A Unified Optimization Model of Feature Extraction and Clustering for Spike Sorting

Libo Huang, *Student Member, IEEE*, Lu Gan, *Senior Member, IEEE*,
and Bingo Wing-Kuen Ling, *Senior Member, IEEE*

***Abstract*—Spike sorting technologies support neuroscientists to access the neural activity with single-neuron or single-action-potential resolutions. However, conventional spike sorting technologies perform the feature extraction and the clustering separately after the spikes are well detected. It not only induces many redundant processes, but it also yields a lower accuracy and an unstable result especially when noises and/or overlapping spikes exist in the dataset. To address these issues, this paper proposes a unified optimization model integrating the feature extraction and the clustering for spike sorting. Unlike the widely used combination strategies, i.e., performing the principal component analysis (PCA) for spike feature extraction and the K-means (KM) for clustering in sequence, interestingly, this paper finds the solution of the proposed unified model by iteratively performing PCA and KM-like procedures. Subsequently, by embedding the K-means++ strategy in KM-like initializing and a comparison updating rule in the solving process, the proposed model can well handle the noises and overlapping interference as well as enjoy a high accuracy and a low computational complexity. Finally, an automatic spike sorting method is derived after taking the best of the clustering validity indices into the proposed model. The extensive numerical simulation results on both synthetic and real-world datasets confirm that our proposed method outperforms the related state-of-the-art approaches.**

***Index Terms*—Spike sorting, PCA, K-means, unified optimization.**

Manuscript received September 20, 2020; revised January 16, 2021; accepted February 1, 2021. Date of publication April 20, 2021; date of current version April 26, 2021. This work was supported in part by the National Nature Science Foundation of China under Grant U1701266, Grant 61372173, and Grant 61671163; in part by the Team Project of the Education Ministry of the Guangdong Province under Grant 2017KCXTD011; in part by the Guangdong Higher Education Engineering Technology Research Center for Big Data on Manufacturing Knowledge Patent under Grant 501130144; in part by the Guangdong Province Intellectual Property Key Laboratory Project under Grant 2018B030322016; in part by the Hong Kong Innovation and Technology Commission, Enterprise Support Scheme under Grant S/E/070/17; and in part by the China Scholarship Council (CSC). *(Corresponding author: Bingo Wing-Kuen Ling.)*

Libo Huang is with the School of Information Engineering, Guangdong University of Technology, Guangzhou 510006, China, and also with the Department of Electrical and Computer Engineering, Brunel University London, London UB8 3PH, U.K. (e-mail: www.huanglibo@gmail.com).

Lu Gan is with the Department of Electronic and Electrical Engineering, Brunel University London, London UB8 3PH, U.K. (e-mail: lu.gan@brunel.ac.uk).

Bingo Wing-Kuen Ling is with the School of Information Engineering, Guangdong University of Technology, Guangzhou 510006, China (e-mail: yongquanling@gdut.edu.cn).

Digital Object Identifier 10.1109/TNSRE.2021.3074162

I. Introduction

EXTRACELLULAR cortical recording with silicon single- or tetrodes-microelectrodes plays an irreplaceable role in studying the information processing mechanisms within the nervous system [1], [2], which are beneficial in broad scopes of applications like neural prosthetics [3], brain-machine interfaces [4], treating epilepsy [5], etc. Although these applications require the neural activity with single-neuron or single-action-potential resolutions, these electrodes record the activities from multiple neurons surrounding the electrode [5], [6]. Fortunately, different neurons generate action potentials with a unique extracellular waveform (i.e., "spike") [7], [8]. With these unique waveforms, assigning each spike to its tentative neuron becomes possible, and this procedure is called spike sorting [7], [9]. Fig.1 shows an example in real scenarios that the recorded signals from single-microelectrode consist of several components such as the spikes from neurons near electrode tip in zone I, the local field potentials from neurons in zone II, and the noises from the distant neurons in zone III and the recording devices. As shown in Fig.1, conventional spike sorting techniques have to filter the signals first to remove most of the unwanted noises and disturbances before performing the spike detection [8]. Here, eight example spikes are detected. Afterward, features extracted in a low dimensional space (in Fig.1 this dimension is 2) and clustering the featured spikes are needed, which are quite challenging compared with the filtering and detection stages [10].

Over the past decades, numerous studies have been devoted to extracting discriminative spike features and clustering the featured spikes. There are two categories of the methods for spike feature extraction: global-features-based and local-features-based ones. The global-features-based methods employ principal components analysis (PCA) [11], Fourier transform [12], wavelet decomposition [13], [14], etc. to extract features. Local-features-based methods are those called locality preserving projection (LPP) [15], Laplacian eigenmaps (LE) [16], and graph-Laplacian (GL) [17], etc. On the other hand, the clustering methods such as K-means (KM) [16], [18], spectral clustering (SC) [15], [19], Gaussian mixture model (GMM) [18], [20] are flourishing. Directly combining the spike feature extraction and the clustering, i.e., separately performing them in sequence, has achieved a certain degree of success.

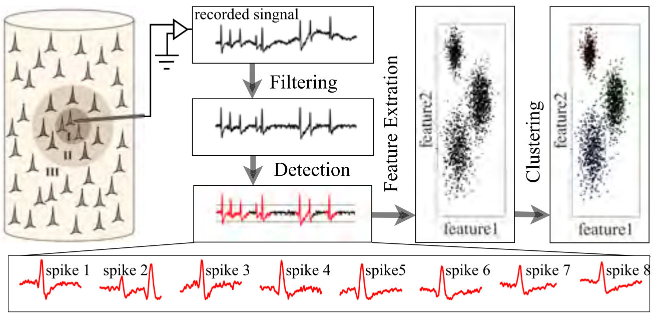
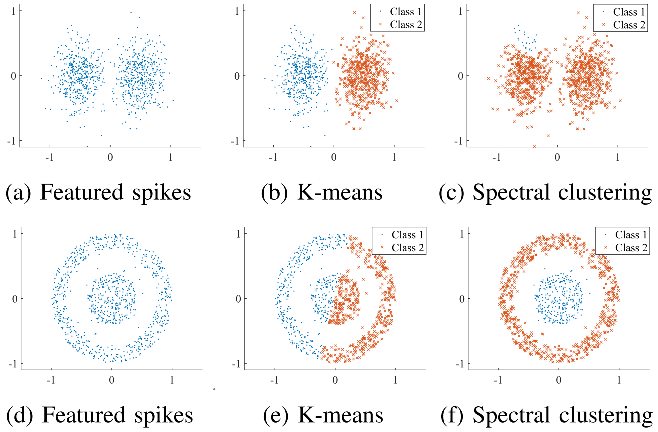


Fig. 1. The conventional spike sorting framework.



(a) Featured spikes (b) K-means (c) Spectral clustering

(d) Featured spikes (e) K-means (f) Spectral clustering

Fig. 2. (a) and (d) are the featured spikes projected to 2 dimensions which are obtained from the global-based and the local-based feature extraction methods, respectively. (b) and (e) show the results from K-means while (c) and (f) are from the spectral clustering. For the spikes distributed in (a), K-means performs better than spectral clustering. Conversely, for the spikes distributed in (d), the conclusion is opposite.

However, it is impractical to exhaust all kinds of potential combinations. What's worse, some combinations may yield unsatisfactory and unstable results for different datasets, especially when there are noises and/or overlapping spikes interference. For an instance, suppose that there are two noise-disrupted datasets obtained from the global-features-based and the local-features-based methods as shown in Fig. 2a and Fig.2d, respectively. For the global-featured spikes as shown in Fig.2a, a satisfactory result is yielded with KM method (Fig. 2b), but an unsatisfactory result is achieved with SC (Fig. 2c). Conversely, for the local-featured spikes as shown in Fig. 2d, the opposite conclusions are drawn. That is, the result with SC is satisfactory (Fig. 2f) while that with KM method is not (Fig. 2e).[1]

Such problems mainly originate from the following three reasons. 1) The internal relationships between the feature extraction and the clustering methods have not been considered when they are directly combined for the spike sorting. 2) The features directly extracted from the detected spikes are sensitive to the interference from noises and/or overlapping spikes but the feedback from the clustering stage has not been considered. 3) The unsupervised clustering methods are sensitive to the initial values and may fall into an unsatisfactory local optimum.

Recently, Keshtkaran and Yang [18], [21] introduced linear discriminant analysis guided by K-means (LDAKM) [22], which tried to join LDA and KM into a coherent framework, to model the feature extraction and clustering problem in the spike sorting. Subsequently, they take an additional GMM process after each KM iteration, proposing LDAGMM [18] to detect the overlapping spikes. Unfortunately, LDAKM confuses the matrix ratio trace together with trace ratio criteria in the framework, resulting in the outputs oscillation or even divergence [23], [24]. Compare with LDAKM, LDAGMM forces different algorithms to be linked together, which may further distract the original LDAKM model. Along with this distraction in LDAGMM, more iterations and computation resources are required. Also, both LDAKM and LDAGMM may still yield unstable results. On the other hand, Hou *et al.* [25] also have tried to join dimensionality reduction and clustering into a unified model by directly combining PCA and KM (PCAKM) with a balance parameter in data mining fields. However, the parameter in PCAKM is hard to determine and significantly affects the final results.

Inspired by these efforts [18], [21], [25], this paper proposes a unified optimization model of the feature extraction and the clustering for spike sorting. In particular, these two stages are with the identical criterion without any balance parameter and alternatively updated using the alternating direction method [26]. In this case, there is a feedback from the clustering in the feature extraction stage. Further, the problem of instability can be solved by embedding the K-means++ in the initialization and one comparison updating strategies [25] in the clustering step. Finally, an automatic spike sorting framework is derived after estimating the number of neurons by engaging the existing clustering validity indices. The contributions of this paper are mainly three-fold,

- This paper proposes an automatic spike sorting framework which not only achieves a high accuracy but also enjoys a low computational complexity. Moreover, the framework guarantees that the results are stable no matter the noises and/or the overlapping spikes exist or not.
- This paper theoretically derives the solution of the proposed unified model and also proves that it can be converted into either the PCA or the KM-like process in each alternate updating. Besides, the analysis of the computational complexity is provided.
- This paper conducts extensive numerical simulation on both synthetic and real-world datasets. The results show that the objective function values of the iteratively solving algorithm are monotonic. Therefore, the efficacy and stability of the proposed method are verified.

The rest of this paper is organized as follows. Section II presents our automatic spike sorting framework. In particular, the unified feature extraction and the clustering model are formulated and the solution is obtained analytically. Section III presents and discusses the numerical simulation results. Finally, this paper is concluded and some possible future works are suggested in Section IV.

[1] Note that different initial values will result in different outputs for both the KM and the SC, and one of them is shown randomly.

II. Proposed Methods

The Filtering and the amplitude threshold detection in [14] and [27] are engaged in our framework. Note that, the particular parameters in these two steps for different datasets are, by default, assigned according to [14], [27], and more details about these default settings will be pointed out in Section III. After spike filtering and detection, each detected spike is stored as a column vector in the matrix $X \in \mathbb{R}^{d\times n}$. Here, d and n stand for the number of variables (i.e., dimensions) and the number of observations (i.e., spikes), respectively.

A. Theoretic Representation and Optimization of PCA and KM

For easy representation, the theoretical representation and optimization of PCA and KM are presented first.

1) PCA: Given a detected spike dataset $X \in \mathbb{R}^{d\times n}$, the target of PCA is to construct an orthogonal linear projection matrix $W \in \mathbb{R}^{d\times m}$ so that the projected spikes in m linear spaces have the largest data variances. Here, m is the reduced dimension. To this end, a centralization matrix is firstly formed by [24],

$$H = I_n - \frac{1}{n}N \in \mathbb{R}^{n\times n}, \tag{1}$$

where I_n and N are, respectively, the identity matrix of size n and the all-ones matrix, in which every element is 1. And H is a symmetric and idempotent matrix [25], i.e., $H = H^T = HH^T$. Then, the spike matrix, X, is centralized by multiplying H in the right hand side, i.e., XH. Finally, the PCA finds W on the scatter matrix, $XH(XH)^T = XHX^T$, as,

$$\begin{aligned} &\max_{W} \operatorname{tr}\left(W^T XHX^T W\right), \\ &s.t., \;\; W^T W = I_m. \end{aligned} \tag{2}$$

where $\operatorname{tr}(\cdot)$ is the trace operator, and the above problem can be solved by performing eigenvalue decomposition (EVD) [25] on XHX^T. Let U be the matrix with its column vectors being the eigenvectors of XHX^T. Let Λ be a diagonal matrix with its diagonal elements being the corresponding eigenvalues of XHX^T. So, $XHX^T = W\Lambda W^{-1}$, and the optimal solution, W^*, of Problem (2) is the m eigenvectors in U corresponding to the largest m eigenvalues in Λ.

2) KM: In KM, a label indicator matrix, $G \in \mathbb{R}^{n\times c}$, is constructed by the rule that $G_{i,j} = 1$ if i-th spike belongs to j-th class and $G_{i,j} = 0$ for others, in which c is the number of classes (i.e., neurons), $G_{i,j}$ is the element in i-th row and j-th column of G for $i = 1, \ldots, n$ and for $j = 1, \ldots, c$. The c centers of all classes are stored as columns in the matrix $M \in \mathbb{R}^{d\times c}$. Then, the entire KM clustering process can be formulated as the following optimization problem and one of the appropriate local optimal solutions can be found using the expectation maximization (EM) algorithm [28], [29],

$$\begin{aligned} &\min_{(G,M)} \left\| XH - MG^T \right\|_F^2, \\ &s.t., \;\; G\mathbf{1}_c = \mathbf{1}_n, \;\; G^T\mathbf{1}_n \succ \mathbf{1}_c, \;\; G_{i,j} \in \{0,1\}, \end{aligned} \tag{3}$$

where $\|X\|_F = \left(\sum_{i=1}^{d}\sum_{j=1}^{n} |X_{i,j}|^2\right)^{\frac{1}{2}}$ is the Frobenius norm on matrix X, $\mathbf{1_c} \in \mathbb{R}^{c\times 1}$ is an all-ones column vector, and $X \succ Y$ indicates $X_{i,j} > Y_{i,j}$ with element-wise inequality.

Theorem 1: (3) is equivalent to the following problem,

$$\begin{aligned} &\min_{G} \operatorname{tr}\left(XH(I_n - G(G^TG)^{-1}G^T)HX^T\right), \\ &s.t., \;\; G\mathbf{1}_c = \mathbf{1}_n, \;\; G^T\mathbf{1}_n \succ \mathbf{1}_c, \;\; G_{i,j} \in \{0,1\}. \end{aligned} \tag{4}$$

Proof: Denote $f(G,M) = \left\|XH - MG^T\right\|_F^2$. Since $\|X\|_F^2 = \operatorname{tr}(XX^T) = \operatorname{tr}(X^TX)$, we have,

$$f(G,M) = \operatorname{tr}\left(XHX^T - 2MG^THX^T + MG^TGM^T\right).$$

Since $\frac{\partial \operatorname{tr}(XY^T)}{\partial X} = Y$ and $\frac{\partial \operatorname{tr}(XYX^T)}{\partial X} = X\left(Y + Y^T\right)$, $\frac{\partial f(G,M)}{\partial M} = 0$ implies that,

$$M = XHG\left(G^TG\right)^{-1}. \tag{5}$$

Here $\partial(\cdot)$ is the differential operator. Substitute (5) into (3), (4) is obtained and the proof is completed. □

Theorem 1 indicates that these two steps in the KM algorithm solved by the EM mechanism with iteratively updating two decision matrices could be simplified to a single step with only one decision matrix.

B. Unification of PCA and KM for Spike Feature Extraction and Clustering

In the conventional spike sorting procedure, KM clustering is performed on the featured spikes, which is obtained by applying PCA on the centralized data matrix, XH. That is, the KM algorithm is operated on W^TXH. Besides, PCA is a maximization problem with a single decision matrix, W. The KM is transformed to a minimization problem with a single decision matrix, G, as indicated in Theorem 1. By employing a trace operation on the ratio of matrices [30], these two optimization problems with individual decision matrix for performing respectively the spike feature extraction and the clustering can be unified to an optimization problem with two decision matrices defined as follows,[2]

$$\begin{aligned} &\max_{(G,W)} \operatorname{tr}\left(\frac{W^TXHX^TW}{W^TXH(I_n - G(G^TG)^{-1}G^T)HX^TW}\right), \\ &s.t., \;\; G\mathbf{1}_c = \mathbf{1}_n, \;\; G^T\mathbf{1}_n \succ \mathbf{1}_c, \;\; G_{i,j} \in \{0,1\}, \\ &W^TW = I_m. \end{aligned} \tag{6}$$

After that, the alternating direction method is employed, which guarantees to converge when the number of decision variables is two [26]. The details for finding the two decision matrices of problem (6) are as follows.

1) Update G: When updating G, the constraint on W can be ignored. Therefore, the maximization problem (6) is equivalent to the following minimization problem. That is,

$$\begin{aligned} &\min_{G} \operatorname{tr}\left(\frac{W^TXH(I_n - G(G^TG)^{-1}G^T)HX^TW}{W^TXHX^TW}\right), \\ &s.t., \;\; G\mathbf{1}_c = \mathbf{1}_n, \;\; G^T\mathbf{1}_n \succ \mathbf{1}_c, \;\; G_{i,j} \in \{0,1\}. \end{aligned} \tag{7}$$

[2] Generally speaking, there are at least four different methods to integrate these two optimization problems with individual decision matrix into a unified optimization problem with two decision matrices. They are ratio trace, ratio determinant, trace linear combination, and trace ratio methods as shown in [30]. However, this paper only focuses on the ratio trace method because it is convenient to optimize and present.

According to Theorem 1, solving the above problem equals to performing the following KM-like procedure,

$$\min_{(G,M)} \left\| \left(W^T XHX^T W\right)^{-\frac{1}{2}} W^T XH - MG^T \right\|_F^2,$$
$$s.t.,\ \ G\mathbf{1}_c = \mathbf{1}_n,\ \ G^T\mathbf{1}_n \succ \mathbf{1}_c,\ \ G_{i,j} \in \{0,1\}. \tag{8}$$

To obtain a better local optimal solution, the following two strategies are employed,

- Different from the random initialization of the class centers, the well-known K-means++ [31] initialization strategy is employed.
- To find a near global optimal solution, a comparison updating rule [25] with the details discussed below is employed.

Updating Rule: Suppose that, after the t-th iteration, we obtain the solutions of Problem (6), W_t^* and G_t^* along with the corresponding M_t^* calculated by the following equation, whose derivation is the same as (5),

$$M_t^* = \left(W_t^{*T} XHX^T W_t^*\right)^{-\frac{1}{2}} W_t^{*T} XHG_t^* \left(G_t^{*T} G_t^*\right)^{-1}.$$

Define

$$g(G,M) = \left\| \left(W^T XHX^T W\right)^{-\frac{1}{2}} W^T XH - MG^T \right\|_F^2.$$

Before updating G at the $(t+1)$-th iteration, $M_{t+1}^1, M_{t+1}^2, \ldots, M_{t+1}^a$ are first initialized using the K-means++ strategy. Then, the corresponding $G_{t+1}^1, G_{t+1}^2, \ldots, G_{t+1}^a$ are calculated. Here, a is the number of initializations, which is chosen as 10 in this paper. And G is updated with the following criterion,

$$G_{t+1}^* = \begin{cases} G_{t+1}^i, & \text{if } g(G_{t+1}^i, M_{t+1}^i) < g(G_t^*, M_t^*); \\ G_{t+1}', & \text{otherwise,} \end{cases} \tag{9}$$

where $i = 1, 2, \ldots, a$, and G_{t+1}' is the solution of the following problem,

$$\min_{(G,M)} \left\| \left(W_t^{*T} XHX^T W_t^*\right)^{-\frac{1}{2}} W_t^{*T} XH - MG^T \right\|_F^2,$$
$$s.t.,\ \ G\mathbf{1}_c = \mathbf{1}_n,\ \ G^T\mathbf{1}_n \succ \mathbf{1}_c,\ \ G_{i,j} \in \{0,1\}. \tag{10}$$

2) Update W: When updating W, taking G as a constant, (6) can be recast as the following optimization problem,

$$\max_W \operatorname{tr}\left(\frac{W^T XHX^T W}{W^T XH(I_n - G(G^TG)^{-1}G^T)HX^T W}\right),$$
$$s.t.,\ \ W^T W = I_m. \tag{11}$$

The above optimization problem is convex with a closed-form solution. Let $S_1 = XH\left(I_n - G\left(G^TG\right)^{-1}G^T\right)HX^T$ and $S_2 = XHX^T$. Then, (11) can be simplified as follow,

$$\max_W \operatorname{tr}\left(\frac{W^T S_2 W}{W^T S_1 W}\right),$$
$$s.t.,\ \ W^T W = I_m. \tag{12}$$

(12) is the same as the (2) when $S_1 = I_d$. Its optimal solution, W^*, can be derived by picking up m generalized eigenvectors corresponding to the largest m eigenvalues of the following generalized eigenvalue problem [18],

$$S_2 W = S_1 W \Gamma \tag{13}$$

where Γ is the diagonal matrix whose diagonal elements are the generalized eigenvalues. Finally, Alg.1 is summarized to find the solution of our proposed model.

Algorithm 1 Algorithm for Finding the Solution of (6)

```
Input: The detected spikes $X \in \mathbb{R}^{d\times n}$ and the number of neurons $c$.
Output: Transformation matrix $W^* \in \mathbb{R}^{d\times m}$ and the allocation matrix of spikes $G^* \in \mathbb{R}^{n\times c}$.
1: Initialize $m = c - 1$ and $t = 1$; Define $H \in \mathbb{R}^{n\times n}$, $W_0 \in \mathbb{R}^{d\times m}$, and $G_0 \in \mathbb{R}^{n\times c}$ according to (1), (2), and (3), respectively.
2: Update $W_t$ based on (13).
3: Update $G_t$ based on (9).
4: Terminate the algorithm if $G_t = G_{t-1}$. Otherwise, increment the value of $t$ and go back to Step 2.
5: Return the matrices $G^* = G_t$ and $W^* = W_t$.
```

C. Descriptions of the Unified Model

In this subsection, the computational complexity of our proposed model is analyzed and the automatic schemes are presented for estimating the number of neurons and determining the dimension of reduced features.

1) Analysis of the Computational Complexity: The computational complexity of our proposed model mainly on the following three parts: the calculation of $W^T XHX^T W$ when W is known, finding the solution of (10) with KM algorithm, and finding the solution of (13) with the generalized EVD. Let d, m, n, and c be the original dimension of each spike, the reduced dimensions of each spike, the total numbers of spikes, and the number of neurons, respectively. In fact, the required computational complexity for calculating $W^T XHX^T W$ is $O(d^2m + m^2d)$ with the constructed $XHX^T \in \mathbb{R}^{d\times d}$. The required computational complexity of KM is $O(nmc)$ [29], while that of the generalized EVD is $O(d^2n)$ [25]. Therefore, the total complexity is $O(b(d^2m + m^2d + nmc + d^2n))$, where b is the time of iteration. Since $n \gg d > m, c$ and $b \approx 10$ hold in the single electrode or the tetrode extracellular recorded signal, it can be further approximated by $O(bd^2n)$ which is linear to the number of the spikes.

2) Automatic Estimation of the Number of Neurons: Although many spike sorting methods determine the number of neurons by manually specifying or graphically merging and decomposing the clusters by hand [17], [27], [32], integrating the existing estimation methods into the model can reduce a lot of unnecessary human intervention [21]. To this end, this estimation is integrated into the initialization stage of our proposed model. This is performed after using the PCA method, (2), to initialize feature extraction before finding the solution of the main model. In other words, the estimation of the number of neurons is independent of finding the solution of the model, and it can be achieved by taking the best of

Algorithm 2 Proposed Automatic Spike Sorting Procedure

```
Input: The recorded spike signal and the initial reduced dimension $m_0$.
Output: The transformation matrix $W^* \in \mathbb{R}^{d \times m}$, the allocation results of spikes $G^* \in \mathbb{R}^{n \times \tilde{c}}$, and the estimated number of neurons $\tilde{c}$.
1: Filter and detect the recorded signal to get the spikes matrix $X \in \mathbb{R}^{d \times n}$.
2: Initialize $t = 1$. Define $H \in \mathbb{R}^{n \times n}$ and $W_0 \in \mathbb{R}^{d \times m_0}$ by finding the solution of (1) and (2), respectively.
3: Estimate the number of neurons $\tilde{c}$ and initialize the indicator matrix $G_0 \in \mathbb{R}^{n \times \tilde{c}}$ simultaneously by finding the solution of (3) with $W_0$ and the existing clustering validity indices.
4: Update $W_t$ based on (13).
5: Update $G_t$ based on (9).
6: Terminate the algorithm if $G_t = G_{t-1}$. Otherwise, increment the value of $t$ and go back to Step 4.
7: Return $G^* = G_t$, $W^* = W_t$ and $\tilde{c}$.
```

various existing clustering validity indices. More specifically, the KM clustering method, (3), is employed to assist this estimation and initialize the indicator matrix. To show its validity, this paper employed several widely used clustering validity indices. For example, Calinski Harabasz [33], [34], gap index [15], [35], [36], Davies-Bouldin [12], Silhouette index [35], Bayesian information criterion (BIC) [37], Akaike information criterion (AIC) [32], l-ratio [38], and isolation distance [39] are employed. For other cluster validity indices used to estimate the number of neurons, please refer to [34]. With initially setting a range of the number of neurons, the KM clustering method is performed first to obtain the corresponding index values. Then one reasonable number of the neurons is picked up based on the best index value [12], [35]. Note that, the initial-reduced dimension, m_0, is only associated with the estimating output and can be used to decide the hyper-parameter in the engaged clustering validity indices (e.g., the degree of freedom in BIC or AIC). So, it can be chosen as an input parameter in our method.

3) Automatic Determination of the Dimension of Features: To efficiently reduce the required computational complexity and further free human intervention, our method automatically determines the dimensions of spikes. Specifically, m_0 is the initially reduced dimensions, and it only affects the estimation of the number of neurons. After estimating the number of neurons $\tilde{c}$, $m = \tilde{c} - 1$ is chosen. Here, m denotes the reduced dimension of each spike in the iteratively solving algorithm. This strategy not only reduces the required computational complexity in each iteration but avoids the occurrence of singularity [23] in the EVD. Finally, the proposed spike sorting method is summarized in Alg.2.

III. Numerical Simulation Results

The proposed automatic spike sorting method is evaluated on both several well-known simulated datasets and a real tetrode recorded dataset. Further, it is compared with several related state-of-the-art spike sorting methods based on different evaluation criteria. All codes of our simulation implemented in MATLAB R2020a were run on a personal computer with Intel Core i7-8750H 2.20 GHz CPU, 32G RAM.

A. Datasets and Evaluation Criteria

Four challenging synthetic neural datasets of Wave_clus [14], [40], which contain ground truth data, are employed in our numerical simulations. They are Easy1_noise*, Easy2_noise*, Difficult1_noise*, and Difficult2_noise*, in which the asterisk stands for the embedded noise standard deviation to the main waveforms (i.e., noise level). Here, the waveforms were issued by three neurons in the neocortex and basal ganglia [40], and the noise was mimicked by randomly selecting the real recorded signal with different amplitudes issued by different neurons [14]. Besides, 'Easy' reflects the used waveforms are easy to distinct (i.e., they are very different from each other) and vice versa for the 'Difficult'. And the noise levels of 'Easy1' range from 0.05 to 0.4 with step-length 0.05, while those of the rest three datasets are in [0.05, 0.2].

On the other hand, the publicly available vivo dataset HC1 [41] is also used in our numerical simulations. HC1 was recorded from the hippocampus of an anesthetized rat using an extracellular tetrode. Here, one extra intracellular electrode was implanted to a single neuron to simultaneously record the release times of its actual action potential. These times are referred to the ground truth and can be used for the evaluation [27], [42].

The performance of determining the number of neurons, clustering accuracy, time consumption, and model's monotonicity are evaluated on the synthetic datasets in our simulations. In particular, to reflect the stability, 20 independent trials were conducted, and the corresponding means and the standard deviations were recorded. As for the real-world dataset HC1, the false-negative rate (FNR, or called miss rate) and the false-positive rate (FPR) criteria are used. Here, the false-negative (FN) represents that the predicted label is negative, but the real class is true, and the false-positive (FP) means the predicted label is positive, but the real class is false. FNR and FPR are reasonable enough to reflect the model quality, and both lower values of them refer to the better performance of the model [18], [27].

B. Numerical Simulations on the Synthetic Dataset

Same as the previous works [14], [40], "Wave_clus" were filtered with passband located at $300 - 6,000$ Hz first. But different from the default configuration that using amplitude thresholds for spike detection, the spikes are acquired by using a 64-sample-length window from the known spike release time in the ground truth. That facilitates to get the consistent spikes to perform a fair comparison with different methods and avoid the detected errors introduced by the data preprocessing.

1) Determination of the Number of Neurons: Four automatic spike sorting methods (LE-KM [16], LPP-SCK [15], LDAKM [21], LDAGMM [18] and the baseline method [14] on Wave_clus (Wave_clu*) are engaged to estimate the number of neurons. Here, SCK is short for landmark-based spectral

TABLE I
THE ESTIMATED NUMBER OF THE NEURONS USING VARIOUS AUTOMATIC SPIKE SORTING METHODS ON WAVE_CLUS DATASETS WITH (WITHOUT) OVERLAPPING SPIKES

			State-of-the-art Methods					Proposed Method with Different Strategies to Estimate the Number of Neurons							
DS	NL	SN(O)	LE-KM	LPP-SCK	LDA KM	LDA GMM	Wave _clu*	Calinski Harabasz	gap index	Davies-Bouldin	Silhouette index	BIC	AIC	l-ratio	isolation distance
Easy1	05	3514(2729)	2(3)	2(10)	2(4)	2(3)	3	3(3)	3(5)	3(3)	4(3)	11(10)	11(11)	3(3)	3(3)
	10	3522(2753)	3(3)	6(9)	2(4)	2(5)	3	3(3)	3(3)	3(3)	3(3)	3(3)	3(3)	3(3)	3(3)
	15	3477(2693)	8(2)	2(4)	2(3)	2(4)	3	3(3)	3(3)	3(3)	3(3)	3(3)	3(3)	3(3)	3(3)
	20	3474(2678)	3(2)	3(7)	2(5)	2(5)	3	3(3)	3(3)	3(3)	3(3)	3(3)	3(3)	3(3)	3(3)
	25	3298(2586)	2(7)	2(2)	2(3)	2(4)	6	3(3)	3(3)	3(3)	3(3)	3(3)	3(3)	3(3)	3(3)
	30	3475(2629)	3(2)	8(3)	2(5)	2(5)	3	3(3)	3(3)	3(3)	3(3)	3(3)	3(7)	3(3)	3(3)
	35	3534(2702)	3(3)	4(2)	2(5)	4(5)	4	3(3)	3(3)	3(3)	3(3)	3(3)	3(3)	3(3)	3(3)
	40	3386(2645)	4(3)	2(2)	2(4)	2(4)	3	3(3)	3(3)	3(3)	3(3)	3(3)	3(3)	3(3)	3(3)
Easy2	05	3410(2619)	7(4)	3(7)	2(5)	2(2)	6	3(3)	3(5)	2(2)	2(2)	11(6)	3(6)	2(3)	4(3)
	10	3520(2694)	3(3)	7(10)	2(3)	2(3)	3	2(3)	3(3)	2(2)	2(2)	3(3)	3(3)	2(2)	3(3)
	15	3411(2648)	8(2)	6(6)	3(3)	2(3)	3	2(2)	3(3)	2(2)	2(2)	3(3)	3(3)	2(2)	3(3)
	20	3526(2715)	3(2)	9(10)	2(3)	2(4)	3	2(2)	3(3)	2(2)	2(2)	2(2)	2(2)	2(2)	3(3)
Difficult1	05	3383(2616)	6(7)	6(10)	2(4)	2(4)	4	3(6)	8(6)	3(3)	3(3)	11(7)	11(8)	3(2)	3(3)
	10	3448(2638)	3(2)	2(3)	2(4)	2(6)	3	3(3)	3(3)	3(3)	3(3)	3(3)	3(3)	3(3)	3(3)
	15	3472(2660)	2(2)	2(3)	2(6)	2(4)	3	3(3)	2(2)	7(7)	3(3)	2(2)	2(2)	3(3)	3(3)
	20	3414(2624)	4(3)	2(2)	2(5)	3(7)	4	3(3)	2(2)	8(10)	3(3)	2(2)	2(2)	8(8)	3(3)
Difficult2	05	3364(2535)	8(3)	3(8)	2(4)	2(3)	4	3(3)	3(3)	2(3)	7(3)	11(3)	11(3)	3(3)	3(3)
	10	3462(2742)	9(3)	3(6)	2(5)	2(4)	3	2(3)	3(6)	2(2)	2(2)	3(4)	3(5)	2(3)	3(3)
	15	3440(2631)	9(3)	2(10)	2(3)	2(3)	4	2(2)	3(3)	2(2)	2(2)	2(4)	2(4)	2(2)	3(3)
	20	3493(2716)	8(4)	2(9)	3(4)	2(4)	3	2(2)	2(2)	2(2)	2(2)	2(2)	2(2)	2(2)	3(3)
Total Number of Hits			8(7)	4(3)	2(6)	2(5)	13	14(15)	16(13)	10(11)	11(13)	11(11)	12(10)	12(13)	19(20)

DS: name of dataset; NL: noise level; SN(O): the number of spikes with overlapping spikes (without overlapping spikes)

clustering in which KM is used to select the landmarks [43]. The proposed method with different strategies to estimate the number of neurons as shown in II-C.2 is compared. Since the existing literatures [44], [45] have reported that the maximal number of neurons detected by extracellular electrodes is between 8 and 10, an initial range of the estimated number of neurons is set between 2 and 10 in this paper. The reduced dimension, m_0, is assigned to 3 at the initialization stage in our method. And, the parameters in all the compared automatic methods are set to the corresponding default values.

Table I presents the results of the estimated number of neurons from different methods on the dataset with (without) overlapping spikes. Note that the distinct spikes in each dataset are generated by three neurons. Overall, our proposed method outperforms LE-KM, LPP-SCK, LDAKM, and LDAGMM methods. Also, it nearly reaches the baseline level especially when the isolation distance strategy is engaged as shown in Table I. Here, the estimated results are 19 (20) out of 20 (20) for the datasets with (without) overlapping spikes. The main reasons for the improvements are as follows. 1) The estimation results with LE-KM and LPP-SCK methods are sensitive to the reduced dimensions of the features. 2) For LDAKM and LDAGMM, the estimation process is mixed in the iteration optimization, making the results susceptible to the model's initial point and oscillating iteration. 3) Our proposed method integrated with various existing indexes succeed mainly with the help of stable input of the estimation from PCA initialization as shown in Eq.(2).

2) Overall Accuracy and Time Consumption: Secondly, the overall accuracy and time consumption of various spike sorting methods are compared. The comparison methods include the global-feature-based methods, PCA [11] combined with either KM (PCA-KM) or GMM (PCA-GMM), and the local-feature-based method, GF [17] combined with SCK [43] (GF-SCK). And the existing combining methods, LE-KM [16] and LPP-SCK [15], and the jointed model methods, PCAKM [25] and LDAKM [18], [21] are compared. Besides, the method named "Proposed*", which excludes the "Updating Rule" shown in II-B.1 compared to our proposed method is also engaged. Note that, due to the space limitation, the results of LDAGMM which are similar to those of LDAKM are not shown here.

For the sake of fair comparison, the number of neurons and the reduced dimensions for each method are assigned as 3 and 2, respectively. Besides, the K-means++ initialization strategy is used for all of those KM-related methods (i.e., PCA-KM, LE-KM, PCAKM, LDAKM), and the number of K-means++ initializations is set to 10 (it is equal to the number of K-means++ initializations in our proposed method as mentioned in subsection II-B.1). Specifically, the number of nearest neighbors in LE-KM is set, by default, to 12. The points of landmarks in SCK and the number of neighbors for each sample in LPP are, respectively, fixed to 1000 and 5 by default. The balance parameter in PCAKM is 0.98, which is determined on Difficult1_noise05 by the strategy provided by the original [25]. As for LDAKM, the open-source code is publicly accessed [46], and the configurations of our proposed method are detailed in Alg.1.[3] Note that 20 independent trials on each dataset for each method are performed, and the final

[3] Please visit https://github.com/HLBayes/Unified-SS to access the Matlab codes about our proposed method.

TABLE II

The Average Accuracy (± Its Standard Deviation) and the Average Processing Time (± Its Standard Deviation) Got From Various Spike Sorting Algorithms After Repeating 20 Times on Each Dataset *without* Overlapping Spike. Here the Number of Neurons and the Reduced Dimension Are Fixed With 3 and 2, Respectively

DS	NL	PCA-KM	PCA-GMM	LE-KM	LPP-SCK	GF-SCK	PCAKM	LDAKM	Proposed*	Proposed (Time)
E1	05	**100.00±0**	**100.00±0**	**100.00±0**	83.02±12.92	84.30±16.07	**100.00±0**	**100.00±0**	**100.00±0**	**100.00±0** (0.06)
	10	**100.00±0**	**100.00±0**	98.95±0	88.34±8.69	78.62±18.08	**100.00±0**	**100.00±0**	**100.00±0**	**100.00±0** (0.06)
	15	**100.00±0**	**100.00±0**	93.13±0	80.75±8.49	88.44±15.01	**100.00±0**	**100.00±0**	**100.00±0**	**100.00±0** (0.07)
	20	99.93±0	99.96±0	87.04±0	80.30±8.01	93.90±8.57	99.93±0	**100.00±0**	**100.00±0**	**100.00±0** (0.11)
	25	99.81±0	99.88±0	87.39±0	85.53±1.26	85.33±12.21	99.85±0	**100.00±0**	**100.00±0**	**100.00±0** (0.11)
	30	99.20±0	98.37±4.68	84.33±0.01	77.60±1.32	88.93±12.30	99.43±0	**100.00±0**	**100.00±0**	**100.00±0** (0.12)
	35	97.11±0	97.39±0.03	79.24±0	69.70±9.85	82.90±19.81	98.30±0	99.99±0.01	99.96±0	**100.00±0** (0.12)
	40	94.33±0	94.72±0.11	74.43±0.03	64.07±8.25	92.57±11.60	97.01±0	99.99±0.02	**100.00±0**	**100.00±0** (0.12)
E2	05	**100.00±0**	**100.00±0**	**100.00±0**	83.44±16.44	87.49±12.59	**100.00±0**	98.60±6.25	**100.00±0**	**100.00±0** (0.06)
	10	99.07±0	99.15±0	**100.00±0**	89.44±12.66	89.61±12.36	99.59±0	**100.00±0**	**100.00±0**	**100.00±0** (0.12)
	15	92.86±0	87.23±10.79	**100.00±0**	85.03±17.43	90.59±15.59	97.17±0	**100.00±0**	**100.00±0**	**100.00±0** (0.17)
	20	85.12±0.01	77.60±6.98	95.54±0	83.65±14.84	86.24±9.28	92.08±0	**100.00±0**	**100.00±0**	**100.00±0** (0.23)
D1	05	97.13±0	98.13±0	**100.00±0**	81.61±19.54	83.93±16.49	99.92±0	**100.00±0**	**100.00±0**	**100.00±0** (0.11)
	10	88.21±0	85.42±8.35	88.17±0.59	98.35±3.77	93.06±13.44	96.55±0	99.12±3.94	99.41±2.64	**100.00±0** (0.18)
	15	70.23±0	60.12±7.60	81.97±0.04	80.93±0.51	81.30±0.61	86.97±0.03	98.98±4.54	**100.00±0**	**100.00±0** (0.30)
	20	56.59±0.02	49.82±5.10	60.94±0	53.65±4.20	56.71±3.07	66.75±0.12	98.86±4.54	**100.00±0**	**100.00±0** (0.60)
D2	05	**100.00±0**	**100.00±0**	**100.00±0**	85.06±16.57	84.84±19.04	**100.00±0**	**100.00±0**	**100.00±0**	**100.00±0** (0.06)
	10	99.85±0	99.89±0	**100.00±0**	88.15±17.65	86.33±14.61	99.96±0	**100.00±0**	**100.00±0**	**100.00±0** (0.12)
	15	92.25±0	87.76±10.52	**100.00±0**	87.53±11.63	91.17±12.44	**100.00±0**	98.51±6.67	**100.00±0**	**100.00±0** (0.18)
	20	74.11±0.01	65.03±5.23	**100.00±0**	81.07±14.72	84.29±14.31	73.42±0	**100.00±0**	**100.00±0**	**100.00±0** (0.44)
Time / s		0.03±0.01	0.03±0.02	0.20±0.03	0.38±0.01	0.43±0.01	1.34±0.09	0.63±0.16	0.14±0.12	0.17±0.14

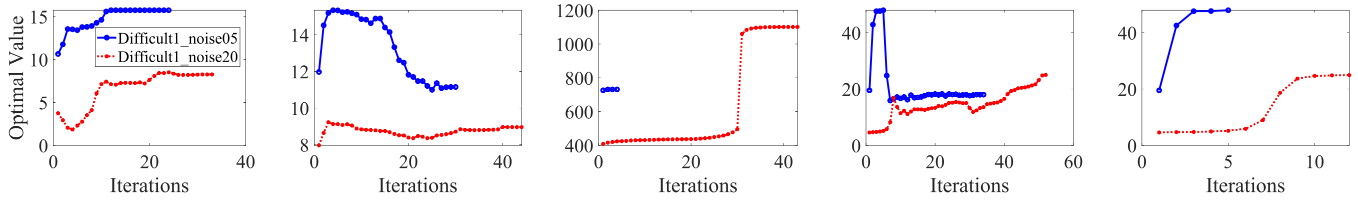
TABLE III

The Average Accuracy (± Its Standard Deviation) and Average Processing Time (± Its Standard Deviation) Got From Various Spike Sorting Algorithms After Repeating 20 Times on Each Dataset *with* Overlapping Spike. Here the Number of Neurons and the Reduced Dimension Are Fixed With 3 and 2, Respectively

DS	NL	PCA-KM	PCA-GMM	LE-KM	LPP-SCK	GF-SCK	PCAKM	LDAKM	Proposed*	Proposed (Time)
E1	05	99.23±0	95.75±10.44	67.60±0.05	96.20±0.71	99.31±0.11	99.23±0	97.85±7.30	99.52±0	**99.52±0** (0.24)
	10	99.40±0	99.52±0	98.38±0	97.91±0.41	99.55±0.11	99.43±0	99.66±0.14	99.77±0	**99.77±0** (0.24)
	15	99.19±0	99.31±0	96.87±0	94.93±0.89	99.62±0.03	99.22±0	99.65±0.09	99.68±0	**99.68±0** (0.25)
	20	99.11±0	97.44±7.41	94.67±0	89.04±3.29	99.57±0.04	99.17±0	99.77±0.06	99.80±0	**99.80±0** (0.24)
	25	99.00±0	99.15±0	90.72±0	81.35±3.42	90.50±14.54	99.03±0	99.64±0.06	99.76±0	**99.76±0** (0.23)
	30	98.04±0	93.55±11.85	86.42±0.01	71.43±3.29	**99.70±0.04**	98.53±0	99.60±0.02	99.60±0	99.60±0 (0.39)
	35	95.90±0.01	93.18±9.85	80.48±0	79.55±0.47	99.61±0.02	97.48±0	99.55±0.05	99.63±0.01	**99.63±0** (0.25)
	40	92.98±0.01	91.32±5.94	73.72±0	70.06±0.61	99.53±0.15	96.31±0	99.69±0.07	99.77±0.03	**99.79±0** (0.29)
E2	05	98.09±0	84.23±14.48	98.80±0	98.57±0.04	99.06±0.18	98.56±0	96.97±8.27	96.45±14.68	**99.74±0** (0.30)
	10	95.94±0	70.25±9.44	99.26±0	99.18±0.02	99.75±0.03	98.38±0	98.52±5.49	99.77±0	**99.77±0** (0.40)
	15	90.00±0	67.53±3.64	98.59±0	98.59±0.11	99.36±0.15	95.72±0	98.37±5.71	99.79±0.02	**99.79±0** (0.33)
	20	82.99±0.01	66.02±1.35	93.85±0	93.81±0.84	91.06±0.91	87.02±3.06	99.60±0.06	99.74±0.01	**99.74±0** (0.41)
D1	05	95.71±0	71.23±11.98	97.37±0	98.44±0.09	97.77±0.12	98.46±0	95.57±10.08	94.44±15.97	**99.35±0** (0.39)
	10	86.06±0.01	59.29±6.44	97.51±0	97.35±0.47	96.33±2.34	95.30±0	96.52±6.86	95.84±9.96	**99.07±0** (0.41)
	15	68.34±0.06	53.33±6.81	76.41±0.05	71.61±7.47	77.07±4.94	85.48±0.18	97.53±2.15	98.95±0.05	**98.96±0.01** (0.56)
	20	55.58±0.03	45.32±3.47	48.18±0.22	45.97±2.65	48.54±3.73	64.65±1.69	98.42±0.08	97.39±6.22	**98.79±0.02** (1.01)
D2	05	98.54±0	96.68±7.53	99.67±0	99.67±0.07	99.66±0.06	98.69±0.01	99.71±0.04	98.12±7.46	**99.79±0** (0.45)
	10	97.66±0	88.12±15.11	99.77±0	99.76±0.02	99.71±0.06	98.99±0	99.86±0.03	99.86±0	**99.86±0** (0.31)
	15	86.65±0.01	65.64±0.14	99.62±0	99.49±0.03	99.63±0.05	94.76±9.04	99.76±0.02	97.35±10.92	**99.80±0** (0.32)
	20	68.31±0	65.80±0.72	99.63±0	99.56±0.03	99.73±0.03	73.75±0	99.85±0.06	95.09±11.71	**99.89±0** (1.10)
Time / s		0.04±0.02	0.05±0.02	0.30±0.02	0.52±0.02	0.62±0.03	2.16±0.12	0.64±0.14	0.41±0.28	0.41±0.24

average results along with the standard deviation are recorded. In these numerical simulations, no matter the overlapping spikes exist or not in the dataset is compared.

As shown in Table II, on the datasets without overlapping spikes while the noise levels are relatively small, nearly all methods could get acceptable results. But when the noise level increases, only LDAKM and our proposed method can maintain high accuracy, while PCAKM and the methods combined feature extraction and clustering in sequence failed. On the one hand, PCAKM significantly relies on its balance parameter. On the other hand, the combined methods have not considered the internal relationships between the feature extraction and the clustering, and they also ignored the feedback from the clustering in the feature extraction stage. After using the K-means++ strategy and setting a sufficient number of initializations, the methods combined with KM get a stable result. However, the standard deviations of accuracy got from LDAKM are still unsatisfactory. For example, 6.25% and



(a) LDAKM (b) LDAGMM (c) PCAKM (d) Proposed* (e) Proposed

Fig. 3. The typical trends about the optimal value versus the times of iteration of methods (a) LDAKM, (b) LDAGMM, (c) PCAKM, (d) "Proposed*" without the *Updating Rule*, and (e) our complete proposed method on two example datasets.

6.67% on Easy2_noise05 and Difficult2_noise15, respectively. It is mainly because LDAKM confuses the ratio trace and trace ratio criteria in the model. For the "Proposed*" method, the outputs of sub-model (8) will easily fall into a local optimum, then the oscillating iteration, more iterations, and unstable results still exist in the model (6). On the contrary, by initializing KM with K-means++ strategy and updating with "Updating Rule", our proposed method obtains the near-perfect accuracies. At the same time, the processing time is less than the others in general. Interestingly, by some simple calculations, we can get that the minimum processing time of our proposed method is equal to the simple combined method PCA-KM. That means the proposed method can automatically adjust the processing time according to the complexity of the dataset, which can be further confirmed from the exact processing time of the proposed method in Table II.

The above findings are consistent with the results on the datasets with overlapping spikes as shown in Table III. More concretely, both the accuracy and the average time of all methods presented in Table III are slightly worse than those shown in Table II as the overlapping spikes exist in datasets. But for the SCK-based methods (i.e., LPP-SCK and GF-SCK), the overlapping spikes on the local-featured space can be well classified by SCK, thus better performance is achieved. Different from KM and GMM, SCK can be regarded as a local sense clustering method and in line with the local features extracted from LPP and GF. Overall, our proposed method performs well on the whole datasets in which the noise and overlapping exist and enjoys a low computational complexity. Since the recording period for each dataset is 60s, it is also potential to implant our method to in real-time projects.

3) Monotonicity Verifications: Finally, to verify the monotonicity of our proposed model, its objective function value versus its iteration times on Difficult1_noise05 and Difficult1_noise20 are graphed. The results on other datasets are similar and ignored here. At the same time, the model-based spike sorting methods, LDAKM, LDAGMM, PCAKM, and Proposed* are also compared. As shown in Fig.3a, although LDAKM has an increasing trend of the objective function values overall, its optimal value is not always increasing as the objective function of the two-step solution for LDAKM is different [24]. Besides, LDAGMM adds GMM operation after each KM [18], making its objective value even iterate toward a decreasing direction. As for the PCAKM, it converged with only four iterations on Difficult1_noise05, but over 40 iterations on Difficult1_noise20. Because its balance parameter is tuned based on Difficult1_noise05 and significantly affects the final results. "Proposed*" performs a heavy oscillating as shown in Fig.3d. Furthermore, LDAKM, LDAGMM, PCAKM and "Proposed*" almost need more iterations (shown in Fig.3a, Fig.3b, Fig.3c, and Fig.3d, respectively), and also more computational resources (shown in Tables II and III). But, for the proposed method, it always iterates towards the increasing objective values and converges after about 10 iterations as shown in Fig.3e.

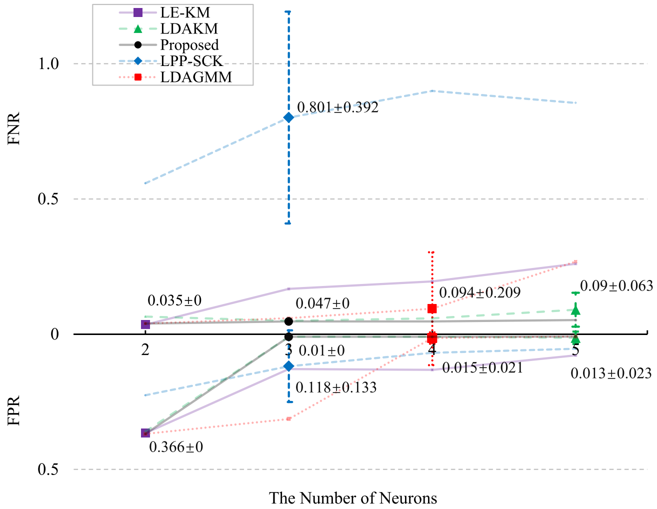
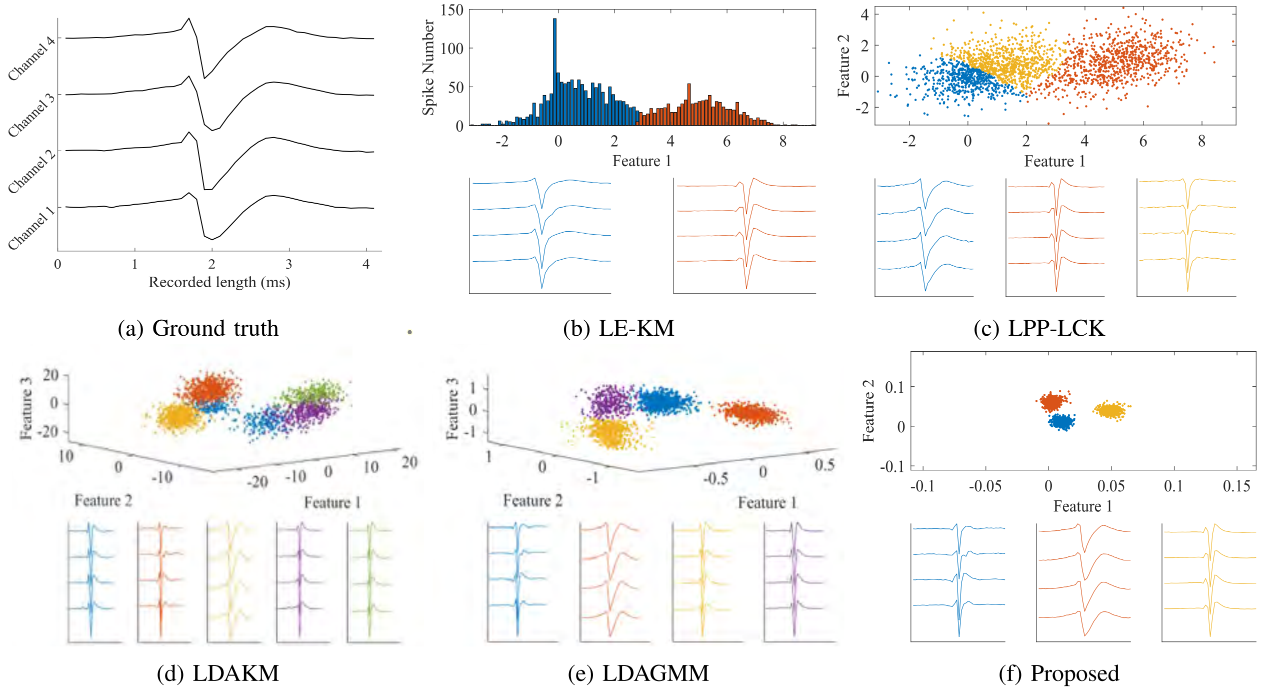


Fig. 4. The FPR and FNR obtained with the automatic spike sorting methods, LE-KM, LDAKM, LPP-SCK, LDAGMM and our proposed one on the HC1 datasets. Here, the results with the estimated number of neurons from 2 to 5 are presented. Among them, the actual estimated results (along with their standard deviation) for each method are indicated with the bold dots.

C. Numerical Simulation on the Real-World Dataset

In this section, the proposed method is performed on the real-world dataset, HC1 [41], and four automatic spike sorting methods, LE-KM [16], LPP-SCK [15], LDAKM [21], LDAGMM [18] are compared. Similar as [18], [27], firstly the raw recorded signal is filtered using a highpass Butterworth filter at 250 Hz with an order 50. Then, spikes were identified by exceeding a threshold, which is assigned by [27]. For more detailed preprocessing about the spike filtering and detection, please refer to the open source [47]. Actually, after these preprocessings, all the ground truth spikes recorded by the intracellular electrode are detected. In other words, there is no miss error introduced by the spike detection process. Then, each spike detected from the tetrode is in 4×41 shape shown in Fig.5a, where 41 is the length of the window. Then each spike is concatenated into a vector for further automatic sorting.



(a) Ground truth

(b) LE-KM

(c) LPP-LCK

(d) LDAKM

(e) LDAGMM

(f) Proposed

Fig. 5. The mean waveform of the ground truth is shown in (a). The low-dimensional distributions of spikes and their corresponding mean waveforms of each tentative neuron got from the automatic spike sorting methods, (b) LE-KM, (c) LPP-LCK, (d) LDAKM, (e) LDAGMM, and (f) our proposed method.

Finally, according to the existing ground truth intracellularly recorded from one neuron, the corresponding FPR and FNR can be calculated.

Fig. 4 shows the results of five automatic spike sorting methods with the estimated number of neurons ranging from 2 to 5. Among them, the actual estimated results are emphasized with bold dots and their numerical values are presented. It is easy to find that the results of LDAKM, LPP-SCK, and LDAGMM are unstable, especially for LPP-SCK the standard deviation of the FNR reaches 39.2%. On the other hand, compared with LE-KM, LPP-SCK, and LDAGMM, LDAKM and our proposed method get better results for different estimated number of neurons, although their actual estimated number is different (the proposed method is 3 while LDAKM is 5).

In addition to FPR and FNR, the distribution of spikes and the mean waveforms of each tentative neuron obtained by different methods are shown in Fig.5. Specifically, assuming the estimated number of neurons is $\tilde{c}$, when $\tilde{c} \leq 3$, the graphical dimension is $\tilde{c} - 1$. And when $\tilde{c} > 3$, this dimension is fixed with 3. As shown in Fig.5, the number of neurons and their waveforms obtained by various methods are different. Relatively speaking, the results of our proposed method are better since the graphical results show three distinctly different clusters and waveforms. In contrast, the results of LE-KM (Fig.5b) and LPP-LCK (Fig.5c) do not separate the spikes reasonably. With further analyzing the average waveforms, it can be seen that the second (red) waveforms from our proposed method are more consistent with the ground truth compared with the third (yellow) waveforms from LDAKM and the second (red) waveforms from LDAGMM. Besides, it appears that the first (blue) and second (red), the fourth (purple) and fifth (green) waveforms of LDAKM (Fig.5d) are very similar and prefer to merge them into an identical cluster, respectively. Similarly, for LDAGMM (Fig.5e), the third (yellow) and fourth (purple) waveforms are also very similar. In summary, our proposed method exhibits excellent accuracy and stability on both synthetic and real-world datasets.

IV. Conclusion

This paper studied an optimization problem for spike sorting, which unified the feature extraction and clustering stages. It was solved basically in an integrated framework with PCA and KM-like procedures. The computational complexity of this model is linear to the number of spikes, which is less than the existing related methods. In particular, an efficient and effective automatic spike sorting method is proposed after integrating existing clustering validity indices in the model to estimate the number of neurons. The yielded satisfactory and stable results on both synthetic and real datasets verified the efficacy of our proposed method.

Currently, our proposed unified model seemingly is the global-sense method from the view of PCA and KM-like solving procedures. One direction of our future work will focus on a more general-sense model. One promising way is jointly modeling by unifying PCA and GMM since the GMM could model non-spherical distribution better compared with KM, or modeling from the view of local sense such as unifying graph-Laplacian and spectral clustering. The other approach is adaptively modeling feature extraction and clustering stages, which takes more consideration of the statistical perspective rather than the distribution property of the dataset. With respect to the data processed, we have handled the signals recorded from single-electrode and tetrodes. Hence, another research direction of our work will extend the method to handle the signals recorded by the multi-electrode arrays which contain several hundred or even thousands of the electrodes.

References

[1] G.-L. Wang, Y. Zhou, A.-H. Chen, P.-M. Zhang, and P.-J. Liang, "A robust method for spike sorting with automatic overlap decomposition," *IEEE Trans. Biomed. Eng.*, vol. 53, no. 6, pp. 1195–1198, Jun. 2006.

[2] Y. Mokri, R. F. Salazar, B. Goodell, J. Baker, C. M. Gray, and S.-C. Yen, "Sorting overlapping spike waveforms from electrode and tetrode recordings," *Frontiers Neuroinform.*, vol. 11, p. 53, Aug. 2017.

[3] M. Linderman *et al.*, "Signal processing challenges for neural prostheses," *IEEE Signal Process. Mag.*, vol. 25, no. 1, pp. 18–28, Dec. 2008.

[4] J. Sanchez, J. Principe, T. Nishida, R. Bashirullah, J. Harris, and J. Fortes, "Technology and signal processing for brain-machine interfaces," *IEEE Signal Process. Mag.*, vol. 25, no. 1, pp. 29–40, Dec. 2008.

[5] S. Gibson, J. W. Judy, and D. Marković, "Spike sorting: The first step in decoding the brain: The first step in decoding the brain," *IEEE Signal Process. Mag.*, vol. 29, no. 1, pp. 124–143, Dec. 2011.

[6] C. Pouzat, O. Mazor, and G. Laurent, "Using noise signature to optimize spike-sorting and to assess neuronal classification quality," *J. Neurosci. Methods*, vol. 122, no. 1, pp. 43–57, Dec. 2002.

[7] M. S. Lewicki, "A review of methods for spike sorting: The detection and classification of neural action potentials," *Netw., Comput. Neural Syst.*, vol. 9, no. 4, pp. R53–R78, Jan. 1998.

[8] H. G. Rey, C. Pedreira, and R. Q. Quiroga, "Past, present and future of spike sorting techniques," *Brain Res. Bull.*, vol. 119, pp. 106–117, Oct. 2015.

[9] D. Carlson and L. Carin, "Continuing progress of spike sorting in the era of big data," *Current Opinion Neurobiol.*, vol. 55, pp. 90–96, Apr. 2019.

[10] S. Mahallati, J. C. Bezdek, M. R. Popovic, and T. A. Valiante, "Cluster tendency assessment in neuronal spike data," *PLoS ONE*, vol. 14, no. 11, Nov. 2019, Art. no. e0224547.

[11] D. A. Adamos, E. K. Kosmidis, and G. Theophilidis, "Performance evaluation of PCA-based spike sorting algorithms," *Comput. Methods Programs Biomed.*, vol. 91, no. 3, pp. 232–244, Sep. 2008.

[12] C. Yang, Y. Yuan, and J. Si, "Robust spike classification based on frequency domain neural waveform features," *J. Neural Eng.*, vol. 10, no. 6, Dec. 2013, Art. no. 066015.

[13] E. Hulata, R. Segev, and E. Ben-Jacob, "A method for spike sorting and detection based on wavelet packets and Shannon's mutual information," *J. Neurosci. Methods*, vol. 117, no. 1, pp. 1–12, May 2002.

[14] F. J. Chaure, H. G. Rey, and R. Q. Quiroga, "A novel and fully automatic spike-sorting implementation with variable number of features," *J. Neurophysiol.*, vol. 120, no. 4, pp. 1859–1871, Oct. 2018.

[15] T. Nguyen, A. Khosravi, D. Creighton, and S. Nahavandi, "Spike sorting using locality preserving projection with gap statistics and landmark-based spectral clustering," *J. Neurosci. Methods*, vol. 238, pp. 43–53, Dec. 2014.

[16] E. Chah, V. Hok, A. Della-Chiesa, J. J. H. Miller, S. M. O'Mara, and R. B. Reilly, "Automated spike sorting algorithmbased on Laplacian eigenmaps and K-means clustering," *J. Neural Eng.*, vol. 8, no. 1, Feb. 2011, Art. no. 016006.

[17] Y. Ghanbari, P. E. Papamichalis, and L. Spence, "Graph-Laplacian features for neural waveform classification," *IEEE Trans. Biomed. Eng.*, vol. 58, no. 5, pp. 1365–1372, May 2011.

[18] M. R. Keshtkaran and Z. Yang, "Noise-robust unsupervised spike sorting based on discriminative subspace learning with outlier handling," *J. Neural Eng.*, vol. 14, no. 3, Jun. 2017, Art. no. 036003.

[19] L. Huang, B. W.-K. Ling, Y. Zeng, and L. Gan, "Spike sorting based on low-rank and sparse representation," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jul. 2020, pp. 1–6.

[20] B. C. Souza, V. Lopes-dos-Santos, J. Bacelo, and A. B. L. Tort, "Spike sorting with Gaussian mixture models," *Sci. Rep.*, vol. 9, no. 1, pp. 1–14, Dec. 2019.

[21] M. R. Keshtkaran and Z. Yang, "Unsupervised spike sorting based on discriminative subspace learning," in *Proc. 36th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC)*, Aug. 2014, pp. 3784–3788.

[22] C. Ding and T. Li, "Adaptive dimension reduction using discriminant analysis and K-means clustering," in *Proc. 24th Int. Conf. Mach. Learn. (ICML)*, 2007, pp. 521–528.

[23] Y. Jia, F. Nie, and C. Zhang, "Trace ratio problem revisited," *IEEE Trans. Neural Netw.*, vol. 20, no. 4, pp. 729–735, Apr. 2009.

[24] F. Wang, Q. Wang, F. Nie, Z. Li, W. Yu, and R. Wang, "Unsupervised linear discriminant analysis for jointly clustering and subspace learning," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 3, pp. 1276–1290, Mar. 2019.

[25] C. Hou, F. Nie, D. Yi, and D. Tao, "Discriminative embedded clustering: A framework for grouping high-dimensional data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 6, pp. 1287–1299, Jun. 2015.

[26] B. He, M. Tao, and X. Yuan, "Alternating direction method with Gaussian back substitution for separable convex programming," *SIAM J. Optim.*, vol. 22, no. 2, pp. 313–340, Jan. 2012.

[27] C. Ekanadham, D. Tranchina, and E. P. Simoncelli, "A unified framework and method for automatic neural spike identification," *J. Neurosci. Methods*, vol. 222, pp. 47–55, Jan. 2014.

[28] F. De la Torre and T. Kanade, "Discriminative cluster analysis," in *Proc. 23rd Int. Conf. Mach. Learn. (ICML)*, 2006, pp. 241–248.

[29] Q. Xu, C. Ding, J. Liu, and B. Luo, "PCA-guided search for K-means," *Pattern Recognit. Lett.*, vol. 54, pp. 50–55, Mar. 2015.

[30] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, 2nd ed. Amsterdam, The Netherlands: Elsevier, 2013.

[31] D. Arthur and S. Vassilvitskii, "K-means++: The advantages of careful seeding," in *Proc. 18th Annu. ACM-SIAM Symp. Discrete Algorithms (SODA)*, 2007, pp. 1027–1035.

[32] C. Rossant *et al.*, "Spike sorting for large, dense electrode arrays," *Nature Neurosci.*, vol. 19, no. 4, pp. 634–641, Apr. 2016.

[33] A. J. Brockmeier and J. C. Principe, "Learning recurrent waveforms within EEGs," *IEEE Trans. Biomed. Eng.*, vol. 63, no. 1, pp. 43–54, Jan. 2016.

[34] J. Zhang *et al.*, "A review on cluster estimation methods and their application to neural spike data," *J. Neural Eng.*, vol. 15, no. 3, Jun. 2018, Art. no. 031003.

[35] T. Nguyen, A. Bhatti, A. Khosravi, S. Haggag, D. Creighton, and S. Nahavandi, "Automatic spike sorting by unsupervised clustering with diffusion maps and silhouettes," *Neurocomputing*, vol. 153, pp. 199–210, Apr. 2015.

[36] R. Tibshirani, G. Walther, and T. Hastie, "Estimating the number of clusters in a data set via the gap statistic," *J. Roy. Stat. Soc., B, Stat. Methodol.*, vol. 63, no. 2, pp. 411–423, 2001.

[37] P. Dan and A. W. Moore, "X-means: Extending K-means with efficient estimation of the number of clusters," in *Proc. 17th Int. Conf. Mach. Learn. (ICML)*, vol. 1, 2000, pp. 727–734.

[38] N. Schmitzer-Torbert and A. D. Redish, "Neuronal activity in the rodent dorsal striatum in sequential navigation: Separation of spatial and reward responses on the multiple t task," *J. Neurophysiol.*, vol. 91, no. 5, pp. 2259–2272, May 2004.

[39] K. D. Harris, H. Hirase, X. Leinekugel, D. A. Henze, and G. Buzsáki, "Temporal interaction between single spikes and complex spike bursts in hippocampal pyramidal cells," *Neuron*, vol. 32, no. 1, pp. 141–149, Oct. 2001.

[40] R. Q. Quiroga, Z. Nadasdy, and Y. Ben-Shaul, "Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering," *Neural Comput.*, vol. 16, no. 8, pp. 1661–1687, Aug. 2004.

[41] 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," *J. Neurophysiol.*, vol. 84, no. 1, pp. 401–414, Jul. 2000.

[42] M. Wehr, J. S. Pezaris, and M. Sahani, "Simultaneous paired intracellular and tetrode recordings for evaluating the performance of spike sorting algorithms," *Neurocomputing*, vols. 26–27, pp. 1061–1068, Jun. 1999.

[43] X. Chen and D. Cai, "Large scale spectral clustering with landmark-based representation," in *Proc. 25th AAAI Conf. Artif. Intell. (AAAI)*, 2011, pp. 313–318.

[44] B. Lefebvre, P. Yger, and O. Marre, "Recent progress in multi-electrode spike sorting methods," *J. Physiol.*, vol. 110, no. 4, pp. 327–335, Nov. 2016.

[45] C. Pedreira, J. Martinez, M. J. Ison, and R. Q. Quiroga, "How many neurons can we see with current spike sorting algorithms?" *J. Neurosci. Methods*, vol. 211, no. 1, pp. 58–65, Oct. 2012.

[46] K. M. Reza. (2017). *LDA-GMM_Spikesort*. Accessed: Jul. 29, 2019. [Online]. Available: https://github.com/mrezak/LDA-GMM_SpikeSort

[47] E. P. Simoncelli and P. H. Li. (2014). *Cbpspikesortdemo*. Accessed: Mar. 1 2019. [Online]. Available: https://github.com/chinasaur/CBPSpikesortDemo