On-chip Machine Learner for Spike Sorting in Implantable Brain Machine Interfaces (BMI)

Analysis & Implementation of Feature Extraction Methods

Swarnima Korde[1], Subhrajit Roy[2], Enyi Yao[3] and Arindam Basu[4]
Electrical & Electronic Engineering
Nanyang Technological University
Singapore
Email: [1]swarnima.k@gmail.com, [2]subhrajit.roy@ntu.edu.sg, [3]eyao1@e.ntu.edu.sg, [4]arindam.basu@ntu.edu.sg

Abstract—**Advances in neuroscience have enabled the rapid development of electronics for prostheses. The neural signals can be detected and amplified with Multi-Electrode Arrays (MEAs) of the order of 1000; and neural amplifiers, respectively. Modern day recordings use single probes for multiple neuron activity, as studying isolated cells does not present the real-life scenario [1]. The issue of accurately identifying neural or 'spike' signals with their corresponding characteristic neurons is known as 'Spike Sorting', and it consists of a two-step process: Feature Extraction and Clustering. The motivation behind this research is to propose novel bio-inspired feature extraction for the purpose of spike sorting. First, results were matched to papers that proved that derivative-based features performed better in terms of noise and error as compared to the established Principal Component Analysis (PCA). Next, a formal spiky neuron model, the Integrate-and-Fire neuron was functionally modeled on software (MATLAB) and implemented at the transistor level on Spice (CADENCE). Output firing rates, or 'Gains' of both were matched and new features were proposed from the outputs of the spiky neuron model. These features were optimized for error and showed promising results for future research in the area.**

Keywords- Prostheses; Neural/Spike signals; Spike Sorting; Feature Extraction; Clustering; Integrate-and-Fire neuron

I. Background And Motivation

Spike Sorting addresses the problem of grouping neural (or 'spike' signals) into clusters based on the similarity of their shapes. Principally, every neuron fires spikes of a particular shape, and so the resulting clusters correspond to the activity of different putative neurons [1].

Intricate brain processes responsible for communication between the brain and the prosthetic are defined by the activity of large neural populations. Studying a single, isolated cell would give a myopic view of the whole picture [2, 3]. *This is the rationale behind Spike Sorting*. Spike sorting consists of two main components: *Feature Extraction* and C*lassification*. In Feature Extraction, a few features offering best cluster distinction and noise immunity are selected from a dataspace of dimension 'm', thus reducing the dimensionality considerably. Classification focuses on optimal clustering algorithms that are *preferably unsupervised*, so as to save on delays and minimize user intervention [4]. With increased scaling of devices nowadays, effective data compression; i.e. extracting useful features from limited data, is crucial. A lower sampling rate can be traded off to achieve higher data compression, especially for a limited power budget. *The motivation behind this research is self-designed, bio-inspired feature extraction based on compressed data information.*

II. Feature Extraction And Pattern Classification; Algorithmic Development

A. Scope and Methodology

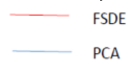
A webpage "Waveclus" [5] introduces new methods for feature extraction. The data provided contains spike potentials recorded from the brain waves of a monkey. It is arranged in varying levels of difficulty; difficulty being inversely proportional to the standard deviation of the added noise, or inverse of signal-to-noise ratio [6]. A superior clustering method, knows as Super Paramagnetic Clustering (SPC) has been used to label the various spike classes, and this is accurate enough to be considered as the 'ground truth' [1, 5].

This section refers to a research paper [6] in accordance with which, two different features are used for clustering the spikes using the unsupervised K-Means method. The deviations between the classifier results and 'spike_class' are reflected as error. This error, known as classification error, is then compared for the two methods/features; namely the newer derivative-based feature: First and Second Derivative Extrema (FSDE), and the common standard Principal Component Analysis (PCA). It is to be noted that two cases are considered: same feature space for both methods, and a higher feature space for PCA.

B. *Results and Discussion*

Out of all the datasets, two have been chosen to best represent the results discussed as follows:

1) *Same feature space: m=3 for FSDE and PCA*



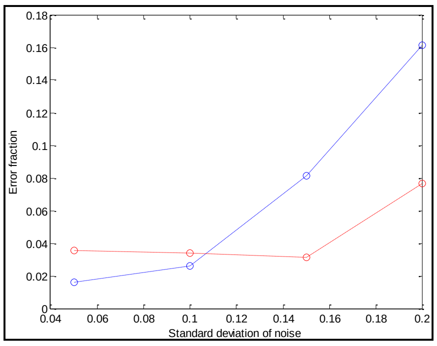


Figure 1:Dataset 2, “Easy 2”

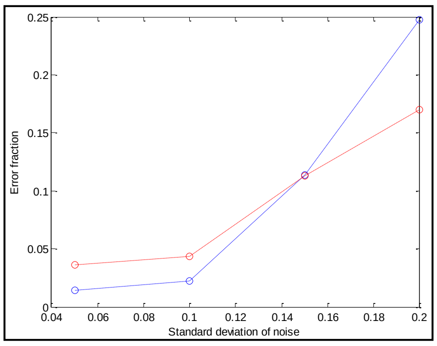


Figure 2: Dataset 4, “Difficult 2”

In general, error is seen to increase with an increase in the level of difficulty as expected, since it gets increasingly more difficult to distinguish between the three clusters. When m=3 for both PCA and FSDE, FSDE performs better than PCA for higher difficulty, as can be deduced from Figures 1 and 2.

2) *Higher feature space: m=10 for PCA, 3 for FSDE*

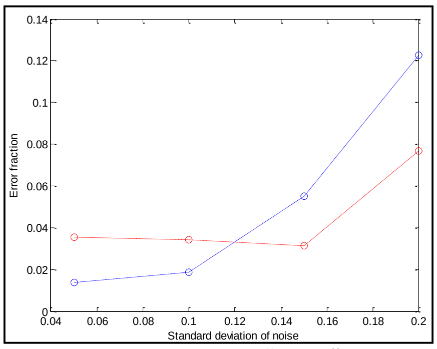


Figure 3:Dataset 2, “Easy 2”

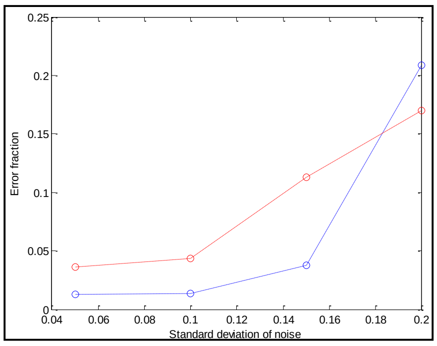


Figure 4: Dataset 4, “Difficult 2”

When m=10 for PCA and m=3 for FSDE, PCA performs better in terms of error.

Therefore, a trade-off exists between performance and data compression. A point in favor of FSDE is that it displays a far better noise immunity than PCA, which is seen to have steep slopes in almost all cases. Moreover, since real-world problems have higher levels of difficulty in segregating the data into clusters, FSDE seems promising Therefore, PCA is not necessarily the optimal method. Other methods such as FSDE, can give lower error and better noise immunity after classification.

III. Pulse-Based Feature Extraction; Software Simulation

The Integrate-and-Fire neuron closely models the working of a formal spiking neuron. A block diagram depicting the simplified working is shown in Fig. 5. The incoming spike (action potential) is first converted to a current ‘I_{input}’ by a capacitor, say ‘$Comp_{cap}$’ Then, ‘I_{input}’ will simply be:

$$I_{input} = Comp_{cap} \; x \; \frac{dV_{Spike}}{dt} \qquad (1)$$

This current charges the capacitor C (Fig. 5) till the voltage across it, ‘V_C’ equals the threshold voltage ‘V_{TH}’. Initially the comparator output is LOW, and the NMOS transistor M1 is OFF. As soon as V_C equals V_{TH}, the comparator output goes to a HIGH and the NMOS is turned ON. This results in the capacitor being discharged, and V_C plunging to the value of the reset voltage [7]. The process involves a conversion from voltage to current and current back to voltage.

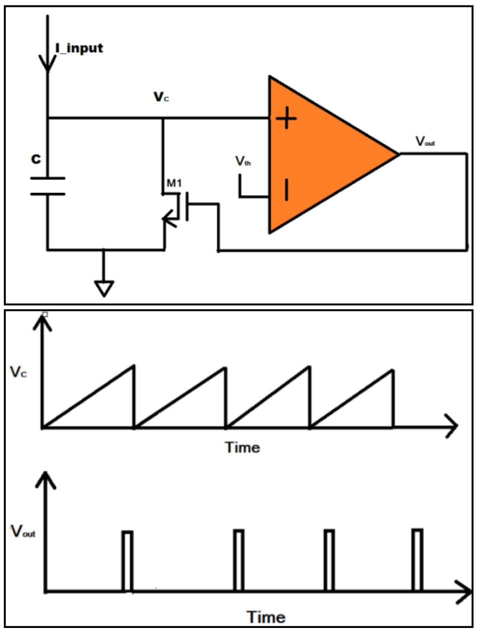


Figure 5: Block diagram of Integrate-and-Fire neuron, waveforms of V_C and V_{OUT}

A. Methodology

The Integrate-and-Fire neuron can be entirely programmed in MATLAB, based on the working that has been described above. Additionally, the current I_{input} is split into positive ('pos') and negative ('neg') halves before being fed independently to the neuron model. This is due to the requirements of the advanced architecture that can then process the positive and negative pulse train outputs separately. The corresponding pulse outputs, 'pos_{sp}' and 'neg_{sp}' obtained from the positive and negative halves respectively, can be seen in Fig. 6. Note that this represents the waveform for one spike.

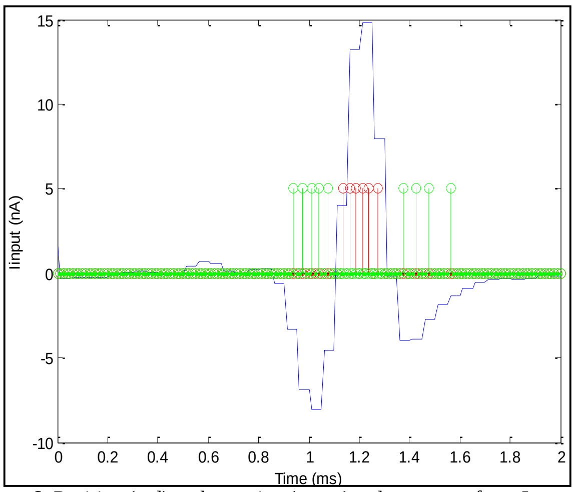


Figure 6: Positive (red) and negative (green) pulse outputs from Integrate-and-Fire neuron for given I_{input}

B. Proposed Feature

The width of a typical spike signal is approximately 2 ms. The total number of 'dt' time (and data) points defining I_{input} is 160, where dt = 12.5 µs. Grouping these into samples of 10 'dt' points each, there would be 16 such samples. Then, it is possible to define vectors P_{Nsp} and N_{Nsp} that count the number of pulses present in each sample, for *pos* and *neg* respectively. So, P_{Nsp} and N_{Nsp} will be vectors of M x 16 each, where M is the number of Spikes given in the data. To illustrate this, the vectors P_{Nsp} and N_{Nsp} for a certain spike action potential are:

***P**$_{Nsp}$*: *0 0 0 0 0 0 0 0 0 5 1 0 0 0 0 0*
***N**$_{Nsp}$*: *0 0 0 0 0 0 0 2 3 0 0 3 1 0 0 0*

The number of pulses generated for the first nine samples or 1.125 ms (90 x dt = 1.125ms) are '0', for P_{Nsp}. The 6th sample however, contains 5 pulses. Looking only at the vector P_{Nsp}, it can be easily inferred that the sixth sample is the place of the maxima, for the positive half of I_{input}. A similar reasoning can be extended towards N_{Nsp}. This information can be used to differentiate between the three types of neuron clusters. A new feature 'Feature1' is defined, which is a concatenation of , P_{Nsp} and N_{Nsp}. To examine the degree to which Feature1 offers 'separateness' between the three neuron clusters, PCA was performed on it, and the resultant scatter plot is seen in Fig. 7. The error can be calculated using K-Means, to be **0.0028**.

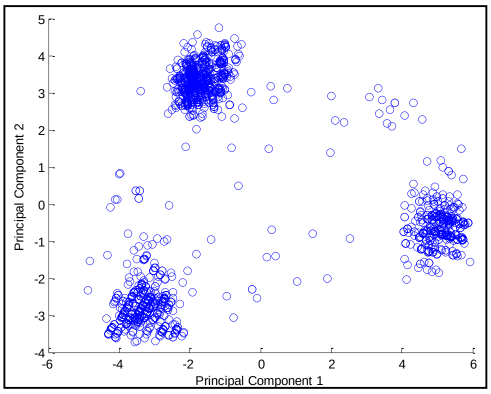


Figure 7: Scatter plot showing clustering accuracy of 'Feature1'

IV. Pulse-Based Feature Extraction; SPICE Simulation

A. Methodology

The Integrate-and-Fire neuron model can also be implemented in the analog domain, on CADENCE. The objective is to obtain results that are similar to those obtained in the software simulations. For the analog circuit, the task of counting the pulses every 125 µs is performed by a Counter, and the task of storing the data is performed by the Shift Register network.

An overall schematic of the circuit can be seen in Fig. 8. It consists of functional blocks, namely the input ‘spike voltage’ source, class B amplifier, Integrate-and-Fire circuit, Counter, and Shift Register Network. The circuit works exactly like the software implementation described previously: The voltage spike is first converted to a current I_{input}, which is separated into positive and negative by a Class B amplifier.

The segregated currents are then fed independently to Integrate-and-Fire neuron circuits. The pulse outputs from both positive and negative currents then go through a counter that counts the number of positive or negative pulses every 125 µs. This ‘refresh’ function is controlled by the ‘Reset’ signal (Fig.8). The 4-bit counter outputs then go through the Shift Register Network, and the final outputs start appearing after 2ms. Note that though the Shift registers have to be synchronous with the counter ‘Reset’, it is imperative to reset the counters after a certain delay, implemented in the form of inverters (Fig.8). This is in accordance with the regular setup time constraints.

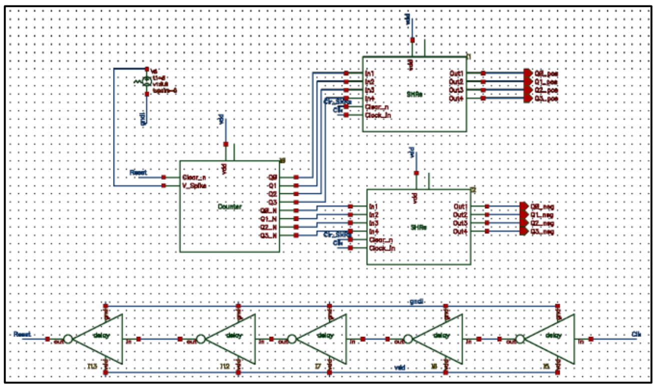


Figure 8: Feature-Extraction Circuit schematic implemented on CADENCE

B. *Results*

The pulse outputs from the Integrate-and-Fire neuron for the positive and negative halves of current I_input can be observed in Fig.9. Fig.10 shows the counter outputs obtained for the positive half of I_{input}, ‘pos’. The 10th sample (1.125-1.250 µs) counts the number of pulses as ‘8’ and the 11th sample (1.250-1.375 µs) counts ‘9’.

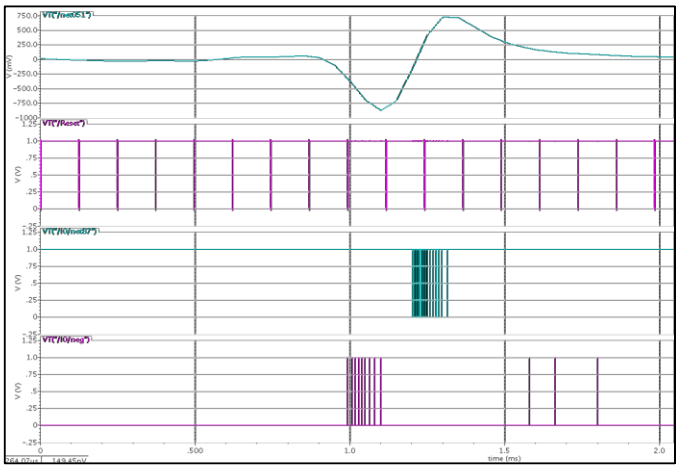


Figure 9: Spice simulation showing Spike (green), Reset/ Clear_n (pink), pulse outputs from positive (cyan) and negative (violet) I_{input}.

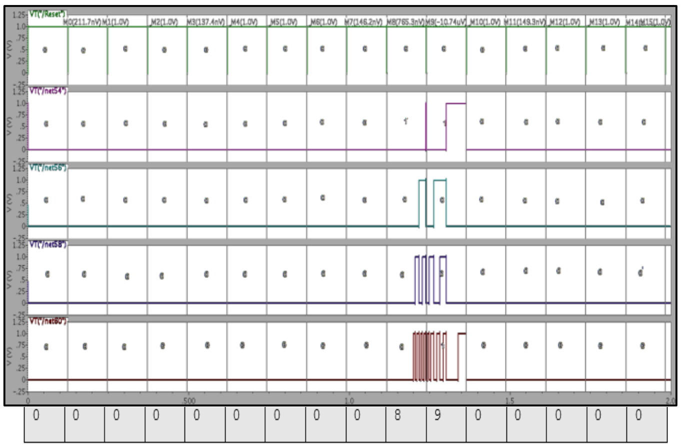


Figure 10: Counter counting pulse outputs for positive I_{input}. Between 2 successive resets (1.125µs -1.250µs), the number of positive pulse outputs are ‘8’. This can be verified by the counter reading: [Q3 Q2 Q1 Q0] = [1 0 0 0]

V. Matching Gains From Software And Spice Simulations

A. *Results and Discussion*

The visible similarity between the pulse outputs obtained from the software and spice simulations (Fig. 11) can be proved quantitatively, by calculating and comparing the firing rate of pulse outputs, or Gain ‘G’ for each. This realization is extremely useful, since the feature proposed programmatically in Section III can now be implemented in the analog domain using Spice.

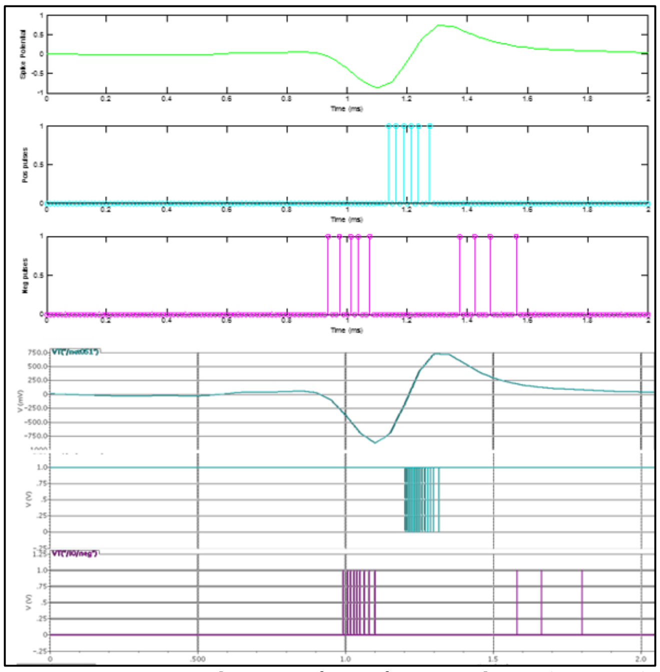


Figure 11: Comparing pulse outputs from software simulations (MATLAB) with Spice simulations (CADENCE)

For the MATLAB implementation, G is calculated by polyfitting the Pulse Frequency vs 'I_{input}' curve, where I_{input} is varied by performing a DC sweep (Fig. 12). This value is calculated to be **6.9465 x 10^{12} Hz/A**. For the analog circuit, Gain 'G' of the Integrate-and-Fire neuron is defined as:

$$Gain = \frac{Firing\ rate\ of\ pulse\ outputs}{I_{input}}$$
(2)

If a ramp function is used as the spike input voltage, the resultant I_{input} is constant, and the pulse outputs obtained are separated evenly. The inverse of this 'time period' would then give P_F (Fig.13). G can be calculated to be **4.072 x 10^{12} Hz/A.**

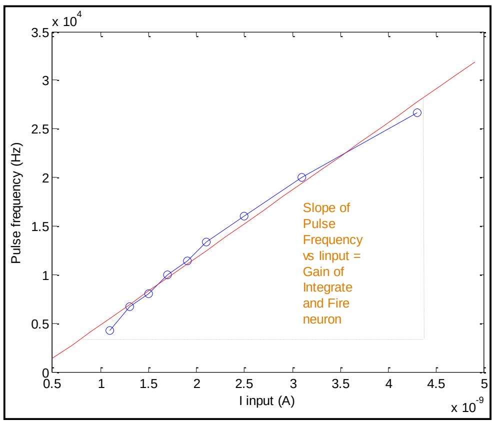


Figure 12: Calculating Gain of Integrate-and-Fire neuron model from slope of 'polyfit' P_F vs I_{input}

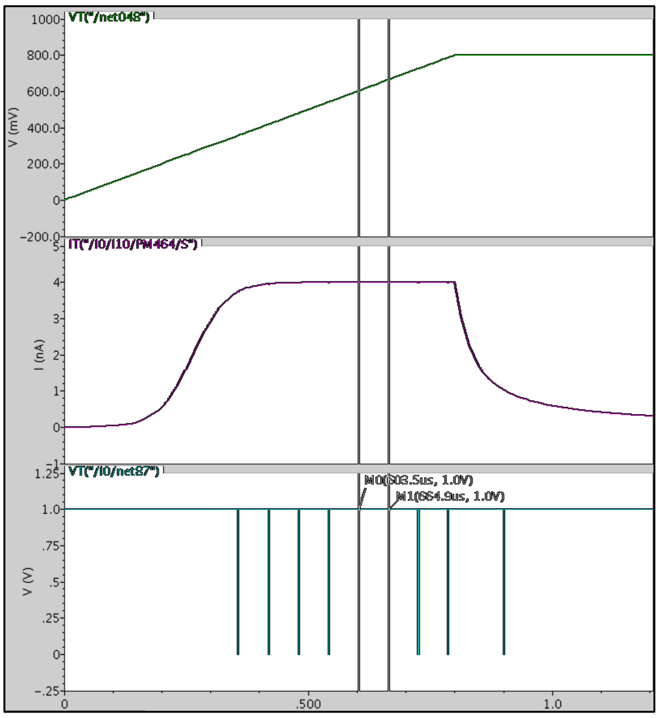


Figure 13: Spice simulation showing ramp function (green), resultant constant current generated I_input (pink) and the pulse outputs from Integrate-and-Fire circuit (cyan).

CONCLUSION

The idea of using electronics to communicate information between brain cells and prosthetics has gained popularity in recent years. Future research in this area relies heavily on the accuracy of identification of a neural signal with its source neuron. Certain problems still persist: overlapping spikes, supervised classification methods etc. *The biggest issue is the lack of availability of a 'ground truth': the actual number of total neurons, and the absolute correct identities of neural signals.* For the purpose of classification, distinguishing between shapes of spike potentials belonging to different clusters may prove to be very useful; i.e. the maxima and minima, the height and width of the peaks and so on. This idea was taken up by some researchers who thought of several features, most of them based on first and second derivatives of the Spike Action Potentials. The classification error obtained by using these methods was compared with that from a reference standard method, PCA. The results showed that for datasets of higher difficulty levels, derivative-based features showed better performance as compared to PCA [6]. This research also proved the same and reiterated the theory proposed by the paper. A feature-extraction based Circuit was designed to implement a novel bio-inspired feature proposed by the authors. This was done in two ways: first, the circuit was programmed functionally on software (MATLAB), and then the analog circuit was designed at the component level on Spice (CADENCE). In both the implementations, the input Spike potentials used produced consistent results, thus allowing for the successful completion of the project.

REFERENCES

[1] Rodrigo Quian Quiroga (2007) Spike sorting. Scholarpedia, 2(12):358

[2] Buzsaki G (2004) Large-scale recording of neuronal ensembles. Nature Neuroscience 7:446-451.

[3] Harris KD (2005) Neural signatures of cell assembly organization. Nat Rev Neurosci 6:399-407.

[4] R. Stufflebeam, "Neurons, Synapses, Action Potentials, and Neurotransmission," in *Consortium on Cognitive Science Instruction*, 2008.

[5] R.Q. Quiroga. (2004). *Unsupervised spike detection and sorting* [Online].Available: http://www.vis.caltech.edu/~rodri/Wave_clus/Wave_clus_home.htm

[6] S.E. Paraskevopoulou, S.Y. Barsakcioglu, M.R. Saberi, A. Eftekhar, and T.G. Constandinou, "Feature extraction using first and second derivative extrema (FSDE) for real-time and hardware-efficient spike sorting," Journal of Neuroscience methods 215 (2013) 29– 37, 2013.

[7] Gerstner and Kistler, "Integrate-and-Fire model," in *Spiking Neuron Models. Single Neurons, Populations, Plasticity* , Cambridge University Press,2002.