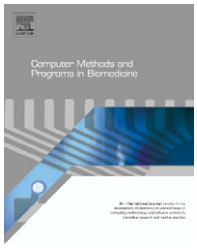
COMPUTER METHODS AND PROGRAMS IN BIOMEDICINE 91 (2008) 232–244

journal homepage: www.intl.elsevierhealth.com/journals/cmpb



Performance evaluation of PCA-based spike sorting algorithms

Dimitrios A. Adamos, Efstratios K. Kosmidis, George Theophilidis*

Laboratory of Animal Physiology, School of Biology, Aristotle University of Thessaloniki, 54 124 Thessaloniki, Greece

ARTICLE INFO

Article history:
Received 12 August 2007
Received in revised form
8 April 2008
Accepted 28 April 2008

Keywords:
Spike sorting
PCA
Clustering
Noise
Single-electrode extracellular recordings

ABSTRACT

Deciphering the electrical activity of individual neurons from multi-unit noisy recordings is critical for understanding complex neural systems. A widely used spike sorting algorithm is being evaluated for single-electrode nerve trunk recordings. The algorithm is based on principal component analysis (PCA) for spike feature extraction. In the neuroscience literature it is generally assumed that the use of the first two or most commonly three principal components is sufficient. We estimate the optimum PCA-based feature space by evaluating the algorithm's performance on simulated series of action potentials. A number of modifications are made to the open source nev2lkit software to enable systematic investigation of the parameter space. We introduce a new metric to define clustering error considering over-clustering more favorable than under-clustering as proposed by experimentalists for our data. Both the program patch and the metric are available online. Correlated and white Gaussian noise processes are superimposed to account for biological and artificial jitter in the recordings. We report that the employment of more than three principal components is in general beneficial for all noise cases considered. Finally, we apply our results to experimental data and verify that the sorting process with four principal components is in agreement with a panel of electrophysiology experts.

1. Introduction

Extracellular recordings of spontaneous nerve activity using either hook or suction electrodes is a common practice for a number of electrophysiological experiments providing valuable information concerning peripheral and central nervous system physiology of vertebrates and invertebrates [1–6]. Extracellular electrodes record voltage potentials representing the activity of an unknown number of activated axons which may serve different functions. It is generally assumed that neurons encode information into series of action potentials (AP), their spike trains, so there is a special interest in reconstructing the waveform of individual neurons from the recorded trace. The procedure of proper assignment of spikes to neurons, in order to draw inferences from neural recordings, is referred to as neural spike sorting. Spike sorting from nerve activity is based on the assumption that the APs of a neuron have the same size and shape as they depend mainly on the axon's diameter and its distance from the electrode. The experimentalist has to identify the number of neurons from the recorded trace and classify each action potential into separate spike trains in a time consuming procedure that grows with the number of axons. The quality of spike sorting depends on the researcher's experience and his objective judgment. Consequently, a significant variability in human spike sorting performance has been noted [7]. Over the last decade, a considerable amount of research has been devoted to computer aided spike sorting

* *Corresponding author*. Tel.: +30 2310 998261; fax: +30 2310 998269.
E-mail addresses: dadam@bio.auth.gr (D.A. Adamos), kosmidef@bio.auth.gr (E.K. Kosmidis), theophil@bio.auth.gr (G. Theophilidis).
0169-2607/$ – see front matter © 2008 Elsevier Ireland Ltd. All rights reserved.
doi:10.1016/j.cmpb.2008.04.011

which is today an indispensable tool in neuroscience research [7–9,11–21].

A generic algorithm for spike sorting, implemented in popular commercial and open source titles, is as follows: spikes are extracted initially from the continuously extracellular recorded signal and a spike vector is created. The classification of detected spikes into multiple groups of neurons is based on spike shape characterization. In order to reduce the dimensionality of the data, features of the shape are selected to be used to represent the most prominent dynamics of spike waveforms. In the final step, clustering techniques are used to achieve the best cluster separation decision making.

Principal component analysis (PCA) is a powerful method employed to automatically select features and use them to create feature vectors [22]. PCA seeks an ordered set of orthogonal basis vectors, the principal components, which capture the directions in the data of largest variation [18]. A smaller subspace created by some of the initial principal vectors is then used to make an approximate projection of the data. In this projection, clusters of different units in the data, corresponding to separate neurons, are revealed. It has been argued that the use of the first two or more commonly three principal components is sufficient to accurately describe the spike [8,10,14–19,23–30]. A very popular choice for the final step is the expectation–maximization (EM) clustering algorithm [31]. The EM algorithm is an ideal candidate for solving parameter estimation problems. It computes probabilities of cluster memberships based on one or more probability distributions while the goal is to maximize the overall probability or likelihood of the final data.

Typical problems in spike sorting are the presence of noise and spike overlaps. Biological noise and noise from the recording devices may introduce problems in spike detection and in spike classification. With noise, similar APs belonging to different neurons may appear the same or APs belonging to the same neuron may appear different. Spike overlaps occur when two or more neurons fire simultaneously or almost simultaneously and produce APs of significant size. Depending on when the peaks and dips of the APs occur the size and shape of the resulting trace will vary. Another inherent weakness in spike sorting is the *a priori* ignorance of the number of active neurons present in the recorded trace. The *a posteriori* supervision of the classification results by the researchers and their decision making on whether more or less clusters should be considered introduce the human error factor in the process.

In this paper we empirically estimate the size of PCA-based feature space in a typical semi-automatic spike sorting approach. Statistical approaches to estimate this subspace dimension in a blind fashion also exist in the literature (for a comparative study see in [32]). We apply the PCA–EM combination to simulated spike trains representing single electrode records from the neural cord of the beetle *Tenebrio molitor*. These neurons display spontaneous activity and multiple Single Fiber Action Potentials (SFAP) can be seen on the trace. We evaluate the optimum volume of principal components that participate in the spike sorting approach under different types and levels of background noise. This is possible based on our *a priori* knowledge of the number of simulated neurons, the exact occurrence timings and overlaps of the APs generated by each one of them. Finally, we apply the method to our experimental data and compare the results with those obtained by a group of neurophysiology experts.

The remainder of this paper is organized as follows: in Section 2, we give the background mathematics on the proposed simulation method and the part of the spike sorting methodology that we exploited during the evaluation procedure; in Section 3, we describe the detailed implementation of our work; and in Section 4, we present the results derived from the evaluation of the spike sorting method on the simulated and experimental data.

2. Background mathematics

2.1. Single fiber action potential model

The model for SFAP is a damped sinusoid as suggested in [33]:

$$f(t) = A \sin\left(\frac{t}{\tau_1}\right) e^{-t/\tau^2} \quad (1)$$

where A, τ_1 and τ_2 are parameters that determine the amplitude, rising phase rate and the total duration of each SFAP, respectively. Spike, AP and SFAP will be used interchangeably throughout this text. A small amount of bound-limited, uniformly distributed jitter was added to all three parameters to account for naturally occurring variability in spike shape of the same axon.

A dead time Poisson process (DTPP) with rate λ, defined for each unit separately, and absolute refractory period $\Delta = 2.5$ ms has been used to generate spike interval times for each unit [34–36].

2.2. Principal component analysis

In PCA a spike vector is constructed in an m-dimensional space, where m is the number of measurement types. Forming an $n \times m$ matrix of values, each of the n rows represents an object that can be regarded as an m-dimensional vector, a row vector in R^m. In R^m, PCA searches for the best-fitting linear combined set of orthogonal axes to replace the initial set of m axes in this space. The idea behind this step is to find a set of $m' < m$ principal axes allowing the objects to be adequately characterized on a smaller (m'-dimensional) space, while the m–m' dimensions may be ignored as describing noise. In PCA, the projections of points on the axis sought for, need to be as elongated as possible, i.e. the variance of the projections needs to be as great as possible. The eigenvectors associated with the m' largest eigenvalues yield the best-fitting m'-dimensional subspace of R^m. For an excellent review on PCA and its applications see in [37].

3. Methodology

3.1. Spike train generation

In order to evaluate the algorithm's performance in more realistic situations we have modeled series of randomly distributed action potentials with different characteristics (A, τ_1

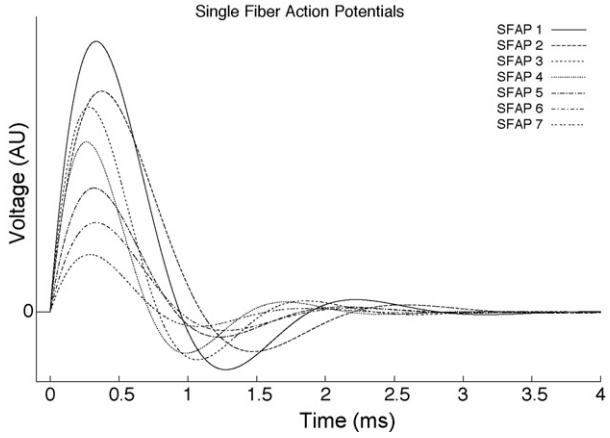


Fig. 1 – Overlap of the seven single fiber actions potentials used in the simulations.

and τ_2) rather than simply generating the waveforms and aligning them to also test the spike extraction step. For each unit, Eq. (1) is altered to

$$f(t) = A \sin\left(\frac{t - t_0}{\tau_1}\right) e^{(t_0 - t)/\tau^2} \quad (2)$$

where t_0 is the action potential triggering time defined by the DTPP for each fiber independently.

In the simulations, $N = 7$ different units have been described. The resulting SFAPs are superimposed in Fig. 1 and their parameters are listed in Table 1.

The traces of all units were added with unitary weights to produce the "recorded trace". White Gaussian and correlated stochastic processes have been added to approximate background activity. White Gaussian noise is widely used in theoretical studies concerning neuronal function [38]. Myelinated axons are considered electrically isolated and have no synaptic interactions within a nerve; thus, their activity is largely independent. In theory, background noise in hook or suction electrode recordings from nerve trunks may be approximated by a Gaussian process. In practice however, ephaptic interactions and the use of filters and digitization according to experimental needs introduce various levels of correlations. Correlated noise is a more realistic choice to describe background neuronal activity and was generated by a dynamical Ornstein–Uhlenbeck (OU) process following the equation:

$$X_{t+dt} = X_t - \frac{X_t}{\tau} dt + dW_t$$

where τ is the time constant of the process, dt is the simulation time step and W_t denotes a Wiener process. The simulation time step was 0.001 ms and the sampling period was 0.05 ms. In order to study the effects of noise amplitude on the performance of PCA, the following procedure was followed: fifteen different simulated traces were generated for statistical purposes using the spike train generation methodology previously described. Single OU and white Gaussian noise implementations have been generated, scaled to yield amplitudes (σ) from 0.05 to 3 in steps of 0.05 and linearly added to the simulated recording trace. Signal-to-noise ratio (SNR) is the average power ratio between a signal and the background noise.

$$\mathrm{SNR\,(dB)} = 10 \log\left(\frac{P_{signal}}{P_{noise}}\right) = 10 \log\left(\frac{mean^2_{signal} + \sigma^2_{signal}}{mean^2_{noise} + \sigma^2_{noise}}\right)$$

Practically, the mean values of the signal and noise traces were very close to zero so the above equation was simplified to:

$$\mathrm{SNR\,(dB)} = 20 \log\left(\frac{\sigma_{signal}}{\sigma_{noise}}\right)$$

A total of 15 × 6 single recording-like simulated spike trains resulted and were subject to the spike sorting methodology in a semi-automatic way for a different number of principal components each time. The entry level for principal components was two and was repeated until seven components were considered. The whole process was repeated three times, one for Gaussian noise and two for correlated noise with τ of 0.01 and 0.1 respectively. The specific values of τ were selected so the resulted traces resembled the experimental traces and did not introduce large non-stationarities. The mean SNR for the 15 traces at the six different levels of noise were the following (all values referred in dB): 19.9 ($\sigma = 0.05$), 14 ($\sigma = 0.1$), 10.7 ($\sigma = 0.15$), 8.5 ($\sigma = 0.2$), 6.9 ($\sigma = 0.25$) and 5.7 ($\sigma = 0.3$).

The detection of spike overlaps was made at the generation time of a spike and was based on the interspike intervals with the last spikes fired by every other fiber. The absolute values of these intervals were compared with spike duration (3.5 ms). In the event of shorter intervals, an index was increased accordingly. In this step, the expected number of spikes for each fiber was calculated by subtracting all spike overlaps with other fibers from the total number of spikes fired by the specific fiber. At the same time, the occurrence times were sorted for further "offline" analysis.

The simulation program was written in C and run on a Pentium–Gentoo linux-based machine.

Table 1 – Parameters of the seven single fiber actions potentials used in the simulations

Fiber	Amplitude (A)	τ_1(ms)	τ_2(ms)	Rate (Hz)
1	15	0.30	0.61	2
2	13	0.35	0.64	4
3	11	0.25	0.54	3
4	9	0.23	0.51	4
5	7	0.29	0.57	3
6	5	0.30	0.60	4
7	3	0.25	0.57	3
Jitter	±0.001	±0.001	±0.005	

3.2. *Spike sorting*

For the implementation of the spike sorting algorithm we utilized the open-source software nev2lkit [29], a preprocessor for the analysis of intracellular or extracellular neuronal recordings that was developed under Cortivis project [39]. Nev2lkit employs PCA feature analysis of the detected spikes, which are extracted from a continuous recording. During

the spike extraction process, the root-mean-square of the continuously extracellular recorded signal is estimated. Based on this value, upper and lower thresholds are set (3× RMS) and events are extracted based on a fixed-length data window (1 ms, 2 ms and 4 ms).

We have altered some parts of the program source code in order to provide extended functionality to the feature extraction procedure. The ability to change the length of the data window following a 10^{-1} ms accuracy was added. The fixed values (1 ms, 2 ms and 4 ms) prevented us from adopting in variable-length spiking activity and inserted spike overlap artifacts in the PCA process. After this modification, the RMS-based spike extraction module was able to correctly discriminate between noise and signal for all noise levels in our simulations.

Furthermore, we have added the ability to compute a user defined number of principal components for spike waveform representation to the nev2lkit software, originally using the first three by default. This number can now be selected between 2 and 7 ($m' \in [2,7]$). Also, the percentage variance explained by this subspace is computed and displayed in the program's graphical interface. The PCA was carried out on the correlation matrix.

Following the feature extraction procedure, the clustering of the data is performed. Nev2lkit uses klustakwik, a program for unsupervised classification of multidimensional continuous data [8], based on the classification expectation maximization algorithm [31]. It consists of a semi-automatic process followed by examination and reassignment by a human operator [21]. This means that the user has to examine the output of the clustering process each time it is performed, i.e. the number of clusters-neurons found. The user can then accept this output, or decide to alter the definable parameters of the clustering process and try again. Tunable parameters of this field are the "maximum number of clusters" and the "penalty mix". The first defines the maximum possible clusters that the process can produce while the second inserts Bayesian information in the clustering process, as a penalty indicator for more clusters. In this step, the functionality of displaying the number of spikes assigned to each cluster, as an additional output of the clustering process, was added to the program.

3.3. *Evaluation process*

During the evaluation process, we attempt to quantify the success of the spike sorting procedure. We examine the output of the spike sorting procedure based on the a priori knowledge of the input. As previously mentioned, the ability to directly compute the number of spikes assigned to a cluster has been added to the nev2lkit software. For each cluster, this number is compared with the number of spikes fired by the corresponding neuron minus the number of overlaps.

We introduce a new cluster error definition to favor over-clustering over under-clustering errors. Our neurophysiology experts pointed out that in the manual human-employed spike sorting process, over-clustering can often be addressed in a less time-consuming way than under-clustering. This is possible using the cluster-merging functionality that most manual spike sorting software titles offer. On the other hand, if under-clustering cannot be addressed by a fine-tuning of the automated clustering process, it can only be addressed by a repeated manual assignment from the experimentalist. In this case, the experimentalist has to separate each spike of the merged cluster based on his subjective judgment of the spike's shape.

There are two important measures when one evaluates a spike sorting method: (a) the number of clusters being decided by the spike sorting procedure versus the actual number of clusters and (b) the number of spikes assigned in each cluster and the corresponding false positive–negative spikes versus the number of spikes that are expected to be classified under each cluster.

If we consider a number of e expected clusters, with n_e being a member of this cluster set ($n_e = 1, 2, \ldots, e$), and a number of c computed clusters by the spike sorting procedure, with n_c being a member of this cluster set ($n_c = 1, 2, \ldots, c$), then we may define the following:

(a) for the e number of clusters, we consider $N(n_e)$ as the number of neurons expected to be classified per cluster, $S(n_e)$ as the number of spikes expected to be classified per cluster and $S_{NOISE}(e)$ as the number of spikes expected to be marked as overlaps or "noise".
(b) for the c number of clusters, we consider $N(n_c)$ as the number of neurons classified per cluster, $S(n_c)$ as the number of spikes classified per cluster and $S_{NOISE}(c)$ as the number of spikes marked as overlaps or "noise" by the classification procedure. We also consider $Fp(n_e)$ as the number of false-positive spikes to n_e that are part of $S(n_c)$ but are seen as "outliers" in n_c either because their corresponding group of spikes belong to n_c' ($n_c' \neq n_c$) or because they originally belonged to $S_{NOISE}(e)$ and they should have been classified as part of $S_{NOISE}(c)$. Accordingly, $Fn(n_e)$ is considered as the number of false-negative spikes of n_e that originally belonged to $S(n_e)$ and should have been classified as part of $S(n_c)$.

For the total spike sorting procedure the following is true:

$$\sum_{n_e=1}^{e} Fn(n_e) = \sum_{n_e=1}^{e} Fp(n_e) + (S_{NOISE}(c) - S_{NOISE}(e)) = \sum_{n_e=1}^{e} Fp(n_e) + \Delta S_{NOISE} \quad (3)$$

Also, since $N(n_e) = 1$, we expect to find one neuron classified under each of the e clusters, and

$$\sum_{n_e=1}^{e} S(n_e) + S_{NOISE}(e) \leq n$$

as the total number of spikes expected to be classified by the spike sorting process can be less or equal to the total n number of SFAPs that were subject to this procedure, due to potential overlaps. This happens when two or more SFAPs overlap, two or more spikes are missing from the per cluster expected classification while only one spike is added to the "noise"

cluster. It is obvious that equality is achieved in absence of overlaps.

Regarding the corresponding c computed values, in order to draw mathematical inferences we have to distinguish three separate cases, depending on c.

(1) $c = e$

In the first and simplest case, every neuron has been properly assigned to a cluster ($N(n_c) = 1$, $n_c = 1, 2, \ldots, c$), which means the number of computed clusters will equal the number of expected clusters, or $c = e$. The issues to be addressed in this case are the number of spikes assigned per cluster and the number of corresponding false-positive spikes. In other words, if each neuron's whole activity is classified under the same cluster, or if some spikes are missing or even if some other spiking activity is classified under the same cluster.

Generally a per cluster percentage error can be defined as:

$$\text{cluster\_error}(n_e) = \frac{\text{Fn}(n_e) + \text{Fp}(n_e)}{\sum_{n_e=1}^{e} S(n_e) + S_{\text{NOISE}}(e)} \times 100\%$$

By definition:

$$S(n_e) - \text{Fn}(n_e) + \text{Fp}(n_e) = S(n_c) \Rightarrow \text{Fn}(n_e) = S(n_e) - (S(n_c) - \text{Fp}(n_e)) = S(n_e) - S(n_c) + \text{Fp}(n_e) \quad (4)$$

Taking into consideration the above equation and since all the c-related values are computed by the spike sorting process, the per cluster percentage error can also be written as:

$$\text{cluster\_error}(n_e) = \frac{S(n_e) - S(n_c) + 2\text{Fp}(n_e)}{\sum_{n_e=1}^{e} S(n_e) + S_{\text{NOISE}}(e)} \times 100\%$$

$n_c, n_e = 1, 2, \ldots, c$ (n_e corresponds with n_c based on the cluster under consideration).

Finally, the overall per spike train percentage error is:

$$\text{error}_{\text{spike\_train}} = \sum_{n_e=1}^{e} \text{cluster\_error}(n_e)$$

(2) $c < e$

The second case is defined by $c < e$; this means that the number of computed clusters is less than those expected which can only happen when the spike sorting process assigns neuronal activity of more than one neurons to the same cluster ($N(n_c) \geq 2$, for at least one cluster). Visually it can be observed when at least one cluster carries spikes whose shapes match the shapes of two or more corresponding neurons' SFAPs. This case could be described as an "under-clustering" case and before the corresponding errors are defined, a few more assumptions should be made. If c is the number of clusters computed by the spike sorting process, we consider c^{p} as the number of clusters properly computed, with n_c^{p} being a member of this cluster set ($n_c^{\text{p}} = 1, 2, \ldots, c^{\text{p}}$ and $c^{\text{p}} \leq c - 1$, since in this case the number of properly computed clusters can only be less than the clusters computed by the spike sorting process). We also consider c^{u} as the number of under-clustering clusters (i.e. the total number of clusters with assigned spikes of more than one neuron), with n_c^{u} being a member of this cluster set ($n_c^{\text{u}} = 1, 2, \ldots, c^{\text{u}}$). Then the following are true:

$$c^{\text{p}} + c^{\text{u}} = c \quad (5)$$

$$\sum_{n_c^{\text{p}}=1}^{c^{\text{p}}} N(n_c^{\text{p}}) + \sum_{n_c^{\text{u}}=1}^{c^{\text{u}}} N(n_c^{\text{u}}) = e \quad (6)$$

$$\sum_{n_c^{\text{p}}=1}^{c^{\text{p}}} N(n_c^{\text{p}}) = c^{\text{p}}, \quad \text{since } N(n_c^{\text{p}}) = 1 \quad (7)$$

Eq. (5) states that the sum of the number of clusters properly computed with the number of under-clustered clusters equals the total number of clusters computed by the spike sorting process. Eq. (6) states that the sum of the total number of neurons classified per properly computed cluster plus the sum of the total number of neurons classified per *under-clustered* cluster equals the total number of clusters that are expected to be found by the spike sorting process, as computed in the spike train generation procedure. Eq. (7) is simply based on the fact that only one neuron is assigned to each of the properly computed clusters, so the sum of assigned neurons equals the number of clusters containing these neurons.

From these three equations, subtracting the first two we get:

$$\sum_{n_c^{\text{u}}=1}^{c^{\text{u}}} N(n_c^{\text{u}}) - c^{\text{u}} = e - c \quad (8)$$

If we take into consideration that

$$\frac{\sum_{n_c^{\text{u}}=1}^{c^{\text{u}}} N(n_c^{\text{u}})}{c^{\text{u}}} \geq 2 \quad (9)$$

which derives from the fact that $N(n_c^{\text{u}}) \geq 2$ (i.e. each of the n_c^{u} clusters should have at least two neurons assigned), Eqs. (8) and (9) give: $c^{\text{u}} \leq (e - c)$. At the same time Eq. (5) restricts the c^{u} range, as $c^{\text{u}} \leq c$. These restrictions define a finite set of solutions based on the fixed $(e - c)$ value and the number of clusters (c) found in the spike sorting process. This means that when the spike sorting process computes the total number of clusters c, there are only certain integer combinations of $N(n_c^{\text{u}})$ and c^{u} which confirm this equation. Depending on which of the two parts seems a more convenient estimation, using the above equation the correct value that fits the other part can be computed. According to this, Eq. (8) can have the following two expressions:

$$\sum_{n_c^{\text{u}}=1}^{c^{\text{u}}} N(n_c^{\text{u}}) = (e - c) + c^{\text{u}} \quad \text{or} \quad c^{\text{u}} = \sum_{n_c^{\text{u}}=1}^{c^{\text{u}}} N(n_c^{\text{u}}) - (e - c)$$

The corresponding percentage cluster error here is defined in two parts; for the first c^{p} clusters we have:

$$\text{cluster\_error}(n_e) = \frac{\text{Fn}(n_e) + \text{Fp}(n_e)}{\sum_{n_e=1}^{e} \text{S}(n_e) + \text{S}_{\text{NOISE}}(e)} \times 100\% \quad \text{or}$$

$$\text{cluster\_error}(n_e) = \frac{\text{S}(n_e) - \text{S}(n_c^{\text{p}}) + 2\text{Fp}(n_c^{\text{p}})}{\sum_{n_e=1}^{e} \text{S}(n_e) + \text{S}_{\text{NOISE}}(e)} \times 100\%$$

n_c^{p}, $n_e = 1, 2, \ldots, c^{\text{p}}$ (n_e corresponds with n_c^{p} based on the cluster under consideration), while for the remaining c^{u} clusters let us also consider firstly its general form:

$$\text{cluster\_error}(n_e) = \frac{\text{Fn}(n_e) + \text{Fp}(n_e)}{\sum_{n_e=1}^{e} \text{S}(n_e) + \text{S}_{\text{NOISE}}(e)} \times 100\%$$

where n_e refers to the expected value of the cluster containing each neuron ($n_e = 1, 2, \ldots, e - c^{\text{p}}$).

As shown in Appendix A, the cluster percentage error can also be written as

$$\text{cluster\_error}(n_e) = \frac{\text{S}(n_c^{\text{u}}) - \text{S}(n_e) + 2\text{Fn}(n_e)}{\sum_{n_e=1}^{e} \text{S}(n_e) + \text{S}_{\text{NOISE}}(e)} \times 100\%$$

where n_c^{u} refers to the under-clustered cluster ($n_c^{\text{u}} = 1, 2, \ldots, c^{\text{u}}$) that contains this neuron.

Again the overall per spike train percentage error is:$\text{error}_{\text{spike\_train}} = \sum_{n_e=1}^{e} \text{cluster\_error}(n_e)$

(3) $c > e$

The third case is defined by $c > e$ when the number of computed clusters is greater than those expected. Furthermore, it is referred to as "over-clustering" ($N(n_c) < 1$, for at least one cluster). We consider the following: if c is the number of clusters computed by the spike sorting process, let c^{p} be the number of clusters properly computed and n_c^{p} being a member of this cluster set ($n_c^{\text{p}} = 1, 2, \ldots, c^{\text{p}}$ and $c^{\text{p}} \le e - 1$, since in this case the number of properly computed clusters can only be less than the clusters expected by the spike sorting process), and c^{o} as the number of over-clustering clusters (i.e. the total number of clusters with assigned spikes of less than one specific neuron), with n_c^{o} being a member of this cluster set ($n_c^{\text{o}} = 1, 2, \ldots, c^{\text{o}}$). Then the following are true:

$$c^{\text{p}} + c^{\text{o}} = c \tag{10}$$

$$\sum_{n_c^{\text{p}}=1}^{c^{\text{p}}} N(n_c^{\text{p}}) + \sum_{n_c^{\text{o}}=1}^{c^{\text{o}}} N(n_c^{\text{o}}) = e \tag{11}$$

$$\sum_{n_c^{\text{p}}=1}^{c^{\text{p}}} N(n_c^{\text{p}}) = c^{\text{p}}, \quad \text{since } N(n_c^{\text{p}}) = 1 \tag{12}$$

As before, from Eqs. (10)–(12) we get:

$$c^{\text{o}} - \sum_{n_c^{\text{o}}=1}^{c^{\text{o}}} N(n_c^{\text{o}}) = c - e \tag{13}$$

and

$$\sum_{n_c^{\text{o}}=1}^{c^{\text{o}}} N(n_c^{\text{o}}) = c^{\text{o}} - (c - e) \quad \text{or} \quad c^{\text{o}} = \sum_{n_c^{\text{o}}=1}^{c^{\text{o}}} N(n_c^{\text{o}}) + (c - e)$$

By definition, in the over-clustering case each neuron that fit in the c^{o} clusters will be divided at least in two parts. This means that the total number of neurons described by the c^{o} clusters cannot be more than $c^{\text{o}}/2$:

$$\sum_{n_c^{\text{o}}=1}^{c^{\text{o}}} N(n_c^{\text{o}}) \le \frac{c^{\text{o}}}{2}$$

The above equation derives from the fact that for a particular under-clustered neuron, the sum of all $N(n_c)$ (referred to the clusters that it is divided) equals to 1. Therefore, the total number of under-clustered neurons that fit in the c^{o} clusters can be noted as $\sum_{n_c^{\text{o}}=1}^{c^{\text{o}}} N(n_c^{\text{o}})$.

So, if we take into consideration that

$$\frac{\sum_{n_c^{\text{o}}=1}^{c^{\text{o}}} N(n_c^{\text{o}})}{c^{\text{o}}} \le \frac{1}{2} \tag{14}$$

Eqs. (13) and (14) give:

$$c^{\text{o}} \le 2(c - e) \tag{15}$$

It is also true, as mentioned before, that $c^{\text{p}} \le e - 1$. From Eq. (10) this means that:

$$c^{\text{o}} \ge (c - e) + 1 \tag{16}$$

Together Eqs. (15) and (16) give:$(c - e) + 1 \le c^{\text{o}} \le 2(c - e)$ or $1 \le c^{\text{o}} - (c - e) \le (c - e)$ and by taking into account Eq. (13) we have:

$$1 \le \sum_{n_c^{\text{o}}=1}^{c^{\text{o}}} N(n_c^{\text{o}}) \le (c - e) \tag{17}$$

Eq. (17) defines a finite set of solutions that is also based on the fixed $(c - e)$ value in the spike sorting process and moreover that the volume of solutions (i.e. possible cases of under-clustered neurons) is equal to $(c - e)$.

The corresponding percentage cluster error here is defined in two parts; for the first c^{p} clusters we have:

$$\text{cluster\_error}(n_e) = \frac{\text{Fn}(n_e) + \text{Fp}(n_e)}{\sum_{n_e=1}^{e} \text{S}(n_e) + \text{S}_{\text{NOISE}}(e)} \times 100\% \quad \text{or}$$

$$\text{cluster\_error}(n_e) = \frac{\text{S}(n_e) - \text{S}(n_c^{\text{p}}) + 2\text{Fp}(n_c^{\text{p}})}{\sum_{n_e=1}^{e} \text{S}(n_e) + \text{S}_{\text{NOISE}}(e)} \times 100\%$$

n_c^{p}, $n_e = 1, 2, \ldots, c^{\text{p}}$ (n_e corresponds with n_c^{p} based on the cluster under consideration), while for the remaining c^{o} clusters let us consider firstly its general form:

$$\text{cluster\_error}(n_e) = \frac{\text{Fn}(n_e) + \text{Fp}(n_e)}{\sum_{n_e=1}^{e} \text{S}(n_e) + \text{S}_{\text{NOISE}}(e)} \times 100\%$$

As shown in Appendix B, this error takes the following form:

$$\text{cluster\_error}(n_e) = \frac{1}{j}\sum_{i=1}^{j}\left[\frac{S(n_e) - S(n_c^o(i)) + 2\text{Fp}(n_e(i))}{\sum_{n_e=1}^{e} S(n_e) + S_{\text{NOISE}}(e)}\right] \times 100\%$$

where n_e refers to the expected value of the cluster containing each neuron ($n_e = 1, 2, \ldots, e - c^p$), $i = 1, 2, \ldots, j$ and n_c^o (i) refer to the j over-clustered clusters that contain parts of this neuron. $\text{Fp}(n_e(j))$ describes the false positive spikes of n_e in respect to each of the j clusters.

Again the overall per spike train percentage error is:

$$\text{error}_{\text{spike\_train}} = \sum_{n_e=1}^{e}\text{cluster\_error}(n_e)$$

Online versions of the derived parametric equations for under- and over-clustering can be found in http://neurobot.bio.auth.gr/ss_clust_model.php.

In each step during the evaluation process, the best cluster fit was achieved by tuning the above-mentioned clustering variables. The best-fit criterion followed a simple algorithm: the number of clusters found by the clustering process ought to be the same with the number of neurons activated in the spike train or at least as many as them and not less, if possible. Fewer clusters found would lead to higher error rates, as defined by the under-clustering case above. This case was considered only when it was the only available outcome of the clustering process.

The implementation of the spike sorting methodology for our simulated spike trains used a default time-window defining the spike waveform length at 3.7 ms. Based on this window, the spike waveforms were extracted from the spike trains and inserted into the PCA process. Under high levels of noise, a few cases occurred where under-clustering was unavoidable no-matter the number of principal components engaged (in the [2–7] range). In these cases, the time-window was decreased in steps of 0.1 ms until under-clustering was avoided for at least one set of principal components. The evaluation process was then performed for this value of window duration.

In the last step and after the cluster decision was taken, the number of spikes for every cluster was estimated. For these cluster memberships, their spike timing information was directly compared to the firing activity of their corresponding neuron in order to estimate the number of false positive/negative spikes and the classification error rates were computed.

3.4. *Experimental data*

We wish to model the spontaneous activity of respiratory motoneurons recorded with "hook" electrodes *in vitro* from the right nerve of the 3rd abdominal ganglion of the beetle *Tenebrio molitor* as described in [5]. An exemplary experimental trace is shown in Fig. 2A where SFAPs with different amplitudes correspond to different axons within the nerve trunk.

3.5. *Enhancing the ability of nev2lkit for parameter space investigation*

The following features, visualized in Fig. 4A, were added to *nev2lkit* spike sorting software, improving its performance and capabilities:

(1) *Variable length data window*: the ability of applying principal component analysis on a variable-length data window was implemented, following 0.1 ms accuracy. The data window defines the spike waveform length that is taken into account in the spike extraction process and afterwards inserted into PCA. With this feature, the software can adopt in variable-length spiking activity and overcome the insertion of spike overlap artifacts.
(2) *Variability of principal components*: we have added the ability to select the number of principal components used for

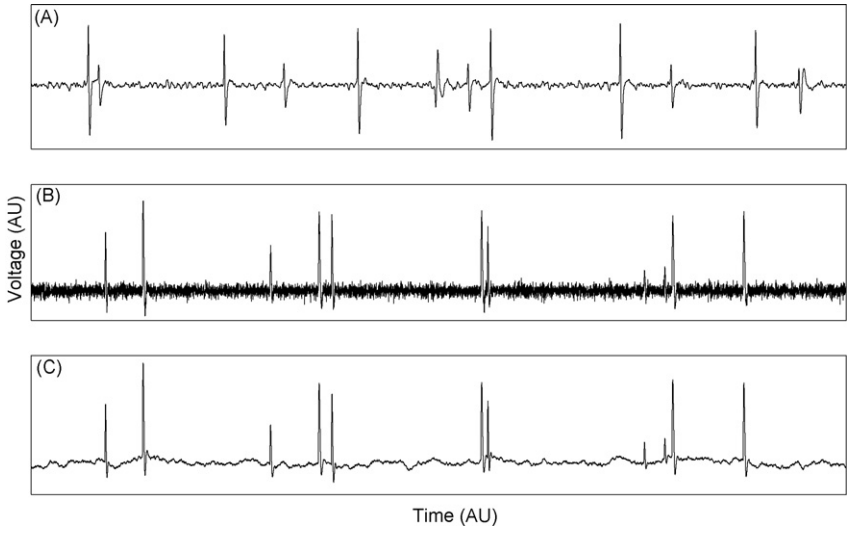


Fig. 2 – Exemplary experimental and simulated traces. (A) Experimental trace, (B) simulated spike train with Gaussian noise ($\sigma = 0.15$) and (C) simulated spike train under Ornstein–Uhlenbeck noise ($\tau = 0.01$ and $\sigma = 0.15$). For all panels, abscissa: time (au), ordinate: voltage (au).

spike waveform representation by PCA. This number can now be selected between 2 and 7.

(3) *Calculation and display of variance*: in spike waveform representation by the principal components, the percentage variance explained by that subspace is computed and displayed in the program's graphical interface.

(4) *Spike count in each cluster*: besides the total count of the detected spikes by the extraction process, the functionality of displaying the number of spikes assigned to each cluster (as an additional output of the clustering process) was added.

The patch including all of the above-mentioned additions to the nev2lkit software is available at http://neurobot.bio.auth.gr/nev2lkit.

4. Results

4.1. Simulation results

As mentioned in Section 2, 15 traces at 6 different noise levels and 3 different noise cases were subjected to the PCA–EM algorithm in a semi-automatic way for a different number of principal components each time. We show realizations of simulated traces with Gaussian noise and with OU noise in Fig. 2B and C, respectively. An exemplary table of results for one trace (Gaussian noise, $\sigma = 0.25$) is shown in Table 2. For each SFAP ($N = 7$, first column) the number of spikes generated (second column) and the ones expected to be detected (third column) are shown. The number of spikes that took part in overlaps is characterized as noise and it is computed separately. The following pairs of columns contain the output of the semi-automatic spike sorting process for each fiber along with the accompanying error percentage. These values are estimated separately for all the different principal component sets, from the first two up to the first seven. Finally, in each case, the amount of variance explained by the principal components that are taken into account is shown, while the overall error percentage is also computed. Table 2 includes an under-clustering case under the three principal components column, where the spikes from fibers 5 and 6 are shown to have been classified under the same cluster. According to the methodology described above, this leads to high error rates as it can be verified in the corresponding column.

The different sets of principal components (PC) used in the spike sorting process along with the different levels of noise and the mean error rates define 3D planes displayed in Fig. 3 for the three noise types. The Gaussian noise case is shown is row A and the OU noise for τ 0.01 and 0.1 in rows B and C, respectively. We only illustrate noise levels from $\sigma = 0.15$ to $\sigma = 0.3$ for clarity purposes.

In the Gaussian noise case the mean error rates are negligible (around 1%) for low levels of noise ($\sigma \leq 0.1$) when the first three or more PCs are used. As the noise level increases the use of the first two PCs leads to the highest error rates. Including the 3rd PC significantly increases the error rates for noise levels higher than $\sigma = 0.2$. The only case where the error rate is kept under 2%, under all noise levels, is when the first four PCs are considered. All other sets of principal components, including less and more than four, result in significant higher error rates. We verified the statistical significance of our conclusion by applying a statistical *t*-test on the error-rate values of the 15 sample spike trains for every noise level. The results for statistical significance at the 5% level are schematically shown in Fig. 3A2. Each of the six tables in this panel compares a set of principal components indicated on top of the table with the other sets (from 2 to 7) at all noise levels. Grey boxes indicate no statistically significant difference ($p > 0.05$), while white indicate that the set of PCs under comparison produce statistically significant ($p < 0.05$) lower error levels than the set they are compared with. Black boxes indicate that the set of principal components under comparison produce statistically significant ($p < 0.05$) higher error levels than the set they are compared with. Again, the use of four PCs provides the best results on average, and especially for higher noise levels, as the corresponding table contains no black boxes. All other sets contain at least one black box, they produce higher error rates that is, when compared to 4 at some noise level. Since different sets of principal components insert different levels of variance of the initial data set into the spike sorting process, it can be inferred for this case that the first four prin-

Table 2 – Exemplary analysis table for one artificial trace

			PC 2	Var. 42%		PC 3	Var. 55%		PC 4	Var. 65%		PC 5	Var. 71%		PC 6	Var. 77%		PC 7	Var. 80%	
A/A	Originated	To be detected	Spikes	Fp		Spikes	Fp		Spikes	Fp		Spikes	Fp		Spikes	Fp		Spikes	Fp	
1	64	57	85	1	0,15%	58	1	0,15%	58	1	0,15%	58	1	0,15%	55	1	0,58%	57	1	0,29%
2	143	123	126	3	0,44%	124	1	0,15%	123	0	0,00%	123	0	0,00%	123	0	0,00%	123	0	0,00%
3	87	77	80	3	0,44%	77	2	0,58%	77	0	0,00%	78	1	0,15%	77	0	0,00%	77	0	0,00%
4	144	122	130	9	1,46%	125	3	0,44%	121	0	0,15%	121	0	0,15%	121	0	0,15%	121	0	0,15%
5	103	86	88	3	0,58%	180	1	13,68%	85	0	0,15%	84	0	0,29%	86	0	0,00%	86	1	0,29%
6	114	95	78	1	1,46%			12,37%	97	2	0,29%	98	3	0,44%	96	1	0,15%	96	2	0,44%
7	90	75	97	4	0,58%	78	2	0,15%	77	2	0,29%	77	2	0,29%	77	2	0,29%	77	2	0,29%
Noise		*52*	*48*	*-4*		*47*	*-5*		*52*	*0*		*51*	*-1*		*55*	*3*	*0,44%*	*53*	*1*	
Total	745	687	696	***20***	5,09%	689	***5***	27,51%	690	***5***	1,02%	690	***6***	1,46%	690	***7***	1,16%	690	***7***	1,46%

Gaussian noise with $\sigma = 0.25$ (SNR = 6.9). The spike sorting process was applied for a variable number [2–7] of principal components. In each line the number of spikes found, the false positives estimated and the percentage error per cluster are shown. In the last cell of the Fp column, the total number of false negatives per process is calculated using Eq. (4). Note that under-clustering has occurred between clusters 5 and 6, when 3 principal components were used in the spike sorting process.

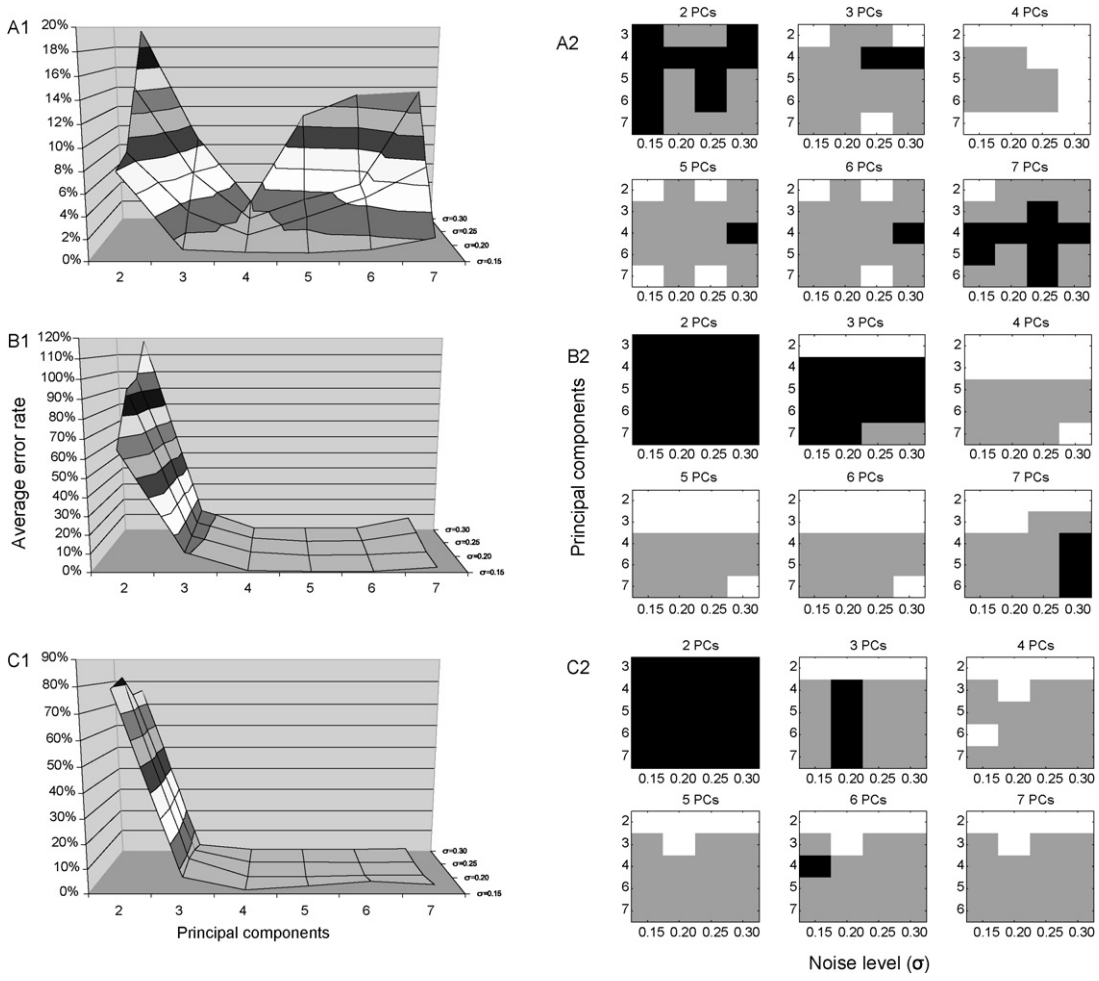


Fig. 3 – (Column 1) 3D graph of the mean error rates *vs* the set of principal components *vs* the noise level. (Column 2) Schematic representation of statistical significance of the mean error rates at the 0.05 level using *t*-test. Grey boxes indicate no statistically significant difference ($p > 0.05$). White (black) boxes indicate that the set of PCs under comparison produce statistically significant ($p < 0.05$) lower (higher) error levels than the set they are compared with. (Row A) Gaussian noise, (Row B) Ornstein–Uhlenbeck noise with $\tau = 0.01$, (Row C) Ornstein–Uhlenbeck noise with $\tau = 0.1$.

cipal components carry the optimum amount of variance for which the spike sorting algorithm reaches to more accurate results.

When correlated noise with a relative low correlation constant ($\tau = 0.01$) is considered the use of two PCs yields excessively high error levels for all noise amplitudes (panel 3B1). Taking into account the third PC dramatically improves the algorithm's performance. Consistently, 4, 5 and 6 PCs display the best performance overall as evident by the statistical comparison in panel 3B2. Increasing the number of PCs to seven has negative effects on the error rate especially for high noise levels.

A 10-fold higher correlation constant ($\tau = 0.1$, panel 3C1) provides a smoother transition. Two PCs display the highest error levels, though lower than the corresponding ones for $\tau = 0.01$, and 3 PCs are inferior to 4–7 PCs only for a specific noise amplitude ($\sigma = 0.2$, panel 3C2). Four PCs provide marginally the best overall performance while seven PCs do not suffer from high noise levels as it was the case for the lower correlation constant value.

White Gaussian noise is characterized by a power spectral density that has equal power in all bands while correlated noises present higher power in low frequencies. The effect of these types of noise on spike shape underlie the observed differences on the algorithm's performance. High frequency components override the spike when white Gaussian noise is added. On the other hand, correlated noise mainly affects the spike with low frequency components changing coarser shape characteristics. The first PC will be assigned to the direction with the largest variation, the second with the second one and so forth until the originally data is fully described. The first few PCs perform better with Gaussian noise of small amplitude as this type of noise does not significantly affect coarse characteristics of the spike, representing the largest variance. However, for large amplitudes a lot of the variance introduced by subsequent PCs is dedicated to describe noise and the error rates rise. On the other hand, the effects of correlated noise on coarse spike characteristics are more prominent; consequently, a small number of PCs is unable to correctly describe the data. Often, the variability introduced by this type

of noise is described by high-ordered PCs yielding excessively high error rates in low dimensional feature spaces. The use of subsequent PCs helps to better describe the spike shape and correctly identify the clusters. The effect of the correlation constant on spike sorting performance will depend on the interplay of the corresponding dominant frequency band and of the spike shape dynamics. In a general consideration, the use of three PCs as suggested in the neuroscience literature

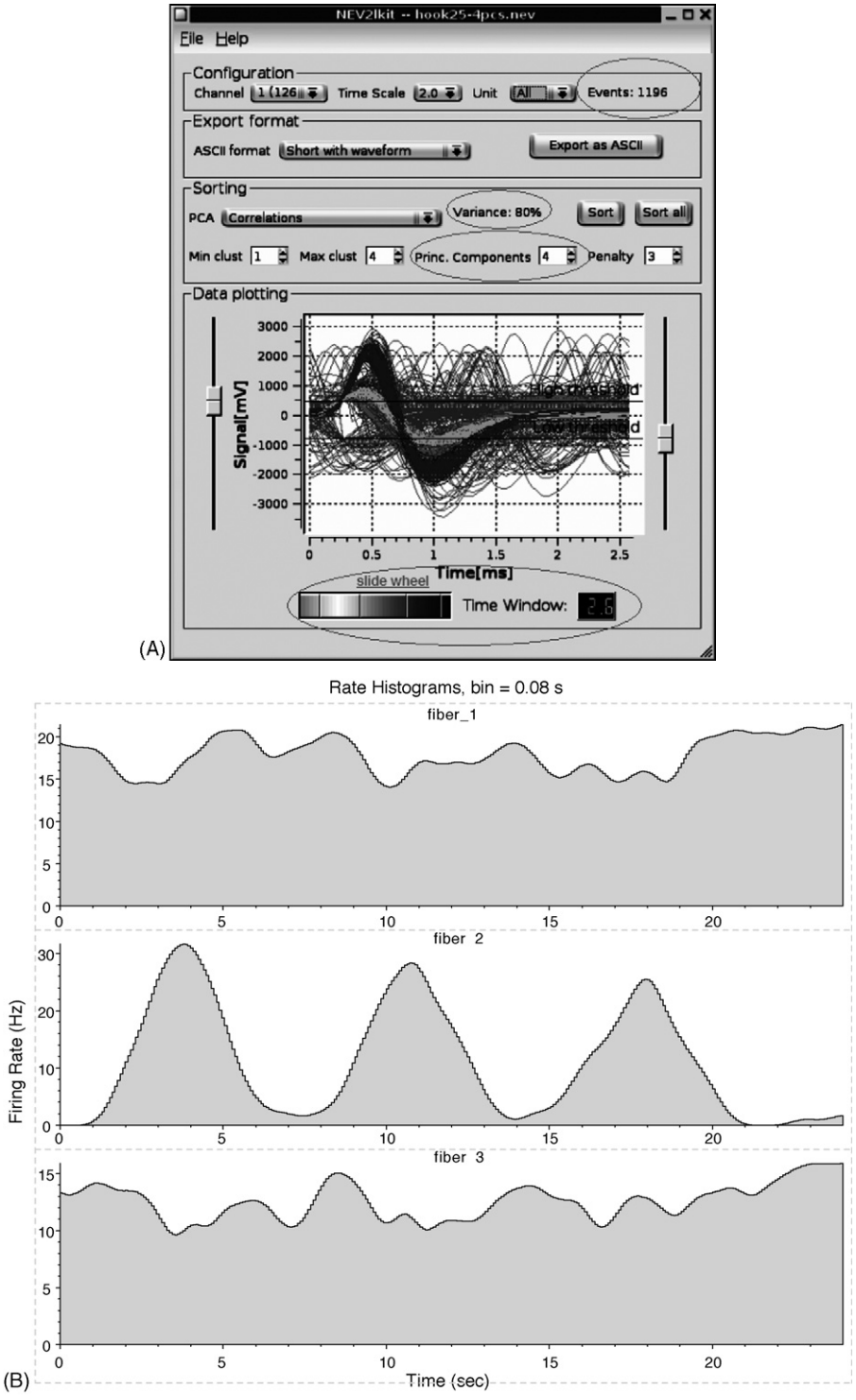


Fig. 4 – (A) An extracellular recording from the beetle's *Tenebrio molitor* peripheral nervous system was opened by the program. The time scale was initially set at 2 ms. In order to account for variable-length spiking activity we demonstrate the modification of the time-window to 2.6 ms, using the slide-wheel. In the next step, we apply PCA to the data using the first four principal components for spike representation. According to the program's calculations, the total amount of variance explained by these four principal components reaches 80%. The program also informs us that the total number of spikes is 1196. (B) The firing rates of fibers 1, 2 and 3 are shown.

is outperformed by the employment of more PCs, considering the type of data we are examining.

4.2. *Experimental results*

Following our simulation results, we applied the spike sorting methodology employing the first four principal components to our laboratory recordings. These are extracellular recordings taken from the right nerve of the 3rd abdominal ganglion, connecting the ganglion with the respiratory muscles of the *T. molitor* beetle. The recorded activity is the summation of the electrical activity of a finite number of motor neuron fibers that control a specific respiratory muscle area of the beetle. An example of such a recording can be seen in Fig. 2A. Experimental traces were analyzed separately by a panel of insect neurophysiology experts in our lab ($n=3$) each of whom has considerable experience with similar data. This analysis consists mainly in measuring the amplitude of each spike and visually inspecting the spike shapes to decide on the number of units present in the recording.

The spike sorting process revealed the presence of three different active neuron fibers and classified their spikes in three different clusters as shown in Fig. 4A. This result was in agreement with our panel of experts. Taking advantage of the information in spike timings extrapolated during the PCA–EM, we reconstructed the time series of each neuron and computed their rate histograms [40]. The histograms were based on the number of spikes per second, as counted in small bins of 80 ms, while the graph[1] was smoothed after computation with a Gaussian window function (Fig. 4B). We can see that fibers 1 and 3 follow a tonic firing pattern, while fiber 2 displays a rhythmic activity. Fiber 2 is identified as a bursting pacemaker neuron representing the driving unit of the respiratory central pattern generator. This analysis provided means of understanding the organization of the respiratory system in an insect from single-electrode extracellular recordings revealing a single pacemaker neuron providing the basic respiratory rhythm.

5. Conclusion

This paper introduces tools to estimate the optimum feature space of a PCA-based algorithm for spike sorting of nerve trunk recordings. We simulated series of randomly distributed action potentials from a total of seven distinct units and introduced a new metric to define clustering errors. We modeled background activity with Gaussian and correlated noises each having different effects on the performance of the algorithm. For moderate noise levels, a statistically important minimum in classification error rates emerged when the set of the four principal components was considered. This result may be valuable to a large number of neurophysiologists working on extracellular nerve trunk recordings with single electrodes. The patch for the nev2lkit program and the newly introduced metric are available online and can be used in the evaluation of PCA-based spike sorting algorithms.

Acknowledgment

This work was implemented in the context of PYTHAGORAS II (EPEAEK) project, a research program funded by the Greek Ministry of National Education and Religious Affairs.

Appendix A

If we consider n_{e1} and n_{e2} as two members of the e cluster set that are falsely merged into n_c^u by the clustering process, a process identified as "under-clustering", then the following is true[2]:

$$S(n_{e1}) - Fn(n_{e1}) + S(n_{e2}) - Fn(n_{e2}) + Fp(n_{e1}, n_{e2}) = S(n_c^u) \quad (A)$$

where $Fp(n_{e1}, n_{e2})$ represents the false positive spikes that are part of $S(n_c^u)$ but can be considered as "outliers" to both n_{e1} and n_{e2}, i.e. they originally belong to some third member of the e cluster set or they could be identified as overlaps.

Again we will consider the cluster error for n_{e1} as (the same apply for n_{e2} respectively):

$$\text{cluster\_error}(n_{e1}) = \frac{Fn(n_{e1}) + Fp(n_{e1})}{\sum_{n_e=1}^{e} S(n_e) + S_{NOISE}(e)} \times 100\%$$

In respect to n_{e1}, spikes belonging to n_{e2} that were classified under the same cluster, defined as $[S(n_{e2}) - Fn(n_{e2})]$, can also be considered as false positives in n_c^u. So,

$$Fp(n_{e1}) = S(n_{e2}) - Fn(n_{e2}) + Fp(n_{e1}, n_{e2}) \quad (B)$$

Taking Eq. (A) into account, Eq. (B) becomes:

$$Fp(n_{e1}) = S(n_c^u) - S(n_{e1}) + Fn(n_{e1})$$

Finally, taking into account all the above, the percentage cluster error becomes:

$$\text{cluster\_error}(n_{e1}) = \frac{S(n_c^u) - S(n_{e1}) + 2Fn(n_{e1})}{\sum_{n_e=1}^{e} S(n_e) + S_{NOISE}(e)} \times 100\%$$

and accordingly for every n_e that is *under-clustered* in n_c^u

$$\text{cluster\_error}(n_e) = \frac{S(n_c^u) - S(n_e) + 2Fn(n_e)}{\sum_{n_e=1}^{e} S(n_e) + S_{NOISE}(e)} \times 100\%$$

In practice, we may well approximate the cluster error with:

$$\text{cluster\_error}(n_e) = \frac{S(n_c^u) - S(n_e)}{\sum_{n_e=1}^{e} S(n_e) + S_{NOISE}(e)} \times 100\%$$

for two reasons: the first is that, when *under-clustering* occurs, the total number of $S(n_{e1})$ and $S(n_{e2})$ belong to $S(n_c^u)$ and we have no partitioning since the clustering process simply fails to partition them. Thus, in this case we have no false negatives. The second is that if we assume that some false-negative spikes existed, they would have a very small contribution to the percentage error comparing to the corresponding false positive

[1] Fig. 4B was created with the demo version of Neuroexplorer Software [41].

[2] The following can easily be shown for more than two members of the e cluster set that are merged in the same cluster.

which includes the total number of spikes belonging to n_{e2} as stated above.

Appendix B

If we consider n_e as member of the e cluster set that are falsely divided (*over-clustered*) into two parts, i.e. $n_c^o(1)$ and $n_c^o(2)$, by the clustering process, then the following is true[3]:

$$S(n_e) = S(n_c^o(1)) - Fp(n_e(1)) + S(n_c^o(2)) - Fp(n_e(2)) \quad (D)$$

$Fn(n_e(j))$ and $Fp(n_e(j))$ describe the false negative and positive spikes of n_e in respect to each of the j clusters.

Also $Fn(n_e(1))$ and $Fn(n_e(2))$ in this case are:

$$Fn(n_e(1)) = S(n_c^o(2)) - Fp(n_e(2)) \quad (E)$$

$$Fn(n_e(1)) = S(n_c^o(2)) - Fp(n_e(2)) \quad (F)$$

since the spikes that are missing from n_e in respect to cluster $n_c^o(1)$ exist in $n_c^o(2)$ and the spikes that are missing from n_e in respect to cluster $n_c^o(2)$ exist in $n_c^o(1)$.

Using Eq. (D), Eqs. (E) and (F) become:

$$Fn(n_e(1)) = S(n_e) - S(n_c^o(1)) + Fp(n_e(1)) \quad (G)$$

$$Fn(n_e(2)) = S(n_e) - S(n_c^o(2)) + Fp(n_e(2)) \quad (H)$$

In order to estimate the percentage cluster error for n_e, generally defined as

$$\text{cluster\_error}(n_e) = \frac{Fn(n_e) + Fp(n_e)}{\sum_{n_e=1}^{e} S(n_e) + S_{NOISE}(e)} \times 100\%,$$

we would not want this error to be proportional to j, i.e. to rise proportionally to j since, as already stated, over-clustering is handled with manual cluster-merging and this can be thought as a process whose difficulty is independent of j.

So we will average the error in respect to j (in our case 2):

$$\text{cluster\_error}(n_e) = \frac{(1/2)[Fn(n_e(1)) + Fp(n_e(1)) + Fn(n_e(2)) + Fp(n_e(2))]}{\sum_{n_e=1}^{e} S(n_e) + S_{NOISE}(e)} \times 100\% \quad (I)$$

Taking into account Eqs. (G) and (H), Eq. (I) becomes:

$$\text{cluster\_error}(n_e) = \frac{1}{2}\frac{[S(n_e) - S(n_c^o(1)) + 2Fp(n_e(1))] + [S(n_e) - S(n_c^o(2)) + 2Fp(n_e(2))]}{\sum_{n_e=1}^{e} S(n_e) + S_{NOISE}(e)}$$

$$\times 100\% = \frac{1}{2}\sum_{i=1}^{2}\left[\frac{S(n_e) - S(n_c^o(i)) + 2Fp(n_e(i))}{\sum_{n_e=1}^{e} S(n_e) + S_{NOISE}(e)}\right] \times 100\%$$

For every n_e that is *over-clustered* in $n_c^o(j)$ clusters

$$\text{cluster\_error}(n_e) = \frac{1}{j}\sum_{i=1}^{j}\left[\frac{S(n_e) - S(n_0^o(i)) + 2Fp(n_e(i))}{\sum_{n_e=1}^{e} S(n_e) + S_{NOISE}(e)}\right] \times 100\%$$

[3] The following can easily be shown if a member of the e cluster set is divided to more than two clusters.

REFERENCES

[1] G. Theophilidis, The femoral chordotonal organs of *Decticus albifrons* (Orthoptera: Tettigoniidae). Part II. Function, Comp. Biochem. Physiol. 84A (3) (1986) 537–543.

[2] H.G. Heinzel, J.M. Weimann, E. Marder, The behavioral repertoire of the gastric mill in the crab *Cancer pagurus*: an in situ endoscopic and electrophysiological examination., J. Neurosci. 13 (1993) 1793–1803.

[3] K. Nishimura, Y. Kanda, A. Okazawa, T. Ueno, Relationship between insecticidal and neurophysiological activities of imidacloprid and related compounds, Pestic. Biochem. Physiol. 50 (1994) 51–59.

[4] D.W. Richter, P. Schmidt-Garcon, O. Pierrefiche, A.M. Bischoff, P.M. Lalley, Neurotransmitters and neuromodulators controlling the hypoxic respiratory response in anaesthetized cats", J. Physiol. 514 (1999) 567–578.

[5] G. Zafeiridou, G. Theophilidis, The action of the insecticide imidacloprid on the respiratory rhythm of an insect: the beetle *Tenebrio molitor*, Neurosci. Lett. 365 (2004) 205–209.

[6] K.-G. Westberg, A. Kolta, P. Clavelou, G. Sandström, J.P. Lund, Evidence for functional compartmentalization of trigeminal muscle spindle afferents during fictive mastication in the rabbit, Eur. J. Neurosci. 12 (2000) 1145–1154.

[7] F. Wood, M.J. Black, C. Vargas-Irwin, M. Fellows, J.P. Donoghue, On the variability of manual spike sorting, IEEE Trans. Biomed. Eng. 51 (2004) 912–918.

[8] K. Haris, D. Henze, J. Csicsvari, H. Hirase, G. Buzsàki, Accuracy of tetrode spike separation as determined by simultaneous intracellular and extracellular measurements, J. Neurophysiol. 84 (2000) 401–414.

[9] M. Sahani, J.S. Pezaris, R.A. Andersen, On the separation of signals from neighboring cells in tetrode recordings, in: Advances in Neural Information Processing Systems, vol. 10, MIT Press, Cambridge, MA, 1998, pp. 222–228.

[10] J. Csicsvari, H. Hirase, A. Czurkó, G. Buzsàki, Reability and state dependence of pyramidal cell–interneuron synapses in the hippocampus: an ensemble approach in the behaving rat, Neuron 21 (1998) 179–189.

[11] G. Zouridakis, D.C. Tam, Identification of reliable spike templates in multi-unit extracellular recordings using fuzzy clustering, Comput. Methods Programs Biomed. 61 (2) (2000) 91–98.

[12] C.M. Stewart, S.D. Newlands, A.A. Perachio, Spike detection, characterization, and discrimination using feature analysis software written in LabVIEW, Comput. Methods Programs Biomed. 76 (3) (2004) 239–251.

[13] M.S. Fee, P.P. Mitra, D. Kleinfeld, Automatic sorting of multiple unit neuronal signals in the presence of anisotropic and non-Gaussian variability, J. Neurosci. Methods 69 (1996) 175–188.

[14] E. Hulata, R. Segev, Y. Shapira, M. Benveniste, E. Ben-Jacob, Detection and sorting of neural spikes using wavelet packets, Phys. Rev. Lett. 85 (2000) 4637–4640.

[15] E. Hulata, R. Segev, E. Ben-Jacob, A method for spike sorting and detection based on wavelet packets and Shannon's mutual information, J. Neurosci. Methods 117 (2002) 1–12.

[16] K.H. Kim, S.J. Kim, Neural spike sorting under nearly 0-db signal-to-noise ratio using nonlinear energy operator and

artificial neural-network classifier, IEEE Trans. Biomed. Eng. 47 (2000) 1406–1411.
[17] K.H. Kim, S.J. Kim, Method for unsupervised classification of multiunit neural signal recording under low signal-to-noise ratio, IEEE Trans. Biomed. Eng. 50 (2003) 421–431.
[18] M.S. Lewicki, A review of methods for spike sorting: the detection and classification of neural action potentials, Netw. Comput. Neural. Syst. 9 (1998) R53–R78.
[19] A. Pavlov, V.A. Makarov, I. Makarova, F. Panetsos, Separation of extracellular spikes: when wavelet based methods outperform the principle component analysis, in: Lecture Notes in Computer Science, vol. 3561, Springer, Berlin/Heidelberg, 2005, pp. 123–132.
[20] S. Shoham, M.R. Fellows, R.A. Normann, Robust, automatic spike sorting using mixtures of multivariate t-distributions, J. Neurosci. Methods 127 (2003) 111–122.
[21] S. Takahashi, Y. Anzai, Y. Sakurai, Automatic sorting for multi-neuronal activity recorded with tetrodes in the presence of overlapping spikes, J. Neurophysiol. 89 (2003) 2245–2258.
[22] M. Abeles, M.H. Goldstein, Multispike train analysis, Proc. IEEE 65 (1997) 762–773.
[23] G. Buzsàki, D.L. Buhl, K.D. Harris, J. Csicsvari, B. Czéh, A. Morozov, Hippocampal network patterns of activity in the mouse, Neuroscience 116 (2003) 201–211.
[24] A. Elhalal, D. Horn, In vitro neuronal networks: evidence for synaptic plasticity, Neurocomputing (2005) 65–66.
[25] S. Furukawa, J.C. Middlebrooks, Cortical representation of auditory space: information-bearing features of spike patterns, J. Neurophysiol. 87 (2002) 1749–1762.
[26] A. Luczak, N.S. Narayanan, Spectral representation—analyzing single-unit activity in extracellularly recorded neuronal data without spike sorting, J. Neurosci. Methods 144 (2005) 53–61.
[27] O. Mazor, G. Laurent, Transient dynamics versus fixed points in odor representations by locust antennal lobe projection neurons, Neuron 48 (2005) 661–673.
[28] P.G. Musial, S.N. Baker, G.L. Gerstein, E.A. King, J.G. Keating, Signal-to-noise ratio improvement in multiple electrode recording, J. Neurosci. Methods 115 (2002) 29–43.
[29] NEV2lkit, A preprocessor for intra- and extra-cellular neuronal recordings distributed under GPL, http://nev2lkit.sourceforge.net.
[30] D.I.G. Wilson, E.M. Bowman, Neurons in dopamine-rich areas of the rat medial midbrain predominantly encode the outcome-related rather than behavioural switching properties of conditioned stimuli, Eur. J. Neurosci. 23 (2006) 205–218.
[31] G. Celeux, G. Govaert, A classification EM algorithm for clustering and two stochastic versions, Comput. Stat. Data Anal. 14 (1992) 315–332.
[32] X. Hu, L. Xu, A comparative investigation on subspace dimension determination, Neural Netw. 17 (2004) 1051–1059.
[33] N. Dalkiliç, F. Pehlivan, Comparison of fiber diameter distributions deduced by modeling compound action potentials recorded by extracellular and suction techniques, Int. J. Neurosci. 112 (2002) 913–930.
[34] P.R. Gray, Conditional probability analyses of the spike activity of single neurons, Biophys. J. 7 (1967) 759–777.
[35] R.P. Gaumond, C.E. Molnar, D.O. Kim, Stimulus and recovery dependence of cat cochlear nerve fiber spike discharge probability, J. Neurophysiol. 48 (1982) 856–887.
[36] D.H. Johnson, A. Swami, The transmission of signals by auditory nerve fiber discharge patterns, J. Acoust. Soc. Am. 74 (1983) 493–501.
[37] F. Murtagh, A. Heck, Multivariate Data Analysis, Kluwer Academic Publishers, Dordrecht, 1987, pp. 20–29.
[38] J.P. Segundo, J.-F. Vibert, K. Pakdaman, M. Stiber, O. Diez-Martinez, Noise and the neurosciences: a long history with a recent revival (and some theory), in: K. Pribram (Ed.), Origins: Brain, and Self-Organization, Lawrence Erlbaum Associates Pub., Hillsdale, NJ, 1994.
[39] CORTIVIS, A project to develop a cortical neuroprosthesis for the blind, Supported by the Commission of the European Communities specific program “quality of life and management of living resources”, QLK6-CT-2001-00279, http://cortivis.umh.es.
[40] P. Dayan, L. Abbott, Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems, The MIT Press, Cambridge, MA, 2001 (Chapter 1.2).
[41] Nex Technologies, http://www.neuro-explorer.com/, 2007.