

Xolotl

A fast and flexible neuronal simulator

Alec Hoyland

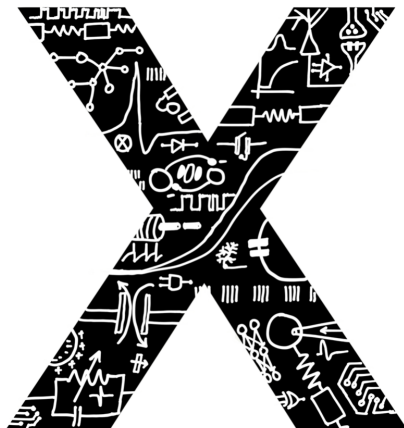
Center for Systems Neuroscience

March 29, 2019

Structure of Talk

"What" more than "Why and How"

- ① What is xolotl?
- ② Features
- ③ Demonstrations
 - ① My first neuron
 - ② My first network
 - ③ Demos & interactive demos

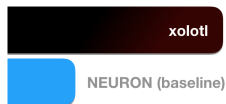


Contact me at: ahoyland@bu.edu

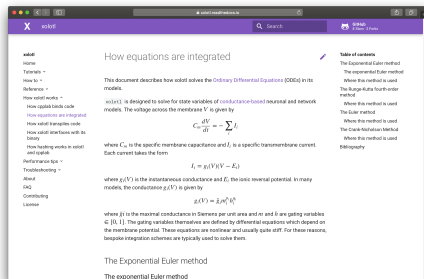
Design Principles

Xolotl should be

- fast
- easy-to-use
- well-documented
- hackable and extensible
- auditable



more than
3x
faster



How xolotl works

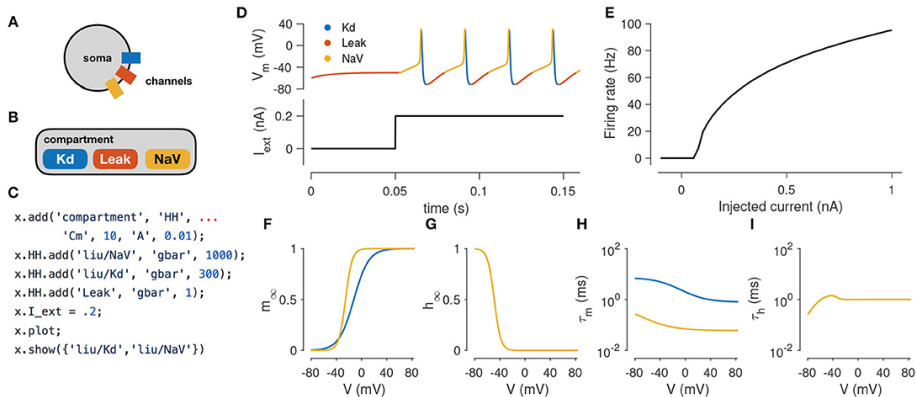


Figure: Model of A & B represented in code C which produces D-I.

Anatomy of a model

Types of Components:

- Compartments
 - ▶ Mechanisms
 - ▶ Conductances
 - ★ Mechanisms
 - ▶ Synapses
 - ★ Mechanisms



Figure: 100+ components are a searchable, indexed feature of the language.

Code to generate an HH model with constant injected current:

```
x = xolotl;  
x.add('compartment', 'HH', 'Cm', 10, 'A', 0.01);  
x.HH.add('liu/NaV', 'gbar', 1000);  
x.HH.add('liu/Kd', 'gbar', 300);  
x.HH.add('Leak', 'gbar', 1);  
x.I_ext = 0.2;
```

% How xolotl prints in the console

```
>> x  
xolotl object with  
-----  
+ HH  
  > Kd (g=300, E=-80)  
  > Leak (g=1, E=-55)  
  > NaV (g=1000, E=30)  
-----
```

Cool features

- puppeteer: real-time parameter manipulation
- xgrid: parallel simulation across a distributed network
- xfit: parameter optimization using particle swarm and genetic algorithms
- xtools: spike counting and data analysis
- model hashing and snapshotting
- control over input and output (clamping, full state matrix)
- automatic component generation from MATLAB
- hyperlinking and tab-completion in the console
- multiple solvers, look-up table caching

Coming Soon

- multi-threading of a single simulation
- server-side compilation / stand-alone integration
- (multi-compartment) server-side GPU computation of Hines matrices
- new compartment types (including low-dimensional models)
- universal support for Runge-Kutta integration schemes
- adaptive time-step solvers (quadrature)
- robust front-end unit support
- Julia front-end (compatible with Python, etc.)

Real-time manipulation

Real-time parameter manipulation. Any numerical xolotl property can be manipulated.

xfit: Parameter optimization

Optimized for:

- 1 Slow-wave troughs at -70 mV.
- 2 Slow-wave peaks at -40 mV.
- 3 Spike downswing ends above slow wave trough.
- 4 Burst frequency of 0.5 Hz.
- 5 Duty cycle of 0.3.

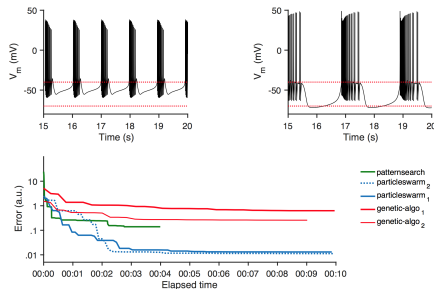


Figure: Fit of 8-conductance model (left to right). PSO #2 shown.

Installing

Acquiring the MATLAB toolbox

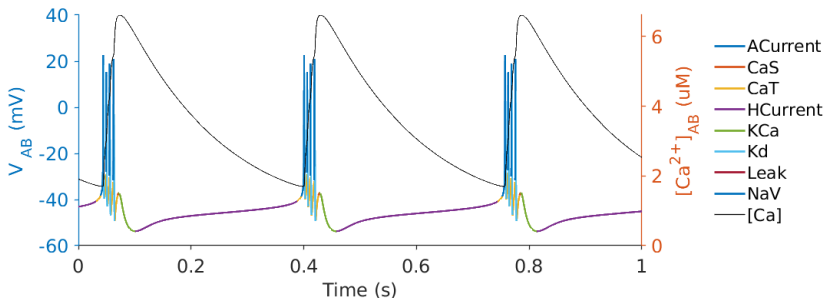
- 1 Go to <https://github.com/sg-s/xolotl/releases/>
- 2 From the 22-March-2019 release, download `xolotl.mltbox`
- 3 Find the file in Downloads and drag it onto your MATLAB workspace
This will install `xolotl`

Run the following commands in MATLAB. You should see this plot.

```
% setup the C++ compiler
mex -setup c++
mex -setup c
% click the link for the MinGW64 Compiler (C++)

% rebuild the component cache
xolotl.rebuildCache

% test to make sure everything is correct
xolotl.go_to_examples
demo_bursting_neuron
```



Demonstrations

Your first neuron

<https://xolotl.readthedocs.io/en/master/tutorials/first-neuron/>

Your first network

<https://xolotl.readthedocs.io/en/master/tutorials/first-network/>

All demos

<https://xolotl.readthedocs.io/en/master/tutorials/built-in-demos/>