



**HAL**  
open science

# A non-parametric method for automatic neural spikes clustering based on the non-uniform distribution of the data

Zoran Tiganj, Mamadou Mboup

► **To cite this version:**

Zoran Tiganj, Mamadou Mboup. A non-parametric method for automatic neural spikes clustering based on the non-uniform distribution of the data. *Journal of Neural Engineering*, 2011, 8 (6), pp.066014. 10.1088/1741-2560/8/6/066014 . hal-00639412

**HAL Id: hal-00639412**

**<https://inria.hal.science/hal-00639412>**

Submitted on 9 Nov 2011

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# A non-parametric method for automatic neural spikes clustering based on the non-uniform distribution of the data

Z. Tiganj<sup>1</sup> and M. Mboup<sup>2,1</sup>

<sup>1</sup> Non-A, INRIA LILLE - NORD EUROPE, parc Scientifique de la Haute Borne 40, avenue Halley, bt A, Park Plaza, 59650 Villeneuve d'Ascq

<sup>2</sup> CReSTIC, UFR SEN, UNIVERSITÉ de REIMS CHAMPAGNE ARDENNE, BP 1039 Moulin de la Housse, 51687 REIMS cedex 2

E-mail: [zoran.tiganj@inria.fr](mailto:zoran.tiganj@inria.fr)

**Abstract.** In this paper we propose a simple and straightforward algorithm for neural spike sorting. The algorithm is based on the observation that the distribution of a neural signal largely deviates from the uniform distribution and is rather unimodal. The detected spikes to be sorted are first processed with some feature extraction technique, such as PCA, and then represented in a space with reduced dimension by keeping only a few most important features. The resulting space is next filtered in order to emphasize the differences between the centers and the borders of the clusters. Using some prior knowledge on the lowest level activity of a neuron, as *e.g.* the minimal firing rate, we find the number of clusters and the center of each cluster. The spikes are then sorted using a simple greedy algorithm which grabs the nearest neighbors.

We have tested the proposed algorithm on real extracellular recordings and used the simultaneous intracellular recordings to verify the results of the sorting. The results suggest that the algorithm is robust and reliable and it compares favorably with the state-of-the-art approaches. The proposed algorithm tends to be conservative, it is simple to implement and is thus suitable for both research and clinical applications as an interesting alternative to the more sophisticated approaches.

## 1. Introduction

### 1.1. Importance of spike sorting

Recording the activity of neurons is typically done with electrodes placed in the extracellular space. An extracellular electrode records a mixture of activities of a vast number of neurons around. Separating the activity of individual neurons from the mixture is the so called *spike sorting problem*.

All the neurons fire morphologically practically the same action potentials. Due to the propagation and the velocity effects, the spikes recorded by extracellular electrodes usually have different shapes and amplitudes if they are coming from different neurons [1]. This fact is used as a base for the development of different spike sorting algorithms (see [2] and [3] and the references therein for a tutorial presentation). In spike sorting it is not necessary to recover the original waveforms of the action potentials, but only to find which neuron fired which of the detected spikes.

### 1.2. Common approach

We consider to have only a single-channel recording, but the spike sorting algorithms that are commonly applied on single channel recordings are applicable on multi-channel recordings as well, usually as a combination with some Source Separation methods (for examples see [4] and [5]). Say we have a single-channel extracellular neural recording that consists of  $N$  samples and let each spike has a duration of  $M$  samples. A generic spike sorting algorithm consists of several steps:

- (i) **Spike detection:** The goal of spike detection is to extract from the recorded signal all the spikes fired by the neurons close to the electrode. For recordings with good Signal to Noise Ratio (SNR) this is usually achieved by a simple thresholding [6]. When the SNR is not good enough, different spike detection algorithms can be applied, *e.g.* [7], [8] and [9]. The result of this step is a  $M \times K$ -matrix, where  $K$  is the number of the detected spikes.
- (ii) **Feature extraction:** Principal Component Analysis (PCA) [10], wavelet decomposition [11], [12] or some other techniques (*e.g.* [13], [14] and [15]) are commonly used to reduce the dimensionality of the  $M \times K$ -matrix by extracting the most important features of the detected spikes. The result is a new matrix of reduced dimension,  $L \times K$ , where  $L < M$  is the number of extracted features per spike.
- (iii) **Finding the number of clusters (the number of neurons close to the electrode) *i.e.* model selection:**  $K$  detected spikes could be fired by one or more neurons. This step is very often done manually: after observing the  $L \times K$  matrix (often,  $L$  is equal to 1, 2 or 3 so the matrix can be visualized), the user chooses the number of clusters. But some general purpose parametric model selection algorithms [16] based on penalization such as Bayesian Information

Criterion (BIC) are also proposed as more efficient (see examples in [17], [18] and [6]). The result of this step is the estimated number of different neurons, call it  $P$ , that fired the  $K$  detected spikes.

- (iv) **Sorting:** Finally, the  $K$  detected spikes are sorted into  $P$  clusters. Again general purpose techniques, such as K-means [4] or Expectation Maximization (EM) [19], are usually applied [2].

Some other, very different approaches, also exist, (e.g. [20]), but they are mostly applicable only in special cases when some specific prior knowledge (e.g. interspike interval histogram) is available. Because such knowledge is available only for some special types of neurons and particular tissues, we will not analyze those methods here, since we aim to address the spike sorting problem in its general form.

### 1.3. Problems of spike sorting

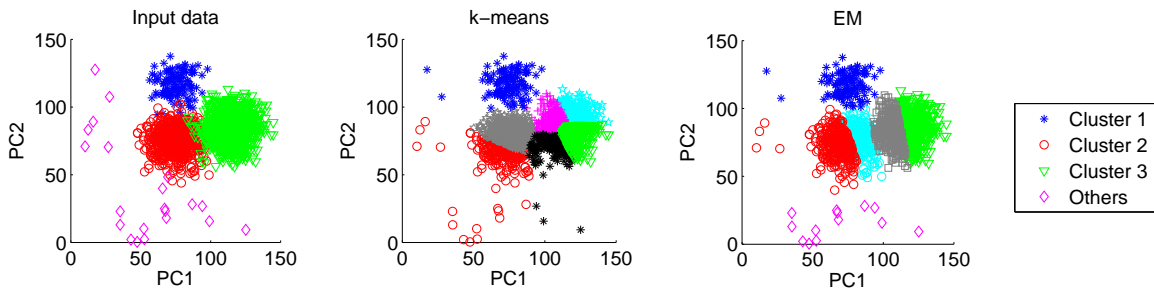
Due to a typically unbalanced structure of the clusters, finding their number and sorting the spikes into them (steps 3 and 4) are usually very difficult. The unbalanced structure is mainly caused by two reasons: 1) The firing rates of different neurons are usually very different (up to hundred times) 2) The distances between the centers of the clusters can also be very different.

To illustrate the difficulty of model selection in neural recordings let us discuss on the implementation of BIC. For each realistic number of clusters (for each different model) we apply EM/k-means on the data and compute the likelihood of the model from the output of EM/k-means. Since increasing the number of clusters naturally increases the likelihood, it is necessary to penalize the increase of the number of clusters. Finding a correct penalization is particularly difficult since it is usually chosen only once for one type of recordings while the morphology of the feature space can vary significantly from recording to recording.

On the other side the accuracy of manual determination of the number of clusters is also not satisfying [21]. Moreover manual approach is very expensive and hard to perform on large sets of data.

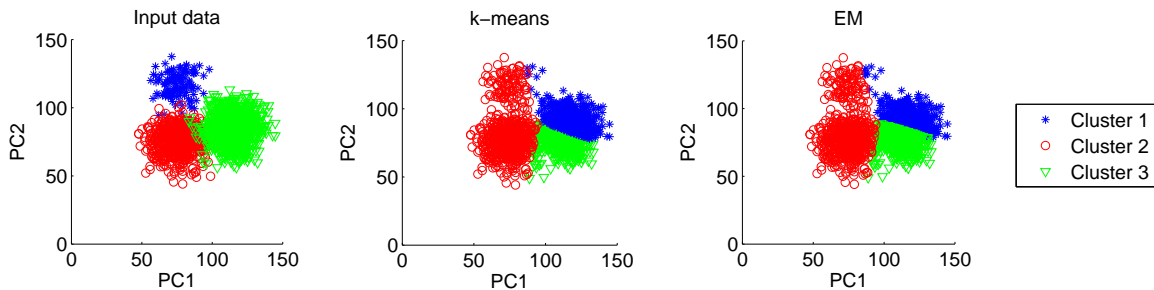
An example of bad results when general purpose clustering methods are applied is shown on figure 1. We used artificial neural data to be able to verify the clustering results. The dimensionality of the spikes is reduced to 2, using PCA. The first plot on the figure 1 displays the original clusters, corresponding to three different neurons (spikes coming from the same neuron are marked with the same shape and color). For model selection we used BIC, as described above, which is known to often overestimate the number of clusters, and obtained the result that the data contain 6 clusters. The results of clustering in 6 clusters with k-means are shown on the second plot on figure 1 and with EM on the last, third plot.

Even if the number of clusters is determined correctly, sorting the spikes into the corresponding clusters is another very difficult problem for neural data. The principal reason for this is that neural data can not be accurately clustered based only on the



**Figure 1.** Sorting example on simulated neural data with combination of BIC with k-means and EM.

distance to the center of a cluster, now this is the most important parameter among the parametric clustering methods. This is illustrated on figure 2 where the data consisting of only three clusters were used. The correct number of clusters was given to both k-means and EM methods, but still the sorting results are not as desired.



**Figure 2.** Sorting example on simulated neural data with using k-means and EM when the number of clusters is priori fixed to 3.

The results are very bad since both methods are not adapted for such data. Single clusters are often divided into two or more. Elements that do not belong to any of the three clusters are often assigned to one of them, instead of being sorted as separate clusters that would later be neglected due to a too small number of members. Finally, we note that although such approaches may be very efficient in general, their application on neural data can lead to bad sorting results.

#### 1.4. Brief description of the proposed method

In this paper we propose a new method for finding the number of neurons that fired the detected spikes and for associating each of the spikes with a corresponding neuron (steps 3 and 4 above). We do not address questions of *e.g.* spike detection or feature extraction, but to demonstrate our results we will use techniques commonly applied for these problems. The proposed method is non-parametric. It is designed especially for neural recordings, with the aim of high robustness with respect to mis-classification.

The method is based on two assumptions on the recorded signal:

a) A lower bound on the number of spikes fired by a single neuron during the recording

time is known. This is a very realistic assumption since neurons (especially those whose activity we actually want to detect) fire usually at least once per second.

b) The recorded signal has a unimodal distribution. This is easy to verify experimentally, but it can also be shown theoretically upon invoking the central limit theorem.

The basic idea behind the proposed method is that most of the spikes fired by the same neuron, after projection on feature extraction vectors, will be concentrated around the center of the corresponding cluster. As we go to the edges of each cluster the density of spikes decreases. Smoothing the vector space that contains the  $K$  detected spikes (represented in  $L$ -dimensions) with a moving average filter allows one to emphasize the differences between clusters and leads to easy non-supervised identification of the centers of the clusters. To assign the spikes to their corresponding clusters we use a simple greedy approach: each cluster is initialized by its singleton center and the clustering is performed by classifying one spike at a time, in such a way that if an element belongs to a cluster then so is its nearest neighbor. This procedure leads to a robust solution, which is not sensitive to unbalanced sizes of clusters.

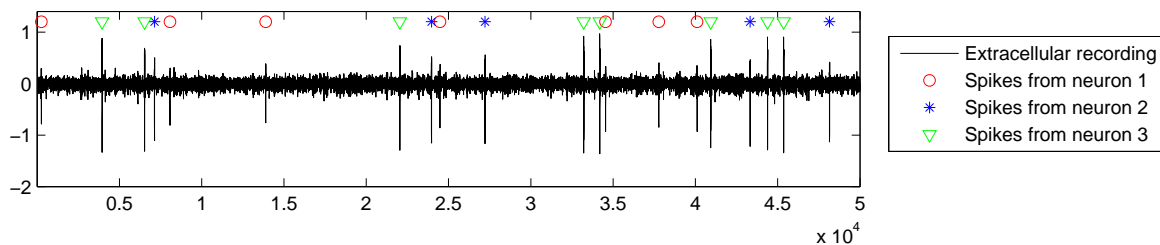
The property of unimodal distribution of neural data is exploited in several neural spike sorting algorithms. In [22] it has been used to find spike templates which are then correlated with all the detected spikes in order to emphasize the differences between clusters. Such template matching can be used as a pre-processing step of the method proposed in this paper, since template matching can generally improve the results obtained by feature extraction methods. Different peak-valley search algorithms, such as histogram peak count, are also based on the distributional properties of neural data (see *e.g.* [23] or the implementation in a commercial system described in [24]). Typically, these algorithms sort all the spikes at once by estimating the morphology of the area with lower spike density, giving the borders between clusters explicitly. The implementation can thus be very complex, especially in higher dimensional space. Also, when more than two clusters exist, the vector space becomes very difficult to analyze with such methods. By clustering one spike at a time, where the borders are given implicitly, with only the filter length and height to set, the proposed method provides a simple and yet robust alternative.

### 1.5. Paper organization

A detailed description of the method is given in section 2. We apply the method on extracellular recordings and use the simultaneous intracellular recordings to verify the results of the sorting for the neuron recorded intracellularly. Such way of validation represents another contribution of this paper, since spike sorting methods are typically validated only on the simulated signal. In section 3 we describe the signal that we use for the validation, which is available online [1], give all the simulation parameters and the results of the comparison with the state-of-the-art approaches. Conclusions and some future perspectives for development of this method are given in section 4.

## 2. Methodology

In this section we will describe our method step-by-step. For the demonstration we use a simulated signal for which the actual activity of each neuron is known. We can therefore display the results of the proposed method. As we said in the previous section, we assume to have an  $N$ -samples long single-channel extracellular neural recording. Our goal is to sort the firing patterns of as many individual neurons as possible (for single-channel recordings this usually means activities of two to three neurons closest to the electrode). Example of simulated extracellular recording is displayed on figure 3. We labeled the spikes that come from different neurons with different symbols.



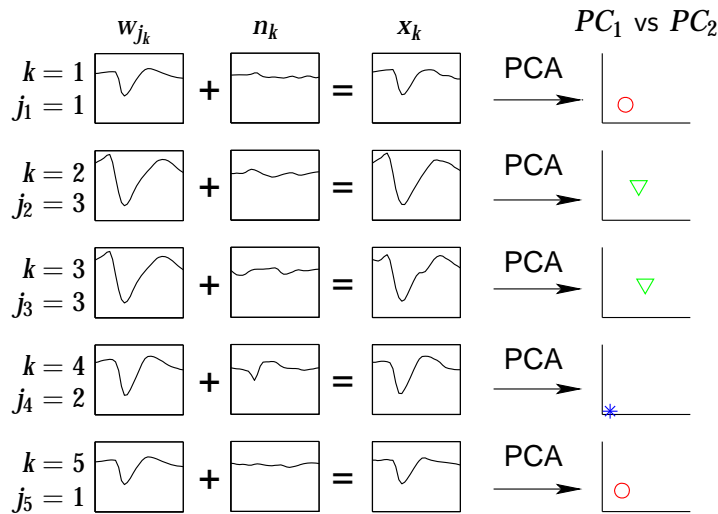
**Figure 3.** Simulated extracellular neural recording with marked spike locations from three neurons closest to the virtual electrode.

### 2.1. Preprocessing: Spike detection and feature extraction

The first preprocessing step is the spike detection. Depending on the SNR, setting a threshold directly on the extracellular recording or applying some more advanced spike detection technique can be considered.

Assume that we detected  $K$  spikes that come from a total of  $P$  neurons (for us  $P$  is unknown). To each detected segment  $x_k, k = 1, \dots, K$ , we associate one neuron, say neuron  $\#j_k, 1 \leq j_k \leq P$ . This is, among all neurons which are active during that segment, the closest one to the electrode. We can then write each segment as:  $x_k = w_{j_k} + n_k$ , where  $w_{j_k}$  is the spike waveform coming from the associated neuron and  $n_k$  is the contribution of all remaining neurons that were active during the detected segment  $x_k$ . Figure 4 shows the first 5 segments  $x_k$  extracted from the signal on figure 3 and displays them as a sum of  $w_{j_k}$  and  $n_k$ .

Since each extracted segment has the same length of  $M$  samples, all the segments can be given in  $M \times K$  matrix. So now we have each detected spike described by  $M$  samples ( $M$  is usually between 50 and 100, depending on the spike duration, which is usually about  $1ms$ , and the sampling period). But each spike can be described by *e.g.* its amplitude, or its minimal and maximal value. Using only these information it can be easier to find which spikes are fired by the same neuron than by observing all the  $M$  samples. Generally speaking, we can try to extract some features that give a compact description of the spikes. There are many techniques for feature extraction that try to extract one or more of the most important features. In this paper we will



**Figure 4.** First 5 spikes from figure 3 shown as a sum of original spike waveforms and noise. Each spike is then projected on the first two principle components. The last column shows the values of these projections in PC1 vs PC2 basis.

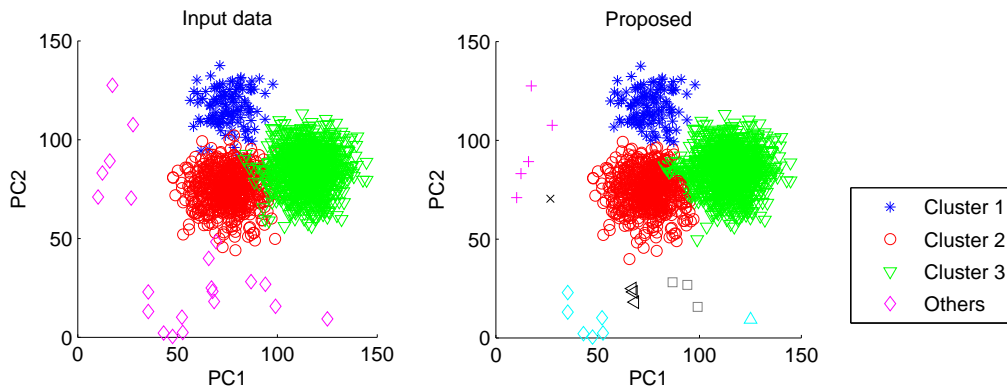
use probably the most popular one, PCA, but also some other techniques could be used (good results are reported *e.g.* when wavelet decomposition is used [11] and new techniques for feature extraction in neural recordings are under constant development see *e.g.* [25]). The algorithm we propose in this paper is compatible with all of these feature extraction methods. PCA takes the variability in the data as the most interesting feature and projects the data in such a way that the first basis vector (principle component) has the variance as high as possible. Each succeeding principle component in turn has the highest variance under the constraint that it is orthogonal to the preceding components. We project our  $M \times K$  matrix on  $L$  principal components. This projection results in new  $L \times K$  matrix. For the simplicity of this description, we will assume the choice of  $L = 2$  (only two features are extracted from each detected segment) and call the corresponding matrix  $D_{2 \times K}$ . An example is shown on figure 4 where the last column gives the projection of  $x_k$  on the first two principle components. Notice that the projections are similar for spikes coming from a same neuron. All the extracted spikes projected on the first two principal components (all the elements of matrix  $D_{2 \times K}$ ) are shown on the left plot on figure 5. Same as for figure 1, the projections of the spikes that belong to different neurons are marked with different symbols and colors.

## 2.2. Unimodal distribution

Now having  $D_{2 \times K}$ , we need to find out how many neurons fired the spikes we detected and to associate each spike to one of the neurons.

To find the centers of the clusters we will first filter the 2D vector space. One key point of the method is to assume that the maxima of the filter's output correspond to the centers of the clusters. Such assumption obviously implies that each cluster has a





**Figure 5.** The left-hand side plot shows the initial data set (same as for figure 1). The right-hand side plot shows sorting results with the proposed method. The elements of each found cluster are labeled with symbols of the same shape/color.

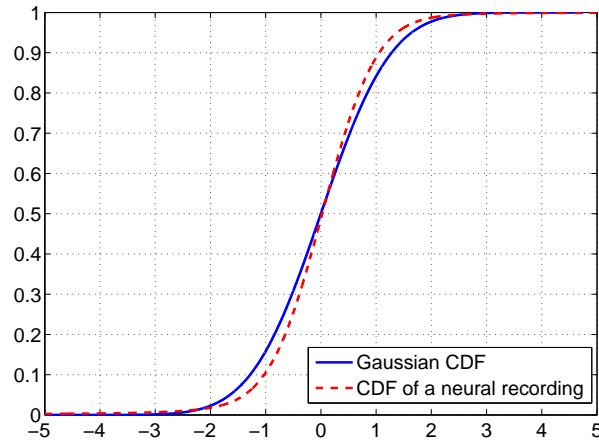
unimodal shape (in  $L$  dimensional space) and also that the centers are far enough so that the unimodal shape is not destroyed by an overlapping of clusters. If this is not the case we would find multiple centers in a single cluster. We will now discuss whether these properties are realistic for neural data.

The application of PCA on each detected segment  $x_k$  can be written as a scalar product between  $x_k$  and, in our case, only the first two principal components ( $PC_1$  and  $PC_2$ ):

$$\langle x_k, PC_i \rangle = \langle w_{j_k}, PC_i \rangle + \langle n_k, PC_i \rangle = \alpha_{j_k,i} + \beta_{k,i}, \quad i = 1, 2.$$

Each spike waveform  $w_{j_k}$ , for the same neuron  $j$ , will be transformed with PCA into exactly the same point in the 2D vector space ( $\alpha_{j_k,i}$ ). As we have mentioned earlier, we consider that these points will be mutually distant enough so that the dispersive effect of the noise cannot induce any overlapping between clusters. So the success in finding the centers of the clusters depends on the difference between the spikes from the neurons we want to sort and the noise level. When performing a recording it is generally possible to influence on both of these parameters by adjusting the electrode position.

Considering the noise part,  $n_k$  is actually a weighted sum of activities from a very large number of neurons. Even though the distribution of the activity of each individual neuron is obviously not unimodal, the distribution of the sum of a large number of these activities tends to be unimodal. This stems from the central limit theorem, assuming that the spiking activity of each neuron is independent to the others. Recall that the many spike sorting algorithms that we have mentioned in the introduction also assume the unimodal property. Moreover, this property is simple to verify experimentally by observing the cumulative distribution function (CDF) of a neural recording. We plot the empirical CDF of a neural recording and a Gaussian CDF as a reference on figure 6. The neural recording has unimodal distribution which is in fact close to Gaussian (notice that a proximity to the Gaussian distribution is not required by the proposed algorithm).



**Figure 6.** Gaussian CDF and empirical CDF of a neural recording.

Since PCA is a linear transformation  $\beta_{k,1}$  and  $\beta_{k,2}$  are just linear combinations of the elements of  $n_k$ , so the distribution of  $\beta_{k,1}$  and  $\beta_{k,2}$  is of the same type as that of  $n_k$ .

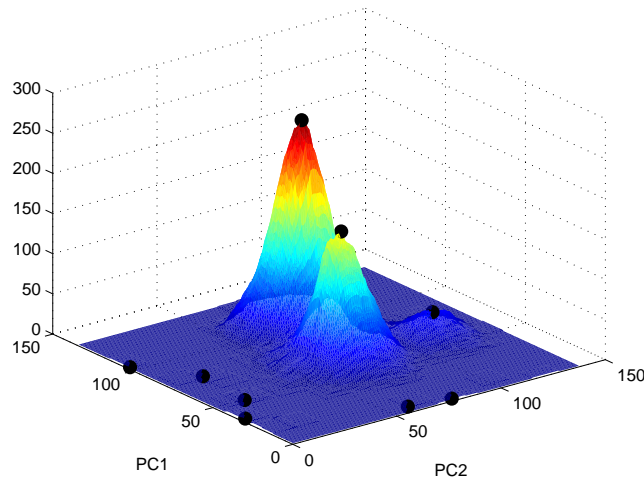
Finally, depending on the conditions mentioned above, we can expect that the density of  $(\langle x_k, PC_1 \rangle, \langle x_k, PC_2 \rangle)$  will have a single maximum for a single cluster. Such morphology suggests the idea of sorting all the spikes into their corresponding clusters using a simple greedy algorithm which grabs the nearest neighbors: since the density of spikes decreases as we go further from the maximum, gathering spikes to the same cluster using the criteria of their proximity to nearest neighbor seems to be a simple solution. This conclusion is on the basis of the proposed spike sorting algorithm.

### 2.3. Finding the centers of the clusters

In this subsection we will describe the algorithm for finding the centers of the clusters. According to the previous subsection this amounts to find the areas of the highest density of the points  $(\langle x_k, PC_1 \rangle, \langle x_k, PC_2 \rangle)$ .

For this, we first assume that each active neuron fires at least, say,  $G$  spikes during the observation window. Most of the neurons fire at least once per second, so depending on the duration of the recording we can easily calculate  $G$ . Now knowing  $G$ , we construct a moving average filter that will smooth the obtained  $2D$   $K$ -elements space without merging different clusters. We assume that the  $2D$  space is normalized so that we can consider a square domain for the filter's support. The filter should emphasize the differences between the centers and the border of each cluster. Its length (height), call it  $R$ , should be adjusted in a way that the filter window does not contain more than *e.g.*  $G/2$  elements. In subsection 3.3 we will show that in practical situations  $R$  can be taken from a large range of values without affecting the clustering results. The result of filtering the data from figure 5 is shown on figure 7. Call the resulting matrix  $D_{2 \times K}^{filt}$ . The centers of the clusters in the data matrix  $D_{2 \times K}^{filt}$  are easily found: any local maximum 1) which is centered in a domain at least larger than the filter window and 2) which is

global in that domain, is a center of a cluster. In such way we are again sure that two clusters will not be merged into one. The centers found in this way are labeled with black dots on figure 7. The figure demonstrates that one center is found for each of the three big clusters. Several centers are found in the area distant from the biggest clusters, these actually represent the activities of neurons whose spikes are only partially detected. As we will see in the next subsection, since only a small number of spikes will be assigned to these clusters (less than  $G$ ) they will be neglected and the spikes assigned to them left unsorted.



**Figure 7.** Filtered vector space from figure 5. Black dots denote clusters maxima.

#### 2.4. Associating each spike with the corresponding center

After finding the centers of the clusters, we have to associate each of the spikes with one of the centers.

To cluster the data we propose a greedy algorithm close in spirit to the agglomerative hierarchical clustering [26]. In spike sorting the agglomerative hierarchical clustering has already been considered in [27] where single spikes are first sorted into an overly large number of clusters and after the clusters are progressively aggregated into a minimal set of clusters. Our approach is close in spirit, but yet significantly different from hierarchical clustering since we sort spikes one by one, instead of merging clusters. Thus we avoid the difficulty of devising a merging criterion for the clusters. Hence, the proposed approach is simpler and it can be implemented in an unsupervised way. We will exploit the fact that we know the centers of the clusters and that they are actually points of the highest density. Our basic statement is that if an element belongs to a given cluster, then its nearest neighbor also belongs to the same cluster. Throughout, we use the Euclidian distance.

Let  $\tilde{x}_k$  be the projection of  $x_k$  on the  $L$  feature extraction vectors:  $\tilde{x}_k = (\langle x_k, PC_1 \rangle, \langle x_k, PC_2 \rangle)$  and call  $\mathfrak{D}$  the set of all  $\tilde{x}_k$ . Let  $c_i, i = 1, \dots, P$  denote the centers

of the clusters, in an  $L$  dimensional space. The pseudocode 1 describes the algorithm for the construction of the cluster  $\mathfrak{C}_i$  that corresponds with the center  $c_i$ . Therein, we use the notation  $\text{dist}$  for  $\text{dist}(x, \mathfrak{C}) = \min_{y \in \mathfrak{C}} \|x - y\|^2$ . For the initialization, we set  $\mathfrak{C}_i = \{c_i\}$  for all clusters.

---

**Algorithm 1** Proposed method for associating spikes with corresponding centers

---

```

while  $\mathfrak{D}$  not empty do
  for  $i = 1$  to  $P$  do
     $\tilde{x}_i^{\min} = \arg \min_{\tilde{x} \in \mathfrak{D}} \text{dist}(\tilde{x}, \mathfrak{C}_i)$            For each cluster  $\mathfrak{C}_i$ , find the closest element
                                                                 $\tilde{x}_i^{\min}$  of  $\mathfrak{D}$ 
  end for
   $j = \arg \min_{1 \leq i \leq P} \text{dist}(\tilde{x}_i^{\min}, \mathfrak{C}_i)$            Find which cluster  $\mathfrak{C}_i$  to update
   $\mathfrak{C}_j \leftarrow \mathfrak{C}_j \cup \tilde{x}_j^{\min}$                        Add the spike to the corresponding cluster
   $\mathfrak{D} \leftarrow \mathfrak{D} \setminus \tilde{x}_j^{\min}$                    Remove the sorted spike from the set  $\mathfrak{D}$ 
end while

```

---

The right-hand side plot on figure 5 was obtained by applying this algorithm, using the centers obtained in the previous step. Comparing with the left-hand side plot on figure 5, we can conclude that the proposed algorithm resulted in a quite good reconstruction of the three clusters. Several elements are misplaced but the result is quite satisfactory, especially in comparison with the one presented on figures 1 and 2. The spikes labeled as "Others" are broken into few clusters any of which is not large enough to exceed  $G$ , so they are all left unsorted. We consider this as a correct solution, since we do not want results that would contain only partially detected activity of some neurons.

In cases where we found only one cluster large enough to be kept, we can go back to the spike detection step and lower down the threshold value aiming to detect more spikes. If detecting much more spikes (extremely low threshold) does not result in detection of more than one large cluster, then it is a good indication that the recorded data are too noisy for such processing. In general, we can say that the cluster which contains most of the spikes with lowest amplitudes should be neglected, especially when high reliability of the results is needed.

### 2.5. Possible situations

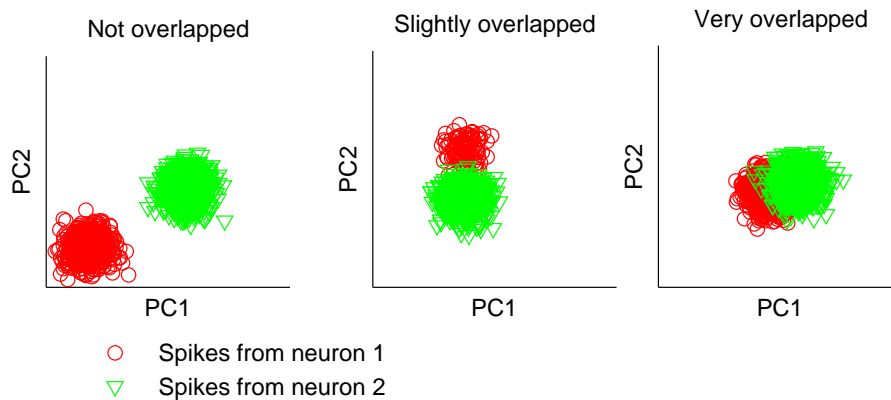
Depending on the measurement conditions, we can have in practice three possible situations:

1) Spikes fired by two given neurons are significantly different. Consequently the extracted features are significantly different too (example is the first plot on figure 8). It is evident that the proposed approach will accurately find the centers and then sort the spikes into their corresponding cluster.

2) Spikes fired by the two neurons do not differ significantly (the second plot on figure 8). In this case the clusters are (slightly) overlapped: some spikes will be wrongly

sorted, but we can expect that the majority will be sorted correctly. Since the density of the projected spikes is higher in the cluster’s center than in the areas between the clusters, the proposed method will accurately find the centers of the clusters.

3) Spikes fired by the two neurons are very similar (the third plot on figure 8). Now finding the centers of the clusters is not possible with the proposed method. Even if the centers were known, an accurate sorting would still not be possible with the proposed greedy algorithm. This situation is typical for very noisy recordings. In such data, the proposed algorithm will usually detect only one cluster. As we have mentioned before, this can be used as an indicator that the recording is too noisy. A potential solution could be to project the data on a different number of principle components, reduce the noise sources, do reposition of the electrode or use multi-channel measurement system.



**Figure 8.** Three generally possible situations considering mutual distance between clusters of the neural data projected on the first two principle components.

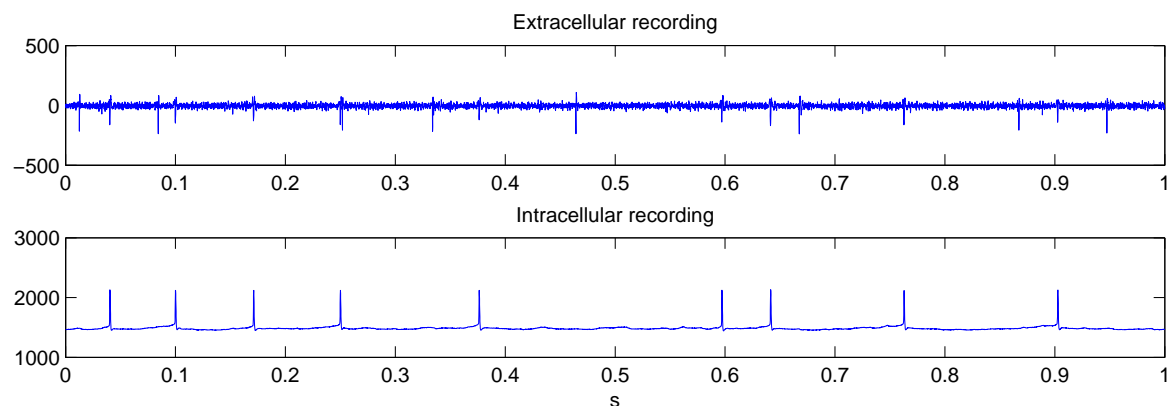
It is important to mention that the results of the proposed method will be relatively conservative. This conclusion comes from the way of finding the centers of the clusters. Since, due to the small value of  $R$ , two clusters can hardly be merged into one, the number of spikes wrongly associated to a particular cluster (false spikes) will be relatively small. Since false spikes can be very confusing and misleading for different algorithms that aim to decode the communication between neurons, in the trade-off between the number of missed spikes and the number of false spikes it is highly preferable to have more missed spikes. See [28] for detailed argumentation and proof of this statement.

### 3. Simulation and results

To analyze the performance of the proposed method we apply it on real extracellular neural recordings. We validate the results using intracellular recordings made simultaneously with the extracellular ones. These results are compared with the results of the state-of-the-art approaches.

### 3.1. Use of simultaneous intracellular and extracellular recordings for the validation and comparison

We use recordings (available at <http://crcns.org/data-sets/hc/hc-1>) from rat hippocampal area CA1 done by Henze et al., described in [1]. These data are particularly interesting because they consist of simultaneous intracellular and extracellular recordings.



**Figure 9.** Simultaneous extracellular and intracellular recording. The two electrodes are very close so the action potentials recorded with the intracellular electrode are clearly visible in the extracellular recording.

Single-channel extracellular spike sorting techniques usually result in sorting activities of two to three neurons. When the extracellular and the intracellular electrode are close enough, one of the neurons whose activity is sorted is probably the one recorded intracellularly. Figure 9 shows such simultaneous recording. The top plot shows the extracellular recording and the bottom plot shows the corresponding intracellular recording. The action potentials fired by the neuron recorded intracellularly are clearly visible on the extracellular recording. This indicates that the electrodes are located very close to each other.

On the plot of the extracellular recording we can also see the activities of other neurons around the extracellular electrode. Notice that many of these spikes have similar amplitudes as the extracellularly recorded spikes that come from the neuron recorded intracellularly. Thus, the task of spike sorting is obviously not trivial in this case as we will have more than one cluster.

Finally, we see that once we apply the spike sorting techniques on the signal recorded extracellularly we can validate the sorting results for one of the sorted neurons, the one which is recorded intracellularly.

From the available database we selected 8 recordings that satisfy the condition that the action potentials from the neuron recorded intracellularly are clearly visible on the extracellular recording. Another criterion for the selection was that the visual inspection of the feature vector space suggests that the sorting problem is not trivial (clusters far from each other) or too challenging (clusters completely overlapped). The

**Table 1.** Variances of the data projected on the first three principle components.

Recording	Variance of PC1	Variance of PC2	Variance of PC3
d533101 1	1.37	0.73	0.42
d533101 2	1.70	0.62	0.35
d533101 3	1.63	0.59	0.34
d533101 4	1.94	0.60	0.38
d12821001 3	5.51	1.88	1.34
d12821001 4	6.15	2.15	1.62
d1122109 4	0.99	0.30	0.24
d1122109 4	1.54	0.43	0.28

recordings we used here have from around 2.5 millions to around 10 millions of samples with 287 to 1197 spikes from the intracellularly recorded neuron and from to 1409 to 4788 detected spikes in total. They are recorded with different types of electrodes and sampling frequency either 10kHz or 20kHz, what gives an important diversity of the data.

### 3.2. Preprocessing of the extracellular recordings

First we detect the spikes from the extracellular recordings. To emphasis the spikes coming from neurons located close to the electrode we apply the spike detection technique described in [9] and [29]. For each of the detected spikes we extract a corresponding 50 samples long data vector.

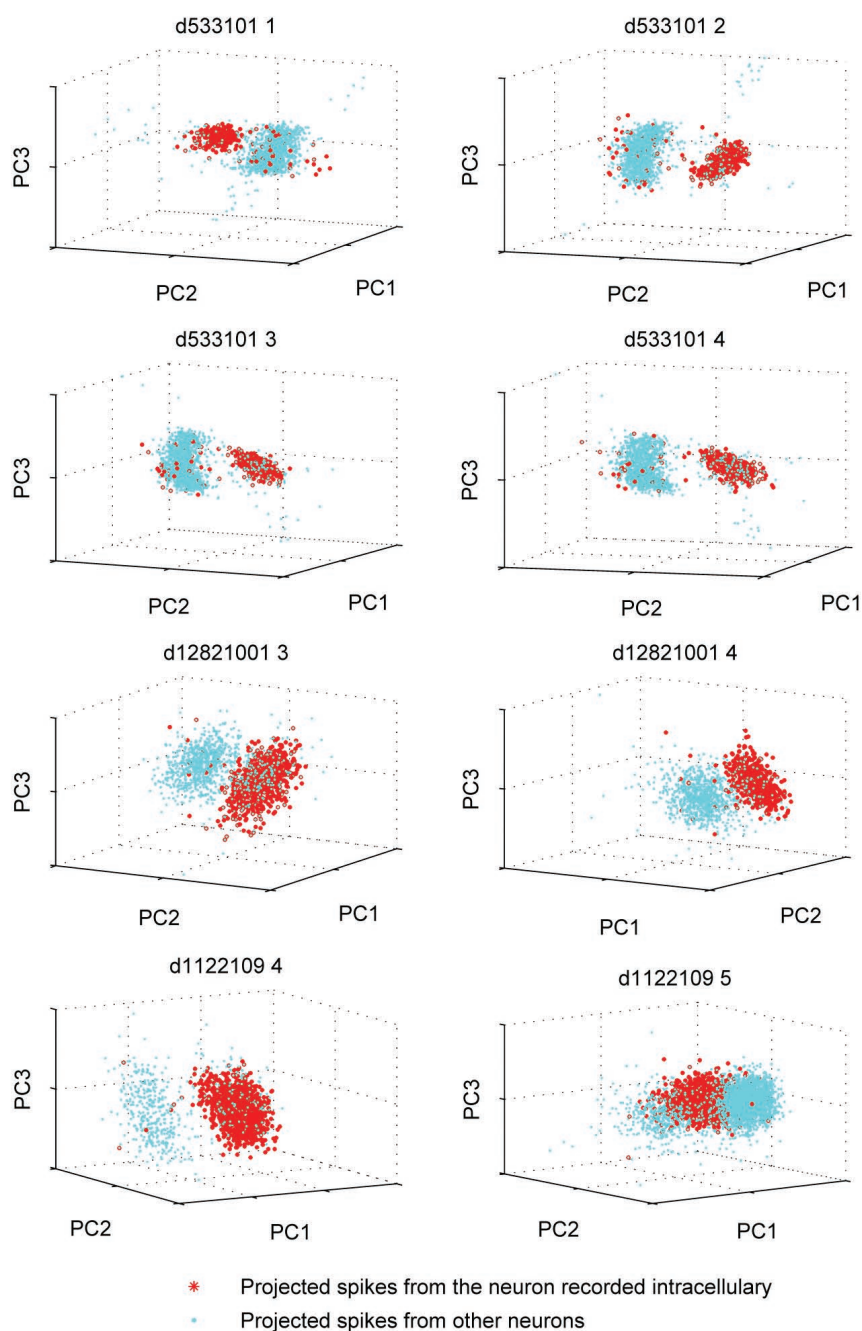
As the second step, we apply PCA for feature extraction. Figure 10 shows 8 plots, each corresponding with one of the signals (we kept the original signal names as in [1]). The plots show the projections on the first three principal components. The spikes that are recognized as coming from the neuron recorded intracellularly are marked with red stars. The rotation angle is adjusted for each plot to make the differences between the clusters more obvious. We can see that the projections on the third principle component do not contribute significantly to the separation of the spikes fired by the neuron recorded intracellularly from the rest of the spikes. On the other side, the projections on the first two principle components are much more informative and enable us to see differences between clusters. This is also elaborated with table 1 which gives variances of the first three principle components for each of the 8 signals. Thus, in the following, we keep only the projections on the first two principle components, that is we set  $L = 2$ .

### 3.3. Clustering: applying the proposed algorithm

To ease the comparisons, we scale  $PC1$  and  $PC2$  in the range  $[0, 100]$ .

We assume a minimal firing rate of 1 spike per second and use a 2D filter with size (window width and length) corresponding to  $R = 8$ . For all the recordings this value of  $R$  and this minimal firing rate meet the requirement of not merging two different clusters.

The centers of the clusters are found as the maxima of the filtered data, with a condition that two maxima are far enough from each other, as it is defined in subsection



**Figure 10.** All the spike from all 8 recordings projected on the first three principle components. Each plot is rotated in order to show differences between spikes from the intracellularly recorded neuron and all other spikes.

2.3. All the spikes are finally clustered using the algorithm 1.



**Table 2.** Comparison of spike sorting on 8 different recordings.  $C$  is the largest number of spikes from the neuron recorded intracellularly that are sorted into the same cluster (correct detections).  $F$  is the number of spikes from other neurons that are wrongly placed into the same cluster for which  $C$  is calculated (false spikes).  $T$  is the total number of spikes fired by an intracellularly recorded neuron. Sorting accuracy ( $SA = 100 * C / (F + C)$ ) is given in percentage (see *e.g.* [14] for details). We also give the percentage of missed spikes as  $MS = 100 * (T - C) / T$

recording	proposed				k-means				EM				superparamag.				$T$
	$C$	$F$	$SA$ %	$MS$ %	$C$	$F$	$SA$ %	$MS$ %	$C$	$F$	$SA$ %	$MS$ %	$C$	$F$	$SA$ %	$MS$ %	
d533101 1	240	64	80	16	237	61	80	17	234	62	79	18	287	1358	17	0	287
d533101 2	356	39	90	12	235	43	85	42	356	57	86	12	339	27	93	16	403
d533101 3	333	61	85	13	335	74	82	12	333	69	83	13	304	36	89	20	381
d533101 4	392	76	84	10	239	66	78	45	151	27	85	65	368	55	87	16	437
d12821001 3	714	69	91	5	724	97	88	4	600	43	93	20	599	28	96	20	753
d12821001 4	438	65	87	4	232	18	93	49	420	57	88	8	134	3	98	71	456
d1122109 4	1065	182	85	0	305	35	90	71	476	60	89	56	157	7	96	85	1070
d1122109 5	795	141	85	34	480	36	93	60	497	33	94	58	1197	3401	35	0	1197

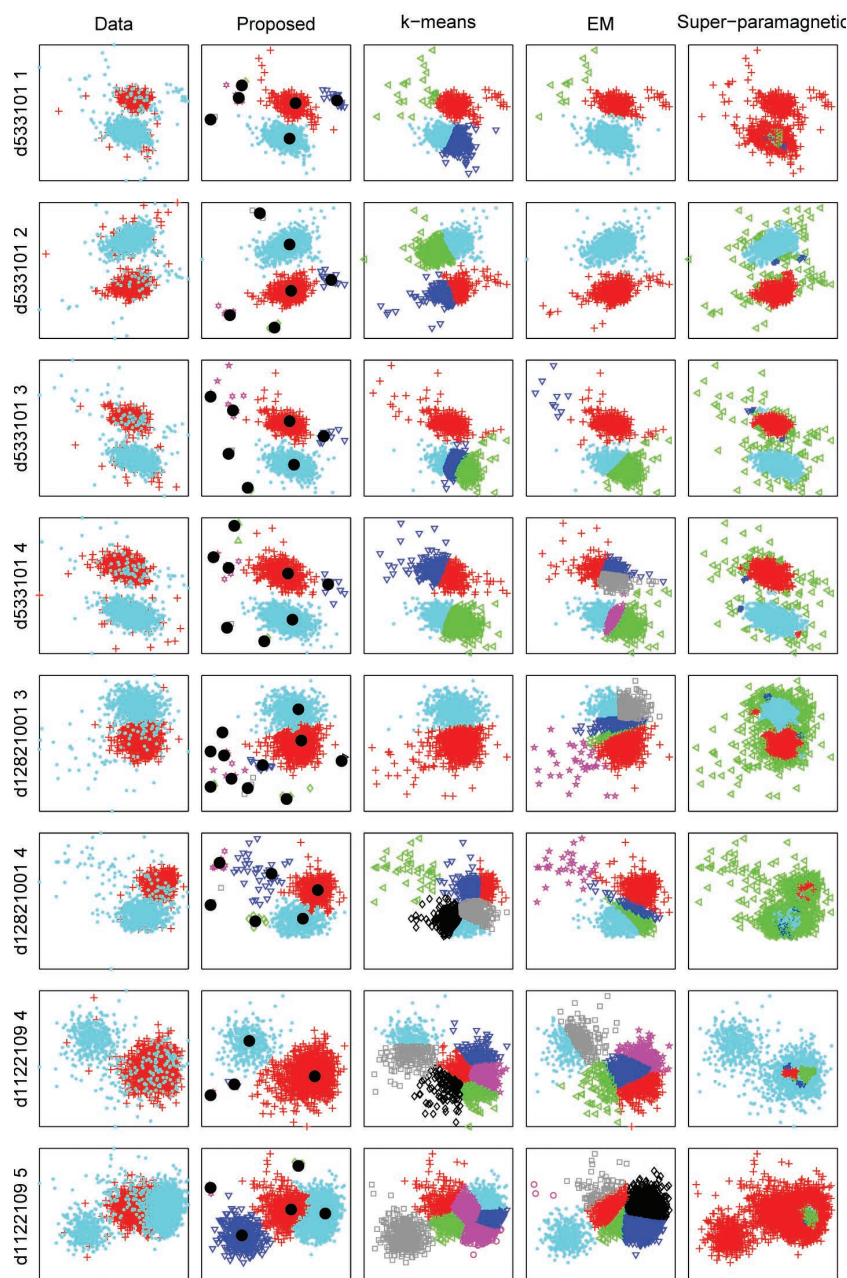
### 3.4. Results

We compared the proposed non-parametric approach with a combination of BIC with k-mean and EM algorithms and also with the super-paramagnetic clustering [11] which is *e.g.* used in spike sorting software called Wave-clus. These stand as the state-of-the-art approaches and are the most widely used among all spike sorting algorithms.

Figure 11 gives comparison of performance of the four algorithms on each of the 8 recordings. The spikes from the neuron which is recorded intracellularly are again represented with red stars in the first column. To quantify the clustering performance we observe how well the different algorithms managed to isolate that cluster. Since we do not have any knowledge on the activities of the other neurons we can compare the algorithms only in their ability to sort the spikes fired by the neurons recorded intracellularly (one neuron for each recording).

By observing closely the first column in figure 11 we see that, after PCA, several spikes from the neuron recorded intracellularly are located far from the area that obviously contains most of the spikes from that neuron. This is especially the case for recordings d533101 1, d533101 2, d533101 3 and d533101 4. We can also notice that several spikes that do not come from the neuron recorded intracellularly are in the middle of the area that contains most of its spikes. The best example is recording d1122109 4. Thus, we can not expect from any sorting algorithm to sort these spikes correctly, since they are obviously misplaced by the feature extraction technique. The results from figure 11 are also presented in table 2.

K-mean and EM based approach separate well spikes from the intracellularly recorded neuron for recordings d533101 1, d533101 3 and d12821001 3. EM based approach performs solid also on recordings d533101 2 and d12821001 4. For all the other recordings these approaches tend to split a single cluster into two or more clusters (*e.g.* the results for recording d1122109 4). Using k-means and EM, isolated elements located far from the cluster are often wrongly associated with that cluster, *e.g.* for k-means recording d12821001 3 and for EM d533101 2. Notice that we could actually



**Figure 11.** Comparison of performance of the four sorting algorithms on 8 recordings. Rows on the figure show the results on different recordings. The first column displays projection of each of the recordings on the first two principle components. Red stars represent spikes that come from the neuron recorded intracellularly while cyan dots represent all other spikes (from unknown number of neurons). The other columns give the sorting results of four different algorithms. Different clusters are marked with different symbols/color. For the proposed algorithm (second column) the black dots mark centers of the detected clusters.

obtain better results if we just fixed the number of clusters to two. However, such results would still be worse than the one of the proposed algorithm since there would be many falsely clustered spikes: all the spikes which seem to naturally fall out of both clusters

would be necessarily clustered in one of the two clusters. One possible solution to this problem could be to increase the fixed number of clusters. However, the simulation on figure 11 suggests that increasing the number of clusters rather results in splitting the clusters through the areas of high spike density, than in affecting the spikes which are far from these areas to the newly added clusters.

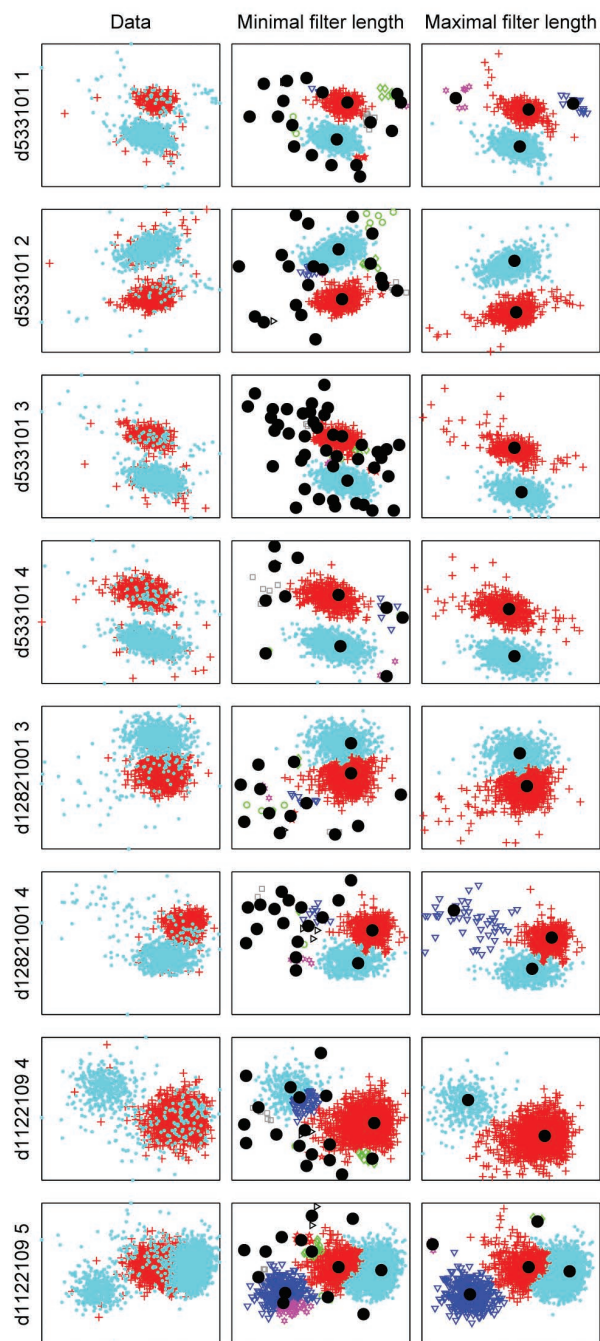
The results of the super-paramagnetic clustering are significantly different from the one of k-means or EM based approaches. They are sometimes very good (d533101 2, d533101 3, d533101 4) so for example the isolated spikes far from the cluster are left outside the cluster. For recording d12821001 3 the results are still satisfactory, but for recordings d533101 1, d12821001 4, d1122109 4 and d1122109 5 they are very bad, since almost all the detected spikes are sorted into same clusters.

The proposed approach performed generally very well. Even though in few situations several spikes that are located a bit far from the cluster are wrongly placed into the cluster (*e.g.* d533101 3 and d1122109 5), the bulk of the spikes that corresponds with the intracellularly recorded neurons are always recognized. Most of the isolated spikes are assigned to their own clusters (notice the black dots that show the locations of the centers of the clusters), which are after neglected since they contain less than  $G$  spikes.

Finally, we can say that the proposed approach compares favorably with all the three approaches. On these data the proposed approach always gave reasonably good results, without big oscillations.

Of course, to obtain a single-channel data where such separation is generally possible, the SNR has to be satisfactory and good performance of feature extraction technique is necessary. These are requirements of all the four approaches.

To analyze the robustness of the proposed approach we change  $R$ , the filter width and height, to see what are the limits for which the algorithm will still give satisfying results. The comparisons are given on figure 12 and in table 3. Small  $R$  leads to more conservative results, since most of the individual spikes, located a bit further from the area of highest density, are detected as clusters. These clusters contain less than  $G$  spikes so they are neglected. For large  $R$ , usually only the most important clusters are detected. Therefore, the results are likely to contain more false alarms (lower  $SA$ ). Table 3 indicates that satisfying results can be obtained for a large range of  $G$ . On vector space scaled on values from 0 to 100, choices of  $R = 7$  and  $R = 8$  lead to good results in all examples even though the clusters have different morphologies. Choosing too large value of  $R$  will usually result in detection of only one cluster what, as we said in subsection 2.4 should indicate that the data are too noisy for the clustering. This significantly increases the reliability of the algorithm, since it is usually preferable to characterize the data as too noisy than to obtain results that merge several clusters into one.



**Figure 12.** Analysis of robustness of the proposed approach. The first column is the same as on figure 11. The second column gives the sorting results for minimal  $R$  and third column for maximal  $R$  which still led to satisfying results.

#### 4. Discussion

In neural recordings the differences between some clusters may be very small. Meanwhile, as the firing rate can vary significantly from one neuron to another, the sizes of the clusters are usually strongly unbalanced. Also, in the spike detection phase,

**Table 3.** Analysis of robustness of the proposed approach. Columns  $\min R$  and  $\max R$  show the results obtained with the minimal and the maximal filter width and height respectively, such that the results are still considered as satisfactory (relatively well clustered activity of the intracellularly recorded neuron).  $G$  is the number of the elements that contained the largest filter window for the given  $R$  and, as before,  $C$  is the number of the correct detections,  $F$  the number of false alarms and  $T$  total number of the spikes fired by the intracellularly recorded neuron.

recording	min $R$						max $R$						$T$
	$R$	$G$	$C$	$F$	$SA$ %	$MS$ %	$R$	$G$	$C$	$F$	$SA$ %	$MS$ %	
d533101 1	4	158	237	43	85	17	14	959	240	61	80	16	287
d533101 2	4	132	351	37	90	13	22	1196	357	60	86	11	403
d533101 3	3	96	311	46	87	18	19	1143	334	80	81	12	381
d533101 4	6	211	390	74	84	11	23	1141	396	92	81	9	437
d12821001 3	7	132	714	49	91	5	22	668	714	100	88	5	753
d12821001 4	5	93	438	64	87	4	19	622	438	65	87	4	456
d1122109 4	5	77	1055	161	87	1	22	781	1065	182	85	0	1070
d1122109 5	5	373	790	111	88	34	8	808	795	141	85	34	1197

the firing activities of most of the remote neurons are very partially captured. Each of those neurons is thus represented by so few spikes that they can not be considered as separate clusters; they rather should be left unsorted. However, the classical solutions for the spike sorting problem are mostly based on the application of a general purpose model selection and sorting algorithms. Now, it is very unlikely that such a general purpose algorithm could comply with all the above mentioned specificities of neural recordings.

Deviating from these solutions, we proposed a non-parametric method that exploits some common properties of the neural signal and thus is particularly adapted for neural spike sorting. We assume that some minimal number of spikes per neuron is given as a prior information (minimal firing rate). This is not a strong requirement since generally we are not interested in isolating the activity of a neuron that fired only few spikes. We also assume that the recorded signal projected on some principle components has a unimodal distribution.

The method we devised has the common preprocessing part as most of spike sorting methods (spike detection and feature extraction). Afterwards, we start with the non-parametric approach by applying a moving average filter in the space projected by the feature extraction vectors (in our simulations principle components). This enhances the centers of the clusters which are then easy to find. We associate all the spikes with their corresponding centers using a greedy sorting algorithm close in spirit to the agglomerative hierarchical clustering and developed particularly for this application. The method can perform with no supervision, using the values of the parameters as given in this paper. However, to increase the accuracy, the user can optimize the filter width and height to the structure of the actual vector space. This optimization can be done for each recording or for a set of recordings when similar properties of the feature vector spaces are expected. Moreover, the filter width and height can be adjusted to obtain more or less conservative results.

We tested the proposed method on a specific type of neural recordings: simultaneous extracellular and intracellular recordings. We applied the method on the extracellular data and validated the results using the intracellular data. Comparing with the state-of-the-art approaches our method leads to significantly better results and shows very reliable performance.

The complexity of the proposed algorithm highly depends on the number of extracted features per spike, since this number defines the dimensionality of the filter. For feature extraction with PCA a good choice was to keep only the projections on the first two principle components which means that we needed to apply a  $2D$  spacial filter. When other feature extraction techniques are used this number of features could be insufficient, so to keep the processing fast an alternative to filtering that would still enhance the centers of the clusters should be implemented. The complexity also highly depends on the scale of the feature extraction space *e.g.* in the presented simulations we scaled the space to values from 0 to 100. If we denote lower and upper bound of  $n^{th}$  scaled feature extraction dimension as  $y_n^{min}$  and  $y_n^{max}$  then the number of operations  $NO_{filt}$  needed for the filtering can be expressed as  $NO_{filt} = \prod_{n=1}^L (y_n^{max} - y_n^{min})$ . The same number of operations is needed for finding the maxima as we need to analyze each element in the vector space. So the total number of operations is given by  $NO_{total} = 2 \prod_{n=1}^L (y_n^{max} - y_n^{min})$ .

In this paper we addressed the question of spike sorting on single-channel recordings. However, in [30] and in the upcoming journal paper we present an algorithm based on an iterative application of ICA on multi-channel recordings. The algorithm uses the method proposed here to cluster the data within each iteration. There, the property of unsupervised clustering is particularly important, since in an iterative algorithm any manual verification of the results within each iteration is not practical.

In its current form, the proposed method is obviously not implementable online. However, we may point out some possible modifications that would allow an online implementation. These concern essentially two points: 1) an initialization period, that provides all the centers of the clusters; 2) an online feature extraction method. These modifications will be investigated in detail in future work.

In order for the result to be reproducible all the Matlab code we used to present the results is available upon request to the email of the corresponding author.

## Acknowledgments

This work was supported by Ministry of Higher Education and Research, Nord-Pas de Calais Regional Council and FEDER through the 'Contrat de Projets Etat Region (CPER) 2007-2013'.

## References

- [1] D. A. Henze, Z. Borhegyi, J. Csicsvari, A. Mamiya, K. D. Harris, and G. Buzsaki. Intracellular Features Predicted by Extracellular Recordings in the Hippocampus In Vivo. *Journal of*

- Neurophysiology*, 84(1):390–400, July 2000.
- [2] M. S. Lewicki. A review of methods for spike sorting: the detection and classification of neural action potentials. *Network (Bristol, England)*, 9(4):53–78, November 1998.
  - [3] S. Gibson, J. W. Judy, and D. Markovic. Comparison of spike-sorting algorithms for future hardware implementation. *Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 1:5015–20, 2008.
  - [4] S. Takahashi, Y. Anzai, and Y. Sakurai. A new approach to spike sorting for multi-neuronal activities recorded with a tetrode—how ICA can be practical. *Neuroscience research*, 46(3):265–272, July 2003.
  - [5] A. Mamlouk, K. Menne, U. Hofmann, and T. Martinetz. Unsupervised spike sorting with ICA and its evaluation using GENESIS simulations. *Neurocomputing*, 65-66:275–82, June 2005.
  - [6] C. Pouzat, O. Mazor, and G. Laurent. Using noise signature to optimize spike-sorting and to assess neuronal classification quality. *Journal of Neuroscience Methods*, 122(1):43–57, Dec 2002.
  - [7] K. H. Kim and S. J. Kim. Neural Spike Sorting Under Nearly 0-dB Signal-to-Noise Ratio Using Nonlinear Energy Operator and Artificial Neural-Network Classifier. *IEEE Transactions on Biomedical Engineering*, 47(10):1406–1411, Oct 2000.
  - [8] Z. Nenadic and J. W. Burdick. Spike detection using the continuous wavelet transform. *IEEE Transactions in Biomededical Engineering*, 52:74–87, 2005.
  - [9] Z. Tiganj and M. Mboup. Spike detection and sorting: Combining algebraic differentiations with ica. In *Independent Component Analysis and Signal Separation, 8th International Conference*, pages 475–482, Paraty, Brazil, 2009.
  - [10] D. A. Adamos, E. K. Kosmidis, and G. Theophilidis. Performance evaluation of pca-based spike sorting algorithms. *Computer Methods and Programs in Biomedicine*, 91:232–244, September 2008.
  - [11] R. Q. Quiroga, Z. Nadasdy, and Y. B. Shaul. Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering. *Neural Computing*, 16(8):1661–1687, August 2004.
  - [12] E. Hulata, R. Segev, and E. Ben-Jacob. A method for spike sorting and detection based on wavelet packets and Shannon’s mutual information. *Journal of neuroscience methods*, 117(1):1–12, May 2002.
  - [13] Y. Ghanbari, L. Spence, and P. Papamichalis. A graph-Laplacian-based feature extraction algorithm for neural spike sorting. In *31st Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 3-6 Sept. 2009*, pages 3142–5, Piscataway, NJ, USA, 2009.
  - [14] E. Chah, V. Hok, A. Della-Chiesa, J. J. H. Miller, S. M. O’Mara, and R. B. Reilly. Automated spike sorting algorithm based on laplacian eigenmaps and k-means clustering. *Journal of neural engineering*, 8(1), February 2011.
  - [15] C. Rogers, J. Harris, J. Principe, and J. Sanchez. A pulse-based feature extractor for spike sorting neural signals. In *3rd Annual International Conference of the IEEE Neural Engineering, 2-5 May 2007*, Kohala Coast, HI, USA, 2007.
  - [16] M. Lavielle. Using penalized contrasts for change-point problem. *Signal Processing*, 85:1501–1510, 2005.
  - [17] D. Novak, J. Wild, T. Sieger, and R. Jech. Identifying number of neurons in extracellular recording. In *4th Annual International Conference of the IEEE Engineering in Medicine and Biology Society on Neural Engineering, April 29, 2009 - May 2, 2009*, pages 742–745, Antalya, Turkey, 2009. IEEE Computer Society.
  - [18] C. Fraley and A. E. Raftery. How many clusters? Which clustering method? Answers via model-based cluster analysis. *The Computer Journal*, 41:578–588, 1998.
  - [19] F. Wood, M. Fellows, J. Donoghue, and M. Black. Automatic spike sorting for neural decoding. In *26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Sep 1-5 2004*, volume 26 I, pages 4009–4012, 2004.
  - [20] M. Delescluse and C. Pouzat. Efficient spike-sorting of multi-state neurons using inter-spike

- intervals information. *Journal of neuroscience methods*, 150(1):16–29, January 2006.
- [21] K. D. Harris, D. A. Henze, J. Csicsvari, H. Hirase, and G. Buzsáki. Accuracy of tetrode spike separation as determined by simultaneous intracellular and extracellular measurements. *Journal of Neurophysiology*, 84:401–414, Jul 2000.
- [22] C. Vargas-Irwin and J. P. Donoghue. Automated spike sorting using density grid contour clustering and subtractive waveform decomposition. *Journal of neuroscience methods*, 164(1):1–18, August 2007.
- [23] D. J. Sebald and A. Branner. Automatic spike sorting for real-time applications. In *3rd Annual International Conference of the IEEE Engineering in Neural Engineering*, pages 670–674, 2007.
- [24] *Cerebus Data Acquisition System - users manual*. Blackrock Microsystems, 2009.
- [25] Y. Ghanbari, P. Papamichalis, and L. Spence. Graph-spectrum-based neural spike features for stereotrodes and tetrodes. In *ICASSP*, pages 598–601, March 2010.
- [26] T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning*. Springer, 2003.
- [27] M. S. Fee, P. P. Mitra, and D. Kleinfeld. Automatic sorting of multiple unit neuronal signals in the presence of anisotropic and non-Gaussian variability. *Journal of Neuroscience Methods*, 69(2):175–188, November 1996.
- [28] N. G. Ilan and H. J. Don. Information theoretic bounds on neural prosthesis effectiveness: The importance of spike sorting. In *ICASSP*, pages 5204–5207, 2008.
- [29] M. Mboup. A volterra filter for neuronal spike detection. *Research report, INRIA*, <http://hal.inria.fr/inria-00347048/fr/>, 2008.
- [30] Z. Tiganj and M. Mboup. Deflation technique for neural spike sorting in multi-channel recordings. *IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2011.