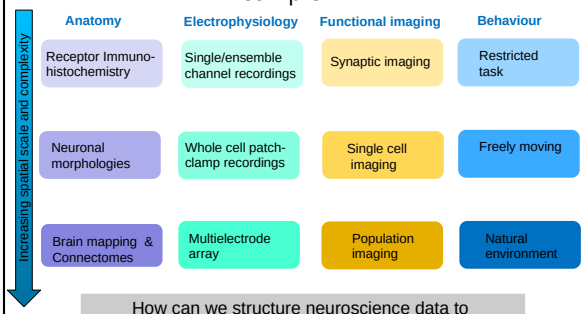
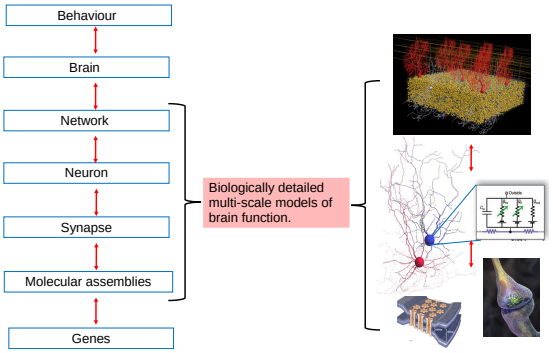
1. A simplified diagram. Actually a lot more complex
2. It is so massive that you can speak to another neuroscientist and not understand a word of what they say—we specialise
3. We won't even discuss dissemination to a non scientific audience today—a completely different problem.

Experimental neuroscience data is heterogeneous, multiscale and analysis is complex

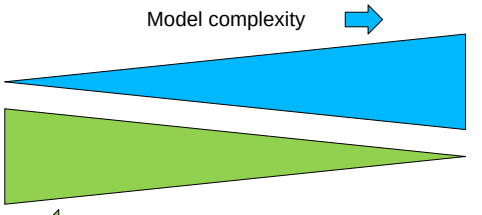


How can we structure neuroscience data to facilitate reuse?

Models of brain function span multiple spatial scales



A scaling problem



Transparency, accessibility, reproducibility, reuse…..and utility as a scientific tool

NWB:N 2.0: An Ecosystem for Neurophysiology Data Standardization

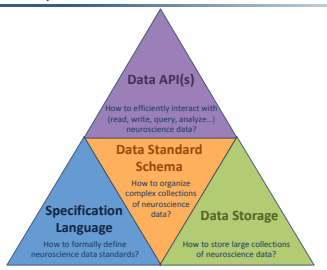
Oliver Rübel

Computational Research Division, Lawrence Berkeley National Laboratory

Open Source Brain Workshop
Alghero, Sardinia
September 10, 2019

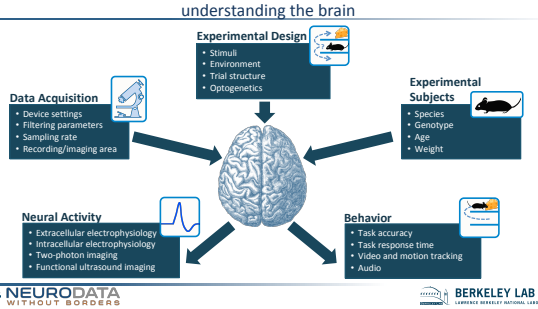
NEURODATA WITHOUT BORDERS BERKELEY LAB

Main components of the NWB:N ecosystem



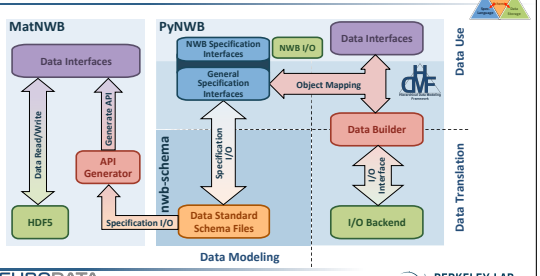
NEURODATA WITHOUT BORDERS BERKELEY LAB

NWB:N supports complex collections of data required for understanding the brain



NEURODATA WITHOUT BORDERS BERKELEY LAB

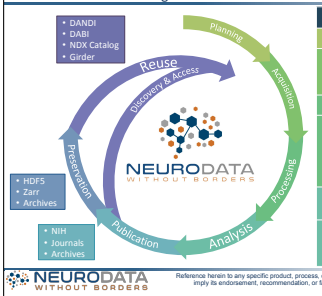
Advanced software architecture for data standardization



NEURODATA WITHOUT BORDERS BERKELEY LAB

Neurophysiology Tools

NWB:N technologies at the heart of the neurodata lifecycle and applications



Electrophysiology	Optical Physiology
DataJoint, NWB Tutorials	
OpenEphys, Plexon, Neuralynx, Intan, TDT, SpikeGadgets, SpikeGLX	MiniScope ScanImage
PyNWB, MatNWB, HDMF, HDF5 tools/libs	
SpikeInterfaces MountainSort, KiloSort, npsorting BrainStorm WaveClust, UltraMegaSort2000, KiloSort Kluster	NoRMCorre CNMF-E CELLMax EXTRACT
NWB-Explorer (OpenSourceBrain) NWB-JupyterWidgets	
BrainStorm, RAVE, Neo, SpikeWidgets, EcogVis, ephys-viz, …	CaImAn

NEURODATA WITHOUT BORDERS

BERKELEY LAB

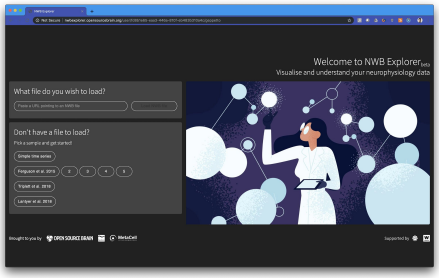
MetaCell

SOFTWARE FOR NEUROSCIENCE

NWB Explorer

Open Source Brain Meeting
Alghero, Italy
10th September 2019

OPEN SOURCE BRAIN



NWB EXPLORER

DEMO

Try it out at

https://nwbexplorer.opensourcebrain.org

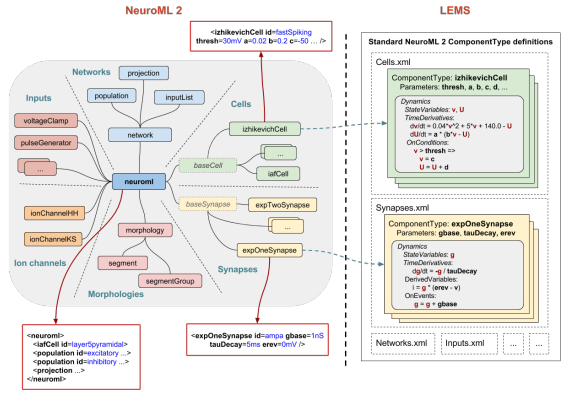
Creating cortical models across scales in NeuroML

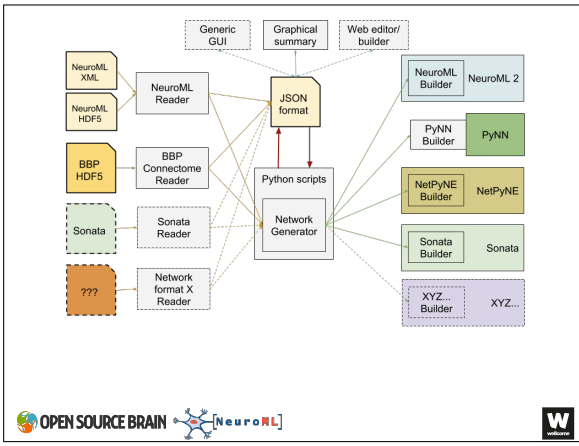
Padraig Gleeson
p.gleeson@ucl.ac.uk
University College London

Open Source Brain workshop 2019

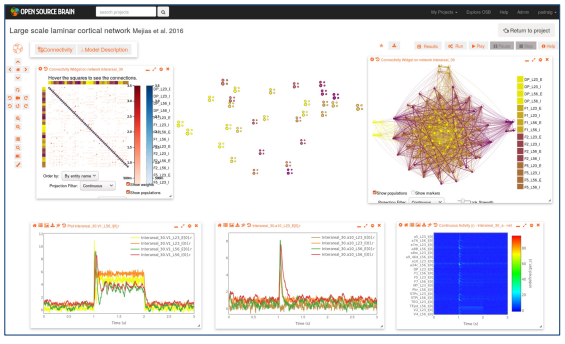
11th Sept 2019

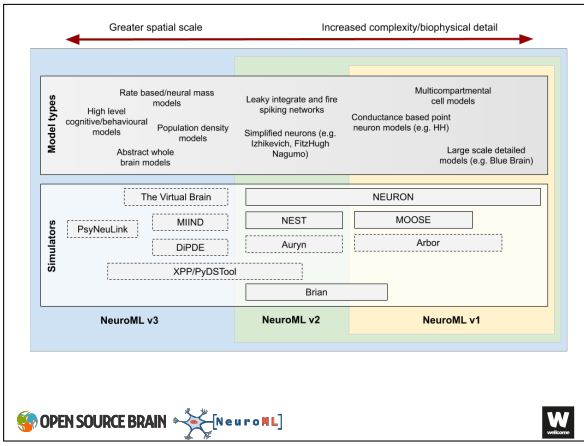
OPEN SOURCE BRAIN NeuroML

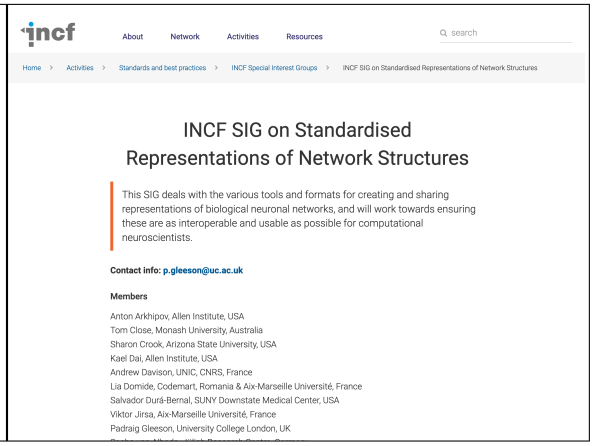




OPEN SOURCE BRAIN NeuroML







Converting simulator specific formats to NeuroML2

Open Source Brain Meeting 2019

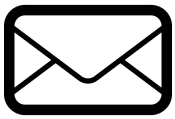
Boris Marin

boris.marin@ufabc.edu.br

Universidade Federal do ABC

Converting NMODL to NeuroML

The Simple™, OSB sponsored way of converting models to NeuroML2



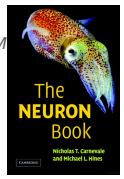
mailto:p.gleeson@ucl.ac.uk

The *NEURON* simulator

Defining models in *NEURON*

https://www.neuron.yale.edu/neuron/

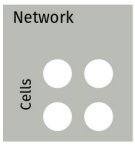
- Cells, Networks: *hoc* language (accessible from Python)
  - morphologies
  - synaptic connections
  - *.hoc* files
- Ion Channels (membrane mechanisms): *NM*
  - *.mod* files



What is NeuroML, and why should I care?

Why can OSB process any NeuroML2 file?

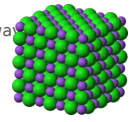
- NML is *structured* (not unlike a *Type System*)



It is all about **Structure**

Structure in NeuroML / NMODL

- A *Type System* (composability rules) is what grants NML its superpowers
- `nmodl` is also powerful, but can be used as a general purpose language
  - `VERBATIM` blocks
  - many different ways of achieving same goal
  - prone to *unstructuredness*
- OSB could in theory treat `nmodl` the same way NML...
  - .... if only people stuck to "good practices"!



Levels of Abstraction

NeuroML2

```
<ionChannelHH id="kChan" conductance="10pS" species="k">

    <gateHHrates id="n" instances="4">
        <forwardRate type="HHExpLinearRate" rate="0.1per_ms" midpoint="-55mV" scale="10mV"/>
        <reverseRate type="HHExpRate" rate="0.125per_ms" midpoint="-65mV" scale="-80mV"/>
    </gateHHrates>

</ionChannelHH>
```

NMODL

```
BREAKPOINT {
        SOLVE states METHOD cnexp
        gk = gkbar * n ^ 4
        ik = gk * (v-ek)
}
INITIAL{
        n = alpha(v) / (alpha(v) + beta(v))
}
DERIVATIVE states{
        n' = (1 - n) * alpha(v) - n * beta(v)
}
```

```
FUNCTION alpha(Vm(mV))(/ms){
        LOCAL x
        UNITSOFF
        x = (Vm + 55) / 10
        if(fabs(x) > 1e-6){
                alpha=0.1*x/(1-exp(-x))
        }else{
                alpha=0.1/(1-0.5*x)
        }
        UNITSON
}
```

Levels of Abstraction

Declarative vs Imperative

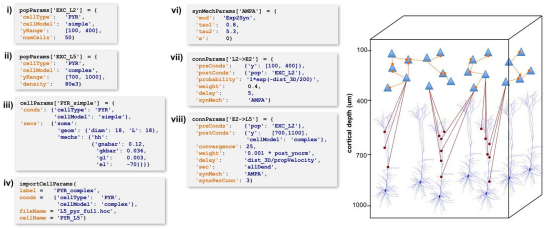
- NeuroML2 operates (at least syntactically) closer to the level of abstraction employed by electrophysiologists
- The gory details exist, but elsewhere: *LEMS*
  - i.e. what to do with α, β; the definition of an `ExpRate`; how all of that is converted to conductances/currents...
- But we seldom need (want!) to interact with that level (look under the hood)

```
<Network ...>
    <Cell ...>
        <Channel ...>
            <Gate ...>
                <Rate ...>
```

```
SOLVE{...} METHOD euler
...
DERIVATIVE {...}
...
FUNCTION trap(v){...}
```

Levels of Abstraction

NetPyNE: structured network specification

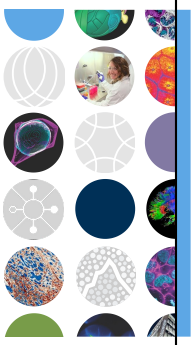


Dura-Bernal, Salvador, et al. "NetPyNE, a tool for data-driven multiscale modeling of brain circuits." Elife 8

ALLEN INSTITUTE

Large-scale Datasets and Modeling Tools from the Allen Institute for Brain Science

Yazan N. Billeh
yazanb@alleninstitute.org



ALLEN INSTITUTE | ALLEN INSTITUTE for BRAIN SCIENCE | ALLEN INSTITUTE for CELL SCIENCE | ALLEN INSTITUTE for IMMUNOLOGY | THE PAUL G. ALLEN FRONTIERS GROUP

- Established 2003 by Paul G. Allen
- South Lake Union, Seattle, WA
- 500 employees++

hard problems
complexity
foundational biology

→ **big science**
team science
open science

→ **data**
knowledge
tools

CORE PRINCIPLES

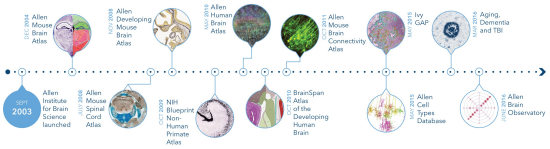
Team Science
Interdisciplinary teams working towards common goal

Big Science
Large-scale projects with robust, massive data

Open Science
All resources available online at brain-map.org or allencell.org

Allen Institute - Online Public Resources

www.brain-map.org

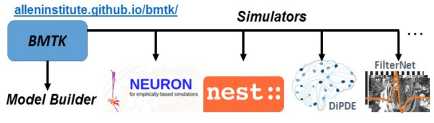


All data are:

- publicly accessible via API as soon as they pass QC
- freely available without any commercial restrictions

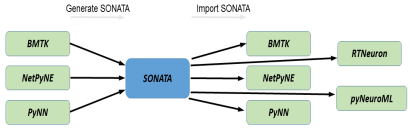
Our Models and Modeling Software Are Freely Available to the Community

Brain Modeling ToolKit (BMTK): https://alleninstitute.github.io/bmtk/



Our Models and Modeling Software Are Freely Available to the Community

Scalable Open Network Architecture TemplAte (SONATA):
https://github.com/AllenInstitute/sonata
An interface between SONATA and the NWB format has been developed as well



Human Brain Project

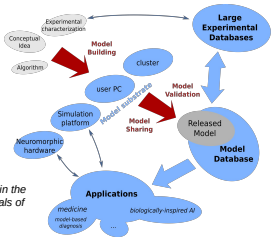
How model standardization enables new tools and applications in neuroscientific research

Insights from the HBP

Yann Zerlaut
Neuroinformatics team / group of A. Davison
Centre National de la Recherche Scientifique, France

Open Source Brain Meeting 2019, Alghero

Motivation



Model production pipeline within the infrastructure and research goals of the Human Brain Project

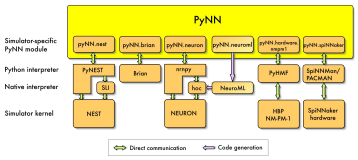
PyNN — A unified interface for neuronal network simulators

Simulator-independent environments for developing neuroscience models:

- keep the advantages of having multiple simulators or hardware devices
- but remove the translation barrier.

Three (complementary) approaches:

- GUI (e.g. neuroConstruct)
- XML-based language (e.g. NeuroML, NineML)
- interpreted language (e.g. Python)



Sonata (pyNN support)

Large-scale simulation of biophysically-detailed neuronal circuits
→ sets specific constraints

the SONATA Data Format emerges as the standard optimized for performance for simulation, analysis and visualization of large-scale circuits
(joint initiative of Blue Brain Project and the Allen Institute for Brain Science)

Export to Sonata format

```
from pyNN.network import Network
from pyNN.serialization import export_to_sonata

sim.setup()
...
# create populations, projections, etc.
...

# add populations and projections to a Network
net = Network(pop1, pop2, ..., prj1, prj2, ...)

export_to_sonata(net, "sonata_output_dir")
```

Import from Sonata format

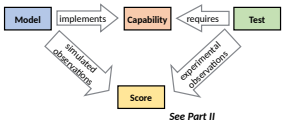
```
from pyNN.serialization import import_from_sonata, load_sonata_simulation_plan
import pyNN.neuron as sim

simulation_plan = load_sonata_simulation_plan("simulation_config.json")
simulation_plan.setup(sim)
net = import_from_sonata("circuit_config.json", sim)
simulation_plan.execute(net)
```

SciUnit

https://github.com/scidash/sciunit

Include a validation framework in model development



See Part II

✓ **What is SciUnit?**

*A **Test**-driven framework for formally validating scientific models against data.*

*It employs the concept of **Capabilities**.*

✓ **What are Tests?**

A procedure intended to establish the quality, performance, or reliability of a model

✓ **What are Capabilities?**

- *interfaces through which the model and the validation framework communicate*
- *implemented as methods (functions) within the model*

Requires the participation of modellers:

- ✓ support to wrap your models for SciUnit
- ✓ add/request new tests to the library
- ✓ critique existing tests
- ✓ suggest new features

Test Packages

The overall test suite has been divided into a number of components, some containing validation tests specific to particular brain regions, others more generic. All validation tests are written in Python, using the SciUnit framework. Some of these are listed below:

Test suites for specific brain regions

- ❑ ***HippoUnit***: `https://github.com/KaliLab/hippounit`
- ❑ ***HippoNetworkUnit***: `https://github.com/pedroernesto/HippoNetworkUnit`
- ❑ ***CerebUnit***: `https://github.com/lungsi/cerebellum-unit`
- ❑ ***BasalUnit***: `https://github.com/appukuttan-shailesh/basalunit`

Test suites for model features, independent of cell type or brain region

- ❑ ***MorphoUnit***: `https://github.com/appukuttan-shailesh/morphounit`
- ❑ ***NetworkUnit***: `https://github.com/mvonpapen/simrest_validation`
- ❑ ***eFELUnit***: `https://github.com/appukuttan-shailesh/eFELunit`

Summary