Efficient Pattern Recognition Using the Frequency Response of a Spiking Neuron

Sergio Valadez-Godínez, Javier González, and Humberto Sossa(✉)

Centro de Investigación en Computación, Instituto Politécnico Nacional,
Av. Juan de Dios Batiz, S/N, Col. Nva. Industrial Vallejo,
07738 Mexico City, Mexico
svaladezg@gmail.com, xvrgonzalez21@gmail.com, hsossa@cic.ipn.mx

Abstract. In previous works, a successful scheme using a single Spiking Neuron (SN) to solve complex problems in pattern recognition has been proposed. This consists in using the firing frequency response to classify a given input pattern, which is multiplied by a weight vector to produce a constant stimulation current. The weight vector is adjusted by an evolutionary strategy where the objective is to obtain an optimal frequency separation. The problem is that the SN has to be numerically simulated several times when the weight vector is being adjusted. In this work, we propose fitting the SN frequency response curve to a piecewise linear function to be used instead of the costly SN simulation. A high fitting degree was found, but, more importantly, the computational cost of the training and testing phases was drastically reduced.

Keywords: Spiking Neuron · Izhikevich · Pattern recognition · Curve fitting · Frequency Response Curve · Piecewise linear function · Firing Rate · Evolutionary strategy · Differential evolution · Computational cost

1 Introduction

In [1–8], a successful scheme using a single SN to solve complex pattern recognition problems has been proposed. This scheme consists in using the firing rate codification to classify a given input pattern. The SN is stimulated by a constant input current to obtain the firing frequency. The frequency represents the class to which the pattern is associated. The necessary current is obtained by applying the dot product between the input and a weight vectors, as is normally done when using the Perceptron [9]. The weight vector is adjusted by an evolutionary strategy to generate the optimal firing rates maximizing the separation among the classes. The problem is that the SN has to be numerically simulated many times while the weight vector is being adjusted. Therefore, depending on the number of individuals and generations in the evolutionary strategy and on the numerical method, integration step and the time window of the SN simulation, the computational cost during training is highly expensive. Although the authors previously obtained a comparable computational cost to other methodologies, such as the Support Vector Machines or Artificial Neural Networks of

J.A. Carrasco-Ochoa et al. (Eds.): MCPR 2017, LNCS 10267, pp. 53–62, 2017.
DOI: 10.1007/978-3-319-59226-8_6

the second generation, their analysis was performed for the testing phase and not for the training one [8]. In the testing phase, only a single SN simulation is utilized whereas, as previously mentioned, the training stage needs several SN simulations.

As the mentioned methodology uses a constant input current and a firing rate encoding, the firing frequency in function of the current, so-called Frequency Response Curve (FRC), can be obtained. Since the influential work of the Nobel laureate E. D. Adrian, the FRC of a sensory nerve ending to an applied stimulus is well known [10]. In the case of SNs, the FRC can be analytically obtained for the Leaky Integrate-and-Fire (LIF) [11,12]. However, that is more complicated for the Izhikevich (IZH) [13] and Hodgkin-Huxley (HH) [14] models. Moreover, the FRC for the HH can be fitted to a logarithmic function [15]. For the case of the IZH, no fitting is known, but its FRC is highly linear and, hence, this can be accurately fitted to a piecewise linear function using the Least Squares Method (LSM). The IZH is prevalently utilized in the previous works [2–8].

In this paper, we propose fitting the FRC of the IZH to a piecewise linear function to be used instead of the costly SN simulation. A high fitting degree was found, but, more importantly, the computational cost in the training and testing phases was drastically reduced. The Differential Evolution (DE) algorithm was selected as the evolutionary strategy [16,17].

The remainder of the paper is organized as follows. Section 2 focuses on the SN scheme to solve complex problems in pattern recognition. Section 3 describes the IZH model. Section 4 shows the fitting procedure. Section 5, gives the results and Sect. 6 presents the conclusions and future work.

2 Pattern Recognition Using a Single Spiking Neuron

In general, the scheme presented in [1–8] is stated as follows: it is supposed SN groups input patterns producing similar firing rates; once this is made, it allows discriminating input patterns giving different firing rates. So, the SN functions as a pattern classifier. A n-dimension input pattern can be represented as $\mathbf{x} = (x_1, x_2, \cdots, x_n)^T \in \mathbf{X} \subset \mathbb{R}^n$ where $x_1, x_2, x_3, \cdots, x_n$ are the pattern features and $\mathbf{X}$ is the dimensional features domain. A set of K input patterns is defined as $\left\{\mathbf{x}^k \in \mathbf{X}^n\right\} \forall k = \{1, 2, \cdots, K\}$ where k is an index and K is the cardinality of the pattern set. A class to which a pattern belongs is defined as $c \in \{1, 2, \cdots, m\}$ where m is the number of classes. Consequently, it is expected that the SN generates m different firing frequencies, each representing a class. For a specific input pattern $\mathbf{x}^k$, there is a class $c^k \forall k = \{1, 2, \cdots, K\}$. Thus, the set A of associations is defined as $A = \left\{\left(\mathbf{x}^k, c^k\right)\right\} \forall k = \{1, 2, \cdots, K\}$. As the SN is not directly stimulated by an input pattern, the input pattern $\mathbf{x}^k$ is multiplied by a weight vector $\mathbf{w} \in \mathbb{R}^n$ to obtain a constant stimulation current $I = \mathbf{x}^k \cdot \mathbf{w}$ to the SN. For example, the LIF is employed in [1] and the IZH in [2–8].

During training, the weight vector is adjusted by using an evolutionary algorithm with the idea of producing the optimal firing rates that allow maximizing separation between classes. For example, Differential Evolution [16,17] is used

in [1–3,6], Cuckoo Search [18] in [4], Particle Swarm Optimization [19] in [5,8] and Artificial Bee Colony [20] in [7]. The firing rate fr^k produced by I for a particular input pattern $\mathbf{x}^k$ is calculated as $fr^k = {}^{\mathrm{N_{sp}}}/_{\mathrm{T}}$ where N_{sp} is the number of spikes that occur in the time span T. Later, the Average Firing Rate $AFR \in \mathbb{R}^m$ of all firing rates fr^k produced by the patterns belonging to the same class is estimated. The class to which an input pattern $\mathbf{x}^k$ is classified is $\widetilde{c} = arg\ min_{c=1}^{m} \left(\left|AFR_c - fr^k\right|\right)$. The evolutionary algorithm minimizes the fitness function $f(\mathbf{w}, A) = 1 - {}^{P_{cc}}/_{P}$ where P_{cc} is the number of input patterns being correctly classified (i.e. $\widetilde{c} = c^k$) and P the number of patterns that actually belong to the given class c^k. This way, the evolutionary algorithm automatically determines the optimal frequencies that maximize the separation among the classes. During testing, once the weight vector $\mathbf{w}$ was adjusted and the AFR calculated, the estimulation current to the SN for a unknown pattern $\widetilde{\mathbf{x}}$ is determined by $I = \widetilde{\mathbf{x}} \cdot \mathbf{w}$, the firing rate by $\widetilde{fr} = {}^{\mathrm{N_{sp}}}/_{\mathrm{T}}$ where N_{sp} is the number of spikes that occur in the time span T and the classification by $\widetilde{c} = arg\ min_{c=1}^{m} \left(\left|AFR_c - \widetilde{fr}\right|\right)$.

3 The Izhikevich Model

This SN is a successful dimensional reduction of the HH [14]. In fact, once the HH was reduced to a two-dimensional system [21,22], Izhikevich analyzed the dynamics in the phase plane [23] to propose a new description [13]. This is

$$\begin{array}{c} \frac{dV}{dt} = 0.04V^2 + 5V + 140 - n + I \\ \frac{dn}{dt} = a\left(bV - n\right) \end{array} \tag{1}$$

where V is the trans-membrane potential, I is the stimulation current, n is a recovery variable representing the potassium activation and sodium inactivation, a is a time scale and b the sensitivity of n to the sub-threshold fluctuations of the trans-membrane potential. When the trans-membrane potential reaches a maximum of 30 mV the voltage V and the recovery variable n are restarted, i.e.

$$If\ V \geq 30\ \mathrm{mV},\ then\ \begin{cases} V \leftarrow c \\ n \leftarrow n + d \end{cases} \tag{2}$$

where c is the restarting potential and d is the restarting of n.

As in [13], the constants were $a = 0.02$, $b = 0.2$, $c = -65$ and $d = 8$ for generating a regular spiking. Figure 1 shows an example of a spike train produced by a constant input current $I = 31\,{}^{\mu\mathrm{A}}/_{\mathrm{cm^2}}$.

4 Obtaining and Fitting the Frequency Response Curve

As in [2–5,7,8], the IZH was simulated with the Forward Euler method [24] but with a step size of 0.05 ms to produce an adecuate simulation due to the IZH not being as efficient as it was thought (see [25,26] for details). To obtain the FRC, the IZH was stimulated with various constant input currents I ranging

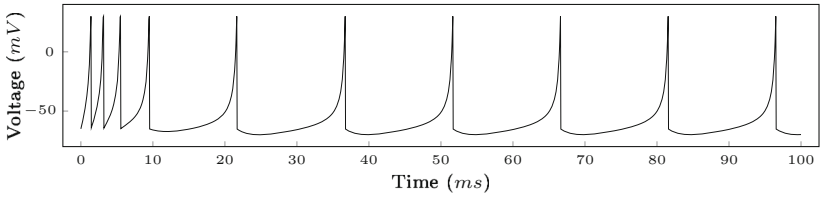

Fig. 1. Spike train generated by the Izhikevich (IZH) model.

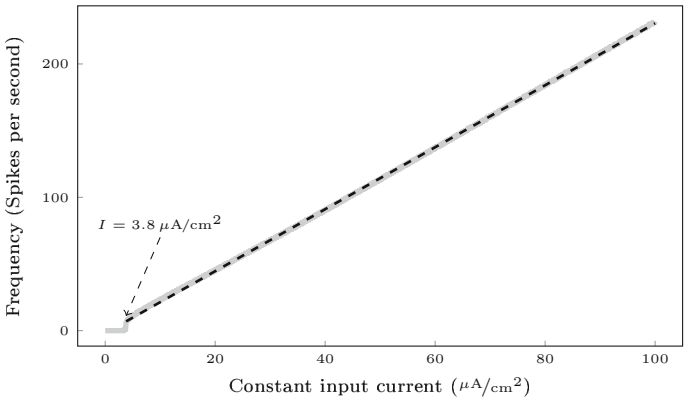

Fig. 2. Linear fitting (dashed line) of the Frequency Response Curve (FRC) (gray continuos line) of the Izhikevich (IZH) model.

from $0\,\mu\text{A/cm}^2$ to $100\,\mu\text{A/cm}^2$ across a current step of $0.1\,\mu\text{A/cm}^2$ and a time span of 1000 ms. The FRC can be seen in Fig. 2 as a gray continuous line.

As can be seen, the FRC is in agreement with a high linear relationship when the IZH produces a firing response for $I \geq 3.8\,\mu\text{A/cm}^2$ (see the arrow in Fig. 2). A strong positive-linear correlation was obtained from the correlation coefficient analysis performed in MATLAB® [27]. The correlation matrix was $\left(\begin{smallmatrix} 1.000 & 0.999 \\ 0.999 & 1.000 \end{smallmatrix}\right)$ for the two variables (current and frequency).

The Curve Fitting Toolbox® software of MATLAB® [27] was employed to fit the FRC from $I \geq 3.8\,\mu\text{A/cm}^2$ to a linear function. As the LSM is sensitive to extreme values in the FRC, the Robust Least Squares was utilized. The Least Absolute Residuals (LAR) method was selected to minimize the influence of such extreme values. With this, a piecewise linear function was obtained

$$fr^k = \begin{cases} 2.324I - 1.898 & I \geq 3.8\,\mu\text{A/cm}^2 \\ 0 & otherwise \end{cases} \tag{3}$$

where fr^k is the frequency and $I = \mathbf{x}^k \cdot \mathbf{w}$ the input current that depends on the k-th pattern.

To prove the validity of (3), the Sum of Squares due to Error (SSE), the Root Mean Square Error ($RMSE$) and the ratio (R^2) of the Sum of Regression Squares and the Total Sum of Squares were obtained. With 95% of confidence, the values were $SSE = 735.572$, $RMSE = 0.875$ and $R^2 = 1.000$. In general, the fitting accuracy gives a good level of confidence due to the ratio R^2 indicating that the variance in the frequency is low. Equation (3) for $I \geq 3.8\,\mu A/cm^2$ is depicted in Fig. 2 as a dashed line and the residual errors are shown in Fig. 3. Thus, the frequency to perform the pattern classification task can be obtained from (3) instead of the IZH numerical simulation.

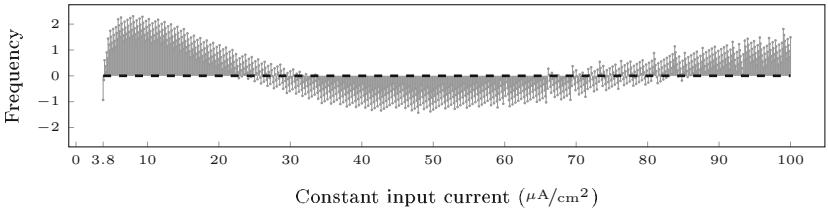

Fig. 3. Residual errors of the linear fitting for the Frequency Response Curve (FRC) of the Izhikevich (IZH) model.

5 Results

Two sets of experiments were carried out to verify that using the piecewise linear function described in (3) is more efficient than using the IZH simulation. See the Appendix for the technical specifications.

The first set of experiments was to obtain a single frequency value. A current $I = 31\,\mu A/cm^2$ was employed in (3) and in (1-2). This experiment was repeated 20 times and the average of the CPU time execution was determined. The IZH was numerically simulated using a time span $T = 1000$ ms and the parameters described in Sects. 3 and 4. The frequencies from (3) and (1-2) were 70.15 and 70 spikes per second. The calculus using (3) consumed on average 2.45 μs (± 0.51) and the IZH simulation 147.15 ms (± 3.26). The frequencies are practically the same, but the difference in the computational cost was four orders of magnitude. The calculus of a single frequency value using (3) is the 0.0017% of the CPU time consumed by the IZH simulation, or, in other words, the computational cost using the IZH simulation is $6,000,000\%$ higher than using the piecewise linear function.

The second set of experiments consisted in solving the well-kown Iris dataset problem [28] by using the methodology described in Sect. 2. To obtain the frequency, in one set of experiments the IZH simulation was employed and in the other, the piecewise linear function in (3). The parameters for the DE [16,17] algorithm were the same as in [2]: $NP = 40$, $MAXGEN = 1000$, $F = 0.9$, $XMAX = 10$, $XMIN = -10$ and $CR = 0.8$. Where NP is the number of

Table 1. Average classification accuracy (± Standard deviation) using the simulation and the piecewise linear function of the Izhikevich (IZH) Frequency Response Curve (FRC).

Dataset	IZH simulation		Piecewise FRC	
	Training	Testing	Training	Testing
Iris	0.9933 (± 0.0023)	0.9867 (± 0.0281)	0.9933 (± 0.0023)	0.9800 (± 0.0322)

Table 2. Average CPU time (± Standard deviation) (h = hours, s = seconds, μs = microseconds) using the simulation and the piecewise linear function of the Izhikevich (IZH) Frequency Response Curve (FRC).

Dataset	IZH simulation		Piecewise FRC	
	Training	Testing	Training	Testing
Iris	1.61 h (± 0.02)	93.48 ms (± 1.95)	46.41 s (± 0.52)	297.60 μs (± 155.87)

individuals in the population, $MAXGEN$ is the maximum number of generations, F is a scaling factor that controls the rate at which the population evolves, $XMAX$ is the lower bound of weights, $XMIN$ is the upper bound of weights and CR is the crossover probability [17]. To validate these experiments, 10-Folds Cross-Validation was used [29]. This kind of validation splits the data into ten subsets; each is employed once for the testing and the remaining for the training. The patterns for each subset were arbitrarily selected. Those patterns were presented to both experiments using the IZH simulation and the piecewise linear function. We obtained the average accuracy from the ten experiments of the 10-Folds Cross-Validation. Also, the average of the CPU elapsed time was determined. The classification accuracy and the CPU time execution are shown in Tables 1 and 2, respectively. It can be observed that the classification accuracy in the training stage is the same for both using (3) and using the IZH simulation. Nevertheless, during testing, the accuracy using the piecewise linear function is slightly less. The CPU elapsed time or computational cost using (3) is drastically reduced in both phases. The CPU time execution using the IZH simulation is roughly $12,000\%$ higher than using the piecewise linear function in the training phase. In the testing phase, the percentage is roughly $31,000\%$.

Figures 4 and 5 show the distribution of a set of training patterns using the IZH simulation and the piecewise linear function of the FRC. As can be seen, the pattern distribution is very similar. The distribution corresponds to the first test in the 10-Folds Cross-Validation. The weight vectors for the IZH simulation and the piecewise linear function were $(-2.79, -2.76\ 10.00, 8.35)^T$ and $(-1.92, -4.10, 10.00, 10.00)^T$.

It can be observed that this scheme is different from that of the traditional Perceptron. The Perceptron distributes the patterns in two regions separated by a hyperplane (see [9]) whereas this proposal distributes the patterns on the

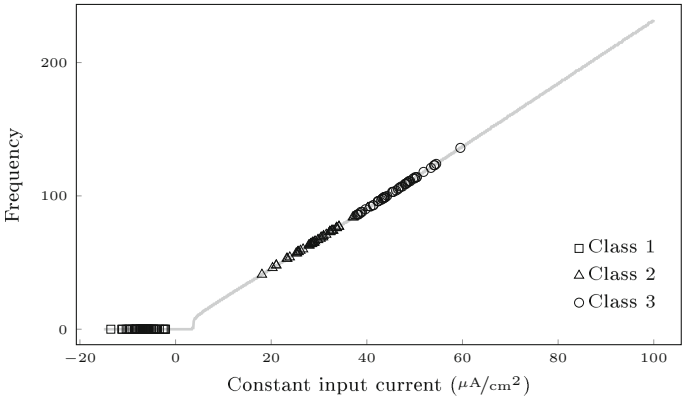

Fig. 4. Pattern distribution using the Frequency Response Curve (FRC) (continuous gray line) obtained from the Izhikevich (IZH) simulation.

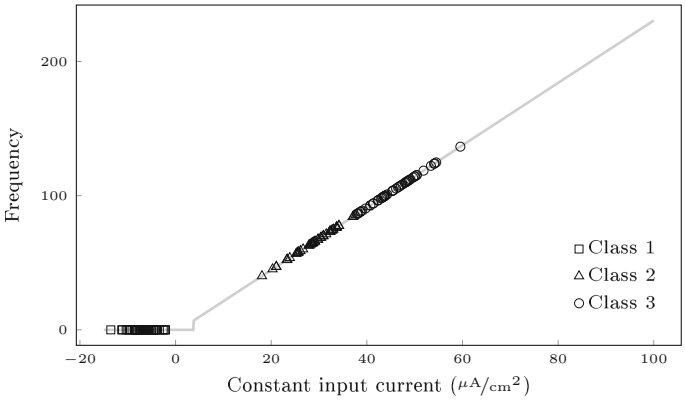

Fig. 5. Pattern distribution using the piecewise linear function (continuous gray line).

FRC. This permits the proposed scheme to solve non-linear problems using a single computational unit. However, as is well-known with a Perceptron, the XOR problem cannot be resolved.

6 Conclusions and Future Work

The single SN scheme consists in using the SN response to group a given input pattern, which is multiplied by a weight vector to produce a constant stimulation current. Depending on the number of individuals and generations in the evolutionary strategy, but also on the numerical method, integration step and the time span of the SN simulation, the computational cost in the training phase is extremely expensive.

The FRC is the relationship between the constant stimulus and the firing frequency emitted by a given SN. The FRC for the IZH model is highly linear. Thus, a piecewise linear function can be fitted to be used instead of the expensive SN simulation. This practically obtains the same accuracy results. Moreover, the computational cost in the training and testing phases was drastically reduced. In the training phase, the CPU time execution using the IZH simulation is roughly $12,000\%$ higher than using the piecewise linear function. In the testing phase, that percentage is approximately $31,000\%$. The computational cost to obtain a single frequency value using the IZH simulation is $6,000,000\%$ higher than using the piecewise linear function. We drastically reduced the computational cost making the scheme very efficient. As the computational cost to simulate the IZH depends on several factors, further research in this respect is needed.

Simulating a SN to solve pattern recognition problems is not recommended when the two following conditions are satisfied: (1) the stimulation current is constant during the entire simulation, and (2) the output signal encoding is the firing rate. SNs are useful for problems where the input pattern is precise in timing, or the output signal encoding is temporal. It is important to highlight that the single SN scheme is interesting due to its capacity to solve complex problems through the optimal classes (frequencies) distribution over the FRC. We believe that in future this scheme will bring new proposals in the research areas of Artificial Neural Networks and Pattern Recognition.

As a future work, we propose extending the experiments to other datasets, SNs, and training algorithms. A gradient descendant algorithm could be implemented due to the piecewise linear function being derivable. Also, this work could represent a first step in linking Deep and Spiking Neural Networks via the Rectified Linear Unit (ReLU) and the FRC linear fitting. Both functions are piecewise linear.

Acknowledgments. S. Valadez-Godínez would like to thank CONACYT and SIP-IPN for the scholarship granted in pursuit of his doctoral studies. J. González would like to thank CONACYT and SIP-IPN for undertaking his Master studies. H. Sossa would like to thank SIP-IPN and CONACYT under grants 20170693 and 65 (Frontiers of Science) to carry out this research. We are also very grateful to reviewers for their helpful comments.

Appendix

All tests carried out on a computer running the Linux Mint 17.3 64 bits operating system on an Intel Core i7-2600 (3.4 GHz with eight cores) processor. The memory of the computer was 8 GB. All software was implemented in MATLAB®. The *tic* and *toc* functions were utilized to calculate the elapsed time of the experiments described in Sect. 5.

References

1. Vazquez, R.A., Cachón, A.: Integrate and Fire neurons and their application in pattern recognition. In: 7th International Conference on Electrical Engineering Computing Science and Automatic Control, pp. 424–428 (2010)
2. Vazquez, R.: Izhikevich neuron model and its application in pattern recognition. Aust. J. Intell. Inform. Process. Syst. **11**, 35–40 (2010)
3. Vázquez, R.A.: Pattern recognition using spiking neurons and firing rates. In: Kuri-Morales, A., Simari, G.R. (eds.) IBERAMIA 2010. LNCS, vol. 6433, pp. 423–432. Springer, Heidelberg (2010). doi:10.1007/978-3-642-16952-6_43
4. Vazquez, R.A.: Training spiking neural models using cuckoo search algorithm. In: 2011 IEEE Congress of Evolutionary Computation (CEC), pp. 679–686 (2011)
5. Vázquez, R.A., Garro, B.A.: Training spiking neurons by means of particle swarm optimization. In: Tan, Y., Shi, Y., Chai, Y., Wang, G. (eds.) ICSI 2011. LNCS, vol. 6728, pp. 242–249. Springer, Heidelberg (2011). doi:10.1007/978-3-642-21515-5_29
6. Matadamas Ortiz, I.C.: Aplicación de las Redes Neuronales Pulsantes en el reconocimiento de patrones y análisis de imágenes. Master's thesis, Instituto Politécnico Nacional, Centro de Investigación en Computación, México (2014)
7. Vazquez, R.A., Garro, B.A.: Training spiking neural models using artificial bee colony. Comput. Intell. Neurosci. **2015**, 14 (2015). Article ID 947098
8. Carino-Escobar, R.I., Cantillo-Negrete, J., Gutierrez-Martinez, J., Vazquez, R.A.: Classification of motor imagery electroencephalography signals using spiking neurons with different input encoding strategies. Neural Comput. Appl., 1–13 (2016)
9. Minsky, M., Papert, S.: Perceptrons: An Introduction to Computational Geometry. The MIT Press, Cambridge (1969)
10. Adrian, E.D.: The Basis of Sensation. The Action of the Sense Organs. Christophers, London (1928)
11. Lapicque, M.L.: Recherches quantitatives sur l'excitation électrique des nerfs traitée comme une polarisation. J. Physiol. Pathol. Gen. **9**, 620–635 (1907)
12. Stein, R.B.: A theoretical analysis of neuronal variability. Biophys. J. **5**, 173–194 (1965)
13. Izhikevich, E.M.: Simple model of spiking neurons. IEEE Trans. Neural Netw. **14**, 1569–1572 (2003)
14. Hodgkin, A.L., Huxley, A.F.: A quantitative description of membrane current and its application to conduction and excitation in nerve. J. Physiol. **117**, 500–544 (1952)
15. Agin, D.: Hodgkin-Huxley equations: logarithmic relation between membrane current and frequency of repetitive activity. Nature **201**, 625–626 (1964)
16. Storn, R., Price, K.: Differential evolution-a simple and efficient adaptive scheme for global optimization over continuous spaces. J. Global Optim. **11**, 341–359 (1997)
17. Price, K.V., Storn, R.M., Lampinen, J.A.: Differential Evolution: A Practical Approach to Global Optimization. Springer, Heidelberg (2005)
18. Yang, X.S., Deb, S.: Cuckoo search via lévy flights. In: 2009 World Congress on Nature Biologically Inspired Computing (NaBIC), pp. 210–214 (2009)
19. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: IEEE International Conference on Neural Networks Proceedings, vol. 4, pp. 1942–1948 (1995)
20. Karaboga, D.: An idea based on honey bee swarm for numerical optimization. Technical report TR06, Erciyes University, Engineering Faculty, Computer Engineering Department (2005)

21. Krinskii, V.I., Kokoz, Y.M.: Analysis of equations of excitable membranes - I. Reduction of the Hodgkin-Huxley equations to a second order system. Biofizika, pp. 506–511 (1973)
22. Kepler, T.B., Abbott, L.F., Marder, E.: Reduction of conductance-based neuron models. Biol. Cybern. **66**, 381–387 (1992)
23. Izhikevich, E.M.: Dynamical Systems in Neuroscience: The Geometry of Excitability and Bursting. The MIT Press, Cambridge (2007)
24. Euler, L.: Institutionum calculi integralis. Volumen primum. Petropoli: Impenfis Academiae Imperialis Scientiarum (1768)
25. Humphries, M.D., Gurney, K.: Solution methods for a new class of simple model neurons. Neural Comput. **19**, 3216–3225 (2007)
26. Skocik, M.J., Long, L.N.: On the capabilities and computational costs of neuron models. IEEE Trans. Neural Netw. Learn. Syst. **25**, 1474–1483 (2014)
27. MATLAB: Version 8.5.0 (R2015a). The MathWorks Inc., Natick, Massachusetts (2015)
28. Lichman, M.: UCI machine learning repository (2013)
29. Alpayd, E.: Introduction to Machine Learning. Second edn. The MIT Press (2010)