2016 International Symposium on Nonlinear Theory and Its Applications,
NOLTA2016, Yugawara, Japan, November 27th-30th, 2016

Two Heuristic Approaches to Parameter Tuning for an Analog Silicon Neuron Circuit

Ethan Green† and Takashi Kohno†

† Institute of Industrial Science, The University of Tokyo
4-6-1 Komaba, Meguro, Tokyo, 153-8505, Japan
Email: green@sat.t.u-tokyo.ac.jp, kohno@sat.t.u-tokyo.ac.jp

Abstract – Neurons play a major role in memory, cognition, sensory processing, body regulation, and a host of other functions vital to organisms. Analog silicon neurons are biologically inspired VLSI (very-large-scale integrated) circuits that mimic the electrophysiological behavior of neurons. This research looks at circuit parameter tuning for an ultra-low power analog silicon neuron designed with qualitative neuronal modeling. A key challenge to operating this circuit is adjustment of the circuit parameters to allow for similar behavior across a range of temperatures and eventually amongst many silicon neuron circuits in a silicon neuronal network. Two heuristic approaches were applied to the silicon neuron to supplement trial-and-error-based tuning of the circuit's parameter voltages. In the future, these two approaches will be combined to create a fully automated tuning algorithm.

1. Introduction

Analog silicon neurons are electronic circuits that adopt the electrophysiological characteristics of neurons, the principle cells of the nervous system. These circuits operate in continuous time, require minimal power, and can be integrated into massively parallel networks [1]. These silicon neuronal networks may form the basis of future computers with neuromimetic architecture that may supplement digital transistor logic with novel computing techniques. Future uses of silicon neuronal networks may include autonomous machines, bio-silico hybrid devices, and ultra-low-power computing platforms.

Qualitative neuronal modeling refers to the use of approximation to reduce the complexity of biophysically accurate ionic-conductance models to create simpler mathematical models which maintain similar neuronal dynamics with fewer variables and simpler formulae. The silicon neuron used in this research was designed with a qualitative modeling approach in order to implement simpler circuitry. The silicon neuron can replicate Class I and Class II spiking as defined by Hodgkin [2] with only two variables [3].

While simpler than conductance-based silicon neuron circuits, our silicon neuron still requires many parameter voltages—bias voltages applied to the components comprising the circuit. A challenge to proper operation of the circuit is finding parameter voltages that can satisfy narrowly defined criteria to yield neuron-like operation. The circuit is equipped with feedback amplifiers that guide finding appropriate parameter voltages with a procedure similar to voltage clamp experiments. Since the circuit uses subthreshold-operated MOSFETs, operation suffers from pronounced temperature sensitivity and requires the parameter voltages to be varied with temperature.

In our previous work [4], we showed that trial and error can be used to find parameter voltage sets for different temperatures. We identified pillar sets of parameter voltages at 17, 22, 27, 32, and 37°C and then interpolated parameters for intermediary temperatures and reported how successfully these sets replicated benchmark circuit behavior.

In this work, we discuss an algorithm that can more effectively generate parameter voltages for pillar temperatures and also be used to tune silicon neuron circuits in a future silicon neuronal network. We show the merits and drawbacks of two algorithms, one based on brute force and the other on Differential Evolution, and then propose a future tuning algorithm that combines the strongpoints of these two approaches. The Spectre circuit simulation platform was used for circuit simulations. The model and circuit of our silicon neuron is explained in the next section. The trial and error approach is reviewed in Sec. 3. The two new approaches are reported in Sec. 4, which is followed by a conclusion.

2. Silicon Neuron Model and Circuit

The silicon neuron circuit [3] is divided into two blocks, a *v*-block and *n*-block, each with a capacitor that is charged and discharged by transconductance circuit components (Fig. 1). Variable *v* represents the membrane potential and variable *n* represents an abstracted ionic activity. Each variable is coded by subtracting the voltage over its capacitor from V_{dd}.

The system equations of the silicon neuron are:

$$C_v \frac{dv}{dt} = f_v(v) - g_v(v) + I_{av} - r(n) + I_{\text{stim}}, \quad (1)$$

$$C_n \frac{dn}{dt} = f_n(v) - g_n(v) + I_{an} - r(n), \quad (2)$$

where $f_x(v)$, $g_x(v)$, and $r(n)$ are the equations governing the transconductance circuit components, I_{ax} is a constant current (x=v,n), and I_{stim} is an externally applied stimulus

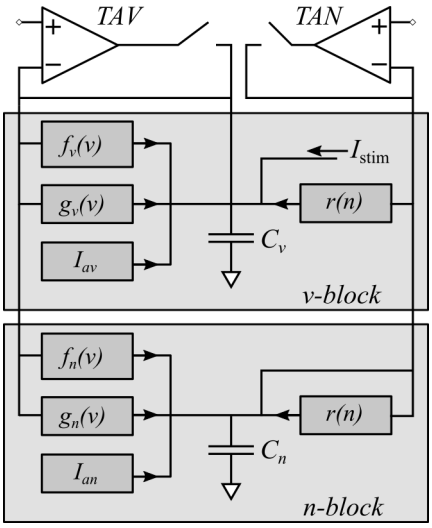

Figure 1: Silicon neuron circuit diagram. Adapted from [4].

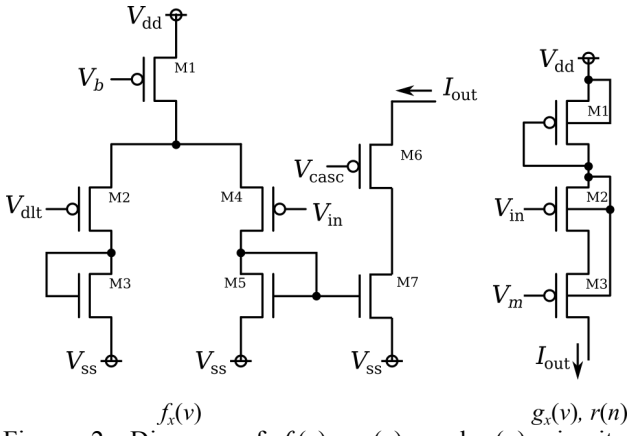

Figure 2: Diagram of $f_x(v)$, $g_x(v)$, and $r(n)$ circuits. Reprinted from [4].

current. The $f_x(v)$ circuit is a differential pair tied to a cascoded current mirror, and $g_x(v)$ and $r(n)$ are modified cascode circuits with source degeneration (Fig. 2). The current-voltage equations for these components are:

$$f_x(v) = \frac{M_x}{1+\exp\left(-\frac{\kappa}{U_T}(v-\delta_x)\right)}, \quad (3)$$

$$g_x(v) = I_0 \sqrt{\frac{\exp\left(\frac{\kappa}{U_T}\theta_x\right)}{1+\exp\left(-\frac{\kappa}{U_T}(v-\theta_x)\right)}}, \quad (4)$$

$$r(n) = I_0 \sqrt{\frac{\exp\left(\frac{\kappa}{U_T}\theta_r\right)}{1+\exp\left(-\frac{\kappa}{U_T}(n-\theta_r)\right)}}, \quad (5)$$

where constants M_x, δ_x, θ_x (x=v,n), and θ_r correspond to externally applied bias voltages (here referred to as parameter voltages), κ is the PMOS capacitive coupling ratio, I_0 is the PMOS off-current, and U_T is the thermal voltage. Eqs. (3)–(5) express sigmoidal relationships [3][5].

All transistors in the silicon neuron circuit are operated in the subthreshold regime, allowing for desirable

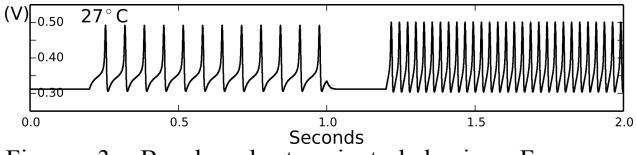

Figure 3: Benchmark transient behavior. Frequency response to a 5 pA and 10 pA sustained stimulus, 15.1 and 36.2 Hz respectively.

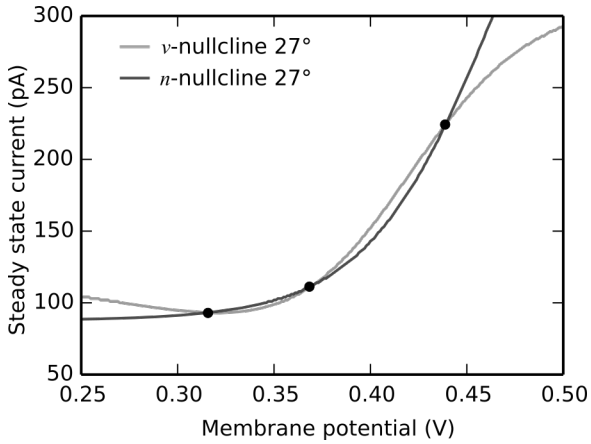

Figure 4: Benchmark nullclines with three intersections highlighted

exponential current-voltage characteristics and total power consumption as low as 3 nW.

The nullclines, curves on which Eqs. (1) and (2) equal zero, can be used to describe the neuron-like dynamics of the circuit [6]. *TAV* and *TAN* are the transconductance amplifiers that measure the nullclines with a voltage clamp technique similar to a DC steady state analysis.

3. Parameter Tuning with Trial and Error

Pronounced temperature sensitivity is a major drawback to operating transistors in the subthreshold regime. Temperature changes of a few degrees can completely disrupt normal operation of the circuit. Furthermore, due to transistor mismatch, individual silicon neurons in a future neuronal network will require a unique set of parameter voltages. These factors illustrate the importance of developing a parameter voltage tuning strategy for proper operation of the silicon neuron.

In [4], we performed Spectre simulations in which we adjusted the parameter voltages of the silicon neuron to yield similar dynamics at different temperatures. We established a benchmark at 27°C in which we recorded the circuit's response to 5 pA and 10 pA sustained stimuli (Fig. 3), and the threshold current necessary to generate an action potential for a 500 μs pulse stimulus (I_{th}). We also plotted the nullclines (Fig. 4). All successive attempts to tune the circuit were judged by these benchmarks. Class I spiking was used because of its readily discernable phase plane with three nullcline intersections (Fig. 4) and simple firing mechanism governed by a saddle-node on invariant circle bifurcation [6].

We then simulated the circuit at 17, 22, 32, and 37°C and adjusted the parameter voltages with trial and error. Five influential parameter voltages were selected with all others held constant. The $g_n(v)$ circuit was turned off. The nullclines were used to find approximate parameter voltages, which were then tuned until circuit operation matched the benchmark transient behavior as closely as possible.

Table 1 shows the parameter voltages found by trial and error for each temperature. Parameter voltages gv_Vm, fn_Vb, Iav_Vin, Ian_Vin, and rn_Vm in the table correspond to constants θ_v, M_n, I_{av}, I_{an}, and θ_n in the circuit equations. The bottom three rows of the table note the transient behavior of each parameter set at its temperature.

4. Heuristic Algorithms

Trial and error allowed us to find adequate parameter voltages for a variety of temperatures, but the method was cumbersome and time consuming. To bring a degree of automation to parameter tuning, we wrote a script which calls Spectre, runs transient or nullcline-drawing simulations, and analyzes results. This script can then be integrated with search algorithms to identify parameter voltages that recreate the benchmark circuit behavior.

4.1. Brute Force

The first search algorithm used was a brute force method. Our aim was to fine-tune the results of the trial and error approach by searching for better parameter voltages in the nearby vicinity of parameter space. For each of the 5 parameter voltages, the brute force algorithm tested 5 possibilities: the original value, ±0.5 mV, and ±1 mV. The circuit was then simulated with all possible parameter combinations. The threshold current (I_{th}) and responses to 5 pA and 10 pA sustained stimuli were recorded. 5^5=3125 simulations required about 3.5 hours of calculation time.

The algorithm searches for the minimum of a cost function, in this case the magnitude of the difference between vectors of the observed circuit behavior and benchmark behavior (Eqs. (6) and (7)). Vector **x** represents the input circuit parameters, function **j**(**x**) is the vector of behaviors as recorded by the circuit simulator, and vector **a** is the benchmark behavior. The threshold index is related to the threshold current.

$$\mathbf{j}(\mathbf{x}) = \begin{bmatrix} \text{threshold index} \\ \text{5 pA stimulus response} \\ \text{10 pA stimulus response} \end{bmatrix}, \quad \mathbf{a} = \begin{bmatrix} 1.21 \\ 15.87 \\ 37.18 \end{bmatrix} \quad (6)$$

$$f(\mathbf{x}) \triangleq \|\mathbf{j}(\mathbf{x}) - \mathbf{a}\| \quad (7)$$

The brute force algorithm includes a polishing step which uses the simulation results to predict the location of the global minimum between parameter combinations.

Table 1: Results of Trial and Error. Reprinted from [4].

Temperature (°C)	17	22	27	32	37
gv_Vm (mV)	388	417	432	443	449.5
fn_Vb (mV)	257	248	237	231	219
Iav_Vin (mV)	434	450	461	468.5	473.5
Ian_Vin (mV)	420	449.5	464.5	475	481.5
rn_Vm (mV)	480.5	462.5	445	430	415
I_{th} (pA)	201	185	178.5	164	147
5 pA response (Hz)	16.3	15.8	15.1	15.7	18.5
10 pA response (Hz)	39	38.8	36.2	38.7	39.4

Table 2: Results of Brute Force

Temperature (°C)	17	22	27	32	37
gv_Vm (mV)	387	416.1	432	442.2	448.6
fn_Vb (mV)	257	247.4	237	230.1	218.7
Iav_Vin (mV)	434.4	451.4	461	469.2	474.2
Ian_Vin (mV)	420.4	451.2	464.5	475.7	482.3
rn_Vm (mV)	478.2	460.3	445	428.6	413.9
I_{th} (pA)	202.5	184.5	178.5	167.5	150.5
5 pA response (Hz)	15.8	16.1	15.1	14.4	17.2
10 pA response (Hz)	37.2	37.1	36.2	37.1	37.4

Table 3: Results of Differential Evolution

Temperature (°C)	17	22	27	32	37
gv_Vm (mV)	394.3	415.7	432	445.3	456.7
fn_Vb (mV)	253.4	245.2	237	228.7	220.3
Iav_Vin (mV)	433.5	450.4	461	468.3	473.6
Ian_Vin (mV)	418.9	449.4	464.5	474.3	481.2
rn_Vm (mV)	478.2	460.3	445	428.6	407.2
I_{th} (pA)	201	181	178.5	179	189
5 pA response (Hz)	17.3	16.9	15.1	9.2	0
10 pA response (Hz)	38.5	34.5	36.2	38.1	35.1

For example, for fn_Vb at 22°C, the algorithm tested 247, 247.5, 248, 248.5, and 249 mV and by interpolation returned 247.4 mV as the global minimum.

Table 2 shows the results of the brute force algorithm. Three circuit behaviors were measured for the four temperature steps. The categories which showed improvement from the original trial and error results are highlighted in gray.

4.2. Differential Evolution

The second search algorithm used was Differential Evolution, an evolutionary algorithm that evolves a population of random solution vectors over many generations to find the global minimum of a cost function [7]. Our algorithm was inspired by a similar algorithm in [8].

Instead of running transient simulations, the algorithm employs the silicon neuron's nullcline drawing feature. The cost function used was the vector magnitude of the mean absolute error of the *v*- and *n*-nullclines from the benchmark nullclines (Eq. 8). **x** again is a vector of the input parameter voltages, $v_i(\mathbf{x})$ and $n_i(\mathbf{x})$ represent points on the *v*- and *n*-nullclines, and b_i and c_i are the corresponding points on the benchmark nullclines. The summation is carried out over each step of a DC steady state analysis.

$$f(\mathbf{x}) \triangleq \sqrt{(\frac{1}{k}\sum_{i=1}^{k}|v_i(\mathbf{x}) - b_i|)^2 + (\frac{1}{k}\sum_{i=1}^{k}|n_i(\mathbf{x}) - c_i|)^2} \quad (8)$$

The algorithm starts with an initial set of random nullclines and gradually fits them to the benchmark curves by rejecting parameter sets which perform poorly while maintaining and evolving sets which perform well. Figure 5 shows the nullclines returned by the algorithm for 32°C overlaid on the benchmark nullclines. One run of the algorithm required about 3 hours of computing time.

The nullclines are independent of the $r(n)$ circuit, so the parameter voltage for rn_Vm—the only parameter voltage that controls the $r(n)$ circuit—was taken from the brute force approach and further tuned by trial and error for 37°C. Table 3 shows these parameter sets along with the results of the transient simulations, which were comparable to the original trial and error approach for 17, 22, and 32°C. The 5 pA stimulus failed to induce firing for 37°C. The measured behaviors that were closer to the benchmark than the trial and error results are highlighted in gray. Fewer categories were closer to the benchmark than for brute force, but the Differential Evolution algorithm was significantly more automated.

5. Conclusion

5.1. Discussion

The brute force algorithm worked well by evaluating the circuit's transient behavior, but suffered from a limited search range, the expansion of which exponentially increases calculation cost. Effective use of this algorithm necessitates a first step which identifies approximate parameter voltages.

The Differential Evolution algorithm on the other hand starts from a random solution and automatically finds parameter voltages that yield accurate nullclines relatively quickly. However, the results of the transient simulations were poor. While the nullclines give good clues about the dynamics of the circuit, they only partially describe the system. Additionally, the algorithm in its current form does not tune $r(n)$ because of circuit constraints. A more effective algorithm must tune $r(n)$ and take transient simulations into account.

5.2. Future Algorithm

We propose a 2-stage heuristic algorithm which combines the merits of Differential Evolution and brute force. The first stage will use Differential Evolution to tune the nullclines of the circuit to best match the benchmark nullclines. The second stage will use brute force to search all nearby parameter combinations for the parameter set which most closely replicates the benchmark transient behavior. The brute force stage may also search a wider range for rn_Vm of $r(n)$ since this parameter is not incorporated in the first stage.

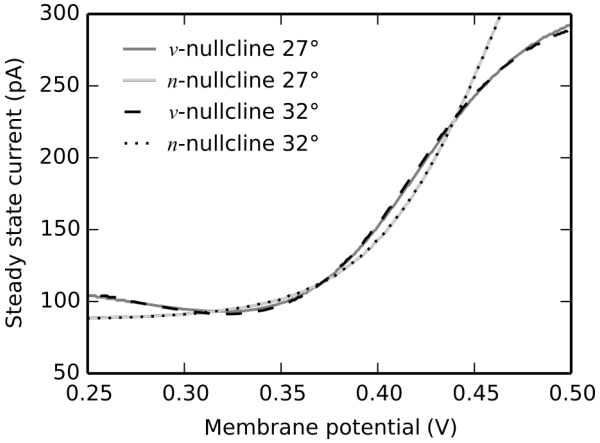

Figure 5: Nullclines from Differential Evolution for 32°C

Such an algorithm may improve the above results. The same approach may also be used to tune different silicon neurons of the same design which vary due to transistor mismatch.

We plan to first simulate this algorithm with Spectre and eventually implement it with our actual VLSI circuit using lab equipment and software such as LabVIEW.

6. Acknowledgements

This study was supported by JST PRESTO and CREST, and VDEC, the University of Tokyo in collaboration with Cadence Design Systems, Inc.

7. References

[1] S. Brink, et al., "A learning-enabled neuron array IC based upon transistor channel models of biological phenomenon," IEEE Transaction Biomedical Circuits and Systems, vol. 7, no. 1, pp. 71-81, Feb. 2013.

[2] A. Hodgkin, "The local electric changes associated with repetitive action in a non-medullated axon," *The Journal of Physiology*, 107, 2, pp. 165-181, March, 1948.

[3] T. Kohno and K. Aihara, "A Qualitative-Modeling-Based Low-Power Silicon Nerve Membrane," *Electronics, Circuits, and Systems (ICECS)*, IEEE, pp. 199-202, December, 2014.

[4] E. Green and T. Kohno, "Compensating Temperature-Dependent Characteristics of a Subthreshold-MOSFET Analog Silicon Neuron," J*ournal of Robotics, Networking and Artificial Life*, vol. 2, no. 4, pp. 209-212, March, 2016.

[5] S. Liu, et al., *Analog VLSI: Circuits and Principles*, MIT Press, 2002.

[6] J. Rinzel and B. Ermentrout, "Analysis of Neural Excitability and Oscillations," *Methods in Neuronal Modeling*, Massachusetts Institute of Technology, pp. 251-291, 1998.

[7] K. Price and R. Storn, *Differential Evolution: A Practical Approach to Global Optimization*, Springer, 2005.

[8] F. Grassia, et al., "Tunable neuromimetic integrated system for emulating cortical neuron models," *Frontiers in Neuroscience,* vol. 5, article 134, December, 2011.