Genetic Algorithm for Optimization and Specification of a Neuron Model

W. C. Gerken[1,2], L. K. Purvis[1,2], R. J. Butera[1,2,3]
[1]Laboratory for Neuroengineering, Georgia Institute of Technology, Atlanta, GA
[2]School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA
[3]Department of Biomedical Engineering, Georgia Institute of Technology, Atlanta, GA

Abstract- **We present a novel approach for neuron model specification using a Genetic Algorithm (GA) to develop simple firing neuron models consisting of a single compartment with one inward and one outward current. The GA not only chooses the model parameters, but also chooses the formulation of the ionic currents (i.e. single-variable, two-variable, instantaneous, or leak). The fitness function of the GA compares the frequency output of the GA generated models to an I-F curve of a nominal Morris-Lecar (ML) model. Initially, several different classes of models compete among the population. Eventually, the GA converges to a population containing only ML-type firing models with an instantaneous inward and single-variable outward current. Simulations where ML-type models are restricted from the population are also investigated. This GA approach allows the exploration of a universe of feasible model classes that is less constrained by model formulation assumptions than traditional parameter estimation approaches. While we use a simple model, this technique is scalable to much larger and more complex formulations.**
***Keywords* – Genetic Algorithm, neuron model**

I. INTRODUCTION

Genetic Algorithms (GAs) are a class of popular, biologically inspired optimization method. The algorithm involves starting with an initial population of models with randomly chosen parameters, and letting the population evolve using mechanisms inspired by the idea of natural selection. Each model in the population is tested for fitness and assigned a score. This fitness score is the basis for that model's likelihood of reproducing (through crossover) and retention for subsequent generations. The desire is that with each generation, characteristics (parameter values and model specification) of the best performing models will combine to produce better models. Researchers have previously used GAs for parameter optimization of neurons models [1]. In this study, we use GAs not only for parameter optimization, but also for model specification. That is, the GA also chooses the type of currents that the model will use to fire action potentials.

II. METHODS

A. Model Selection

To implement use of a genetic algorithm to identify both model parameters and specification, the chromosome must contain information about parameter values as well as model structure. Model parameters are encoded by chromosomes, a concatenation of binary strings of given length, and decoded into real numbers in a specified interval using Gray decoding. The model specification uses two two-bit sections to select the formulation of the inward and outward currents. Conceptually, one can think of each individual as choosing one inward and one outward current from each set of four possibilities. The four current types are listed below.

Type 1.
Two state variable current

$$I_x = \bar{g} x_1 x_2 (V - E_x)$$

Type 2.
Single state variable current

$$I_x = \bar{g} x (V - E_x)$$

Type 3.
Instantaneous variable current

$$I_x = \bar{g} x_\infty (V)(V - E_x)$$

Type 4.
Leak current

$$I_x = \bar{g} (V - E_x)$$

The membrane potential is found using the differential equation

$$\frac{dV}{dt} = \frac{1}{C_m}\left(-I_{inward} - I_{outward} - I_{leak} + I_{app}\right)$$

where V is the membrane potential, C_m is the whole cell capacitance, t is time, and I_{app} is the applied stimulus current. The range of values for each parameter in the model is given in Table 1.

Parameter	Min. Value	Max. Value	Precision (bits)
Channel Type	1	4	2
Maximum Conductance	0 nS	10 nS	10
Leak Reversal Potential	-80 mV	-40 mV	10
Half Activation	-77 mV	50 mV	10
Slope factor	+/- 3	+/- 20 mV	10 + 1(sign bit)
Applied Current	2	20	1.0 increments
Inward Reversal Potential	+50 mV	-	-
Outward Reversal Potential	-77 mV	-	-
Membrane Capacitance	1 nF	-	-

Table 1- Parameter ranges and constant values.

B. Morris-Lecar model

The Morris-Lecar (ML) model is a relatively simple Hodgkin-Huxley style model originally developed for modeling barnacle muscle fiber action potentials [2]. This

model has two voltage dependent conductances, a fast (instantaneous) inward Ca^{2+} and a (single-variable) slow outward K^{+}, capable of producing neural-like oscillations.

Firing rate encoding uses the frequency of action potentials in response to a stimulus as the measure of neuron information. By sweeping the applied stimulus current through a range of values, the generated current-frequency (I-F) curve can be used to describe the neuron's oscillatory activity.

C. Fitness Function

An ideal fitness function succinctly quantifies the optimality of the solution. The function must strongly correlate with the goal of the algorithm. Since we are interested in finding models that produce non-linear dynamic behavior, we can use the I-F curves to quantify the spiking rate as a function of input. This approach closely mirrors experimental methodology. Using a 'goal' I-F curve generated from a nominal ML model, we compute the fitness function using a normalized sum of the squares error (SSE) measure at a fixed set of applied current levels.

$$SSE = \frac{\sum (F_{goal,i} - F_{sim,i})^2}{n}$$

Non-spiking simulations are given an extremely poor (high) fitness score. This measure captures the dynamics of the system with minimum computation. The SSE measure heavily weights outliers which can help to capture the nature of bifurcation as outliers will occur around the point where the model starts to spike. Additionally, this measure converges to zero as the models behavior more closely approximates the goal of the algorithm. This convergence is important, as the genetic algorithm needs a measure that can convey the relative success of achieving the algorithm's goal.

D. Genetic Algorithm Parameters

Each individual is defined by a 168-bit binary chromosome. The parameters are determined by a fixed segment of the chromosome. The simulation consists of 100 generations. The initial generation starts with 1000 randomly created models. This population size should be adequate given our chromosome size [3]. Each model is simulated and scored as described previously. The ranking function ranks individuals represented by their score, to be minimized, and returns the corresponding individual fitnesses. Using the results of the ranking, the best 200 models are selected to remain in the population. Using stochastic universal sampling to select 'parents', the next generation is created using crossover and mutation. The new population then consists of the previous 200 best models and 800 children models. The toolbox default was used for crossover and mutation rates of 70% and 0.4%, respectively.

E. Simulations

MATLAB (Mathworks, VA) along with "The MATLAB Genetic Algorithm Toolbox" [4] was used for running the GA. Model simulations were performed using the interactive differential equation simulation package XPP [5]. Each generation was run in parallel on a Beowulf computing cluster consisting of 54 2.4GHz (or higher) PCs running Redhat Linux 9.0.

III. RESULTS

Fig. 1 represents the output of the firing neuron models produced by the GA after 100 generations. The desired I-F curve generated by the ML model is represented by the black curve, and all other color lines are members of the population that were able to produce spikes at generation 100. Not all 1,000 members of the final population are able to fire action potentials. For the simulation results shown in Fig. 1, there are about 350 firing neuron models. Many of these models produce I-F curves that closely resemble the ML model. The class of firing neuron models present at generation 100 is only the ML-type models, that is, an instantaneous inward and single-variable outward current.

It is interesting to observe the classes of models that compete during the first 100 generations. Fig. 2 illustrates this competition between the three classes of models that were able to produce firing neuron models. The three classes of firing neuron models are a model with: 1) a single-variable inward and an instantaneous outward current (blue line), 2) an instantaneous inward and a two-variable outward current (red line), and 3) an instantaneous inward and single-variable outward current (black line). By generation 100, it is the 3rd class of neuron models that triumphs. This is the ML-type model (instantaneous inward and single-variable outward current). If the GA is allowed to proceed until generation 500, the ML-type model continues to be the only class of firing model in the population.

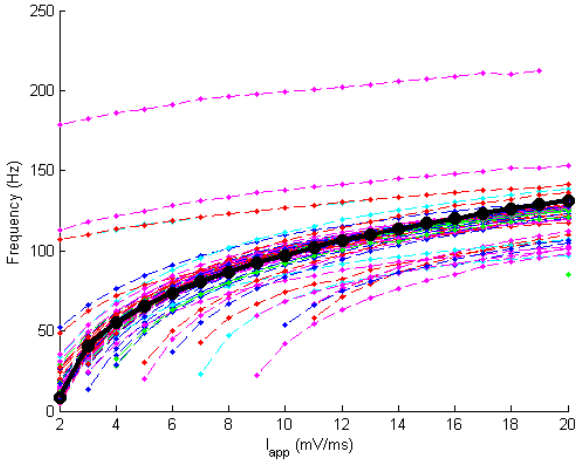

Fig. 1. I-F curve of the ML model (black line) and all firing neuron models produced by the GA (color lines) at generation 100.

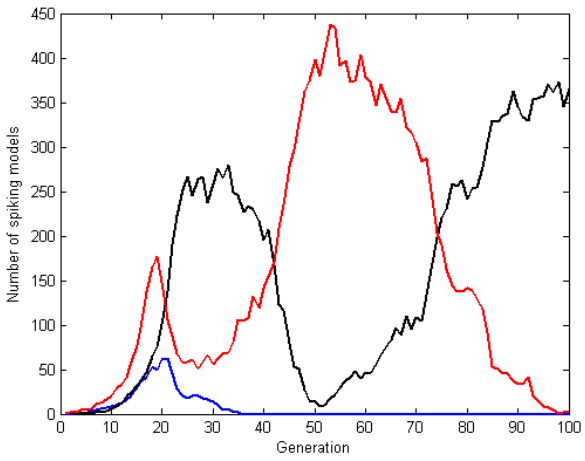

Fig. 2. Demographics graph showing the three classes of firing neuron models generated by the GA. The three classes of models are a single-variable inward and an instantaneous outward current (blue line), an instantaneous inward and a two-variable outward current (red line), and an instantaneous inward and single-variable outward current (black line).

Additional simulations were run where no ML-type models were allowed to enter the population in order to study if other potential model specifications were 'crowded out' by the true specification. The resulting population was dominated by the instantaneous inward and two-variable outward current models. In this simulation, the models score better, sooner than in the original simulation. Presumably this is due to the reduced diversity among the population, which promotes more reliable crossover. However, exclusion of the ML-type models results (after 100 generations) in GA produced models with inferior scorers compared to the original simulation. Further limiting the simulation to not allow either the ML-type or the instantaneous inward and two-variable outward current model results in only the single-variable inward and instantaneous outward models being produced. This again scores better, sooner than the original simulation, but ultimately results in inferior scoring models, as well as fewer firing neuron models.

IV. SUMMARY

We have shown that a GA can be used for model specification in addition to parameter optimization. Our results also demonstrate the ability of a GA to produce a diversity of solution possibilities. Providing the GA the freedom to choose the formulation for each current allowed the GA to find the best class of model for the task. Since a ML model was used to generate the target I-F curve, it should not be surprising that the GA eventually chose a ML-type model. Limiting the class of models available results in a lower final fitness, confirming that incorrect assumptions about model formulation will result in inferior models. These results are useful for future modeling studies where the desired formulation of the model is unknown. This technique is also scalable to much larger and more complex models.

ACKNOWLEDGMENT

This work was supported by a grant from The National Institutes of Health.

REFERENCES

[1] M.C. Vanier, J.M. Bower, "A comparative survey of automated parameter-search methods for compartmental neural models," *J Comput Neurosci*, 7, pp. 149-71, 1999.
[2] C. Morris, H. Lecar, "Voltage oscillations in the barnacle giant muscle fiber," *Biophys J*, 35, pp. 193-213, 1981.
[3] D.E. Goldberg, K. Deb, J.H. Clark, "Genetic algorithms, noise, and the sizing of populations," *Complex Systems*, 6. pp. 333-362, 1992.
[4] A.J. Chipperfield, P.J. Fleming, "The MATLAB Genetic Algorithm Toolbox," *IEE Colloquium on Applied Control Techniques Using MATLAB*, 1995/014, 26 Jan, 1995.
[5] B. Ermentrout *Simulating, Analyzing, and Animating Dynamical Systems. A Guide to XPPAUT for Researchers and Students*. Philadelphia: SIAM, 2002.