Journal of Bioinformatics and Computational Biology
Vol. 14, No. 3 (2016) 1642003 (16 pages)

DOI: 10.1142/S0219720016420038

World Scientific
www.worldscientific.com

ModFossa: A library for modeling ion channels using Python

Gareth B. Ferneyhough, Corey M. Thibealut, Sergiu M. Dascalu
and Frederick C. Harris Jr.*

Department of Computer Science and Engineering
University of Nevada Reno, NV 89557, USA
**Fred.Harris@cse.unr.edu*

Accepted 10 December 2015
Published 29 February 2016

The creation and simulation of ion channel models using continuous-time Markov processes is a powerful and well-used tool in the field of electrophysiology and ion channel research. While several software packages exist for the purpose of ion channel modeling, most are GUI based, and none are available as a Python library. In an attempt to provide an easy-to-use, yet powerful Markov model-based ion channel simulator, we have developed ModFossa, a Python library supporting easy model creation and stimulus definition, complete with a fast numerical solver, and attractive vector graphics plotting.

Keywords: Ion channel modeling; Markov models; Python.

1. Introduction

Ion channels are trans-membrane proteins that control the passive flow of ions into, and out of a biological cell. They exist in all cell types and play crucial roles in cellular sensing and communication, maintenance of the membrane potential, and regulation of cell volume.

For example, voltage-gated sodium, potassium, and calcium channels, among others, work together to shape the automatically recurring cardiac action potential which gives our hearts their sure, and usually steady, beat. Philosophy aside, the very thoughts passing through the consciousness of a scientist as he reads this page exist as nothing more than electrochemical signals in the brain. Specifically, voltage-gated sodium and potassium channels integrated in the cell membrane of all neurons allow electrical signals to propagate rapidly down the axon in the form of an action potential. When the action potential reaches the end of the axon, voltage-gated calcium channels regulate the release of special chemical messengers called neurotransmitters into the synapse, the structure that connects neurons to one another.

*Corresponding author.

As the neurotransmitters diffuse across the synapse, they bind to ligand-gated ion channels on the post-synaptic neuron. Depending on the type of ion channel and the type of neurotransmitter, the channels may open or close, thereby altering the membrane voltage. If the post-synaptic neuron receives enough excitatory stimulus from the many pre-synaptic neurons in which it is connected, its membrane potential will depolarize to a point and it too will experience an action potential, thus passing the signal along to other neurons.

As further evidence of their vital importance, ion channels are specifically targeted by several naturally occurring toxins and poisons, including venom from spiders, snakes, scorpions, and fish. It comes as no surprise that the development of new types of drugs which alter the behavior of certain channels is a major motivator behind ion channel research. As in most scientific fields, researchers often develop mathematical models to assist in their understanding of the results. Ion channels are no exception; several classes of ion channel models exist which allow for the study of ion channel behavior under varying conditions. One such generalization involves modeling channel gating using continuous-time Markov processes, and solving the underlying ordinary differential equations that arise. A number of commercial as well as open-source software programs exist for this purpose, but it is also common for scientists to implement their own solution using the differential equations solvers in Matlab, for example.

In order to provide an elegant and simple solution for the creation and simulation of Markov model-based ion channels, we present ModFossa, a simulator written in C++ with an easy-to-use Python interface. Several published models of ion channel gating have been reproduced using our simulator; these results, along with performance analysis are presented in the following. In Sec. 2, we present several mathematical, biological, and electrical concepts related to ion channel modeling. In Sec. 3, we present the design and implementation of ModFossa. Results are provided in Sec. 4. Finally, we conclude with a discussion of applications and future work in Sec. 5.

2. Background

2.1. *Biology background*

Salts, ions, and the cellular solutions: Ions, charged atoms, arise from a variety of natural processes, including the dissolution of salts in water. For example, sodium chloride (NaCl) dissolves in water, sodium cations with one positive charge (Na^+) and chloride anions with one negative charge (Cl^-) become dissociated and free to move around independently. Similarly, the salts potassium chloride (KCl) and calcium chloride ($CaCl_2$) dissolve into solutions of free K^+, Cl^-, and Ca^{2+} ions.

To see the importance of ions in the role of cellular function, imagine a bath containing two solutions separated by a membrane which is perfectly permeable to K^+. At time $t = 0$, a concentrated solution of KCl is introduced to the left side of the

bath, while a weaker solution of KCl is introduced to the right side. The instant after the solutions are added to the bath, K^+ will begin to move across the membrane from left side to the right side, driven by the diffusion force. As this movement of charge continues, the right side will become more positive. Eventually, the buildup of positive charge will deter K^+ ions from moving to the right side. At this point, the system is at equilibrium. The chemical gradient pushing K^+ to the right is balanced by the electrical force opposing the movement of charge. The voltage at which the bath reaches equilibrium is called the Nernst potential, and is calculated for a generic ion species S

$$E_S = \frac{RT}{zF} \ln\left(\frac{[S]_e}{[S]_i}\right), \tag{1}$$

where R is the gas constant, T is the absolute temperature in Kelvin, z is the charge on ion S, F is Faraday's constant, and $[S]_e$ and $[S]_i$ represent the extracellular and intracellular concentrations of ion S.

The combination of the diffusion and electrical forces is called the *electrochemical gradient*, and can be thought of as a form of potential energy for a cell.

Cell membrane: The cell membrane separates the intracellular and extracellular regions of a cell and is composed of a thin double layer of lipids about 7.5 nm thick.[10] An important characteristic of the cell membrane is that it is selectively permeable, which is facilitated through several mechanisms including ion channels — trans-membrane proteins which facilitate the passive diffusion of select ion species down their electrochemical gradient. In the open configuration, channels allow ions to pass through at speeds of 10^8 ions per second, near diffusion speed.[3] From the perspective of the whole cell, the large number ion channels together can be said to affect the *permeability* of the membrane to a certain ion species.

Membrane potential: Given various intracellular and extracellular ionic concentrations, as well as the membrane's relative permeability to these ions, one can calculate the resting membrane potential of a cell. First, some basic concepts in electricity need to be refreshed.

Ohm's law defines the relationship between voltage, current, and conductance

$$V = \frac{I}{G}. \tag{2}$$

Considering the simplification that the membrane is permeable only to ion species S, the membrane's conductance, G, is directly related to how easily S passes though its ion channels. This is called channel conductance, and is a function of the opening and closing, referred to as gating from now on, of the channels as well as their type. I_S is the current through the membrane due to the movement of species S. In this case, the voltage, V, is referred to as the *driving force*. The driving force is different for each different ion species in the cell — Na^+ ions, for example, experience a force greater in magnitude and opposite in direction compared with K^+ ions.

To summarize, the current from ion S is equal to the driving force of S times the membrane conductance to S. Formally,

$$I_S = G_S(\text{Driving force}_S). \tag{3}$$

Driving force is determined by the membrane voltage, V_m, and E_S, the equilibrium potential (calculated using Eq. (1)) for ion S

$$\text{Driving force}_S = (V_m - E_S). \tag{4}$$

Substituting Eq. (4) in Eq. (3)

$$I_S = G_S(V_m - E_S). \tag{5}$$

Equation (5) is used by the software described in Sec. 3 to calculate the ionic currents given a known membrane voltage and simulated channel conductances.

2.2. *Modeling ion channels*

The modeling of ion channels dates back over 100 years.[12] As is with many mathematical models, the motivations driving the development of ion channel models include: (i) striving to infer fundamental knowledge of the underlying physical processes, (ii) matching experimental data to the model's predictions, (iii) using the model to predict behaviors which are difficult to observe, and (iv) developing common terminology and knowledge among researchers.

The interpretation of ion channel models during the pre-molecular era provided the primary source of information about channel structure.[12] However, current technology such as molecular biology and X-ray diffusion are beginning to prove atomic-resolution detail of bacterial as well as some mammalian channels, though the fraction of channels with known structure remains low.[3,12]

Hodgkin and Huxley's landmark model describing the ionic mechanisms behind the action potential of a squid giant axon provided the foundation for many modern practices in electrophysiology and ion channel modeling.[8,10] Their model describes the voltage-dependent and time-dependent behaviors of Na^+ and K^+ conductances using a coupled set of ordinary differential equations. While the Hodgkin–Huxley model is still used today due to its simplicity and low number of parameters, it does exhibit several shortcomings. These include, for example, a lack of connectivity between activating and inactivating gates in the Na^+ channel, as well as the premise that the inactivation gate can only close after the activation gate opens.[7]

Markov models: Continuous-time Markov processes are commonly used to simulate stochasticity in a variety of fields including Biology, Chemistry, Physics, and Economics. As the name implies, a continuous-time Markov process is similar to a discrete time Markov chain, with the exception that transitions between states can occur at any time, with an exponentially distributed probability. Both discrete and continuous-time Markov processes satisfy the Markov property which says that the probability distribution of future states depends only on the current state, and not on the preceding events. This "memoryless" behavior is what gives the Markov

processes their stochastic behavior. Continuous-time Markov chains are easily described using weighted directed graphs, where vertices represent the states, and the edges are labeled with the *rates* of transitioning between states.

Markov models for ion channels are an extension of the Hodgkin–Huxley formalism, and are good for modeling single channel data, gating currents, and drug interactions. Unlike Hodgkin–Huxley models, Markov models show the state dependence of activation and inactivation.[6]

Theory: Ion channels can be modeled as continuous-time Markov processes, where a simplification of the channels functional physical shape is represented by states in the Markov chain. A model can have several open, closed, deactivated, and inactivated states, as well as states which represent the binding of ligands. The rates connecting the various states are time-independent kinetic rates which can be constant, or dependent on the membrane voltage, or intra and extra-cellular ionic concentrations.

Criticism: Markov models are not without critics. In a one page perspective titled *Are rate constants constant?*, Jones challenges the community to examine the benefits of time-dependent rate constants over Markov models.[9] Jones cites the findings by Uebachs[17] who reports that the rate of recovery from inactivation in a type of Ca^{+2} channel depends on the length of depolarization, which goes against the basic principle of chemical kinetics that rate constants remain constant in a constant environment.[9]

Although Markov models are memoryless by definition, the time-dependent behavior described by Uebachs can be reproduced by having a long chain of open or closed states.[9,13] Nevertheless, Jones concludes that time-dependent rate constants as used by fractal channel models, would provide a more explicit memory than multistate Markov models.

Fractal Models: Liebovitch has been the main proponent of fractal ion channel models for over 15 years.[13–15] One of his main arguments questioning the use of Markov models is found in the body of evidence depicting ion channel proteins as complex structures with a continuum of states, exhibiting large and small motions over varying time scales. Fractal models as proposed by Liebovitch contain a large number of conformational states with kinetic rates connecting them. The rates, however, are linked and not independent as in Markov models. This has the effect of giving the channel *memory*.[13]

An additional concern raised by Liebovitch is the use of exponential rate constants. He provides a friendly reminder to his colleagues that the fitting of experimental data to sums of exponentials is famously ill-conditioned.[1,13] As confirmation, he provides a quote from a numerical methods text which states that those who try to determine the parameters of such equations from experimental data "must be spanked or counseled. At the very least, keep them from obstructing Progress and the computer!"[1] In their defense, other researchers have argued that Markov models with exponential rate constants and a small number of states fit the experimental data better than fractal models.[16] Nevertheless, fractal models continue as the number one challenger to Markov models.

3. ModFossa

3.1. *Overview*

In this section, we present ModFossa, a fast and easy-to-use Python library for creating and simulating ion channel kinetics using continuous-time Markov processes. A brief discussion of ModFossa's features, components, are provided here. For more details regarding the design of ModFossa, the reader is referred to Ref. 4 and for the source code, the reader is referred to Ref. 5.

For those interested, the name ModFossa was formed from two Latin words — mod, and fossa, which can be interpreted as *open channel* when written together. Before we begin the design overview, Table 1 compares the basic features of several simulators, including our solution, ModFossa.

It is worth noting that ModFossa is unique in that it was designed to be a Python library. All of the other major simulators are GUI based and only one of them QUB (in its newest version) now has the ability to be used in a script. ModFossa was designed as a Python library so that it can be scripted or used in other programs and large parameter spaces could be explored. Also the graphing capabilities of ModFossa were designed around vector graphics so that it could produce much better graphs than the bitmap graphics produced by the GUI based simulators.

3.2. *Features*

Python interface: ModFossa provides easy, yet versatile and powerful ion channel modeling through its novel Python interface. Python is well-used in the scientific and academic communities; however, we are unaware of any Python libraries which facilitate the creation and simulation of ion channel kinetics. The popularity of Python as a research tool stems from its simple syntax and extensive standard library, and the availability of many high-quality open-source numeric and scientific libraries. Additional qualities of Python include easy third party library installation; strong, dynamic typing; and an interactive interpreter. ModFossa can also be used directly as a C++ library, allowing for more flexibility.

Easy model creation: ModFossa allows the user to define an ion channel's Markov model using states, rates, and connections. States and rates are referred to

Table 1. Feature comparison of several ion channel simulators which use Markov models.

Name	Model	GUI	Script	Plotting, format	Simulation type	OS	License
IonChannelLab	Markov	Yes	No	Interactive bitmap	Numerical integration, Monte Carlo, Q-Matrix	Win.	Free, no source available
ChannelLab	Markov	Yes	No	Yes bitmap	Numerical integration, Monte Carlo	Win.	Proprietary
QUB	HMM	Yes	Yes	Interactive bitmap	Numerical integration	Win., Mac, Linux	GPL
ModFossa	Markov	No	Yes, Python	Yes, vector	Numerical integration	Linux	GPL

by their user-supplied names. This allows the user to create the model in any order. For example, connections can be defined before the definition of the states and rate constants.

States represent the various channel states as defined by a continuous-time Markov chain. All states are referred to by unique name which is assigned by the user during creation. States are created as either non-conducting or conducting. Additionally, the gating of conducting states can by specified during construction. The gating parameter is used to simulate states that are partially conducting. For example, assigning a gating value of 0.5 to a state in a particular model may be used to simulate channel blocking by 50%.

Rates represent the kinetic rate constants which define the rate of transition between two states. Rates are referred to by unique name. ModFossa includes the following rate constant types: constant, Boltzmann voltage dependence, exponential voltage dependence, and ligand-gated. The equations for each rate constant are presented below.

Constant rate constant are defined by a single parameter, k,

$$\text{rate} = k. \tag{6}$$

Exponential voltage-dependent rate constants depend exponentially on the membrane voltage

$$\text{rate}(V) = a * \exp(k * V), \tag{7}$$

where V is the membrane voltage, exp is the exponential function, and a and k are parameters.

Boltzmann voltage dependence is defined using the sigmoid-like Boltzmann equation

$$\text{rate}(V) = \frac{a}{1 + \exp[(V - V_{0.5})/k]}, \tag{8}$$

where V is the membrane voltage, $V_{0.5}$ is the half-maximal activation voltage (given as a parameter), exp is the exponential function, and a and k are parameters (Taken from Angermann *et al.*[2]).

Ligand-gated rate constants depend on the concentration of a particular ionic species, S,

$$\text{rate}([S]) = k * [S]^n, \tag{9}$$

where $[S]$ is the concentration of ligand S, n is the ligand power, and k is a parameter.

Connections define a transition from one unique state to another using a specified rate constant. States can have multiple ingoing and outgoing connections. Rate constants may also be used in multiple connections.

Experiment definition: An experiment consists of a channel Markov model, a voltage protocol, and an optional concentration protocol. Voltage protocols are a fundamental technique in electrophysiology used to measure the current response of ion channels. In the laboratory, a small electrode is inserted into the cell, through the

cell membrane. The electrode, and therefore the cell membrane, is held constant at a desired voltage for a length of time, as defined by the voltage protocol.

A concentration protocol defines the concentration values, extracellular or intracellular, of a certain ligand. Like a voltage protocol, ModFossa's concentration protocols are defined using *hold stages*. However, there is no support for stepped concentration protocol stages. During simulation, the simulator will run the model through the entire voltage protocol using a single stage from the concentration protocol. Next, the concentration protocol will advance to the next stage, and the voltage protocol will be run using the new concentration value. This continues until there are no stages left in the concentration protocol.

Data analysis and plotting: ModFossa supports the creation of a number of useful plots using the Python plotting library, PyPlot. Unlike other simulators reviewed in Sec. 2, the plots can be saved easily in a vector graphics format. Visual examples of each plot type are found in Sec. 4. To assist the user in creating a valid simulation, ModFossa provides validation methods which return detailed messages regarding any problems with the model and experiment definitions.

3.3. *Implementation*

Sundials (SUite of Nonlinear and DIfferential/ALgebraic equation Solvers) is a C library developed by Lawrence Livermore National Laboratory. ModFossa uses Sundials to solve the system of ordinary differential equations governing the ion channel state probabilities. Armadillo, a C++ linear algebra library provides easy-to-use matrix structures and linear solvers, and is also used by ModFossa.

4. Results

Several example models of varying complexity have been created and simulated using ModFossa. A description of the each model is provided, along with a listing of the Python source code used define the model. Visual results are provided in the form of plots generated by the code.

4.1. *Simple model*

A simple three state model is presented in order to give the reader an introduction to creating Markov models using ModFossa's Python interface. This models has little biological relevance; for results from a published calcium-gated chloride ion channel model, see Sec. 4.2. This section follows a tutorial format, with explanations provided alongside a sequence of short source code snippets.

Let us create a simple model that will show some features we can use to define real ion channel models, such as voltage steps, gating variables, and voltage and ligand-gated rate constants. For this model, we need three states: C_1 is a closed state, while O_1 is a partially conducting state with a gating variable of 0.5, and O_2 is a fully conducting, open state. The rate constants for the model are provided in Table 2.

Table 2. Rate constants used in the three state model example.

Name	Type	Parameters	Units
k_1	Ligand-gated	$k = 10e^8$, ligand = Ca^{2+}, power = 1	$M^{-1}s^{-1}$
k_2	Constant	$k = 1$	s^{-1}
k_3	Sigmoidal voltage-gated	$k = 1$, a=100, $V_{0.5} = 50$	s^{-1}
k_4	Constant	$k = 4$	s^{-1}

Let us examine the code for the three state model. First, we define our states and their gating variables. The closed state does not conduct, so we do not have to specify its gating variable, it is 0 by default. State O_1 is a partially conducting state, and O_2 is fully conducting.

```
from modFossa import *
state ('C')
state ('O1', gating = 0.5)
state ('O2', conducting=True)
```

Next, connect the states.

```
connect (from_state = 'C', to_state = 'O1', rate = 'k1')
connect (from_state = 'O1', to_state = 'O2', rate = 'k2')
connect (from_state = 'O1', to_state = 'C', rate = 'k3')
connect (from_state = 'O2', to_state = 'O1', rate = 'k4')
```

Now we define our rate constants. We have a ligand-gated rate constant which depends on the concentration of calcium. This could be any ligand that we chose, but we must define its concentration in the experiment sweep, or ModFossa will give us an error.

```
rate ('k1', type = 'ligandGated', k = 10e8, ligand = 'Ca', power = 1)
rate ('k2', type = 'constant', k = 1)
rate ('k3', type = 'sigmoidal', k = 1, a = 100, v_half = 50)
rate ('k4', type = 'constant', k = 4)
```

Now, the remainder of the model definition.

```
initialState('O2')
startAtSteadyState(False)
maxChannelConductance(1.16)
reversalPotential(0)
membraneCapacitance(100)
```

Next, define the voltage protocol and experiment sweep. The voltage protocol holds the voltage at −50 mV for 500 ms, then depolarizes the membrane by jumping to 50 mV and holding for another half of a second. The concentration of calcium is set to 500 nM in the experiment sweep. This is necessary because k_1 is a ligand-gated rate dependent on [Ca].

```
voltageProtocol('vp')
voltageProtocolAddStage('vp', 'hold1', voltage=-50, duration=500)
voltageProtocolAddStage('vp', 'hold2', voltage=50, duration=500)
experimentSweep('three_state', 'vp', Ca=500e-9)
```

Finally, validate and run the experiment and plot the results.

```
validate()
run()
states=plotStates ('three_state')
```

Figure 1 shows the results from the three state model. Once again, we did not start in steady state, so we see the state probabilities moving toward steady state under the experiment sweep conditions of a −50 mV membrane voltage, and 500 nM [Ca]. At 500 mS, the voltage protocol stage 'hold2' is activated, and the voltage is increased immediately to positive 100 mV. The effect of this jump in voltage is evident in Fig, 1. The jump in the probability of state O_1 is due to a decrease in the value of the sigmoidal rate constant, k_3.

4.2. *Angermann CaCl currents model*

Our final example presents an implementation of the calcium-activated chloride channel model developed by Angermann *et al.*[2] A brief description of the study is provided, along with the model's description and implementation details. Later, the Python source code used to implement the model in ModFossa is listed, along with several figures and accompanying discussion.

Study details: In Ref. 2, Angermann *et al.* attempt to determine how phosphatase activity influences calcium-activated chloride channels in rabbit pulmonary

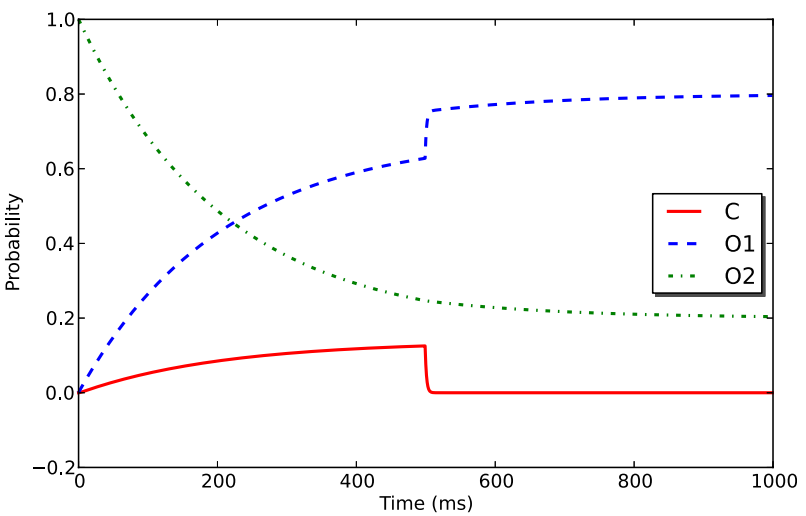

Fig. 1. A plot showing the evolution of the three state model state probabilities as generated by ModFossa.

artery myocytes (smooth muscle cells). Whole cell patch clamp experiments were conducted to measure the calcium-activated chloride current (I_{ClCa}) under intracellular [Ca^{2+}] ranging from 20 mM to 1000 mM. The control group contained 3 mM ATP in the pipette solution, while the test group used 3 mM adenosine $5'-(\beta,\gamma-imido)$-triphosphate (AMP-PNP).

Under both groups, the maximum I_{ClCa} is greater at higher intracellular Ca^{2+} concentrations, as is expected for such channels. However, the currents in test group (containing AMP-PNP) exhibited behaviors indicative of a negative shift in voltage dependent activation. The maximal conductance of I_{ClCa} was more than three-fold larger in the test group. Additionally, the activation times were less and the deactivation time greater in the test group compared to the control. These observations led the authors to the conclusion that the increased I_{ClCa} activity in the test group (pipette containing AMP-PNP) is not explained by Ca^{2+} channel sensitivity, nor the number of Ca^{2+} activating the channel, but instead by a negative shift in the voltage-dependent activation.

Model description: To simulate the behavior of the test and control groups, Angermann *et al.* created a Markov chain kinetic model based on the work by Kuruma and Hartzell.[11] Slightly different parameters were chosen for the test and control groups in an attempt to make clear the difference in behavior between the two. The model, shown in Fig. 2, consists of four closed states and three open states. Calcium binding occurs in the closed states with rates directly proportional to [Ca]. State C_1 is representative of the channel's state when no Ca^{2+} ions are bound to the receptor sites, while state C_4 represents the channel with all three Ca^{2+} binding sites occupied. The closed states with at least one binding site occupied can transition into open states with constant channel opening rates. Closed states with more occupied binding sites have higher values for the channel opening rates, giving the model its concentration-dependent behavior. Voltage-dependent channel closing rates take the model from open to closed states.

In Ref. 2, the behavior of the channel in the presence of AMP-PNP was simulated by setting the gating variables of all three of the channel's open states to 1. Conversely, to simulate the channel in the presence of ATP, only channel O_1 was assigned a gating variable of 1, while the rest of the open states were given a value of

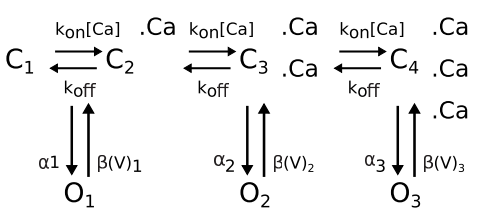

Fig. 2. Angermann CaCl Markov model. The model has four closed states and three open states. State C_2 represents the channel with a single Ca^{2+} binding site occupied, while state C_4 represents the channel with all three binding sites occupied. Note that the model can only transition to an open state in the presence of at least one bound Ca^{2+}.

zero. This reproduces the blocking effect by phosphorylation that the authors hypothesize as a possible explanation for the inhibition of I_{ClCa} while the channel is subjected to ATP. In addition to reducing the channel gating variables to model the effect of ATP, the authors also increased the magnitude of the channel closing rates and shifted the voltage dependence toward more positive potentials.

By altering the Markov model's parameters in an insightful manner, Angermann *et al.* were able to simulate the behavior of Ca^{2+}-activated Cl^- currents in the presence of both AMP-PNP and ATP.

Model implementation in ModFossa: The calcium activated chloride channel Markov model shown in Fig. 2 was implemented in ModFossa using parameter values chosen to match the behavior of I_{ClCa} in the presence of ATP, and were modified slightly by Thibeault from Angermann *et al.*[2]

ModFossa output: ModFossa can generate a number of I–V curves at a given time, and place them on a single plot, as shown in Fig. 3. This figure shows the *late* I–V curves, measured at 1099 ms into the simulation, which is the moment of maximum channel current.

To show the effects of both ligand concentration and voltage on the channel conductance, ModFossa's *plotGversusConcentration* method can be used to generate a chord conductance plot (not shown). This plot contains 13 traces of conductance as a function of the internal calcium concentration. Each trace corresponds to a different step value in the voltage protocol.

A second chord conductance plot was generated using ModFossa (Fig. 4). This plot shows channel conductance as a function of voltage, for six different concentration values. The simulated data (Fig. 4) match the experimental data well.

5. Discussion

5.1. *Summary*

We have identified and addressed a gap in ion channel simulation software by developing ModFossa, a Python library dedicated to the construction and simulation of ion channels based on continuous-time Markov processes. While several recent software tools have been released for this purpose, none are Python libraries. Python is a popular tool among research scientists, therefore we hypothesize that a versatile and easy-to-use ion channel simulation library has the potential for providing a valuable service to those developing and testing Markov models of ion channels. In summary, ModFossa allows for the creation of channel models using simple Python syntax. Several rate constant types are supported, including exponential and Boltzmann voltage-dependent rates, as well as ligand-gated rates. Voltage and current protocols are defined easily. Some common plots including I–V curves, state probabilities, conductance versus voltage, and conductance versus ligand concentration are generated using a single function call. Finally, ModFossa's core is implemented in C++ using the Sundials solver, providing the benefit of fast execution times. When comparing it with similar Matlab code, ModFossa was 17 times faster.

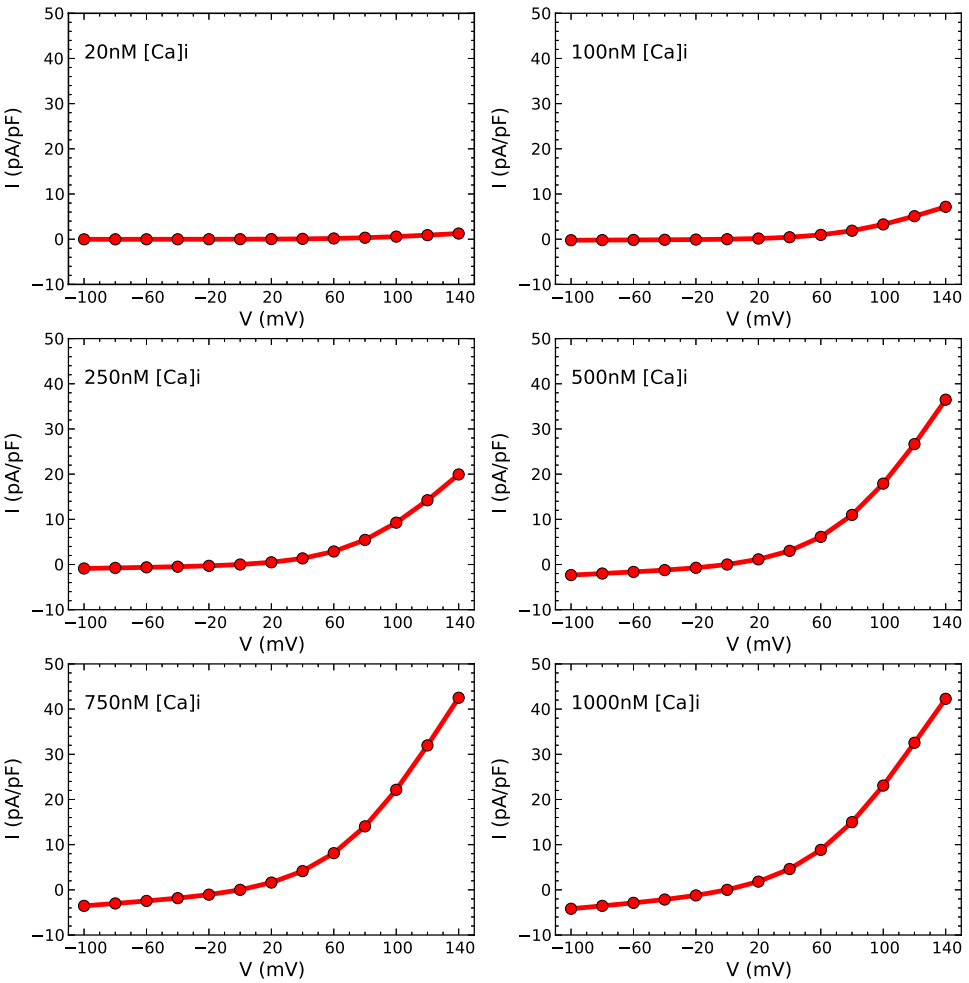

Fig. 3. Angermann CaCl late *I–V* curves given various internal Ca^{+2} concentrations. This figure was generated using ModFossa's *plotIV* method at time 1099 ms.

5.2. *Applications and Speed*

ModFossa can provide rapid model development and testing for researchers with varying programming skill. A common task in ion channel model development is parameter searching, which requires the modeler to find the rate constant parameters which best fit the experimental data. Because ModFossa is usable from Python, existing machine learning libraries such as scikit-learn can be leveraged to perform parameter searches automatically. This situation is where ModFossa's fast execution speed provides a great benefit, as it is not uncommon for modelers to leave parameter searches running for several days, or weeks. When comparing ModFossa with similar Matlab coe, we found a 17× speedup in our experiments.

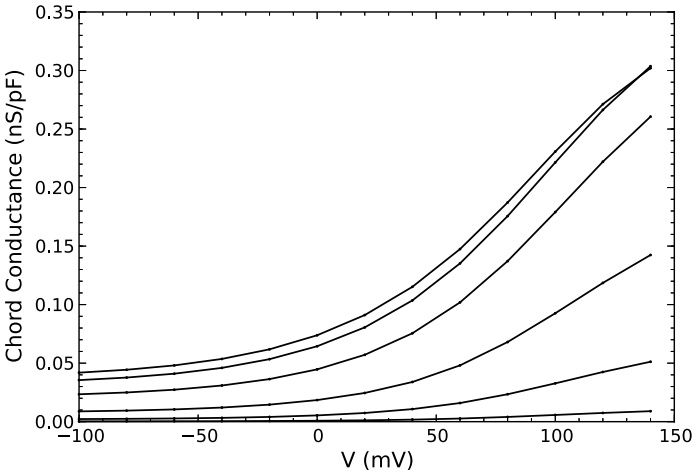

Fig. 4. Simulated Angermann CaCl conductance at varying concentrations as a function of membrane voltage. This figure was generated using ModFossa's *plotGversusV* method.

5.3. *Future work*

Several ideas for additional development of ModFossa are presented here. First, some minor enhancements, such as the addition of more sample models, improved customization of plotting functions (plot size, legend, colors, data range, etc.), and standardization of the data structures used in the results module would add value to the software by ensuring a smooth experience for new users. Additionally, creating and distributing a Debian package for ModFossa would ease installation significantly.

Several preliminary users have expressed interest in user-defined rate constant equations. A custom rate equation could be written by the user in Python using either a decorator function, or by inheriting from a base class. During simulation, the simulator would call the user's custom rate equation, passing in the current state of the simulation (membrane voltage, concentrations, time, etc.), and use the returned value as the rate for whichever transitions the user specified. The performance hit due to calling a Python function from C++ during the simulation would have to be quantified, but it is not expected to be significant.

Lastly, the idea of offering ModFossa as a web service with a rich graphical user interface has been explored. Doing so would encourage use from a wider audience, particularly from those with limited or no programming experience.

Acknowledgments

This paper is based in part upon work supported by The National Science Foundation under Grant No. IIA-1329469. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

1. Acton FS, *Numerical Methods That (Usually) Work*, Harper and Row, New York, 1970.
2. Angermann JE, Sanguinetti AR, Kenyon JL, Leblanc N, Greenwood IA, Mechanism of the inhibition of Ca^{2+}-activated Cl^- currents by phosphorylation in pulmonary arterial smooth muscle cells, *J Gen Physiol* **128**(1):73–87, 2006.
3. Capener CE, Kim HJ, Arinaminpathy Y, Sansom MS, Ion channels: Structural bioinformatics and modelling, *Human Mol Genet* **11**(20):2425–2433, 2002.
4. Ferneyhough GB, A Python library for ion channel modeling, Master's Thesis, Department of Computer Science and Engineering, University of Nevada, 2013.
5. Ferneyhough GB, A Python library for ion channel modeling, https://github.com/gareth-ferneyhough/modfossa, 2015.
6. Fink M, Noble D, Markov models for ion channels: Versatility versus identifiability and speed, *Philos Trans R Soc A, Math Phys Eng Sci* **367**(1896):2161–2179, 2009.
7. Gurkiewicz M, Korngreen A, A numerical approach to ion channel modelling using whole-cell voltage-clamp recordings and a genetic algorithm, *PLoS Comput Biol* **3**(8):e169, 2007.
8. Hodgkin AL, Huxley AF, A quantitative description of membrane current and its application to conduction and excitation in nerve, *J Physiol* **117**(4):500–544, 1952.
9. Jones SW, Are rate constants constant?, *J Physiology* **571**(3):502, 2006.
10. Keener JP, Sneyd J, *Math Physiol*, Springer, New York, 2009.
11. Kuruma A, Hartzell HC, Bimodal control of a Ca^{2+}-activated Cl^- channel by different Ca^{2+} signals, *J Gen Physiol* **115**(1):59–80, 2000.
12. Levitt DG, Modeling of ion channels, *J Gen Physiol* **113**(6):789–794, 1999.
13. Liebovitch L, Testing fractal and markov models of ion channel kinetics, *Biophys J* **55**(2):373–377, 1989.
14. Liebovitch LS, Scheurle D, Rusek M, Zochowski M, Fractal methods to analyze ion channel kinetics, *Methods* **24**(4):359–375, 2001, doi: 10.1006/meth.2001.1206, http://www.sciencedirect.com/science/article/pii/S104620230191206X.
15. Lowen SB, Liebovitch LS, White JA, Fractal ion-channel behavior generates fractal firing patterns in neuronal models, *Phys Rev E* **59**(5):5970–5980, 1999.
16. McManus O, Weiss D, Spivak C, Blatz A, Magleby K, Fractal models are inadequate for the kinetics of four different ion channels, *Biophys J* **54**(5):859–870, 1988.
17. Uebachs M, Schaub C, Perez-Reyes E, Beck H, T-type Ca^{2+} channels encode prior neuronal activity as modulated recovery rates, *J Physiol* **571**(3):519–536, 2006.

Gareth B. Ferneyhough received a B.S. and an M.S. in Computer Science and Engineering from the University of Nevada, Reno in 2010 and 2013, respectively. His research is in the area of Computational Neuroscience. He is currently working as a Software Engineer in a private sector.

Corey M. Thibeault received a B.S. degree in Computer Engineering from the Rochester Institute of Technology, Rochester, NY, USA, in 2004, the B.S. and M.S. degrees in Mechanical Engineering from Minnesota State University, Mankato, MN, USA, in 2006 and 2009, respectively, and the M.S. degree in Computer Engineering and the Ph.D. degree in Biomedical Engineering from the University of Nevada, Reno, NV, in 2012. He is currently a Senior Data Scientist at Neural Analytics, Los Angeles, CA, USA.

Sergiu M. Dascalu is a Professor in the Department of Computer Science and Engineering at the University of Nevada, Reno, USA, which he joined in 2002. In 1982, he received a Master's degree in Automatic Control and Computers from the Polytechnic University of Bucharest, Romania and in 2001, a Ph.D. in Computer Science from Dalhousie University, Halifax, NS, Canada. His main research interests are in the areas of software engineering and human–computer interaction. He has published over 140 peer-reviewed papers and has been involved in numerous projects funded by industrial companies as well as federal agencies such as NSF, NASA, and ONR.

Frederick C. Harris Jr. is a Professor in the Department of Computer Science and Engineering and the Director of the High Performance Computation and Visualization Lab and the Brain Computation Lab at the University of Nevada, Reno, USA. He received his B.S. and M.S. degrees in Mathematics and Educational Administration from Bob Jones University in 1986 and 1988, respectively, and his M.S. and Ph.D. degrees in Computer Science from Clemson University in 1991 and 1994, respectively. He is a Senior Member of ACM and ISCA. His research interests are in parallel computation, computational neuroscience, computer graphics, and virtual reality.