1

Xolotl: An Intuitive and Approachable Neuron & Network Simulator in MATLAB

Alec Hoyland [1†]**, Srinivas Gorur-Shandilya** [1†]**, and Eve Marder** [1*]

[1] *Volen Center for Complex Systems and Biology Department*
Brandeis University
Waltham, MA

† *These authors have equally contributed to this article.*

Correspondence*:
Eve Marder
Marder Lab, Brandeis University, Biology Department and Volen Center for
Complex Systems, Waltham, MA, USA
`marder@brandeis.edu`

An Intuitive Neuronal Simulator

2 **ABSTRACT**

3   `xolotl` is a free and open-source neuronal simulator written in `C++` with `MATLAB` wrappers.
4 Biophysically-detailed models of networks can be designed efficiently using an intuitive lan-
5 guage tightly coupled to the object-based architecture of the underlying `C++` code. Models
6 can be specified by adding conductances to compartment objects. The structure is mod-
7 ular, serialized, and searchable, permitting high-level programmatic control over nearly all
8 features of the models. `C++` templates are provided for developing new conductances, com-
9 partments, and integration schemata. It also includes a customizable graphical user interface
10 (GUI) for rapid prototyping and hand-tuning conductances in real-time. The modular structure
11 and accessibility to all parameters, variables, and dynamics of the model network in `MATLAB`
12 facilitate rapid construction and assessment of model networks. `xolotl` is freely available at
13 `https://github.com/marderlab/xolotl`. This tool provides straightforward implementa-
14 tion and fast simulation of neuronal models while permitting full control over every aspect of the
15 network and integration.

16 **Keywords: simulator, MATLAB, C++, conductance-based, neuron, network, keyword**

1   INTRODUCTION

17 `xolotl` (`https://github.com/marderlab/xolotl`) is a fast single-compartment and multi-
18 compartment simulator in `C++` with `MATLAB` wrappers. Written with an emphasis on ease-of-use,
19 `xolotl` can simulate single-compartment conductance-based models, networks of these, and detailed
20 multi-compartment models. `xolotl` exploits a novel automatic type system, `cpplab`, which binds
21 `MATLAB` code to `C++` header files, creating objects and classes *ad libitum* in `MATLAB` which reflect
22 the underlying object-oriented code. `xolotl` implements `cpplab` to represent the nested structure of
23 conductance-based models, and exploits the computational efficiency of the low-level programming lan-
24 guage to quickly integrate models. For this reason, models can be implemented entirely in `MATLAB` with
25 a few lines of code.

26   Models are specified in `MATLAB` by a nested structure. The `xolotl` object contains compartments
27   which themselves contain conductances. Synapses belong to the `xolotl` object and connect compart-
28   ments together. The high-level specification supports arbitrarily large network and multi-compartment
29   morphologies.

30   The software has been implemented in `MATLAB` due to its ease-of-use and popularity among neuroscien-
31   tists. `cpplab` provides a powerful backend for specifying and integrating models without relying on the
32   significantly slower and limiting `MATLAB` `codegen`. While automated `C++` transpiling from `MATLAB`
33   using the proprietary `codegen` can drastically improve performance over loops through strong typing
34   and memory pre-allocation, supervenience of `MATLAB` over `C++` prevents efficient use of low-level fea-
35   tures, such as passing by reference and object-oriented programming. Minimal experience with `MATLAB`
36   is required to use `xolotl`, and all equations and integration methods are provided transparently to the
37   end user. No string parsing of equations is required (Sherfey et al. 2018; Stimberg, D F M Goodman, et al.
38   2014; Stimberg, D F Goodman, et al. 2013)

39   `xolotl` comes packaged with visualization functions and a graphical user interface (GUI) for real-
40   time manipulation of model parameters. Plotting of voltage, intracellular calcium, conductance gating
41   functions, and time constants is provided by built-in `xolotl` methods. The GUI permits real-time tuning
42   of any network parameters using numerical sliders in a graphical interface which displays the resultant
43   membrane potential and intracellular calcium traces. The ease-of-use of these tools lends them to peda-
44   gogical applications and rapid exploration of toy models. This tool aims to simplify the investigation of
45   dynamics of complex neural network models, facilitate collaborative modeling, and complement other
46   tools being developed in the neuroinformatics community.

2   DESIGN GOALS

47   `xolotl` is designed to be easy-to-use without sacrificing speed.

48   The software has been designed in `MATLAB` due to its popularity among neuroscientists for pedagogy
49   and research. `xolotl` capitalizes on `MATLAB`'s straightforward structure array syntax to permit rapid
50   prototyping and experimentation, especially for small neuronal networks of complex models. Parameters
51   of conductances, neuronal compartments, and simulations may all be edited in the structure before any
52   calls to integration functions. The underlying code is written in `C++` for speed and memory optimization,
53   and while models can indeed be integrated using the compiled binary, symbolic manipulations can be
54   readily performed in `MATLAB` without ever touching the foundational code.

2.1   FEATURES

55   `MATLAB` provides a high level programmatic and graphical interface for implementing, manipulating, and
56   visualizing models without sacrificing the enhancements of the underlying `C++` code.

57   *Modular structure.* Models are specified by adding compartments and synapses to the `xolotl` object.
58   Conductances are added to compartments and controllers can be added to conductances. This modular
59   structure recapitulates the biophysics of the Hodgkin-Huxley formalism and obviates the need to explicitly
60   write out equations, which in `xolotl` are contained within the conductance header files.

61   *Interface between C++ and MATLAB.* `xolotl` relies on `cpplab` constructions, which allow the user
62   to exploit the efficiency of low-level `C++` code. `MATLAB` treats `cpplab` objects as fully-typed variables
63   allowing for symbolic manipulation using only the high-level programming language and graphical inter-
64   faces. `xolotl` is fast because all time-intensive code is written in `C++`. While automated `C++` transpiling
65   from `MATLAB` using the proprietary `codegen` can drastically improve performance over loops through
66   strong typing and memory pre-allocation, supervenience of `MATLAB` over `C++` prevents efficient use of
67   low-level features, such as passing by reference and object-oriented programming. `C++` provides speed

68 improvements beyond the benefits of translating `MATLAB` features into low-level code. For this reason,
69 `cpplab` has been designed to provide an interface for constructing, transpiling, and compiling `C++` code
70 to be called from within `MATLAB`. `xolotl` simulations are run entirely from `C++` executables.

71 *Automatic and efficient compiling.* `xolotl` automatically handles transpiling and compiling `MATLAB`
72 code into `C++`. The MD5 algorithm hashes the network to compile a new binary and `MEX` bridge file only
73 if needed and to confirm that the correct binary fetched during execution.

2.2 SYNTAX

74 `MATLAB` can easily control the `cpplab` objects using the standard, flexible data structure notation popular
75 in high-level scripting languages.

76 *Adding features.* The `add` function creates a compartment, conductance, or controller and affixes it as
77 a field in the `xolotl` network structure. This function generates a `MATLAB` `struct` that faithfully
78 represents the underlying `C++` code. Compartments add to the `xolotl` object and conductances add to
79 compartments. Specific properties can be specified using key-value pair arguments (e.g. Figure 1A).

80 *Finding features.* `cpplab` comes with several features which simplify the handling of complexly-nested
81 models. The `find` function acquires a cell array of all properties of the network which satisfy a search
82 condition. For example, one can find all paths to maximal conductances within the `'HH'` compartment
83 by:

84   `x.find('HH*gbar');`

85   To extract a vector of the maximal conductances:

86   `gbars = x.get('HH*gbar');`

87   To set the maximal conductances all at once:

88   `x.set('HH*gbar', gbars)`

89 *Compartments.* A model neuron consists of one or more compartments, each representing a section of
90 membrane with capacitance and surface area. Isopotential models require one compartment, whereas
91 models with multiple neurons, units, or non-trivial morphology require multiple compartments. All
92 specifiable properties of compartments are shown in Supplementary Table 1.

93 *Synapses.* `xolotl` provides some features for generating complex models. Synapses can be added
94 with the `connect` function. At minimum synapses possess identifiers to presynaptic and postsynap-
95 tic compartments and default to electrical synapses. All specifiable properties of synapses are shown in
96 Supplementary Table 2. To create axons or transport chains, the `slice` function splits a compartment
97 into *n* discrete segments and adds these compartments to the network connected by electrical synapses.

98 *Conductances and controllers.* All conductances contain fields for maximal conductance and reversal
99 potential. Conductances with activation and inactivation variables include them as *m* and *h* respectively.
100 Gating functions and their respective time constants are contained within the conductance header file.
101 `xolotl` comes packaged with conductances from several dozen papers (Supplementary Table 3).

102 *Creating custom* `cpplab` *objects.* `xolotl` contains template header files for producing custom con-
103 ductances. The template contains instructions on how to design novel conductances with arbitrary
104 specifications.

105 *Simulation.* Models are simulated in `xolotl` with the `integrate` function which outputs as time series
106 the membrane potentials, intracellular calcium concentrations, controller states, intrinsic currents, and
107 synaptic currents. The `integrate` function also accepts an argument which specifies injected current or
108 clamped voltage.

109 *Numerical integration.* `xolotl` uses the exponential Euler method for single compartment models, for-
110 ward Euler for gating variables, and a Crank-Nicholson regime for electrically-coupled compartments
111 (Butcher 2016; Dayan and Abbott 2001; Oh and French 2006) These defaults provide a mix of speed,
112 accuracy, and stability, and are built into the `cpplab` header files. Custom `cpplab` header files can
113 be customized with any iterative integration method. The simulation time-resolution can be specified to
114 target arbitrary precision, and an output time step can be selected to support automatic down-sampling for
115 memory considerations.

116 *'Closed-loop' vs. 'open-loop.'* Simulations can be run in 'closed-loop' mode where each simulation be-
117 gins by resetting all dynamical variables to their initial conditions at instantiation, or 'open-loop' mode
118 which begins simulation with the current network state.

119 *Using the graphical interface to manipulate parameters.* `xolotl` comes packaged with a graphical user
120 interface for visualizing parameter changes in real-time. The `manipulate` function opens the GUI,
121 which displays a figure plotting the membrane potential and intracellular calcium concentration of all
122 compartments as time series, and a dialog box with customizable sliders for all parameters of the model,
123 much like the `Manipulate` function in Wolfram `Mathematica`. Moving the sliders integrates the
124 model in 'open-loop' mode with the new parameters. The parameters available in the sliders can be cus-
125 tomized by passing a cell array to manipulate. For example, to only see sliders for maximal conductances
126 of the `HH` compartment, call `x.manipulate('HH*gbar')`. Closing the GUI saves the network state
127 of the model to the `xolotl` object. This is particularly helpful for rapid prototyping of models.

128 *Optimizing parameters.* `xolotl` can use the Global Optimization toolbox for `MATLAB` to optimize any
129 accessible `xolotl` parameters. The toolbox is algorithm-agnostic and accepts any function in `MATLAB`
130 with a scalar first output as the objective function. Simulations run on multi-core processors or high-
131 performance computing clusters using the Parallel Computing toolbox.

2.3 LIMITATIONS

132 The focus on ease-of-use and speed means some features were elided in the streamlining process.

133 *Reliance on compiled `C++` code.* While `MATLAB` comes with robust features for compiling `C` and `C++`
134 code, `xolotl` cannot run without `C++` compilation. For users, this necessitates the additional step of
135 setting up the `mex` compiler which can be problematical, especially for nonstandard (e.g. Arch-based
136 Linux). Secondly, compilation adds a small amount to total processing time. Longer simulations (> 1000
137 time-steps) minimizes this effect. Adding new conductances also requires writing some `C++` code. For
138 model conductances in the Hodgkin-Huxley formalism (Dayan and Abbott 2001; Hodgkin, Huxley, and
139 Katz 1952) adjustments consist of changing default values in a template `C++` header file. Implementing a
140 new integration scheme requires much more in-depth usage of `C++`.

141 *Limited to conductance-based models.* `xolotl` has been developed specifically for conductance-based
142 models. It does not currently support rate- or current-based models.

143 *Limited numerical integration strategies.* While the exponential Euler method performs well in neuronal
144 models (Dayan and Abbott 2001; Oh and French 2006) it may be desirable to use other methods un-
145 der certain conditions. `xolotl` does not currently support other integration schemes for its built-in
146 conductances, nor does the software support error-sensitive variable step-sizes 'out-of-the-box.'

147 *Inefficient tools for handling large networks.* While `xolotl` can integrate large networks (> 1000 com-
148 partments), `xolotl` uses string-based comprehension for labeling compartments which is suited to
149 descriptive naming, but prohibits vector operations over compartments.

3   USAGE EXAMPLES

150 In `MATLAB`, users create a `xolotl` object and populate it with `cpplab`-generated objects which describe
151 compartments, conductances, synapses, and controllers. The model is integrated with the `integrate`
152 function where the membrane potential, intracellular calcium concentration, controller states, intrinsic
153 currents, and synaptic currents can be outputs.

154   `xolotl` comes packaged with a library of pre-existing conductance and synapse objects which greatly
155 simplify the task of constructing model neurons. These objects can be referenced by name and added
156 directly to a compartment. Novel conductance dynamics can be easily written by modifying a template
157 header file contained in the `xolotl` distribution, or designed entirely from scratch.

3.1   SIMULATING A HODGKIN-HUXLEY MODEL

158 The seminal Hodgkin-Huxley model of action potentials in the squid giant axon (Hodgkin and Huxley
159 1952; Hodgkin, Huxley, and Katz 1952) contains a fast inactivating sodium conductance (`NaV`), a non-
160 inactivating delayed rectifier (`Kd`), and a passive leak current (Figure 1A). A compartment, `HH`, with
161 membrane capacitance (`Cm`) and surface area (`A`) can be specified by Figure 1B. Network properties can
162 be set during construction or afterwards using dot-notation in `MATLAB` (e.g. `x.HH.Cm`). Figure 1C shows
163 the `MATLAB` command prompt after invoking the `xolotl` object `x`, displaying the hierarchical structure
164 inherent in conductance-based treatments of neurodynamics.

165   This model was constructed using conductances from Liu et al. 1998 based on electrophysiological
166 recordings from the lobster stomatogastric ganglion (Turrigiano, LeMasson, and E. Marder 1995) In
167 the absence of applied positive current, the model is quiescent. When 0.2 nA is injected, the model
168 tonically spikes (Figure 1D). The `integrate` function takes the applied current as an argument (e.g.
169 `x.integrate(Iapp)`), so that the `xolotl` object is agnostic to integration-specific perturbations.
170 The `plot` function generates voltage and intracellular calcium traces, where the voltage trace is colored
171 by the dominant current. If the membrane potential is increasing, the strongest instantaneous inward cur-
172 rent colors the trace. Conversely, if the membrane potential is decreasing, the strongest outward current
173 colors the trace instead. Figure 1F-I display the results of the `show` function. Activation and inactivation
174 steady-states and the voltage-dependent time constants of these gating variables describe the conductance
175 dynamics in absence of other channel types.

3.2   PERFORMING A VOLTAGE CLAMP EXPERIMENT *IN-SILICO*

176 `xolotl` can recapitulate the results of voltage clamp experiments (Destexhe and Bal 2009; Swensen and
177 E. Marder 2000, 2001; Turrigiano, LeMasson, and E. Marder 1995) Figure Figure 2 displays steps in the
178 procedure to clamp the membrane potential of a cell with delayed rectifier potassium conductance. Dur-
179 ing an *in-vitro* experiment, confounding currents would be pharmacologically-blocked and two-electrode
180 voltage clamp used to record tail currents at fixed membrane potential (Connor and Stevens 1971a,b)

181   A single-compartment model with a delayed-rectifier conductance is simulated at stepped membrane
182 potentials. The model is simulated using the `integrate` function. The second argument determines the
183 clamped voltage and the fourth output is the current trace.

184   ```
[V, Ca, ~, I] = x.integrate([], clamped_voltage)
```

185      Currents under voltage clamp approach the steady-state holding current (Figure 2D-E). The current-
186 voltage relation is the steady-state current over the clamped voltage, and the effective conductance is the
187 derivative of that relation (Figure 2F-G). Since the effective conductance is the product of the maximal
188 conductance and the gating variables (Dayan and Abbott 2001; Turrigiano, LeMasson, and E. Marder
189 1995) and the tail current is monotonically-increasing with time under voltage clamp, the current can be
190 represented as non-inactivating. Fitting a sigmoid to various powers yields a model for the current dy-
191 namics (Figure 2H-I). These figures describe graphically the theoretical underpinnings of current analysis
192 through voltage clamp and can serve as an effective pedagogical tool for computational and quantitative
193 neuroscience.

### 3.3   SIMULATING NETWORK MODELS

194 Network models in `xolotl` consist of compartment objects connected by synapses. Synapses are stored
195 in a vector array as a field of the `xolotl` object in `MATLAB`. Presynaptic and postsynaptic labels in-
196 dicate the connectivity of the synapse. Figure Figure 3 implements a model of the triphasic pyloric
197 rhythm in the stomatogastric ganglion of crustaceans. The pyloric model contains three compartments
198 and seven synapses (Figure 3A). This structure is reciprocated in the hierarchy of the `xolotl` object,
199 where conductances are contained within compartments (Figure 3B).

200      Representing the network in `xolotl` requires constructing three compartments and eight conductances
201 in each using the `add` function.

```
202        x.add('AB', 'compartment', 'Cm', 10, 'A', 0.628, ...)
203        x.AB.add('prinz/NaV', 'gbar', 1000, 'E', 50)
204        ...
```

205   Synapses are upper-level properties of the network which point between two compartments (Figure
206 3C). This exploits vectorized operations in `MATLAB` and does not require each synapse to possess a
207 unique name. The `connect` function adds synapses to the network. The first two arguments specify
208 the presynaptic and postsynaptic compartments. The third dictates the type of synapse. All others follow
209 the `'keyword'`, `value` paradigm and preset parameters of the synapse.

```
210        x.connect('AB', 'LP', 'Chol', 'gbar', 30)
```

3.4   SIMULATING INTEGRAL CONTROL

211 `xolotl` can implement homeostatic tuning rules as integral control. The controller computes an error
212 signal (typically a function of intracellular calcium concentration), and adjusts the conductance or synapse
213 it controls accordingly (O'Leary et al. 2013) In `xolotl`, integral controllers are `cpplab` objects added
214 to the conductance or synapse they regulate.

215   In a demonstration adapted from O'Leary et al. 2013, integral control changes maximal conductances to
216 bring a neuron from quiescence into a bursting regime. Calcium sensors supervene on maximal con-
217 ductance density (Figure 4) to change neuronal activity. Each conductance in the `xolotl` structure
218 contains a calcium-sensitive controller (Figure 4B-C). Maximal conductances increase from random ini-
219 tial conditions to a set which elicits the desired network output by minimizing the error signal (Figure
220 4D-F).

3.5   USING THE GUI TO MANIPULATE PARAMETERS

221 The `manipulate` function opens the GUI which permits visualization of changing parameters in real-
222 time. Moving sliders representing the values of network parameters updates a plot (Figure 5B). By default,
223 the function opens a figure displaying the results of the `plot` function, which shows the voltage and

224 intracellular calcium traces for each compartment (Figure 5A). `manipulate` grants slider control over
225 all `xolotl` parameters by default, but specific ones can be selected by passing them as arguments. For
226 example, to manipulate only the maximal conductances and visualize using the `myPlot` function

227
```
x.manipulate('*gbar', @myplot)
```

4 BENCHMARKS

228 To assess speed and accuracy, `xolotl`, `DynaSim` (Sherfey et al. 2018) and `NEURON` (Hines and
229 Carnevale 1997) were compared in simulations over varied time resolution, simulation time, and number
230 of compartments (Figure 6).

231 Single-compartment Hodgkin-Huxley-like models were generated using conductance dynamics from
232 Liu et al. 1998 in the simulation environments. `xolotl` uses the exponential Euler method for in-
233 tegrating membrane potential (Dayan and Abbott 2001) `DynaSim` was implemented with a 2nd-order
234 Runge-Kutta integration scheme as recommended for high-performance in the documentation. `NEURON`
235 used an implicit Euler regime (Hines and Carnevale 1997)

236 To compare the integration methods, models were simulated for 5 s at varying time-resolution (Figure
237 6A). The ratio between 'simulated' time and actual runtime was defined as the speed factor. Higher values
238 indicate faster simulations. The coincidence factor determines the correlative overlap between two spike
239 trains (Jolivet et al. 2008) To assess accuracy over decreasing time-resolution for the three simulation
240 environments, spike trains at each resolution were compared to a 'canonical' spike train (exponential
241 Euler at a time-step of $dt = 0.001$ ms).

242 To assess the performance of the simulators in absence of set-up overhead, models were simulated with
243 a time-resolution of 0.1 ms over increasing simulation time (Figure 6B). The speed factor was defined as
244 the ratio between time represented in the simulation and actual runtime (simulation-time). Therefore, the
245 speed factor represents how many times faster the simulation is than a real-time observation.

246 Many simulators perform well in simulations of many compartments (Brette et al. 2007; Delorme
247 and Thorpe 2003; Sherfey et al. 2018; Vitay, Dinkelbach, and Hamker 2015) Networks of up to 1,000
248 Hodgkin-Huxley cells were simulated for 5 s at a time-resolution of 0.1 ms (Figure 6C).

249 DESCRIBE BENCHMARK RESULTS

5 DISCUSSION

5.1 REPRODUCIBILITY

250 `xolotl` fosters reproducibility in science. While the availability of hosting sites with version control (viz.
251 GitHub (`https://github.com`), GitLab (`https://gitlab.com/`), and Open Science Frame-
252 work (`https://osf.io/`)) and the push for reproducibility in computational science (Baker 2016;
253 Eklund, Nichols, and Knutsson 2016; Stodden et al. 2016) has resulted in the availability of source code,
254 much of this code base is bespoke and difficult to implement (Sedano 2016; W Xu, D Xu, and Deng 2017)

255 To this end, `xolotl` provides an environment with readability and reproducibility in mind. Each net-
256 work is hashed to provide a unique alphanumeric identifier. Conductance header files are easily viewed in
257 the `xolotl` source files; conductances in `MATLAB` contain links to the full path of the generating file.

5.2 CIRCUMVENTING LANGUAGE TRADEOFFS

258 Executing `C/C++` code in higher-level languages such as `MATLAB` or `Python` often provides speed
259 improvements for iterative code in algorithms.

260 `C` is statically-typed, with procedural syntax that provides low-level access to memory (Kernighan and
261 Ritchie 1978) providing significant advantages for time-intensive computations. Unfortunately, automatic
262 code-generation is limited by the supervening language. `MATLAB`, for instance, cannot use pointers or pass
263 by reference, which limits the efficiency of `C` code automatically generated from `MATLAB`. Conversely,
264 custom `C/C++` code provides significant increases in performance and memory conservation, but lacks
265 the ease-of-use and flexibility of scripting languages.

266 `xolotl` handles this problem through symbolic manipulation of `C++` objects in `MATLAB`. Built from
267 the ground up in `C++`, `xolotl` maintains all the advantages of custom compiled code, but can run in
268 `MATLAB` without the user having to touch the `C++` code. `xolotl` represents compartment, conductance,
269 synapse, and controllers as `cpplab` objects, which map to underlying `C++` header files. In this way,
270 properties of the `xolotl` network can be examined and changed using object-oriented paradigms. The
271 object specifies the `integrate` function, not the other way around.

5.3 APPLICATIONS OF `CPPLAB`

272 NEED TO WRITE THIS

CONFLICT OF INTEREST STATEMENT

273 The authors declare that the research was conducted in the absence of any commercial or financial
274 relationships that could be construed as a potential conflict of interest.

AUTHOR CONTRIBUTIONS

275 SG-S designed and implemented the core of the `xolotl` toolbox. AH contributed to the code base,
276 created the online user documentation, and wrote the manuscript. EM supervised the project. All authors
277 reviewed the paper.

FUNDING

ACKNOWLEDGMENTS

SUPPLEMENTAL DATA

282 Tables including all conductances packaged with `xolotl` should be put in the supplementary material.

DATA AVAILABILITY STATEMENT

283 The code to generate all figures is available at (https://github.com/marderlab/
284 xolotl-paper). xolotl is frely available at (https://github.com/marderlab/xolotl).

REFERENCES

285 Baker, Monya (Sept. 13, 2016) "Why Scientists Must Share Their Research Code". In: *Nature*. ISSN:
286 1476-4687. DOI: 10.1038/nature.2016.20504. URL: http://www.nature.com/
287 doifinder/10.1038/nature.2016.20504 (visited on 05/30/2018)
288 Brette, Romain et al. (Dec. 1, 2007) "Simulation of Networks of Spiking Neurons: A Review of Tools
289 and Strategies". In: *Journal of Computational Neuroscience* 23.3, pp. 349–398. ISSN: 1573-6873. DOI:
290 10.1007/s10827-007-0038-6. URL: https://link-springer-com.resources.
291 library.brandeis.edu/article/10.1007/s10827-007-0038-6 (visited on
292 05/01/2018)
293 Butcher, J. C. (2016) "Numerical Differential Equation Methods". In: *Numerical Methods for Ordi-
294 nary Differential Equations*. Third. Wiley-Blackwell, pp. 55–142. ISBN: 978-1-119-12153-4. DOI: 10.
295 1002/9781119121534.ch2. URL: https://onlinelibrary.wiley.com/doi/abs/
296 10.1002/9781119121534.ch2 (visited on 05/29/2018)
297 Connor, J. A. and C. F. Stevens (Feb. 1971a) "Inward and Delayed Outward Membrane Currents in
298 Isolated Neural Somata under Voltage Clamp". In: *The Journal of Physiology* 213.1, pp. 1–19. ISSN:
299 0022-3751. pmid: 5575338. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/
300 PMC1331719/ (visited on 05/26/2018)
301 — (1971b) "Voltage Clamp Studies of a Transient Outward Membrane Current in Gastropod Neural So-
302 mata". In: *The Journal of Physiology* 213.1, pp. 21–30. ISSN: 1469-7793. DOI: 10.1113/jphysiol.
303 1971.sp009365. URL: https://physoc.onlinelibrary.wiley.com/doi/abs/10.
304 1113/jphysiol.1971.sp009365 (visited on 05/26/2018)
305 Dayan, Peter and L. F. Abbott (2001) *Theoretical Neuroscience*. Computational neuroscience. Cambridge,
306 Mass.: Massachusetts Institute of Technology Press. xv+460. ISBN: 978-0-262-04199-7.
307 Delorme, Arnaud and Simon J. Thorpe (Jan. 1, 2003) "SpikeNET: An Event-Driven Simulation Package
308 for Modelling Large Networks of Spiking Neurons". In: *Network: Computation in Neural Systems* 14.4,
309 pp. 613–627. ISSN: 0954-898X DOI: 10.1088/0954-898X_14_4_301. pmid: 14653495. URL:
310 https://doi.org/10.1088/0954-898X_14_4_301 (visited on 05/01/2018)
311 Destexhe, Alain and Thierry Bal (Mar. 11, 2009) *Dynamic-Clamp: From Principles to Applications*.
312 Springer Science & Business Media. 428 pp. ISBN: 978-0-387-89279-5.
313 Eklund, Anders, Thomas E. Nichols, and Hans Knutsson (June 28, 2016) "Cluster Failure: Why fMRI In-
314 ferences for Spatial Extent Have Inflated False-Positive Rates". In: *Proceedings of the National Academy
315 of Sciences*, p. 201602413. ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.1602413113.
316 pmid: 27357684. URL: http://www.pnas.org/content/early/2016/06/27/
317 1602413113 (visited on 05/30/2018)
318 Hines, M. L. and N. T. Carnevale (Aug. 1, 1997) "The NEURON Simulation Environment". In: *Neural
319 Computation* 9.6, pp. 1179–1209. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.6.1179. URL:
320 https://doi.org/10.1162/neco.1997.9.6.1179 (visited on 04/30/2018)
321 Hodgkin, A. L. and A. F. Huxley (Apr. 1952) "The Components of Membrane Conductance in the
322 Giant Axon of Loligo". In: *The Journal of Physiology* 116.4, pp. 473–496. ISSN: 0022-3751. pmid:
323 14946714.
324 Hodgkin, A. L., A. F. Huxley, and B. Katz (Apr. 28, 1952) "Measurement of Current-Voltage Relations in
325 the Membrane of the Giant Axon of Loligo". In: *The Journal of Physiology* 116.4, pp. 424–448. ISSN:
326 0022-3751. pmid: 14946712. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/
327 PMC1392219/ (visited on 11/16/2017)

Jolivet, Renaud et al. (Apr. 30, 2008) "A Benchmark Test for a Quantitative Assessment of Simple Neuron Models". In: *Journal of Neuroscience Methods* 169.2, pp. 417–424. ISSN: 0165-0270. DOI: 10.1016/j.jneumeth.2007.11.006. pmid: 18160135.

Kernighan, Brian and Dennis M. Ritchie (1978) *The C Programming Language*. Prentice hall.

Liu, Z. et al. (1998) "A Model Neuron with Activity-Dependent Conductances Regulated by Multiple Calcium Sensors". In: *The Journal of Neuroscience: The Official Journal of the Society for Neuroscience* 18.7, pp. 2309–2320. ISSN: 0270-6474. pmid: 9502792.

O'Leary, Timothy et al. (July 9, 2013) "Correlations in Ion Channel Expression Emerge from Homeostatic Tuning Rules". In: *Proceedings of the National Academy of Sciences* 110.28, E2645–E2654. ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.1309966110. pmid: 23798391. URL: http://www.pnas.org/content/110/28/E2645 (visited on 02/14/2018)

Oh, Jiyeon and Donald A. French (Jan. 1, 2006) "Error Analysis of a Specialized Numerical Method for Mathematical Models from Neuroscience". In: *Applied Mathematics and Computation* 172.1, pp. 491–507. ISSN: 0096-3003. DOI: 10.1016/j.amc.2005.02.028. URL: http://www.sciencedirect.com/science/article/pii/S0096300305002183 (visited on 05/29/2018)

Prinz, Astrid A., Cyrus P. Billimoria, and Eve Marder (Dec. 2003) "Alternative to Hand-Tuning Conductance-Based Models: Construction and Analysis of Databases of Model Neurons". In: *Journal of Neurophysiology* 90.6, pp. 3998–4015. ISSN: 0022-3077. DOI: 10.1152/jn.00641.2003. pmid: 12944532.

Prinz, Astrid A., Dirk Bucher, and Eve Marder (Dec. 2004) "Similar Network Activity from Disparate Circuit Parameters". In: *Nature Neuroscience* 7.12, pp. 1345–1352. ISSN: 1097-6256. DOI: 10.1038/nn1352. pmid: 15558066.

Sedano, T. (Apr. 2016) "Code Readability Testing, an Empirical Study". In: *2016 IEEE 29th International Conference on Software Engineering Education and Training (CSEET)* 2016 IEEE 29th International Conference on Software Engineering Education and Training (CSEET) pp. 111–117. DOI: 10.1109/CSEET.2016.36.

Sherfey, Jason S. et al. (2018) "DynaSim: A MATLAB Toolbox for Neural Modeling and Simulation". In: *Frontiers in Neuroinformatics* 12. ISSN: 1662-5196. DOI: 10.3389/fninf.2018.00010. URL: https://www.frontiersin.org/articles/10.3389/fninf.2018.00010/full (visited on 04/30/2018)

Stimberg, Marcel, Dan F. M. Goodman, et al. (2014) "Equation-Oriented Specification of Neural Models for Simulations". In: *Frontiers in Neuroinformatics* 8. ISSN: 1662-5196. DOI: 10.3389/fninf.2014.00006. URL: https://www.frontiersin.org/articles/10.3389/fninf.2014.00006/full (visited on 05/01/2018)

Stimberg, Marcel, Dan FM Goodman, et al. (July 8, 2013) "Brian 2 - the Second Coming: Spiking Neural Network Simulation in Python with Code Generation". In: *BMC Neuroscience* 14 (Suppl 1) P38. ISSN: 1471-2202. DOI: 10.1186/1471-2202-14-S1-P38. pmid: null. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3704840/ (visited on 05/01/2018)

Stodden, Victoria et al. (Dec. 9, 2016) "Enhancing Reproducibility for Computational Methods". In: *Science* 354.6317, pp. 1240–1241. ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.aah6168. pmid: 27940837. URL: http://science.sciencemag.org/content/354/6317/1240 (visited on 05/30/2018)

Swensen, A. M. and E. Marder (Sept. 15, 2000) "Multiple Peptides Converge to Activate the Same Voltage-Dependent Current in a Central Pattern-Generating Circuit". In: *The Journal of Neuroscience: The Official Journal of the Society for Neuroscience* 20.18, pp. 6752–6759. ISSN: 0270-6474. pmid: 10995818.

— (June 1, 2001) "Modulators with Convergent Cellular Actions Elicit Distinct Circuit Outputs". In: *The Journal of Neuroscience: The Official Journal of the Society for Neuroscience* 21.11, pp. 4050–4058. ISSN: 1529-2401. pmid: 11356892.

Turrigiano, G., G. LeMasson, and E. Marder (May 1995) "Selective Regulation of Current Densities Underlies Spontaneous Changes in the Activity of Cultured Neurons". In: *The Journal of Neuroscience:*

380   *The Official Journal of the Society for Neuroscience* 15 (5 Pt 1) pp. 3640–3652. ISSN: 0270-6474. pmid:
381   7538565.
382 Vitay, Julien, Helge Ülo Dinkelbach, and Fred H. Hamker (2015) "ANNarchy: A Code Generation
383   Approach to Neural Simulations on Parallel Hardware". In: *Frontiers in Neuroinformatics* 9. ISSN:
384   1662-5196. DOI: 10.3389/fninf.2015.00019. URL: https://www.frontiersin.
385   org/articles/10.3389/fninf.2015.00019/full (visited on 05/26/2018)
386 Xu, W., D. Xu, and L. Deng (July 2017) "Measurement of Source Code Readability Using Word Concrete-
387   ness and Memory Retention of Variable Names". In: *2017 IEEE 41st Annual Computer Software and*
388   *Applications Conference (COMPSAC)* 2017 IEEE 41st Annual Computer Software and Applications
389   Conference (COMPSAC) vol. 1, pp. 33–38. DOI: 10.1109/COMPSAC.2017.166.

FIGURE CAPTIONS

Figure 1: `xolotl` can quickly set up and simulate conductance-based models. (A) Cartoon of a Hodgkin-Huxley single-compartment neuron model with fast sodium, delayed rectifier, and leak currents. (B) Code snippet in MATLAB used to implement D, F-I. (C) `xolotl` schematic displayed in the MATLAB command prompt. (D) Simulated voltage trace of a Hodgkin-Huxley model with three conductances and 0.2 nA of injected current. Colors indicate the dominant current (gold is fast sodium, blue is delayed rectifier, red is leak). (E) Firing rate-input relation displaying firing rate as a function of injected current current. (F-G) Steady-state gating functions for activation (m) and inactivation (h) gating variables. (H-I) Voltage-dependence of time constants for activation (m) and inactivation (h) gating variables.

Figure 2: Simulating a voltage-clamp experiment. (A) Cartoon of a cell with delayed rectifier potassium conductance (Liu et al. 1998) with experimentally-fixed voltage. (B) Structure of `xolotl` object in A. (C) Code snippet depicting integration under voltage clamp. (D-E) Current response to steps in voltage from a holding potential of $V_m = -60$ mV. (F) Current-voltage relation of the steady-state current ($t = 400$ ms) indicating a reversal potential of $E = -80$ mV and no inactivation. (G) Conductance-voltage relation at steady-state takes the form of a sigmoid. (H) Sigmoids m fit to the model as m^n data indicating that $n = 4$ is the best fit. (I) Goodness of fit vs. exponent n, suggesting $n = 4$ as the best fit to the data.

Figure 3: Simulating a network of conductance-based model neurons. (A) Diagram of a network model of the pyloric rhythm in the crustacean stomatogastric ganglion (Prinz *et al.* 2004). (B) Each neuron is modeled as a single compartment with 7-8 intrinsic conductances and 1-3 post-synaptic conductances. (C) `xolotl` implements conductances as fields of compartments and synapses as connections between compartments. (D-F) Simulated voltage trace of a model network for the three compartments. (G) Time series of synaptic currents in the simulated network can be obtained from the integration.

Figure 4: Simulating neurons under homeostatic regulation. (A) Cartoon of a model neuron (Liu et al. 1998) with integral control (O'Leary et al. 2013) (B) Hierarchical structure of a neuronal network considers controllers as components of compartments which act on conductances. (C) `xolotl` implements controllers as properties of conductances and synapses. (D) Calcium sensors change maximal conductances to move a neuron from quiescence to a bursting state. (E) Voltage trace shows regular bursting activity after integral control.

Figure 5: Using the GUI to manipulate neuron parameters. (A) Real-time output of the `plot` function displaying voltage (colored) and intracellular calcium (black) traces of a bursting neuron model (Prinz, Billimoria, and Eve Marder 2003; Prinz, Bucher, and Eve Marder 2004) Colors indicate the dominant current. (B) Sliders control the maximal conductances, which updates on the figure.

Figure 6: `xolotl` benchmarked against `DynaSim` and `NEURON`. (A) Ratio of 'simulated' time to runtime (speed factor) and accuracy, measured by spike train coincidence plotted against decreasing time-resolution. (B) Speed factor for models at increasing simulation times. (C) Speed factor over number of compartments.