Optimisation of stimulation patterns for specific questions in electrophysiology experiments : a Python framework

Damien Drix, Thomas Nowotny

June 8, 2011

In order to investigate the properties and function of electrically active membranes, in particular neurons, scientists use electro-physiological recordings in two established experimental modes : voltage-clamp and current-clamp. In both cases, a variety of stimulation patterns is used to elicit salient responses. The stimulation patterns are typically generated via a small number of established methods. The most common voltage clamp protocols use a combination of square steps and pre-steps to characterise the ionic conductances in a membrane ; in other cases sinusoids and noise sources have been used. The choice of a particular stimulus depends on the aim of the experiment ; for instance, one can select the amplitude and duration of voltage steps so as to highlight the effect of a potassium conductance, or apply high-frequency sinusoidal current injections to determine the membrane capacitance. Similarly, white noise has been used in model fitting tasks.

Here we propose a set of generic methods for systematically generating stimuli tailored to specific tasks, implemented within a simulated electrophysiology framework in Python. Our approach involves the use of splines to represent stimulation patterns, and genetic algorithms to evolve patterns that are particularly suitable for an arbitrary given task. Splines permit defining a curve through a number of given support points. By manipulating the positions of the support points and the type of spline interpolation, steps and sinusoids can be approximated, as well as other arbitrary shapes. This versatile encoding allows the optimisation algorithm to exploit any pattern that yields the desired result.

We developed a Python framework called `evopy` to handle the evolutionary search. This framework is able to leverage Enthought's Traits attributes to generate a real-valued or binary genetic encoding for arbitrary objects. Thus, the allowed parameter ranges and their inclusion or exclusion in the search can be easily controlled via metadata, with any change in the object model seamlessly reflected in the genetic encoding, encouraging experimentation. The optimisation itself is performed with a multi-objective genetic algorithm. Particle Swarm Optimisation and an interface to `scipy.optimize` routines are also available.

Since speed is a concern with meta-heuristic optimisation methods, we use a mixture of Python code to manage the experiments and analysis together with C and Cython components for critical sections. We define our Hodgkin-Huxley neuron models and simulated electrophysiology components as Cython classes. Similarly, ODE solvers from the GNU GSL library and spline interpolation code from the ALGLIB library are used through Cython language bindings. Thus, the entire simulation runs in native code, avoiding costly Python calls during integration.

We demonstrate the use of this framework to generate stimuli optimised to isolate the contributions of various Hodgkin-Huxley parameters to the dynamics of a model neuron in voltage- and current-clamp. Future directions using active stimuli in a dynamic clamp setup are also presented.