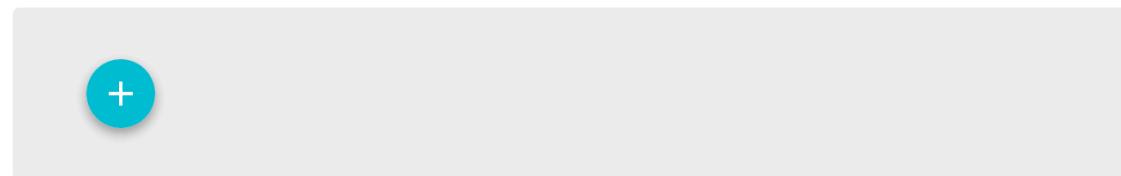
Mechanism	Value
pas	
g	0.0004
e	-70

# NetPyNE GUI Tutorial: Simple cell net

## 13) Add an IClamp (current clamp) source of stimulation

### Stimulation sources

Define here the sources of stimulation in your network



IClamp1

Point process used as stimulator

IClamp



Current clamp delay (ms)

20

Current clamp duration (ms)

10

Current clamp amplitude (nA)

0.1

# NetPyNE GUI Tutorial: Simple cell net

14) Create a stimulation target rule to place IClamp1 on the cell dendrite:

## Stimulation target rules

Define here the rules to connect stimulation sources to targets in your network

The screenshot shows the 'Stimulation target rules' section of the NetPyNE GUI. On the left, there is a list of existing rules: 'IClamp1 -> cell0'. A blue circular button with a '+' sign is available to add new rules. On the right, a detailed configuration panel for the 'IClamp1 -> cell0' rule is displayed. The panel includes tabs for 'General' (selected) and other categories. The configuration fields are as follows:

- Stimulation source:** IClamp1
- Target section:** dend
- Target location:** 1.0

# NetPyNE GUI Tutorial: Simple cell net

15) Place the IClamp1 just on one of the cells (with global index 0) using the target rule ‘conditions’:

## Stimulation target rules

Define here the rules to connect stimulation sources to targets in your network

Stimulation target rules

Define here the rules to connect stimulation sources to targets in your network

+

IClamp1 -> cell0

General

Conditions

Target population  
E

Target cell model

Target cell type

Range of x-axis locations

Range of y-axis locations

Range of z-axis locations

Add new Target cell global indices (gids)

0

# NetPyNE GUI Tutorial: Simple cell net

16) Set the simulation duration to 200ms in ‘Simulation configuration’

## Simulation configuration

Define here the configuration options for the simulation

General Record Save Configuration Error Checking Network Attributes

Duration (ms)  
200

Time step, dt  
0.025

Interval to print run time at (s)

Set global parameters (temperature, initial voltage, etc)  
{ "clamp\_resist":0.001, "celsius":6.3, "v\_init":-65 }

Randomizer seeds  
{ "loc":1, "stim":1, "conn":1 }

Create NEURON objects  
 Create Python structure  
 Add synaptic mechanisms  
 Include parameter rule label  
 Show timing  
 Verbose mode  
 Use compact connection format (list instead of dicT)  
 Select random sections from list for connection  
 Print population average firing rates  
 Print total connections  
 Gather only simulation output data  
 use CVode cache\_efficient  
 use CVode

# NetPyNE GUI Tutorial: Simple cell net

17) Record soma and dendrite voltage traces from from cell with id 0:

Simulation configuration  
Define here the configuration options for the simulation

The screenshot shows the 'Record' tab of the NetPyNE GUI. At the top, there are tabs for General, Record (which is selected), Save Configuration, Error Checking, and Network Attributes. Below the tabs, there are sections for 'Add new Cells to record traces from' and 'Add new Record LFP electrode locations'. The 'Add new Cells to record traces from' section has a button to add a cell (labeled '0') and a button to remove it. The 'Add new Record LFP electrode locations' section has a button to add a location and a question mark icon. On the right, there is a section titled 'Traces to record from cells' which contains a Python dictionary: `{'V_soma': {'var': 'v', 'sec': 'soma', 'loc': 0.5}, 'V_dend': {'var': 'v', 'sec': 'dend', 'loc': 1}}`. A red oval surrounds this dictionary. Below it, there is a 'Time step for data recording (ms)' input field set to '1' and a checkbox for 'Record spikes of artificial stimulators (NetStims and VecStims)'. A red arrow points from the text 'Specify traces to record using Python dictionary format:' below to the red oval.

Specify traces to record using Python dictionary format:

```
{'V_soma': {'var': 'v', 'sec': 'soma', 'loc': 0.5},  
 'V_dend': {'var': 'v', 'sec': 'dend', 'loc': 1.0}}
```

# NetPyNE GUI Tutorial: Simple cell net

17) Record soma and dendrite voltage traces from from cell with id 0:

Simulation configuration  
Define here the configuration options for the simulation

General Record Save Configuration Error Checking Network Attributes

Add new Cells to record traces from

0 -

Add new Record LFP electrode locations

Store LFP of individual cells

Traces to record from cells

{'V\_soma': {'var': 'v', 'sec': 'soma', 'loc': 0.5}, 'V\_dend': {'var': 'v', 'sec': 'dend', 'loc': 1}}

Time step for data recording (ms)  
1

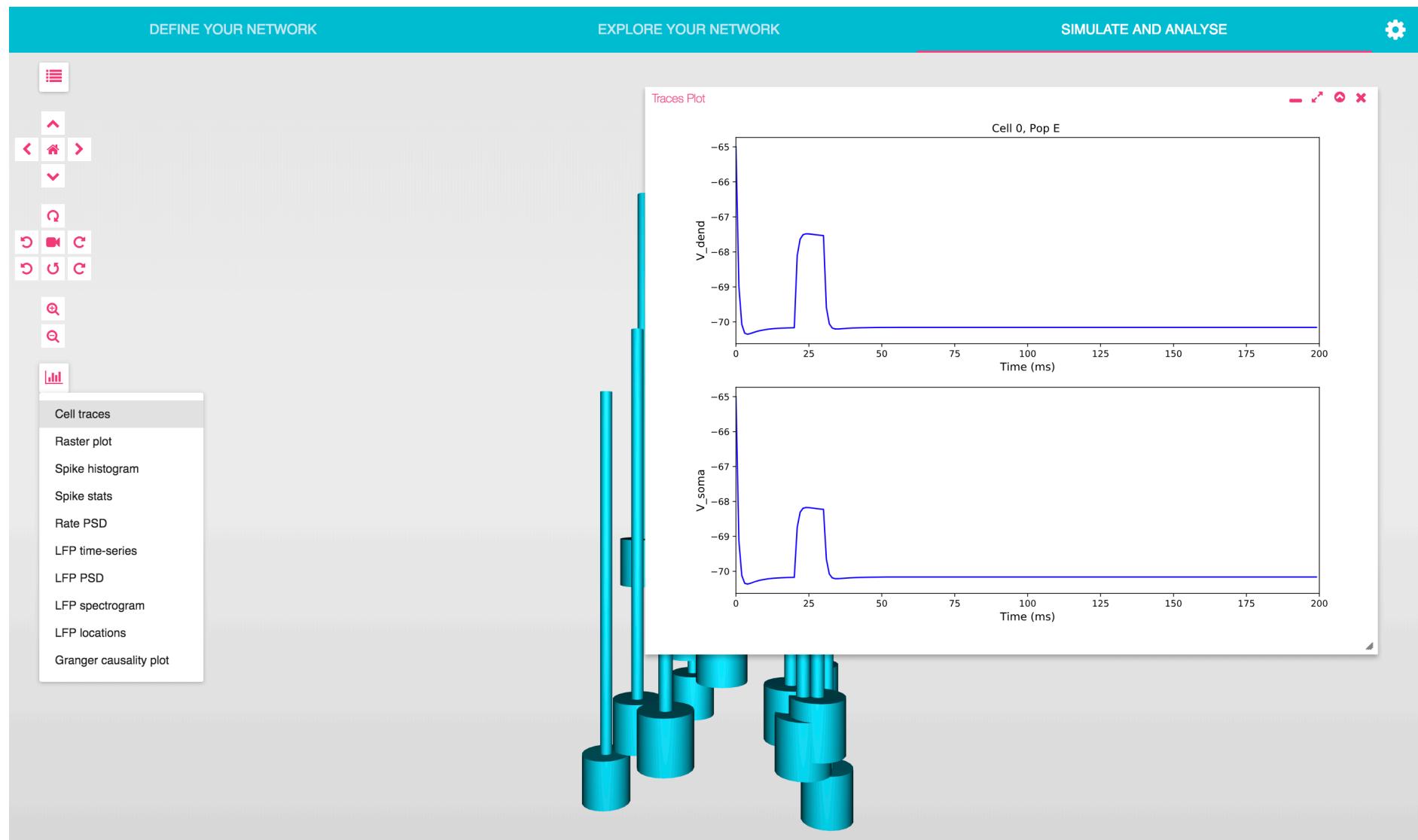
Record spikes of artificial stimulators (NetStims and VecStims)

Specify trace s to record using Python dictionary format:

```
{'V_soma': {'var': 'v', 'sec': 'soma', 'loc': 0.5},  
 'V_dend': {'var': 'v', 'sec': 'dend', 'loc': 1.0}}
```

# NetPyNE GUI Tutorial: Simple cell net

18) ‘Simulate and Analyze’ the network and plot ‘Cell traces’

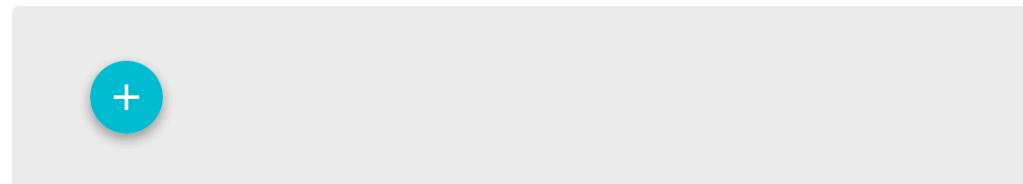


# NetPyNE GUI Tutorial: Simple cell net

19) Increase 'IClamp1' amplitude so generate a spike (set to 0.6 nA)

## Stimulation sources

Define here the sources of stimulation in your network



IClamp1

Point process used as stimulator

IClamp



Current clamp delay (ms)

20

Current clamp duration (ms)

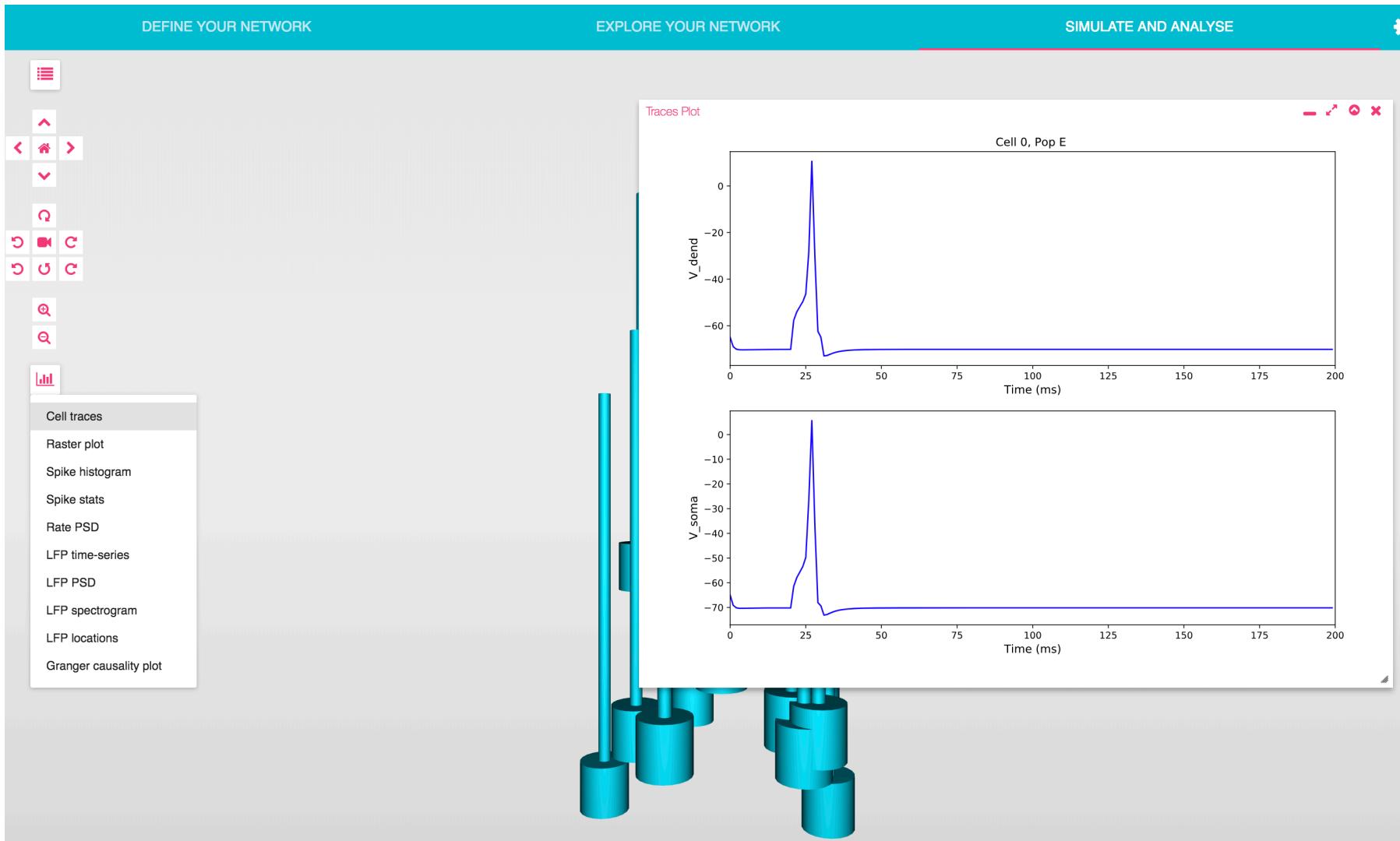
10

Current clamp amplitude (nA)

0.6

# NetPyNE GUI Tutorial: Simple cell net

21) Simulate and plot traces (dendrite current clamp, soma spike and back-propagation to dendrite)



# NetPyNE GUI Tutorial: Simple cell net

22a) Create recurrent connections (E->E) rule; first set weight to 0.03

## Connectivity rules

Define here the rules to generate the connections in your network

The screenshot shows the 'Connectivity rules' section of the NetPyNE GUI. On the left, there is a circular icon labeled 'E->E'. To its right is a large grey button with a blue '+' sign. Above the main panel, there are three tabs: 'General' (selected), 'Pre-synaptic cells conditions', and 'Post-synaptic cells conditions'. The 'General' tab has the following fields:

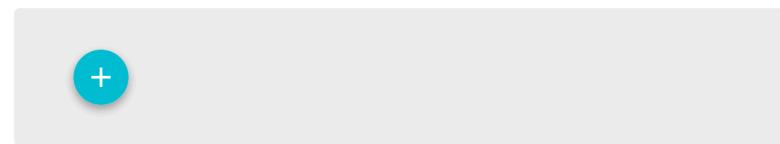
- 'The name of the connectivity rule': E->E
- 'Postsynaptic neuron section': soma
- 'Add new Postsynaptic neuron location (0-1)': A button with a blue '+' sign and a question mark icon.
- 'Synaptic mechanism': A dropdown menu with a question mark icon.
- 'Number of individual synaptic contacts per connection': A field with a question mark icon.
- 'Weight of synaptic connection': 0.03
- 'Add new Connection delay (ms)': A button with a blue '+' sign and a question mark icon.

# NetPyNE GUI Tutorial: Simple cell net

22b) Make presynaptic cells condition be 'E' population

## Connectivity rules

Define here the rules to generate the connections in your network



General



Pre-synaptic cells  
conditions



Post-synaptic cells  
conditions

Population (multiple selection available)

E

Cell model (multiple selection available)

Cell type (multiple selection available)

Range of x-axis locations

Range of y-axis locations

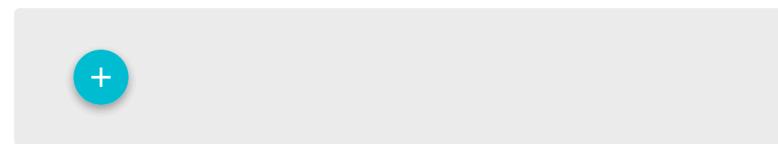
Range of z-axis locations

# NetPyNE GUI Tutorial: Simple cell net

22b) Make presynaptic cells condition be 'E' population

## Connectivity rules

Define here the rules to generate the connections in your network



General



Pre-synaptic cells  
conditions



Post-synaptic cells  
conditions

Population (multiple selection available)

E

Cell model (multiple selection available)

Cell type (multiple selection available)

Range of x-axis locations

Range of y-axis locations

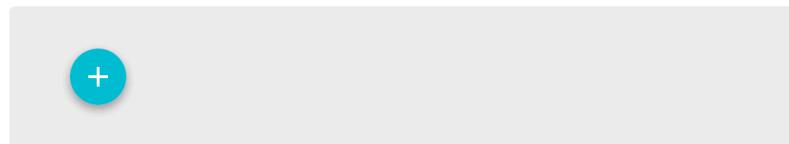
Range of z-axis locations

# NetPyNE GUI Tutorial: Simple cell net

22c) Make postsynaptic cells condition be 'E' population

## Connectivity rules

Define here the rules to generate the connections in your network



General



Pre-synaptic cells  
conditions



Post-synaptic cells  
conditions

Population (multiple selection available)

E

Cell model (multiple selection available)

Cell type (multiple selection available)

Range of x-axis locations

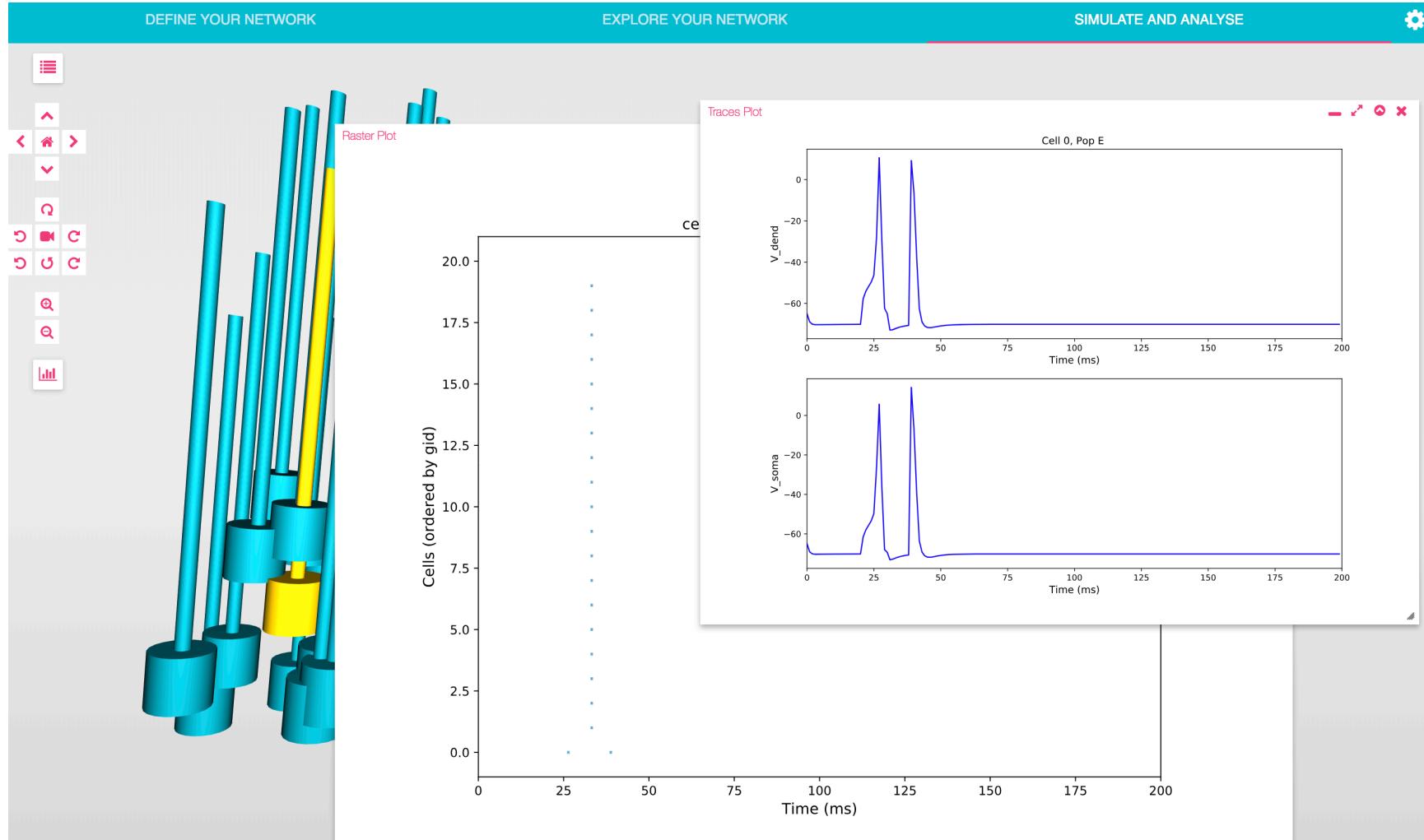
Range of y-axis locations

Range of z-axis locations

# NetPyNE GUI Tutorial: Simple cell net

23) Simulate and plot traces and raster plot;

Cell 0 spikes due to IClamp -> triggers spikes in other cells due to conns -> cell 0 spikes again



# NetPyNE GUI Tutorial: Simple cell net

Note: If you have any errors with step-by-step, try loading the “**simple cell net**” tutorial directly from file

## Method 1:

The screenshot shows the 'IMPORT' tab of the NetPyNE GUI dialog box. It contains fields for NetParams and SimConfig paths, module names, variables, and a checkbox for compiling mod files.

NetParams path	SimConfig path
/home/jovyan/netpyne_workspace	/home/jovyan/netpyne_workspace

NetParams module name	SimConfig module name
gui_tut1	gui_tut1

NetParams variable	SimConfig variable
netParams	simConfig

**Compile mod files**

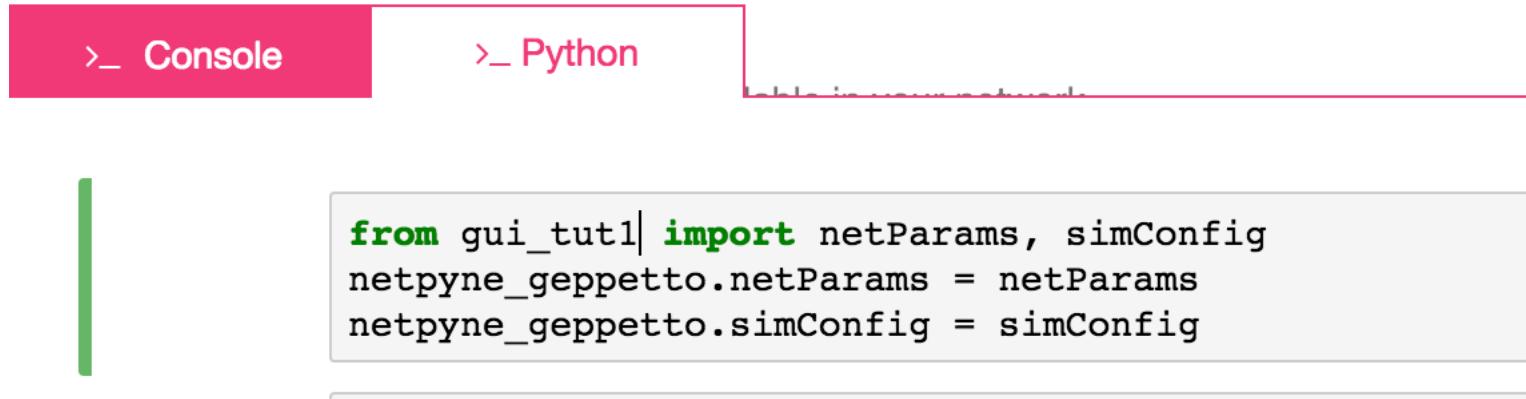
Mod path folder

**CANCEL** **IMPORT**

# NetPyNE GUI Tutorial: Simple cell net

Note: If you have any errors with step-by-step, try loading the “**simple cell net**” tutorial directly from file

## Method 2:



The screenshot shows the NetPyNE GUI interface. At the top, there are two tabs: "Console" (pink background) and "Python" (white background). Below the tabs is a code editor window containing Python code. A vertical green bar is positioned to the left of the code editor.

```
from gui_tut1 import netParams, simConfig
netpyne_gepetto.netParams = netParams
netpyne_gepetto.simConfig = simConfig
```

Copy paste from here:

```
from gui_tut1 import netParams, simConfig
netpyne_gepetto.netParams = netParams
netpyne_gepetto.simConfig = simConfig
```

To execute press Shift + Enter

# NetPyNE GUI Tutorial: Complex cell net

1) Reload webpage to start from scratch

localhost:8888/gepetto?

Apps Google Maps cosas Neurosim NYC COURSES Google Scholar Sci-Hub: removing... SciWrite Coursewar... edX Lecture 1D - The hi... weather brooklyn ... G

**DEFINE YOUR NETWORK**

**EXPLORE YOUR NETWORK**

**SIMULATE AND ANALYSE**

**Populations**  
Define here the populations of your network

**Cell rules**  
Define here the rules to set the biophysics and morphology of the cells in your network

**Synaptic mechanisms**  
Define here the synaptic mechanisms available in your network

**Connectivity rules**  
Define here the rules to generate the connections in your network

**Stimulation sources**  
Define here the sources of stimulation in your network

**Stimulation target rules**  
Define here the rules to connect stimulation sources to targets in your network

**Simulation configuration**  
Define here the configuration options for the simulation

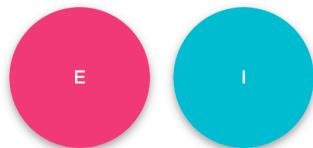
**Plots configuration**  
Define here the options to customize the plots

# NetPyNE GUI Tutorial: Complex cell net

- 2) Add 2 populations of 3 cells :  
- 'E' (excitatory) of cell type 'PT' (pyramidal-tract corticospinal)  
- 'I' (inhibitory) of cell type 'FS' (fast-spiking interneuron)

## Populations

Define here the populations of your network



The name of your population

E

Cell type

PT

General  
Spatial Distribution  
Cell List

Spatial Distribution

Cell List

Cell model

Number of cells

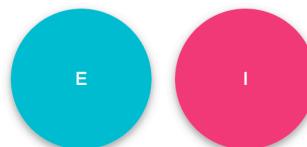
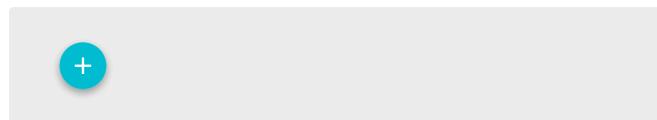
Number of cells

Number of cells

3

## Populations

Define here the populations of your network



The name of your population

I

Cell type

FS

General  
Spatial Distribution  
Cell List

Spatial Distribution

Cell List

Cell model

Number of cells

Number of cells

Number of cells

3

?

# NetPyNE GUI Tutorial: Complex cell net

## 3) Create PT rule and import from template (need to copy files to workspace)

### Cell rules

Define here the rules to set the biophysics and morphology of the cells in your network

The screenshot shows the NetPyNE GUI interface. On the left, there's a list of cell rules with a red callout bubble pointing to the 'PT\_rule' entry. To the right, a modal dialog is open for creating a new cell rule.

**Cell rule creation:**

- Name:** PT\_rule
- Conditions:** Cell type PT

**Import Cell Template:**

Python or Hoc files

Absolute path to file: /home/jovyan/netpyne\_workspace/cells/PTcell.py

Cell template/class name: PTcell

Path to mod folder: /home/jovyan/netpyne\_workspace/mod

Import synaptic mechanisms    Compile mod files

Buttons at the bottom: CANCEL, IMPORT, IMPORT TEMPLATE

# NetPyNE GUI Tutorial: Complex cell net

4) Create FS rule and import from template (need to copy files to workspace)

Cell rules  
Define here the rules to set the biophysics and morphology of the cells in your network

The screenshot shows the NetPyNE GUI interface. On the left, under 'Cell rules', there is a large grey box with a '+' button for adding new rules. Below it are two circular icons: a teal one labeled 'PT\_rule' and a pink one labeled 'FS\_rule'. On the right, there are several input fields and buttons. At the top right is a field 'The name of the cell rule' containing 'FS\_rule'. Below it is a 'Conditions:' section with 'Cell type' set to 'FS'. Further down are fields for 'II model' and 'population'. At the bottom right are two buttons: 'SECTIONS' (highlighted in teal) and 'IMPORT TEMPLATE'. A large dialog box in the foreground is titled 'Import Cell Template' and specifies 'Python or Hoc files'. It contains fields for 'Absolute path to file' (set to '/home/jovyan/netpyne\_workspace/cells/FScell.py'), 'Cell template/class name' (set to 'FScell'), and 'Path to mod folder'. It also includes checkboxes for 'Import synaptic mechanisms' and 'Compile mod files'. At the bottom of the dialog are 'CANCEL' and 'IMPORT' buttons.

The name of the cell rule  
FS\_rule

Conditions:

Cell type  
FS

II model

population

SECTIONS IMPORT TEMPLATE

Import Cell Template

Python or Hoc files

Absolute path to file  
/home/jovyan/netpyne\_workspace/cells/FScell.py

Cell template/class name  
FScell

Path to mod folder

Import synaptic mechanisms    Compile mod files

CANCEL IMPORT

# NetPyNE GUI Tutorial: Complex cell net

## 5) Visualize network

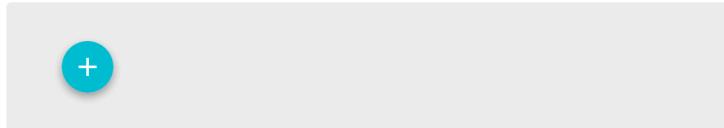


# NetPyNE GUI Tutorial: Complex cell net

## 6) Add AMPA and GABA synapses

### Synaptic mechanisms

Define here the synaptic mechanisms available in your network



#### AMPA

NMODL mechanism name

Exp2Syn



Time constant for exponential 1 (ms)

0.5

Time constant for exponential 2 (ms)

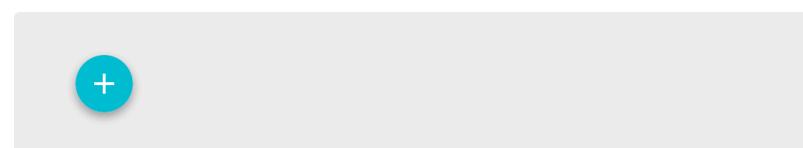
1

Reversal potential (mV)

0

### Synaptic mechanisms

Define here the synaptic mechanisms available in your network



#### GABA

NMODL mechanism name

Exp2Syn



Time constant for exponential 1 (ms)

0.5

Time constant for exponential 2 (ms)

1

Reversal potential (mV)

-100

# NetPyNE GUI Tutorial: Complex cell net

7a) Add background stimulation Netstim (spike generator) to PT cells

## Stimulation sources

Define here the sources of stimulation in your network

The screenshot shows the NetPyNE GUI interface for defining stimulation sources. On the left, there is a list of stimulation sources, with one named "bkg" highlighted by a pink circle. To the right of the list are several configuration parameters:

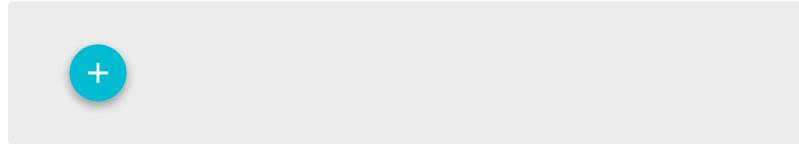
- bkg**: A text input field containing the name of the stimulation source.
- Point process used as stimulator**: A dropdown menu set to "NetStim".
- Interval between spikes (ms)**: A text input field containing the value "50".
- Maximum number of spikes**: A text input field.
- Start time of first spike**: A text input field containing the value "1".
- Noise/randomness fraction (0-1)**: A text input field containing the value "0".

# NetPyNE GUI Tutorial: Complex cell net

## 7b) Add background stimulation Netstim (spike generator) to PT cells

### Stimulation target rules

Define here the rules to connect stimulation sources to targets in your network



General

Conditions

bkg->PYR1

Stimulation source

bkg

Target section

soma

Target location

0.5

Target synaptic mechanism

AMPA

Weight of connection between NetStim and cell

0.1

Delay of connection between NetStim and cell

5

Number of synaptic contacts per connection between NetStim and cell

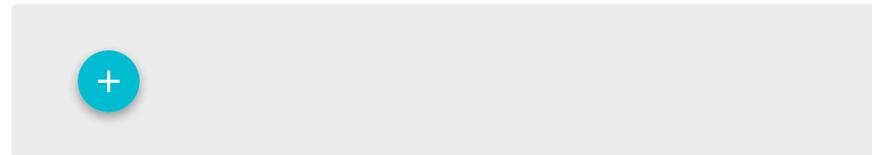
1

# NetPyNE GUI Tutorial: Complex cell net

7c) Add background stimulation Netstim (spike generator) to PT cells (E population)

## Stimulation target rules

Define here the rules to connect stimulation sources to targets in your network



Target population

E



Conditions

Target cell model

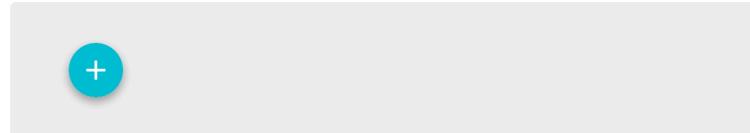
Target cell type

# NetPyNE GUI Tutorial: Complex cell net

## 8) Connect E->I

### Connectivity rules

Define here the rules to generate the connections in your network



Pre-synaptic cells conditions

Post-synaptic cells conditions

The name of the connectivity rule

E->I

Postsynaptic neuron section

soma

Add new Postsynaptic neuron location (0-1)

0.5

Synaptic mechanism

Number of individual synaptic contacts per connection

1

Weight of synaptic connection

0.03

Add new Connection delay (ms)

5



Pre-synaptic cells  
conditions

Post-synaptic cells  
conditions

Population (multiple selection available)

E



Pre-synaptic cells  
conditions

Post-synaptic cells  
conditions

Population (multiple selection available)

I

# NetPyNE GUI Tutorial: Complex cell net

9) Set duration to 200 ms and time step to 0.1

## Simulation configuration

Define here the configuration options for the simulation

The screenshot shows the 'General' tab of the simulation configuration panel. At the top right are three icons: a grey circle labeled 'Record', a question mark icon, and a small Greek letter epsilon icon. Below the tabs, there are two input fields: 'Duration (ms)' containing '200' and 'Time step, dt' containing '0.1'. Each input field has a question mark icon at its right end.

General

Record

?

ε

Duration (ms)

200

?

Time step, dt

0.1

?

# NetPyNE GUI Tutorial: Complex cell net

## 10) Record voltate trace from cell 0

### Simulation configuration

Define here the configuration options for the simulation

The screenshot shows the NetPyNE GUI interface for setting up a simulation. At the top, there are several tabs: General, Record (which is currently selected), Save Configuration, Error Checking, and Network Attributes. Below the tabs, there's a section titled "Add new Cells to record traces from" with a plus sign (+) and minus sign (-) button. A list shows one cell, "0", followed by a dotted line. To the right, there's a section for "Traces to record from cells" containing the JSON object: {"V\_soma": {"var": "v", "loc": 0.5, "sec": "soma"}}. Below that, a section for "Time step for data recording (ms)" has the value "1".

General

Record

Save Configuration

Error Checking

Network Attributes

Add new Cells to record traces from

0

-

Traces to record from cells

{"V\_soma": {"var": "v", "loc": 0.5, "sec": "soma"}}

Time step for data recording (ms)

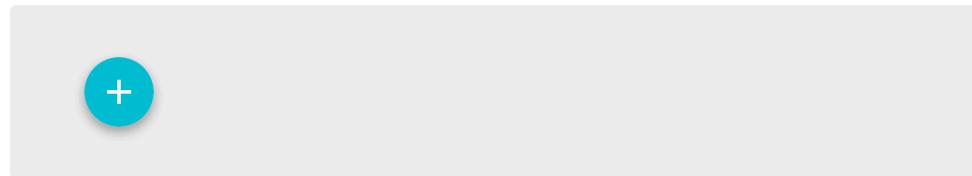
1

# NetPyNE GUI Tutorial: Complex cell net

## 11) Set spike histogram options

### Plots configuration

Define here the options to customize the plots



Add new Cells to include

E    I  

Time range [min,max] (ms)

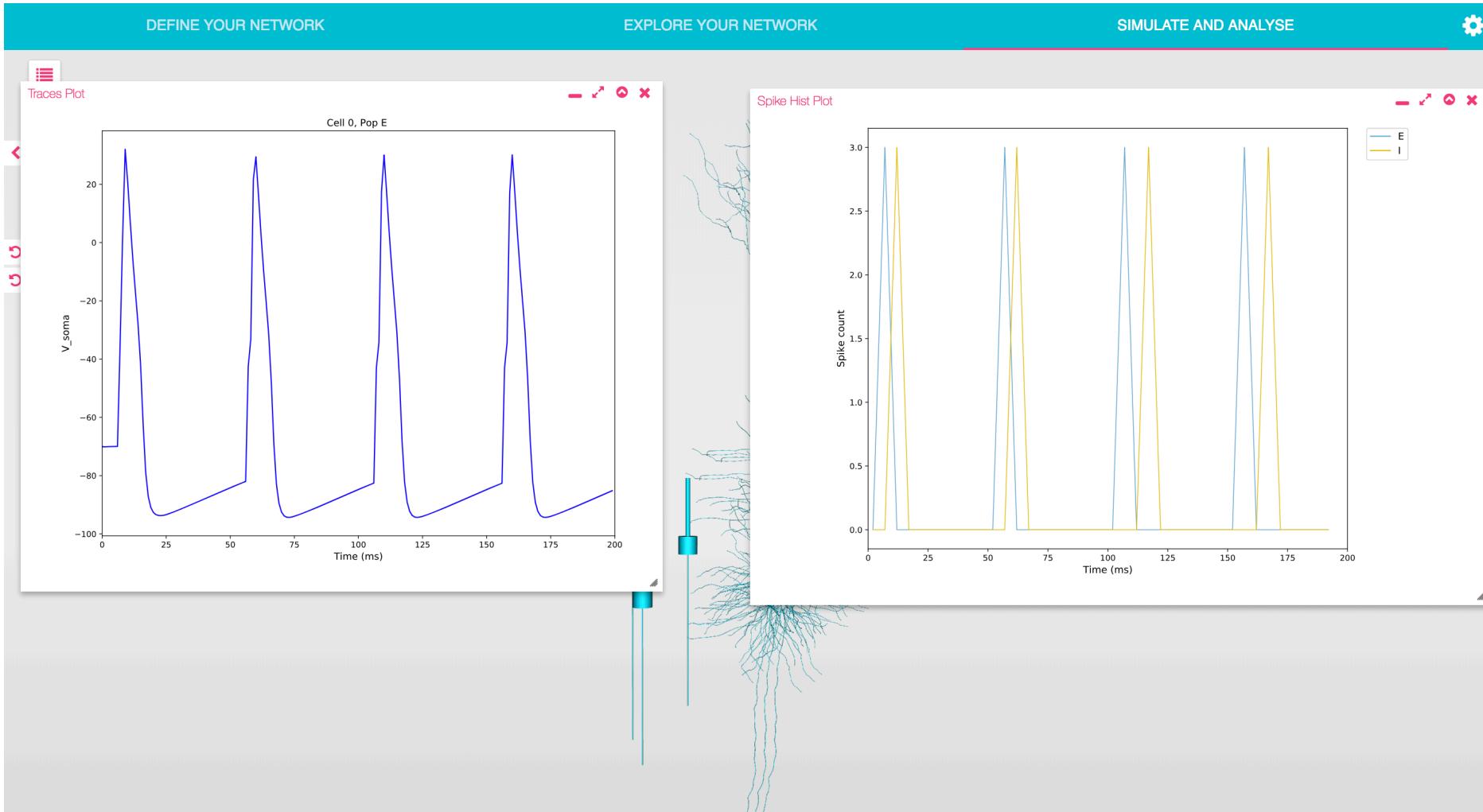
bin size for histogram

type of Graph

axis units  
count

# NetPyNE GUI Tutorial: Complex cell net

## 11) Simulate and visualize traces and spike histogram

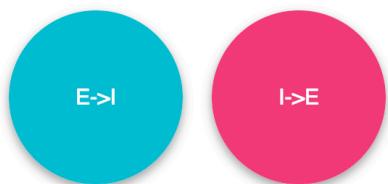
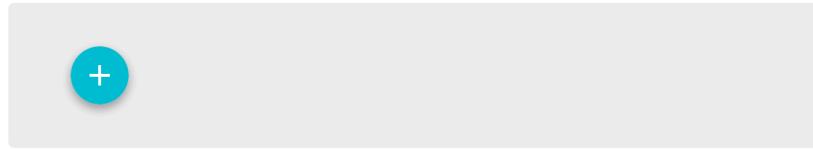


# NetPyNE GUI Tutorial: Complex cell net

## 12) Connect I->E with delay of 40ms

### Connectivity rules

Define here the rules to generate the connections in your network



General



Pre-synaptic cells  
conditions



Post-synaptic cells  
conditions

The name of the connectivity rule

I->E

Postsynaptic neuron section

soma

Add new Postsynaptic neuron location (0-1)

0.5

Synaptic mechanism

Number of individual synaptic contacts per connection

1

Weight of synaptic connection

0.09

Add new Connection delay (ms)

40



General



Pre-synaptic cells  
conditions



General



Pre-synaptic cells  
conditions



Post-synaptic cells  
conditions

Population (multiple selection available)

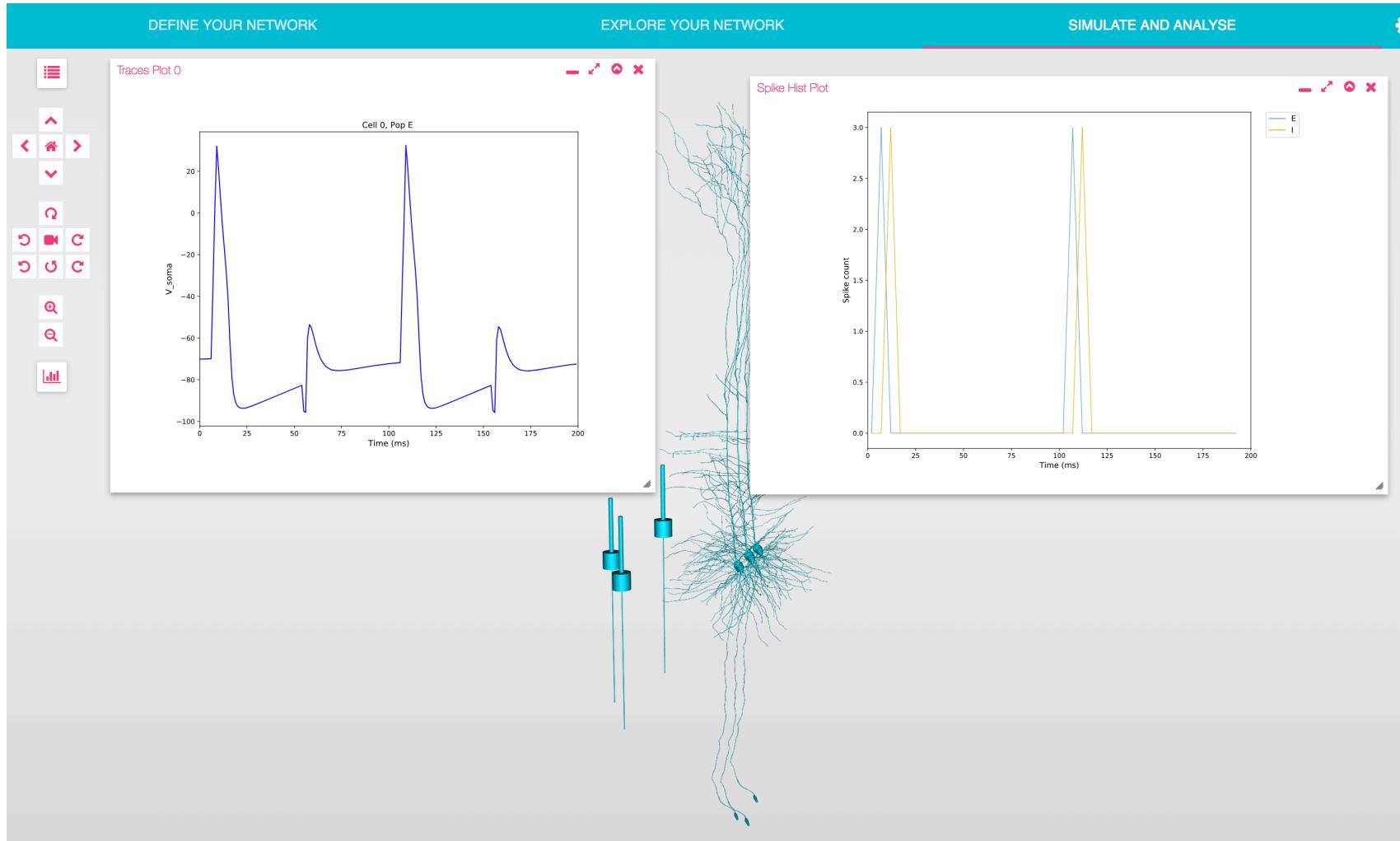
|

Population (multiple selection available)

E

# NetPyNE GUI Tutorial: Complex cell net

## 13) Simulate and plot traces and spike histogram



# NetPyNE GUI Tutorial: Complex cell net

Note: If you have any errors with step-by-step, try loading the “complex cell net” tutorial directly from file

## Method 1:

**IMPORT**      **EXPORT**

NetParams path /home/jovyan/netpyne_workspace	SimConfig path /home/jovyan/netpyne_workspace
NetParams module name gui_tut2	SimConfig module name gui_tut2
NetParams variable netParams	SimConfig variable simConfig

**Compile mod files**

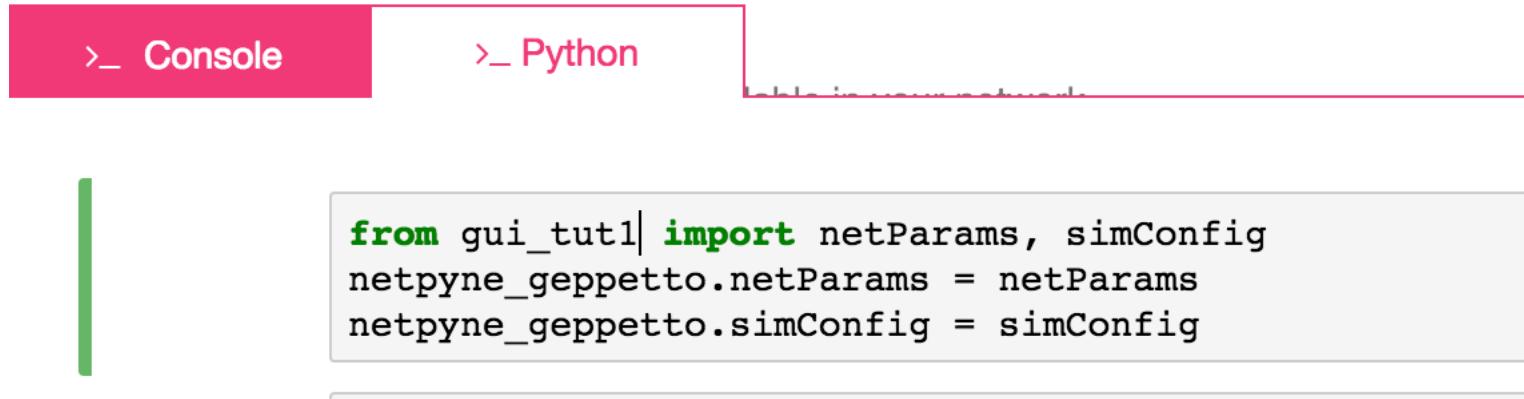
**Mod path folder**  
me/jovyan/netpyne\_workspace/mod

**CANCEL**      **IMPORT**

# NetPyNE GUI Tutorial: Complex cell net

Note: If you have any errors with step-by-step, try loading the “**simple cell net**” tutorial directly from file

## Method 2:



```
from gui_tut1 import netParams, simConfig  
netpyne_gepetto.netParams = netParams  
netpyne_gepetto.simConfig = simConfig
```

Copy paste from here:

```
from gui_tut2 import netParams, simConfig  
netpyne_gepetto.netParams = netParams  
netpyne_gepetto.simConfig = simConfig
```

To execute press Shift + Enter

# NetPyNE GUI Tutorial: Multilayer net

Load the “multilayer net” tutorial directly from file

## Method 1:

**IMPORT**      **EXPORT**

NetParams path <code>/home/jovyan/netpyne_workspace</code>	SimConfig path <code>/home/jovyan/netpyne_workspace</code>
NetParams module name <code>gui_tut3</code>	SimConfig module name <code>gui_tut3</code>
NetParams variable <code>netParams</code>	SimConfig variable <code>simConfig</code>

**Compile mod files**

**Mod path folder**  
`me/jovyan/netpyne_workspace/mod`

**CANCEL**      **IMPORT**

# NetPyNE GUI Tutorial: Multilayer net

Load the “multilayer cell net” tutorial directly from file

**Method 2:**

>\_ Console

>\_ Python

```
from gui_tut3 import netParams, simConfig  
netpyne_gepetto.netParams = netParams  
netpyne_gepetto.simConfig = simConfig
```

Copy paste from here:

```
from gui_tut3 import netParams, simConfig  
netpyne_gepetto.netParams = netParams  
netpyne_gepetto.simConfig = simConfig
```

To execute press Shift + Enter

# NetPyNE GUI Tutorial: Multilayer net

## 1) Simulate and plot raster and LFP

