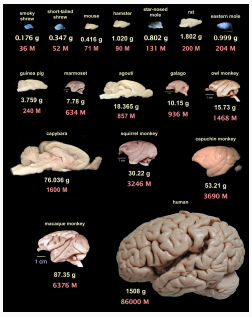
The NeuroML ecosystem for standardised multi-scale modelling in neuroscience

Ankur Sinha
Silver Lab
Department of Neuroscience, Physiology, & Pharmacology
University College London

2024-02-26

An understanding of the brain



- ~86B neurons
- ~100T synapses
- also ~85B glia
- specialised circuits
- different neuronal types
- synaptic connections
- complex sub-cellular processes

[1] Herculano-Houzel, S. **The human brain in numbers: a linearly scaled-up primate brain.** *Frontiers in human neuroscience* **3**, 31 (2009)

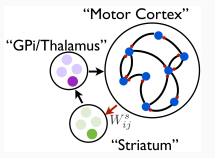
[2] von Bartheld, C. S. *et al.* **The search for true numbers of neurons and glial cells in the human brain: A review of 150 years of cell counting.** *Journal of Comparative Neurology* **524**, 3865–3895. ISSN: 1096-9861 (June 2016)

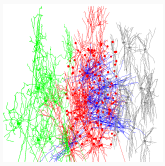
Models complement experimental neuroscience

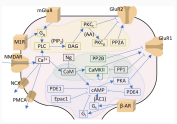
Models are fully observable, controllable.

- Combine individual experimental results into unified theories
- Explore generalisability of experimental results over wider range of conditions
- Generate new experimentally testable, physically plausible hypotheses: dictate experiment design

Models: different scales







[1] Murray, J. M. **Local online learning in recurrent networks with random feedback.** *eLife* **8** (eds Latham, P. et al.) e43299. ISSN: 2050-084X (2019)

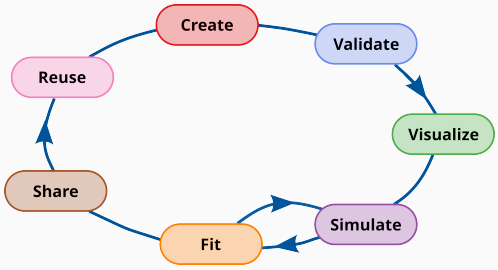
[2] Schirner, M. *et al.* **Learning how network structure shapes decision-making for bio-inspired computing.** *Nature Communications* **14**. ISSN: 2041-1723 (May 2023)

[3] Yao, H. K. *et al.* **Reduced inhibition in depression impairs stimulus processing in human cortical microcircuits.** *Cell Reports* **38**. ISSN: 2211-1247. https://doi.org/10.1016/j.celrep.2021.110232 (Jan. 2022)

[4] Mäki-Marttunen, T. *et al.* **A unified computational model for cortical post-synaptic plasticity.** *eLife* **9** (eds Shouval, H. Z. et al.) e55714. ISSN: 2050-084X. https://doi.org/10.7554/eLife.55714 (July 2020)

A *mechanistic* understanding of the brain requires biophysically detailed modelling

The model life cycle



Computational modelling software ecosystem is fragmented

- many specialist tools:
  - NEURON, NEST, Brian, GENESIS, MOOSE, STEPS, ANNarchy, TVB, LFPy, NeuroLib, EDEN, Arbor, NetPyNE…
- but:
  - different APIs, syntax:
    - increased difficulty for users
  - not well defined model descriptions:
    - models cannot be easily validated
  - custom machine readable internal representations:
    - models cannot be easily inspected/analysed
  - ad-hoc utilities:
    - cannot be used with all tools

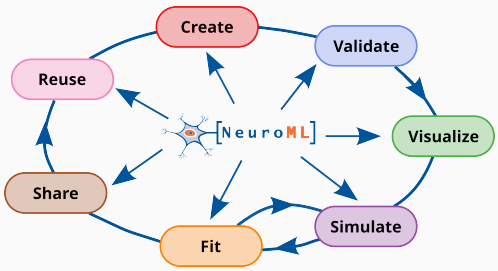
Makes computational neuroscience models
less
FAIR
(Findable, Accessible, Interoperable, Reusable)

Standards enable FAIR neuroscience

[1] Abrams, M. B. *et al.* **A Standards Organization for Open and FAIR Neuroscience: the International Neuroinformatics Coordinating Facility.** *Neuroinformatics* **20**, 25–36. issn: 1559-0089. https://doi.org/10.1007/s12021-020-09509-0 (2022). https://incf.org

[2] COmputational Modeling in Biology NEtwork (COMBINE): https://co.mbine.org/

NeuroML ecosystem supports all stages of the model cycle



NeuroML ecosystem

- standard/specification
- software ecosystem

NeuroML standard

Model specification (schema: XSD)

- elements
- attributes
- hierarchical relationships

Dynamics (LEMS component type definitions)

- dynamical behaviour

NeuroML standard: schema: XSD

Way of specifying the structure of an XML document.

- allows defining types and extensions/restrictions on types to create new types.
- allows generation of APIs

A model description can be validated against the schema before simulation

[1] https://www.w3.org/TR/xmlschema-1/

NeuroML standard: schema: XSD

```
<xs:simpleType name="Uml2Quantity_voltage"> <!-- For params with dimension voltage -->
  <xs:restriction base="xs:string">
    <xs:pattern value="-?([0-9]*(\.[0-9]+)?)([eE]-?[0-9]+)?[\s]*(V|mV)"/>
  </xs:restriction>
</xs:simpleType>

<xs:complexType name="Izhikevich2007Cell">
  <xs:annotation>
    <xs:documentation>Cell based on ...>
    </xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="BaseCellMembPotCap">
      <xs:attribute name="v0" type="Uml2Quantity_voltage" use="required"/>
      <xs:attribute name="k" type="Uml2Quantity_conductancePerVoltage" use="required"/>
      <xs:attribute name="vr" type="Uml2Quantity_voltage" use="required"/>
      <xs:attribute name="vt" type="Uml2Quantity_voltage" use="required"/>
      <xs:attribute name="vpeak" type="Uml2Quantity_voltage" use="required"/>
      <xs:attribute name="a" type="Uml2Quantity_pertime" use="required"/>
      <xs:attribute name="b" type="Uml2Quantity_conductance" use="required"/>
      <xs:attribute name="c" type="Uml2Quantity_voltage" use="required"/>
      <xs:attribute name="d" type="Uml2Quantity_current" use="required"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

[1] Izhikevich, E. M. ***Dynamical systems in neuroscience***. (MIT Press, 2007)

NeuroML standard: LEMS component type definitions

Low Entropy Model Specification language

- domain independent
- allows creation of "Component Types" (classes) from which "Components" (objects) can be instantiated by providing the necessary parameters
- provides a reference implementation/simulator
- machine readable: translatable into other formats

[1] Cannon, R. C. *et al.* **LEMS: a language for expressing complex biological models in concise and hierarchical form and its use in underpinning NeuroML 2.** *Frontiers in Neuroinformatics* **8** (2014)

NeuroML standard: dynamics (LEMS)

```
<ComponentType name="izhikevich2007Cell" extends="baseCellMembPotCap"
    description="Cell based ...">

    <Parameter name="v0" dimension="voltage" description="Initial membrane potential"/>

    <!--
    Defined in baseCellMembPotCap:
    <Parameter name="C" dimension="capacitance"/>
    -->
    <Parameter name="k" dimension="conductance_per_voltage"/>

    <Parameter name="vr" dimension="voltage" description="Resting membrane potential"/>
    <Parameter name="vt" dimension="voltage" description="Spike threshold"/>
    <Parameter name="vpeak" dimension="voltage" description="Peak action potential value"/>

    <Parameter name="a" dimension="per_time" description="Time scale of recovery variable u"/>
    <Parameter name="b" dimension="conductance" description="Sensitivity of recovery variable u to subthreshold
    ↪  fluctuations of membrane potential v"/>
    <Parameter name="c" dimension="voltage" description="After-spike reset value of v"/>
    <Parameter name="d" dimension="current" description="After-spike increase to u"/>

    <Attachments name="synapses" type="basePointCurrent"/>

    <Exposure name="u" dimension="current" description="Membrane recovery variable"/>

    <Dynamics><!-- snipped --></Dynamics>

</ComponentType>
```

NeuroML standard: XSD and LEMS

XSD:

```
<xs:attribute name="v0" type="Nml2Quantity_voltage" use="required"/>
<xs:attribute name="k" type="Nml2Quantity_conductancePerVoltage" use="required"/>
<xs:attribute name="vr" type="Nml2Quantity_voltage" use="required"/>
<xs:attribute name="vt" type="Nml2Quantity_voltage" use="required"/>
<xs:attribute name="vpeak" type="Nml2Quantity_voltage" use="required"/>
<xs:attribute name="a" type="Nml2Quantity_pertime" use="required"/>
<xs:attribute name="b" type="Nml2Quantity_conductance" use="required"/>
<xs:attribute name="c" type="Nml2Quantity_voltage" use="required"/>
<xs:attribute name="d" type="Nml2Quantity_current" use="required"/>
```

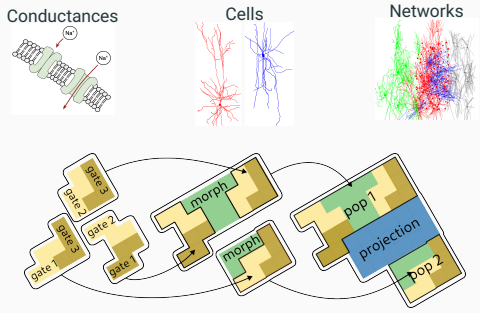
LEMS:

```
<Parameter name="v0" dimension="voltage" description="Initial membrane potential"/>
<Parameter name="k" dimension="conductance_per_voltage"/>
<Parameter name="vr" dimension="voltage" description="Resting membrane potential"/>
<Parameter name="vt" dimension="voltage" description="Spike threshold"/>
<Parameter name="vpeak" dimension="voltage" description="Peak action potential value"/>
<Parameter name="a" dimension="per_time" description="Time scale of recovery variable u"/>
<Parameter name="b" dimension="conductance" description="Sensitivity of recovery variable u to subthreshold
↪ fluctuations of membrane potential v"/>
<Parameter name="c" dimension="voltage" description="After-spike reset value of v"/>
<Parameter name="d" dimension="current" description="After-spike increase to u"/>
```

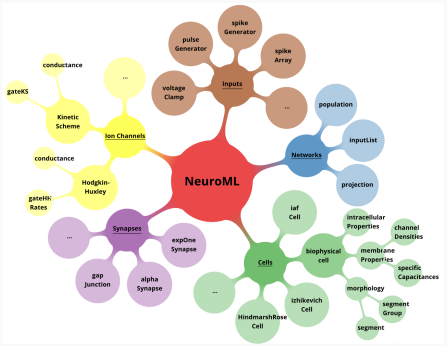

NeuroML standard: dynamics (LEMS)

```
<ComponentType name="izhikevich2007Cell" extends="baseCellMembPotCap"
    description="Cell based ...">
    <!-- snipped ... -->
    <Attachments name="synapses" type="basePointCurrent"/>

    <Exposure name="u" dimension="current" description="Membrane recovery variable"/>

    <Dynamics>
        <StateVariable name="v" dimension="voltage" exposure="v"/>
        <StateVariable name="u" dimension="current" exposure="u"/>

        <DerivedVariable name="iSyn"  dimension="current" exposure="iSyn" select="synapses[*]/i" reduce="add" />

        <DerivedVariable name="iMemb" dimension="current" exposure="iMemb" value="k * (v-vr) * (v-vt) + iSyn - u"/>

        <TimeDerivative variable="v" value="iMemb / C"/>
        <TimeDerivative variable="u" value="a * (b * (v-vr) - u)"/>

        <OnStart>
            <StateAssignment variable="v" value="v0"/>
            <StateAssignment variable="u" value="0"/>
        </OnStart>

        <OnCondition test="v .gt. vpeak">
            <StateAssignment variable="v" value="c"/>
            <StateAssignment variable="u" value="u + d"/>
            <EventOut port="spike"/>
        </OnCondition>

    </Dynamics>
</ComponentType>
```


NeuroML is declarative, modular, structured, hierarchical

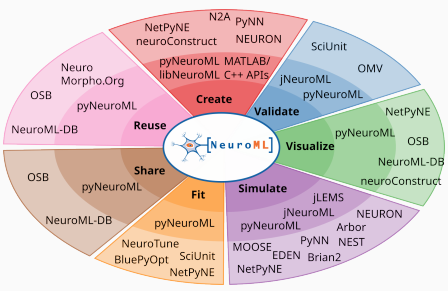


NeuroML provides users with a set of curated model elements

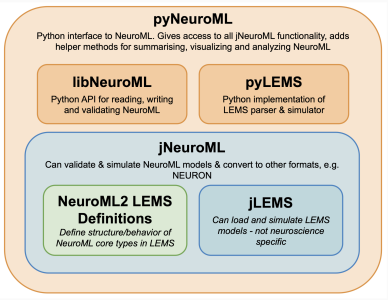


[1] Full standard is at: https://docs.neuroml.org/Userdocs/Specification.html

NeuroML software ecosystem



NeuroML software ecosystem: core tools




```
pip install pyneuroml
```


NeuroML software ecosystem: pyNeuroML

```
# validation
validate_neuroml2("file.nml")
doc.validate(recursive=True)

# inspection
element.info()
summary(doc)
nml2_to_png(doc)
nml2_to_svg(doc)
generate_nmlgraph(doc)

# visualisation/analysis
plot_2D(cell)
plot_interactive_3d(cell)
plot_interactive_3d(network)

plot_channel_densities(cell)
plot_time_series(file)


# simulation
run_lems_with_jneuroml("sim.xml")
run_lems_with_jneuroml_neuron("sim.nml")
run_on_nsg("jneuroml_neuron", "sim.xml")

# sharing
create_combine_archive("sim.xml")
```

```
$ pynml "file.nml" -validate


$ pynml-summary "file.nml"
$ pynml -png "file.nml"
$ pynml -svg "file.nml"
$ pynml "file.nml" -graph
$ pynml "file.nml" -matrix 1
$ pynml-plotmorph "cell.nml"
$ pynml-plotmorph -i "cell.nml"
$ pynml-plotmorph -i "network.nml"
$ pynml-channelanalysis "channel.nml"
$ pynml-plotchan "channel.nml"
$ pynml-plotspikes "sim.xml"
$ pynml-plottimeseries "sim.xml"
$ pynml-plottimeseries "*.dat"


$ pynml "sim1.xml"
$ pynml "sim1.xml" -neuron -run


$ pynml-archive "file.xml"
```


NeuroML: creating/simulating models

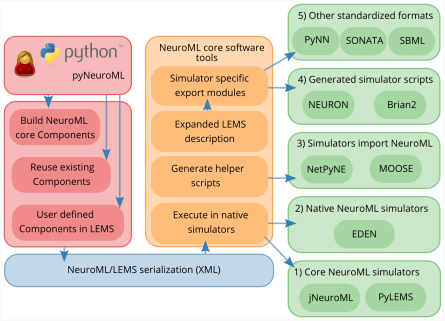
Python script to create a new network, and validate it:

```
from neuroml import * # NeuroML API libNeuroML

newdoc = NeuroMLDocument(id="new_doc")
newcell = IafTauCell(id="cell_0", leak_reversal="-60mV", thresh="0mV", tau="5ms", reset="-70mV")
newdoc.add(newcell)

network = newdoc.add(Network, id="new_net", validate=False)
population = network.add(Population, id="new_pop", size=10, component=newcell.id)

# Helper method to ensure all parameters
# present and appropriate
newdoc.validate(recursive=True)
```

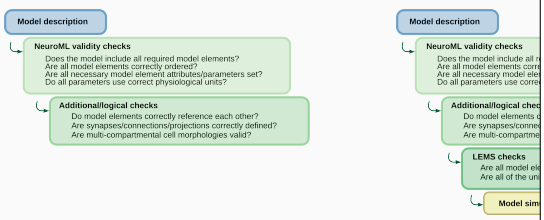
Resultant NeuroML XML serialization:

```
<neuroml id="new_doc">
  <iafTauCell id="cell_0" leakReversal="-60mV" thresh="0mV" reset="-70mV" tau="5ms"/>
  <network id="new_net">
    <population id="new_pop" component="cell_0" size="10"/>
  </network>
</neuroml>
```

NeuroML: creating/simulating models



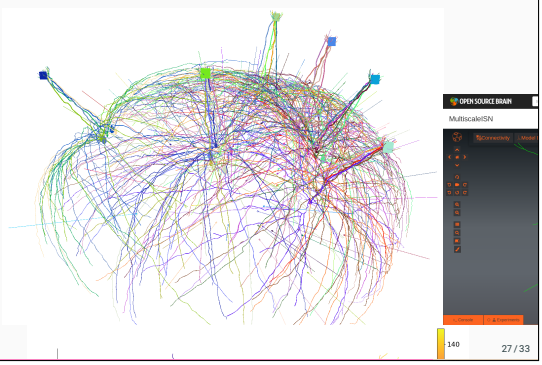
NeuroML: validating models



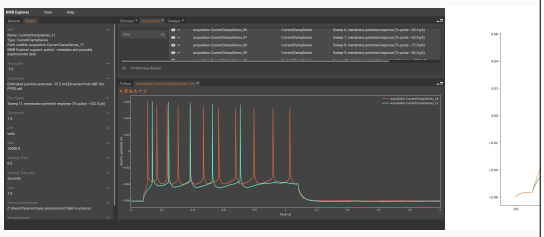
[3] https://github.com/OpenSourceBrain/osb-model-validation

[4] https://sciunit.io

NeuroML: visualising/analysing models



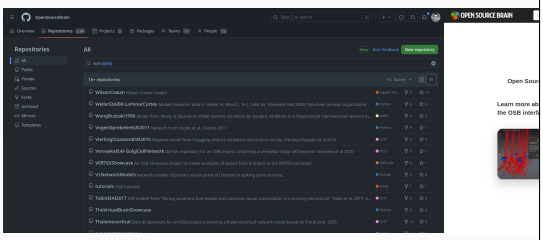
NeuroML: fitting models



[1] Visualising Lanore et al. [12] in NWB Explorer on Open Source Brain

[2] Izhikevich cells fitted using NeuroML fitting pipeline: NeuroTune, using Inspyred

NeuroML: sharing and re-using models



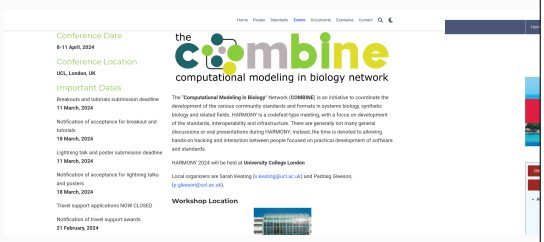
[1] Standardized models on Github: Open Source Brain: https://github.com/OpenSourceBrain

[2] Standardized models on Open Source Brain v1: https://v1.opensourcebrain.org

[3] Standardized models on Open Source Brain v2: https://v2.opensourcebrain.org

[4] Standardized models on NeuroML-DB: https://neuroml-db.org

NeuroML: community: events



[3] https://co.mbine.org/events/

[4] https://www.cnsorg.org

NeuroML: projects: GSoC

- Open source, cross simulator, large scale network models in NeuroML and PyNN
- Implementation of SWC to NeuroML converter in PyNeuroML
- Incorporate new features into an advanced, cross-platform 3D viewer for NeuroML cells and networks

https://summerofcode.withgoogle.com/programs/2024/organizations/incf

NeuroML: closing the neuroscience research loop with OSB

- Open Source Brain Video

NeuroML: resources

Sinha, A. *et al.* **The NeuroML ecosystem for standardized multi-scale modeling in neuroscience.** *bioRxiv.* eprint: https://www.biorxiv.org/content/early/2023/12/11/2023.12.07.570537.full.pdf. https://www.biorxiv.org/content/early/2023/12/11/2023.12.07.570537 (2023)(in review)

https://docs.neuroml.org

https://opensourcebrain.org