SCIENTIFIC REPORTS

OPEN

Spike sorting with Gaussian mixture models

Bryan C. Souza[1], Vítor Lopes-dos-Santos[1,2], João Bacelo[1] & Adriano B. L. Tort[1]

Received: 11 May 2018
Accepted: 5 February 2019
Published online: 06 March 2019

The shape of extracellularly recorded action potentials is a product of several variables, such as the biophysical and anatomical properties of the neuron and the relative position of the electrode. This allows isolating spikes of different neurons recorded in the same channel into clusters based on waveform features. However, correctly classifying spike waveforms into their underlying neuronal sources remains a challenge. This process, called spike sorting, typically consists of two steps: (1) extracting relevant waveform features (e.g., height, width), and (2) clustering them into non-overlapping groups believed to correspond to different neurons. In this study, we explored the performance of Gaussian mixture models (GMMs) in these two steps. We extracted relevant features using a combination of common techniques (e.g., principal components, wavelets) and GMM fitting parameters (e.g., Gaussian distances). Then, we developed an approach to perform unsupervised clustering using GMMs, estimating cluster properties in a data-driven way. We found the proposed GMM-based framework outperforms previously established methods in simulated and real extracellular recordings. We also discuss potentially better techniques for feature extraction than the widely used principal components. Finally, we provide a friendly graphical user interface to run our algorithm, which allows manual adjustments.

Analyzing the activity of neurons recorded with extracellular electrodes is one of the main techniques to study the brain *in vivo*. To that end, it is necessary to reliably identify spikes of different neurons recorded from the same electrode. In principle, this can be partially achieved because the extracellular waveform of action potentials varies depending on biophysical and morphological properties of the cells, as well as on the relative position of the electrode[1–3]. Thus, the detected spikes can be assigned into clusters of similar waveforms that correspond to different neurons. This clustering procedure is also known as 'spike sorting' and constitutes a crucial step before spike train analysis.

Many algorithms have been developed to deal with this problem[4–6]. One of the main challenges is to automatically identify which features of the spike waveform should be used for classification. In fact, as important as the clustering algorithm per se is the preceding processing step, often referred to as dimensionality reduction[7]. This step aims at transforming the set of waveforms into a small representation in order to reduce noise and provide the most informative components to be used by the clustering algorithm.

A largely used technique for dimensionality reduction is the principal component analysis[4,8,9] (PCA). Representing the data in terms of a principal component (PC) means that each waveform is redefined as a weighted sum of its values, a linear combination determined by the PC weights; this transforms the entire waveform into a single number. By definition, the first PC is the axis (or linear combination) which captures most variance of the data. The second PC is the axis that captures most variance orthogonal to the first PC, and so on. In this way, waveforms might be represented by a small set of uncorrelated (orthogonal) components that capture most of their variance.

Although it is convenient to represent the data in a set of uncorrelated components, the main caveat of PCA-based sorting is the core assumption that feature variance would be proportional to its capacity of isolating neurons (clustering separability), which may not be the case. Alternatively, one can rescale the data in order to bias PCA to primarily extract multimodal components relevant for clustering, a technique called weighted-PCA[10–12] (wPCA). In this framework, each time point (sample) of the waveform is divided by its variance (across waveforms) and multiplied by an estimate of its clustering separability. The challenge, then, is to find the optimal method to make such estimate in an unsupervised manner.

[1]Brain Institute, Federal University of Rio Grande do Norte, Natal, Brazil. [2]Present address: MRC Brain Network Dynamics Unit, Department of Pharmacology, University of Oxford, Oxford, UK. Correspondence and requests for materials should be addressed to B.C.S. (email: bryancsouza@neuro.ufrn.br)

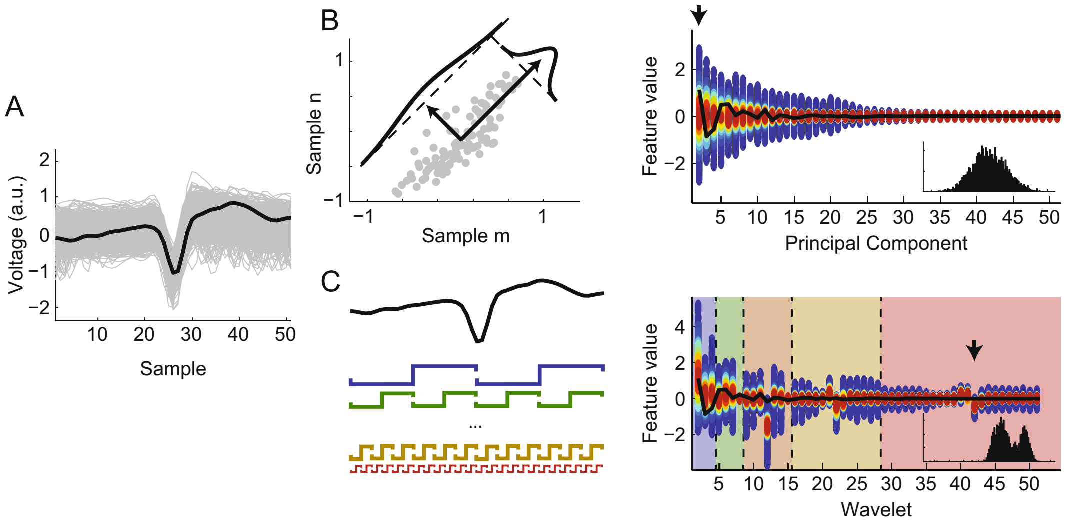

Figure 1. Schematic representation of wavelet decomposition and principal component analysis. (**A**) Detected spike waveforms (gray traces) and a particular example (black line). (**B**) Principal component analysis of the waveforms. (Left) Scatter plot of two sample points of the waveforms. The principal components (PCs) are successively defined as orthogonal directions maximizing the data variance (black arrows in this example). (Right) Color-coded histogram of the PC scores extracted from the waveforms in A. Warmer colors code for a higher density of waveforms with the same score value. Black line shows the PC scores of the highlighted example in A. Inset shows the histogram of the first PC scores (black arrow). (**C**) (Left) Haar wavelet decomposition. Each waveform was decomposed using a set of Haar wavelets of multiple scales and non-overlapping translations (colored functions). (Right) Color-coded histogram of the wavelet coefficients separated by different scales (background colors). Black line shows the wavelet coefficients of the highlighted example in A. Black arrow indicates the wavelet coefficients whose distribution is shown in the inset.

A common alternative to PCA is the wavelet decomposition (WD)[13–16], which provides a time-frequency representation of the signal. The WD transforms each waveform into a set of wavelet coefficients, which isolate frequency (or time scale) components at a certain location in time. Similarly to PCA, one can then obtain a small set of wavelet coefficients that capture localized frequency components relevant for spike classification, thus reducing dimensionality. As in the wPCA, estimates of clustering separability are required to select a mixture of relevant components (see Materials and Methods).

Once the complexity of the data is reduced, the next step is to perform clustering with the selected features. Manual selection of each cluster with the aid of visualization tools is a difficult task when there are too many neurons, besides introducing human bias. To automatize this step, previous approaches have used unsupervised clustering based on template matching[17], the distance between cluster centers[4,13] (i.e., k-means), or statistical models of the data (see refs[4,11,18–21]).

In this work, we propose a spike sorting framework using Gaussian mixture models (GMMs), a statistical model that fits the data using a mixture of Gaussian distributions. We combined GMMs with different feature extraction techniques (PCA, wPCA and WD) and explored the multiple ways of reducing dimensionality to relevant features. Additionally, GMMs were also used to estimate the number of clusters and classify spikes. We tested our approach using datasets with simulated and actual waveforms and compared its performance with two other mixture-models-based spike sorting algorithms – EToS[10,11], which is based on t-Student distributions, and KlustaKwik[8,9], also based on Gaussian distributions. Our method showed similar or better performance concerning benchmark attributes such as signal-to-noise ratio, stability and symmetry. We finish by presenting a MATLAB graphical user interface to perform spike sorting based on GMMs, which we made freely available online.

Materials and Methods

Theoretical background.

Principal components analysis. Principal components analysis (PCA) is a linear transformation that maps the data **X** into a new orthogonal basis maximizing the variance of each dimension (Fig. 1B). It is formally defined by $\mathbf{Y} = \mathbf{XA}$, where **X** is an s-by-n data matrix with s observations (i.e., number of spikes) of n dimensions (i.e., number of time samples from each waveform), and **A** is an n-by-n matrix whose columns are formed by the eigenvectors of the covariance matrix of **X**, which are called principal components (PCs). Each column of **Y** corresponds to the scores of a PC, that is, the projection of an **X** column over the direction defined by the PC. The eigenvalue associated to the eigenvector defining the PC denotes the variance of the data in that direction. Notice that projecting the data in a PC direction is equivalent to performing a linear combination of the n dimensions of **X**. The PCs are usually sorted in a decreasing order of eigenvalues so that the scores of the first PCs (columns of **Y**) carry most of the variance of the data, and are thus used for further analyses (dimensionality reduction).

A related procedure, called weighted-PCA[11,12] (wPCA), relies on first normalizing the variance of each dimension *n* before performing the PCA. This can be used to give more weight to the dimensions that are deemed more relevant (Supplementary Fig. S1).

Wavelet decomposition. A discrete wavelet transform is a time-frequency decomposition that consists in computing the inner product of each waveform *i* (a row of **X**, denoted as $x_i[n]$) with scaled and translated versions of the mother wavelet function. This wavelet decomposition (WD) is formally defined as:

$$\text{Wavelet coefficient}\{x_i\}(j, k) = \sum_n x_i[n] \cdot \psi_{j,k}[n]$$

with

$$\psi_{j,k}[n] = \frac{1}{\sqrt{2^j}}\psi\left(\frac{n - k \cdot 2^j}{2^j}\right)$$

where ψ is the mother wavelet and *j* and *k* are integers representing the scaling and translation factors, respectively. Convolving the signal with the wavelet of a given scale can be seen as a filtering procedure, with the wavelet as a kernel. Here the WD was computed using the multi-resolution approach based on Haar wavelets[22] (Fig. 1C) due to its computational efficiency and simplicity. Briefly, this approach uses a Haar wavelet to filter the signal (high-pass filter) and downsamples it by 2 to obtain the wavelet coefficients of the fast scale. The original signal is also filtered using a complementary kernel (a quadrature mirror filter of the wavelet) so that it separates the low-frequency components of the signal (low-pass filter). The procedure is successively repeated using as input the low-frequency signal downsampled by 2, until the desired level of filtering (4 in our case). Because of the downsampling, the next level of filtering captures wavelet coefficients on a slower scale.

Gaussian mixture models. A Gaussian mixture model (GMM) is a probabilistic model that assumes the data comes from a combination of *k* Gaussians distributions, with independent means, variances and weights (amplitudes). The GMM for an empirical distribution of feature values can be formally defined as:

$$p(x) = \sum_{i=1}^{k} \alpha_i \cdot N(x| \mu_i, \Sigma_i)$$

where μ_i and Σ_i, are the mean and covariance matrix of the ith Gaussian, respectively, and α_i is the probability that *x* belongs to the ith Gaussian, or the Gaussian weight.

Except for the number of Gaussians, the parameters of the model are iteratively defined in a data-driven manner. We estimated these three parameters θ (mean, covariance and Gaussian weight) using an expectation-maximization (EM) algorithm that searches for the maximum *a posteriori* probability[23]. Briefly, the EM consists in two steps that are iterated until convergence. In the first step the expected value of *p(x,z)* - where *z* is the (unknown) membership of the observation *x* - is estimated given an initial set of θ. The second step consists in updating the parameters so that it maximizes the probability of *x* in the model.

Because the resulting GMM depends on the initial conditions, we computed 10 replicates for each fitting and kept the ones with highest log likelihood. We set the stopping criterion as 10^4 interactions or a convergence threshold (10^{-6} percentage of change in *log p(x)*). In a later step of our algorithm (see below), we used a modified version of the EM in which we could set the Gaussian centers of the model. In this version, which we refer to as "fixed-mean GMM", only the α and Σ parameters were updated in the first step so that the initial and final μ were the same.

GMM-based spike sorting. Our proposed algorithm can be separated in two main steps: feature extraction and clustering. Each step uses GMMs, as detailed below:

Feature extraction. For feature extraction, we performed either PCA or WD on the set of waveforms (Fig. 1). Once the waveforms are transformed into such features (PC scores or wavelet coefficients), a subset of them are selected for clustering. This selection is crucial since the quality of clustering is highly dependent on the amount of spike identity information conveyed by the features. We investigated selection criteria based on GMM, which we explain next.

The probability function for values of a given feature (PC score or wavelet coefficient) was estimated by fitting a GMM with eight Gaussian functions to its empirical distribution. We used the fitted probability function to compute four different metrics of clustering separability. Three of them were based on GMM, which depended on the (1) peaks and (2) inflection points of the mixture, as well as on the (3) distance between the individual Gaussians (Fig. 2). The peak and inflection points were found by first discretizing the fitted probability function into 100 points in the range of the data and then numerically assessing the first and second derivatives. The peak- (I_{peak}) and inflection-based (I_{inf}) metrics were defined as:

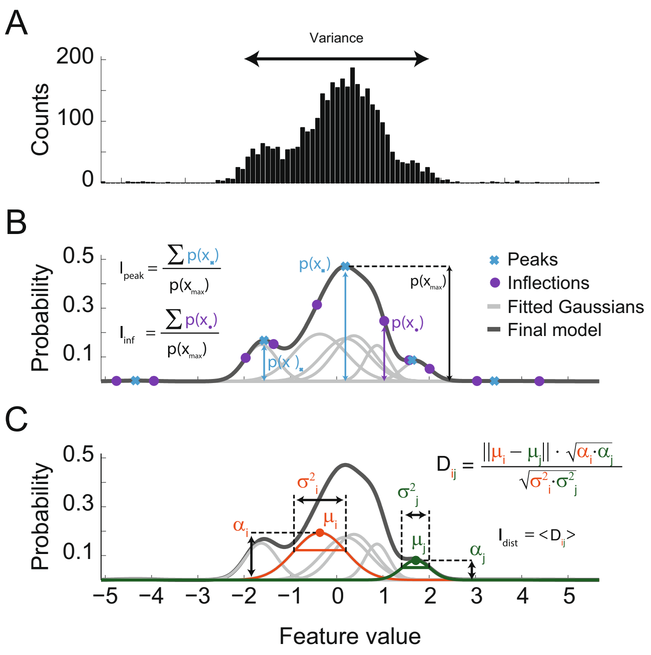

Figure 2. Estimating clustering information of a feature with Gaussian mixture models. (**A**) Histogram of the values of a waveform feature (e.g., PC scores or wavelet coefficients). Note multimodal distribution. The variance of the feature values (var) was one of the 4 separability metrics studied in this work. (**B**) Gaussian mixture model of the feature values in A. The black line shows the probability distribution function of the mixed model composed by eight Gaussians (gray lines). Crosses and disks mark the peak and inflection points of the model, respectively, which were used to compute two of the separability metrics (I_{peak} and I_{inf}). (**C**) Schematic representation of the normalized distance between Gaussian pairs (D_{ij}); μ: Gaussian mean; σ: standard deviation; α: Gaussian weight. Another separability metric was defined as the median distance (I_{dist}).

$$I_{peak} = \frac{1}{max p(x)} \sum_{i \in peak} p(x_i)$$

$$I_{inf} = \frac{1}{max p(x)} \sum_{i \in inf} p(x_i)$$

where $p(x_i)$ is the probability of the model at either a peak or an inflection point x_i (see Fig. 2B). Intuitively, these metrics are the sum of the model probability at the peaks or inflection points normalized by the highest probability. The third metric, referred to as the distance metric (I_{dist}), was computed from the normalized distance between each pair of Gaussians:

$$D_{ij} = \frac{|\mu_i - \mu_j| \cdot \sqrt{\alpha_i \cdot \alpha_j}}{\sqrt{\sigma_i^2 \cdot \sigma_j^2}}, \quad I_{dist} = \langle D_{ij} \rangle$$

where μ, σ and α are, respectively, the mean, standard deviation and weight of Gaussians *i* and *j* (Fig. 2C), and the brackets denote the median over all *i* and *j*. In other words, D_{ij} measures how distant two Gaussians are after correcting them by their standard deviation. Because the amplitude of each Gaussian can influence the amount of data they separate, we also weighted them by their amplitude, so that a Gaussian pair with high amplitude is more distant than one with low amplitude.

The rationale underlying these metrics is that, intuitively, features exhibiting multimodal distributions are better for separating spikes than unimodal ones. Moreover, the number of modes in the distribution is related to the number of clusters the feature may separate. Although using a normality metric to evaluate features can also assess multimodality, this approach does not give information about the potential number of clusters, as opposed to considering the Gaussian centers, peaks and inflections as done here.

The fourth metric we used was the variance of the features, which is independent of the GMM. For each of the four metrics, we selected the 5 highest-ranked PC scores or wavelet coefficients to perform clustering.

In addition to pure WD, we also implemented a third technique for feature extraction using wPCA in the wavelet coefficients. To that end, each coefficient was z-scored and multiplied by one of the three GMM-based metrics (I_{peak}, I_{inf} or I_{dist}) prior to computing PCA (notice that we did not use the variance metric since this would oppose the z-score normalization). The first 5 weighted PC scores (ranked by variance) were then selected for clustering (see Supplementary Fig. S1).

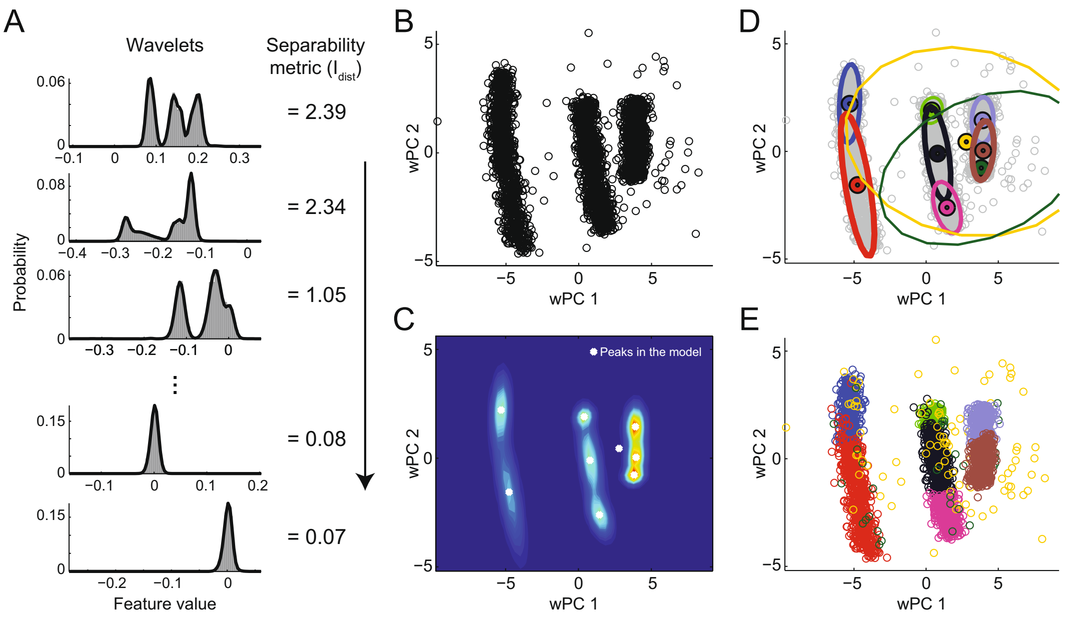

Figure 3. GMM-based spike sorting framework. (**A**) Example of feature values (wavelet coefficients) and their GMM fittings. In the proposed framework, wavelet coefficients (or PC scores) are ranked according to a clustering separability metric; 4 metrics were investigated: var, I_{peak}, I_{inf} or I_{dist} (see Materials and Methods). The example depicts wavelet coefficient distributions ranked by the I_{dist} metric. Note that unimodal distributions have lower I_{dist} values. (**B**) The first two wPCs of the coefficients in A. Weighted PCA was obtained by normalizing the variance of wavelet coefficients by I_{dist} and applying PCA. Clustering was done using the first 5 wPCs. For pure PCA and WD, we used the first 5 features according to the separability metric. (**C**) The probability density function of the 5-dimension GMM computed from the feature subspace (same dimensions as in B) and its peaks (white dots). The GMM was computed with 12 (Dataset A) or 20 (Dataset B) Gaussians. (**D**) Representation of a fixed-mean GMM with Gaussians centered at the peaks in C. Dots and ellipses denote the center and the 2-standard-deviation boundary of each Gaussian; line thickness represents Gaussian amplitude. (**E**) Final classification of the waveforms using the fixed-mean GMM. Each waveform was assigned to the Gaussian with higher probability in the model in D. Colors in D and E represent different Gaussians/clusters. For this example, the spikes of three neurons were analyzed. Notice that the algorithm performs "overclustering", capturing changes in the waveform shape of a same neuron due to bursting activity. Clusters can be merged during post-processing adjustments.

Clustering. After feature extraction (5 features for each combination of technique and clustering metric), we first estimated the number of clusters in the data using an overclustering approach. To do so, we fitted a GMM for the data projected into the 5 selected features with a large number of Gaussians (12 for Dataset A and 20 for Dataset B; see below) (Fig. 3B,C). In this model, each Gaussian was defined by their mean (center) and covariance in the 5-feature space. We then searched for peaks in the GMM probability function surface by running a Nelder-Mead simplex algorithm[24] for 12 or 20 initial conditions, defined by the Gaussian centers. Peaks distant by less than 1% of the data range were merged and counted as one. Finally, each peak of the mixture was then regarded as a cluster center (Fig. 3C).

To classify the waveforms into different clusters, we performed yet another GMM fitting using the cluster centers found in the previous step as fixed means (i.e., a fixed-mean GMM in the 5-feature space). In this case, the number of Gaussians was equal to the number of cluster centers. The Gaussians in this model defined cluster probabilities in the feature space for clustering (Fig. 3D). Thus, we assigned each 5-dimensional point (representing a waveform) to the cluster of highest *a posteriori* probability (Fig. 3E).

Comparing sorting performance. We compared the different strategies (PCA, WD and wPCA) of our GMM approach within themselves (I_{peak}, I_{inf}, I_{dist} and variance), with each other, and with two other spike sorting methods based on mixture models: KlustaKwik[9] and EToS[11]. Briefly, the KlustaKwik algorithm (version 3.0; "classic" mode) uses PCA combined to GMMs and a cluster penalty criterion to avoid overclustering. Since we only analyzed tetrodes and single wire recordings, we used its simpler unmasked version. The EToS algorithm used here (version 3.1.4) is based on WD using CDF97 wavelet combined with a wPCA approach for feature extraction and a variational Bayes t-Student mixture model for clustering. In each case, the standard parameters of each algorithm were used. To compare the algorithms, we analyzed two datasets available online that were used in previous spike sorting studies[8,14] (Supplementary Table S1).

Dataset A was composed of 25 simulated recordings, each containing three different neurons[14]. Briefly, background activity consisted of spikes randomly drawn from a database containing 594 average waveforms. Three waveforms from the same database were superimposed on ~60 s of background activity at random times with different signal-to-noise ratios.

Dataset B was composed of simultaneous intra and extracellular recordings of CA1 pyramidal cells in anesthetized rats[25]. In this case, the intracellular recording served as ground truth for the cellular identity of one of the extracellular spikes. We tested the spike sorting methods using tetrode recordings available in this dataset. The extracellular signals were band-pass filtered (300–3000 Hz). A detailed description of datasets A and B can be found in refs[8,14], respectively.

For both datasets, a threshold of 3–7 standard deviations was used to detect the spikes in the extracellular field potentials. The waveforms were centered at their peaks and classified according to the simulated spike times (Dataset A) or to the detected intracellular spikes (Dataset B). For Dataset B, spike times were detected individually in each tetrode, and the waveforms were concatenated across channels before feature extraction. Intracellular spikes that could be associated with more than one extracellular spike within 2 ms were discarded.

Measuring classification performance. We used the mutual information[26] (MI) between the real and assigned spike classes to measure classification performance (Supplementary Fig. S2). MI is defined as:

$$\mathrm{MI}_{\mathrm{X,Y}} = \sum_{\mathrm{x}\in\mathrm{X}} \sum_{\mathrm{y}\in\mathrm{Y}} p(\mathrm{x},\ \mathrm{y}) \cdot \log_2\left(\frac{p(\mathrm{x},\ \mathrm{y})}{p(\mathrm{x})p(\mathrm{y})}\right)$$

where X and Y are the real and assigned classes. The MI measures how mixed the real classes are within the assigned classes, providing the entropy shared by them. This is particularly valuable since unsupervised clustering can assign waveforms into a different number of clusters than the real one. Supplementary Fig. S2 shows how the MI is affected by adding more assigned classes or mixing them. In order to compare the MI between different waveform sets, we normalized each MI by its maximum possible value (i.e., the entropy of the real class). This yields the percentage of extracted spike information ($\mathrm{MI}_{\mathrm{norm}}$):

$$\mathrm{MI}_{\mathrm{norm}} = 100\frac{\mathrm{MI}_{\mathrm{X,Y}}}{\mathrm{MI}_{\mathrm{X,X}}}$$

We computed MIs through the Information Breakdown toolbox[27]. For comparison, we also report the error rates, which were computed using the predominant neuron of each cluster as its identity (Supplementary Figs S3 and S4).

For Dataset A, we subsampled spikes to investigate the effect of unbalanced clusters (different firing rates) on classification performance. More specifically, we initially set the three neurons to have the same number of spikes; we next subsampled the waveforms of one of the neurons to a percentage of its total number of spikes, which defined the "symmetry index" (i.e., a symmetry index of 10% corresponds to a reduction to 10% of the total number of spikes). Each of the three neurons had their spikes subsampled individually, with symmetry indexes varying logarithmically from 1 to 100%. For this analysis, we also computed the $\mathrm{MI}_{\mathrm{norm}}$ using the mutual information between the classification of the subsampled neuron against the others ($\mathrm{MI}_{\mathrm{norm}}$ of the smallest cluster).

All analyses were implemented in MATLAB.

Results

We first used simulated data (Dataset A) to investigate how well the spike sorting approaches separate the neurons. To that end, we used the mean percentage of extracted spike information (mean $\mathrm{MI}_{\mathrm{norm}}$) over the multiple runs of each set of waveforms (Fig. 4A). We found that PCA-based strategies had variable performance regardless of the information metric used, and presented a broad distribution of mean $\mathrm{MI}_{\mathrm{norm}}$. On the other hand, WD and wPCA had mean $\mathrm{MI}_{\mathrm{norm}}$ values concentrated around 90%, except when using WD with variance, whose distribution was similar to PCA-based strategies. We found that EToS also had a good performance (~90% of mean extracted spike information), while KlustaKwik had lower performance (~60%). These results were similar when analyzing the error rates instead of the $\mathrm{MI}_{\mathrm{norm}}$ (Supplementary Fig. S3).

Since the initial conditions of the algorithm may influence the final classification, we next investigated the consistency of the sorting results across runs, defined as the variance of $\mathrm{MI}_{\mathrm{norm}}$, which measures if the algorithm provides similar $\mathrm{MI}_{\mathrm{norm}}$ values over multiple runs using the same data. The PCA-based strategies and the WD-variance case had lower consistency (high variance of $\mathrm{MI}_{\mathrm{norm}}$) than wPCA and the other WD strategies in simulated data (Fig. 4B). EToS showed almost no variance, while KlustaKwik had low consistency across runs. We also investigated the mean number of clusters found in each run. Among GMM methods, the PCA-based strategies and WD-variance had the lowest mean number of clusters, followed by wPCA and the other WD strategies (Fig. 4C). EToS had the lowest number of clusters, while KlustaKwik the highest. Based on the mean and variance (consistency) of extracted spike information, we selected the variance metric for PCA and the $\mathrm{I}_{\mathrm{dist}}$ metric for either WD or wPCA as the best strategies (highlighted in Fig. 4).

We performed the same analysis for real neuron data recorded from tetrodes (Dataset B). To that end, we combined intracellularly confirmed single units from different recordings in groups of 5 or 10. We found that the mean extracted spike information (mean $\mathrm{MI}_{\mathrm{norm}}$) decreased as the number of neurons increased (Fig. 5A; see Supplementary Fig. S3 for mean error rates), and that the best strategies selected for the simulated dataset were also among the best in the real dataset (highlights in Fig. 5). The best wPCA and PCA approaches had similar mean $\mathrm{MI}_{\mathrm{norm}}$, slightly outperforming the best WD method. Differently from Dataset A, KlustaKwik had higher performance than EToS in Dataset B. The relative performance among methods (i.e., consistency and mean number of clusters; Fig. 5B,C) was comparable between the datasets, so that a metric with good consistency in Dataset A (Fig. 4B) also had good consistency in Dataset B (Fig. 5B). One exception, however, was that KlustaKwik exhibited the highest consistency only in Dataset B.

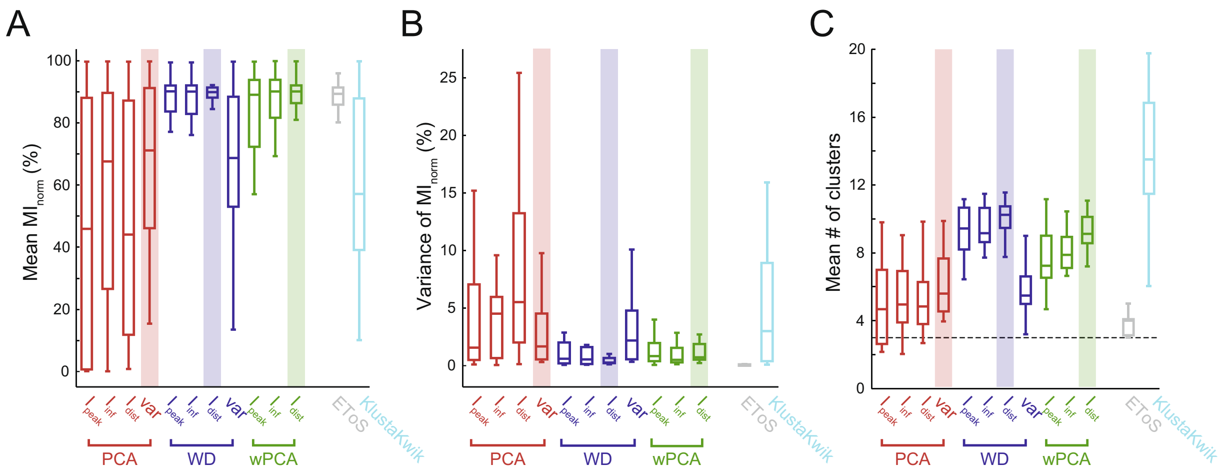

Figure 4. Spike sorting performance in a simulated dataset. (**A**) Boxplots show the distributions of the mean spike information (MI$_{norm}$) for each GMM-based feature extraction strategy. Twenty-five sets of 3 neurons from Dataset A were analyzed; for each set, the mean MI$_{norm}$ was computed using 25 runs. Mean MI$_{norm}$ values for ETоS and KlustaKwik are also shown. (**B**,**C**) Boxplots of MI$_{norm}$ variance (**B**) and mean number of detected clusters (**C**) across the 25 runs of each set of neurons. Note that since the GMM-based methods are based on overclustering, the number of clusters was always higher than the true number of neurons (dashed line). Merging of clusters is a common post-processing step for several sorters. Highlights show the best (high performance and low variance) metric of cluster information for each feature extraction approach. The var metric was the most informative for PCA (red), while I$_{dist}$ was the best metric for WD (blue) and wPCA (green).

We next compared the performance of the GMM-based best strategies (variance for PCA and I$_{dist}$ for WD and wPCA) with ETоS or KlustaKwik in a pairwise manner for Dataset A (Fig. 6) and Dataset B (Fig. 7). For Dataset A, we found that WD, wPCA and PCA outperformed KlustaKwik, and that WD and wPCA were similar to ETоS (Fig. 6). For Dataset B, we found that WD was similar to ETоS while the PCA and wPCA strategies had mean MI$_{norm}$ values higher than ETоS and slightly better than KlustaKwik (Fig. 7). Exploring different parameters for ETоS and KlustaKwik yielded similar results (Supplementary Fig. S5).

We then investigated how each algorithm performs at the different signal-to-noise ratios present in Dataset A (Fig. 8A). In this analysis, only the neuronal sets with more than one noise level were used (20 first rows of Supplementary Table S1). We computed the mean MI$_{norm}$ and the number of clusters for each set of waveforms at varying degrees of noise levels (Fig. 8B,C). The MI$_{norm}$ values of both PCA-variance and KlustaKwik were the most sensitive to signal-to-noise ratio, rapidly decreasing as the noise level increased. Regarding the mean number of clusters, the ETоS algorithm was the most robust across noise levels, and provided the best estimates for the number of clusters in the data. For the PCA, WD and wPCA strategies, the mean number of clusters tended to decrease with the noise level, whereas KlustaKwik, which had the overall highest number of clusters (see Fig. 4C), exhibited the opposite trend, increasing the number of clusters with noise.

Asymmetrical cluster sizes arise whenever neurons have very different firing rates (e.g., pyramidal cells and interneurons), which is often the case. We have thus analyzed the effects of unbalanced cluster sizes on clustering performance. To that end, we defined a symmetry index (see Materials and Methods), and randomly subsampled the datasets to match different values of symmetry index prior to performing the sorting procedure. We found that the mean MI$_{norm}$ was relatively uncorrelated with the symmetry index (Fig. 9A). However, when we computed a modified MI$_{norm}$, which takes into account only the information extracted from the smallest cluster, we found that this metric decreased for lower symmetry indexes (Fig. 9B), indicating that spikes from smaller clusters tend to be integrated into larger ones. In other words, the capacity of discriminating the smaller cluster lowers as the asymmetry in cluster size increases. WD and wPCA approaches were the least affected by unbalanced cluster sizes.

Since the GMM-based wPCA-I$_{dist}$ exhibited the highest extracted information and consistency across datasets, we consider it the best strategy under an overclustering scenario, in which clusters might be merged in a posterior step. Nevertheless, to investigate whether the feature extraction step could by itself improve the performance of another algorithm, we repeated the analyses of Figs 4 and 5 using the features defined by wPCA-I$_{dist}$ as input to KlustaKwik (Supplementary Fig. S4). For Dataset A, the modified KlustaKwik-wPCA had higher percentage of extracted information and consistency than the original KlustaKwik, with values approaching wPCA-I$_{dist}$ and WD-I$_{dist}$. The mean number of clusters remained roughly the same. However, for Dataset B the mean percentage of extracted information and consistency for KlustaKwik-wPCA remained the same as for KlustaKwik, while the mean number of clusters slightly increased. A pairwise comparison between KlustaKwik-wPCA and KlustaKwik is shown in Supplementary Fig. S6.

At last, we created a friendly graphical user interface to run the wPCA-I$_{dist}$ algorithm and inspect the resulting classification (Supplementary Fig. S7), which also allows for manual adjustments of cluster identity (e.g., merging

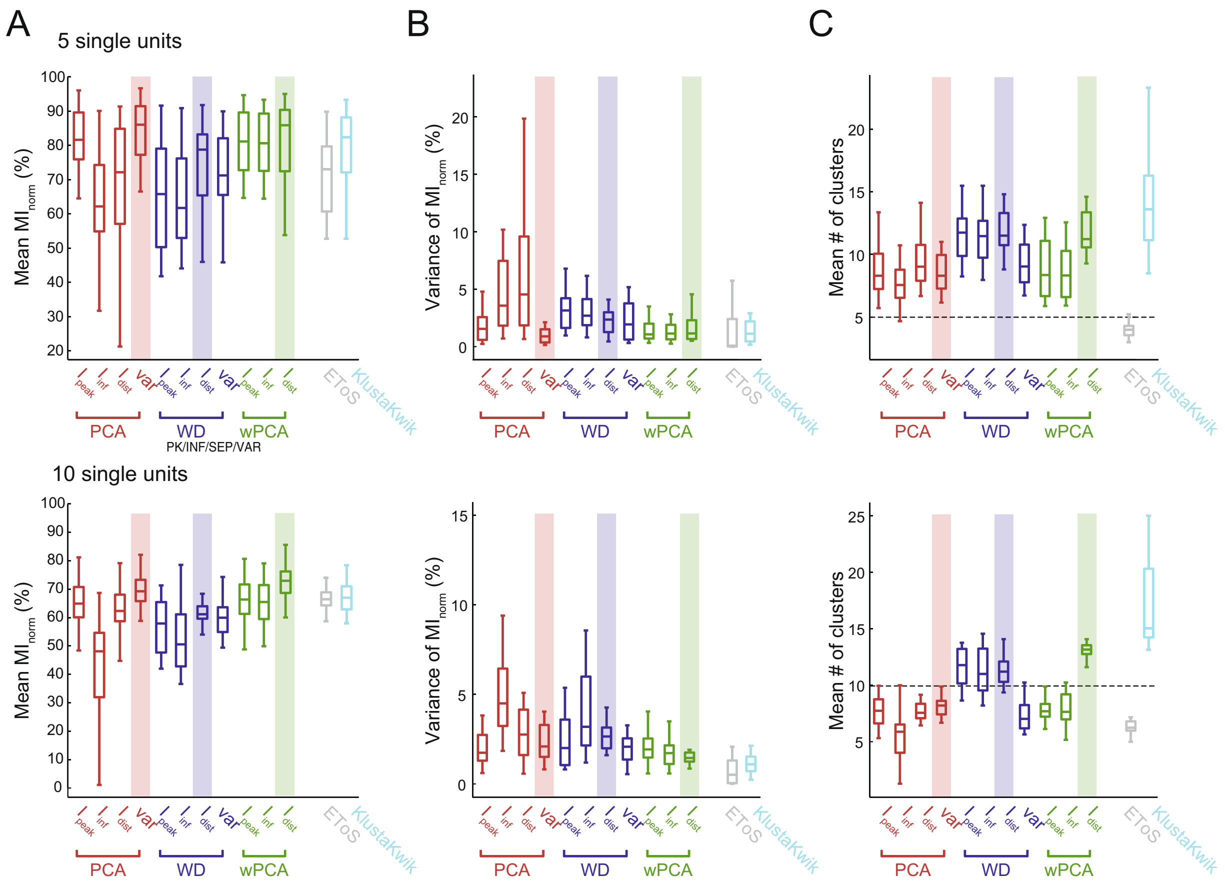

Figure 5. Spike sorting performance using real waveforms. Single units from Dataset B were pulled together in groups of 5 (top) or 10 (bottom) and used to test spike sorting performance. In this dataset, the identity of the extracellularly recorded waveforms was confirmed through simultaneous intracellular recordings. (**A**–**C**) Mean spike information (MI_{norm}) (**A**), MI_{norm} variance (**B**), and the mean number of detected clusters (**C**) are shown as in Fig. 4.

of clusters). We also provide an integrated pipeline with other spike sortings. The codes and instructions are freely available at https://github.com/tortlab/GMM-spike-sorting.

Discussion

In this work, we explored multiple strategies of feature extraction and selection using GMMs. We proposed GMM-based approaches to classify features and estimate the number of clusters in a data-driven way. These were compared to two spike sorting algorithms that are also based on mixture models[9,11].

The clustering step proposed here is based on an initial overclustering of the data (Fig. 3). We first built a GMM of the selected features which overestimated the number of clusters, resulting in a mixture model with more Gaussians than the real number of neurons. Then, we estimated the cluster number – along with their centers – through the peaks of the mixed probability of the model. This is possible because adding new Gaussians decreases the error of the model. Thus, the few extra Gaussians (beyond the actual number of clusters) can be seen as a smoothing of the probability function of the mixed model that does not change the number and position of its peaks. Using the peak positions as new Gaussian centers, we recalculated the GMM and defined the cluster regions based on the new Gaussian distributions. These determined the maximum *a posteriori* probability used for clustering.

We used known feature extraction techniques (PCA, WD and wPCA) and combined them with three unsupervised estimators of clustering separability. The variance, which is widely used in standard PCA approaches, was used as a fourth clustering metric. We found that the variance was the best estimator of separability for the PCA-based approaches. This is in accordance to the standard use of the PCA in feature extraction, which selects the PC scores of highest variance[4,9,21]. However, for the simulated dataset (Dataset A), the PCA-variance performance was below WD and wPCA approaches, especially when combining them with the I_{dist} metric (Fig. 4A). This supports previous work showing that WD can outperform PCA for spike sorting[13,14,28]. Nevertheless, the standard PCA presented better performance than WD in a second dataset composed of actual neurons (Dataset B; Fig. 5A). Only wPCA, which combines both PCA and WD into a single approach, was among the best methods in both datasets.

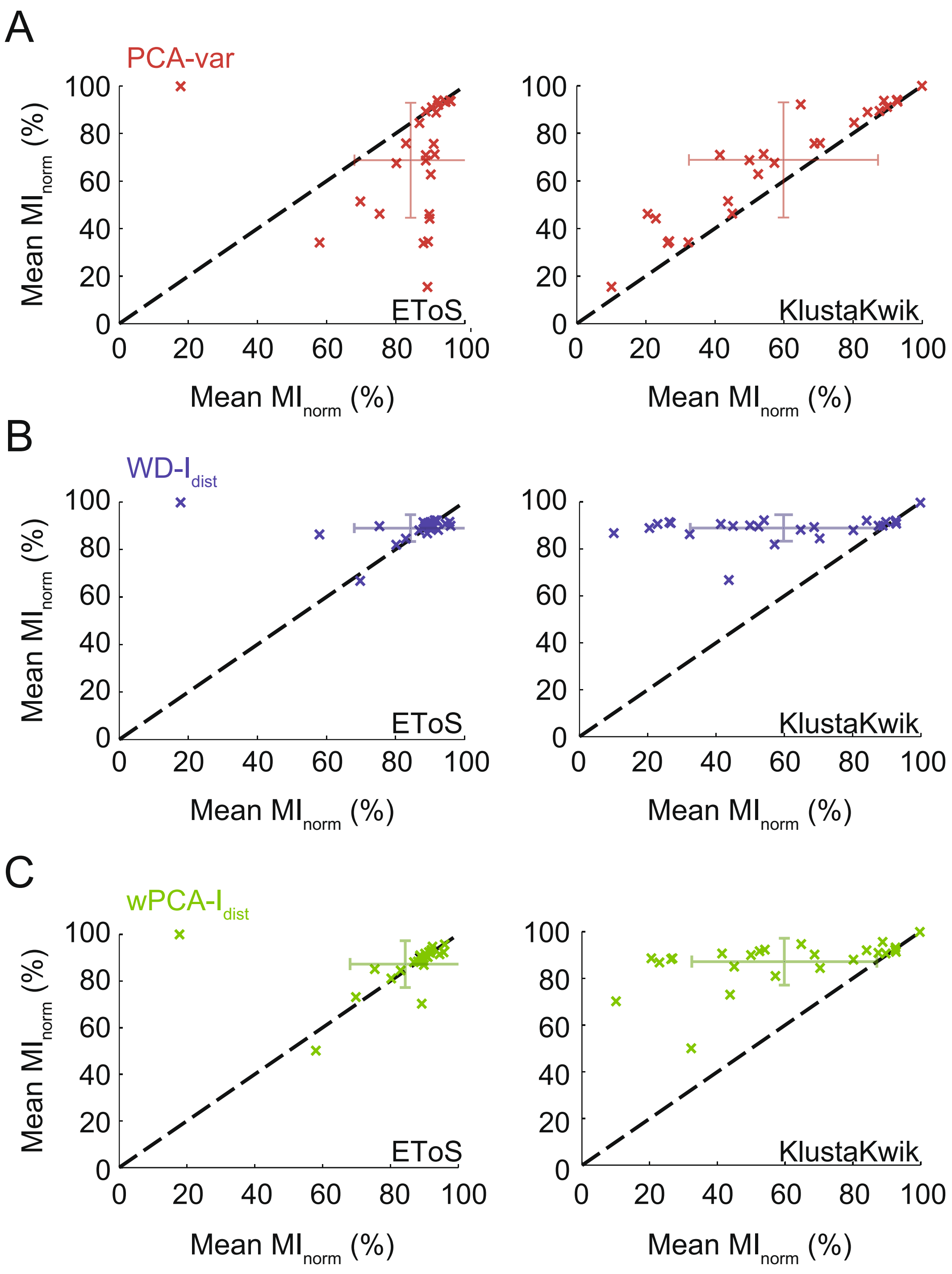

Figure 6. Comparing sorting performance in Dataset A. (**A**–**C**) Pair-wise comparison between the best strategy for PCA (**A**), WD (**B**) or wPCA (**C**) and EToS or KlustaKwik. Errorbars denote standard deviation around the mean. All strategies performed better than KlustaKwik, while WD-I_{dist} and wPCA-I_{dist} performed similarly to EToS.

We compared the performance of our sorting approaches to previously established algorithms: EToS and KlustaKwik, which are also based on mixture models[9,11]. Their performance varied according to the dataset used. KlustaKwik, which is based on PCA, performed better for Dataset B (Fig. 5A), as it was the case of our own PCA approach. This might be explained by the fact that earlier versions of KlustaKwik were developed using Dataset B. On the other hand, EToS, which also uses a combination of wavelets and wPCA, was best suited for Dataset A (Fig. 4A), similar to our WD approach. For each of the two datasets, our wPCA approach yielded better or comparable results to EToS and KlustaKwik, indicating it to be a good alternative for spike sorting (Figs 6 and 7).

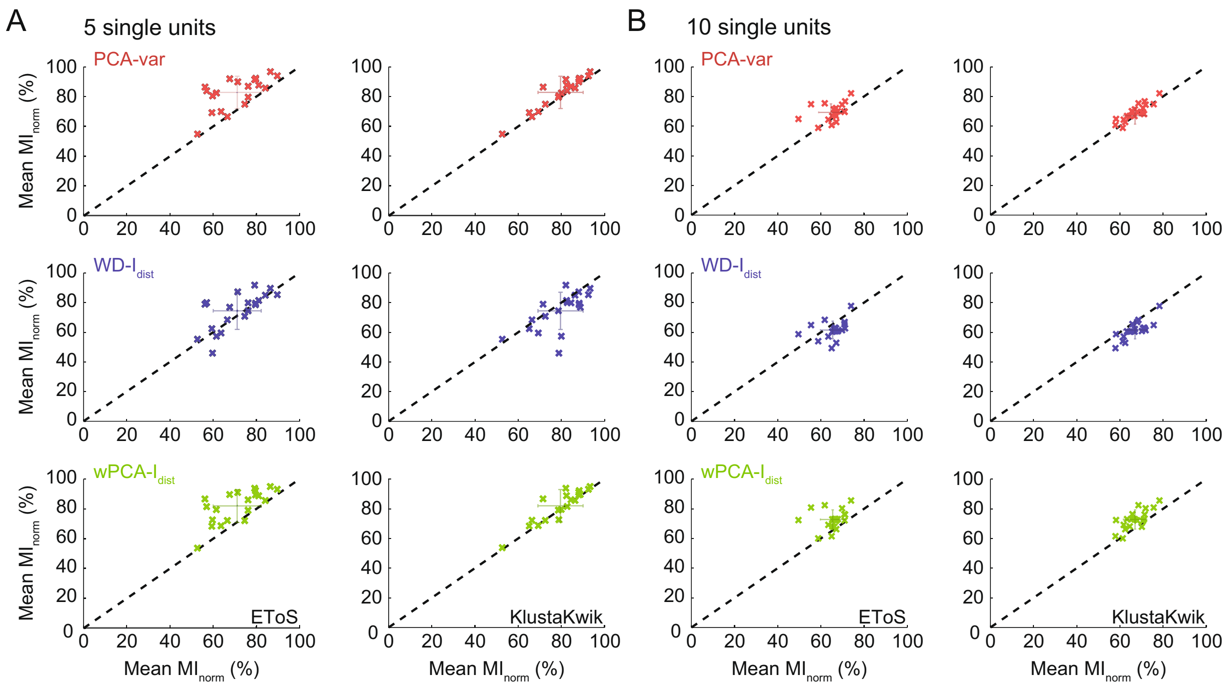

Figure 7. Comparing sorting performance in Dataset B. (**A,B**) Pairwise comparison between the best PCA, WD or wPCA strategies and EToS or KlustaKwik for groups of 5 (**A**) and 10 (**B**) single units in Dataset B. Errorbars denote standard deviation around the mean. PCA-var and wPCA-I_{dist} performed better than EToS and slightly better than KlustaKwik, while WD had similar performance to these methods.

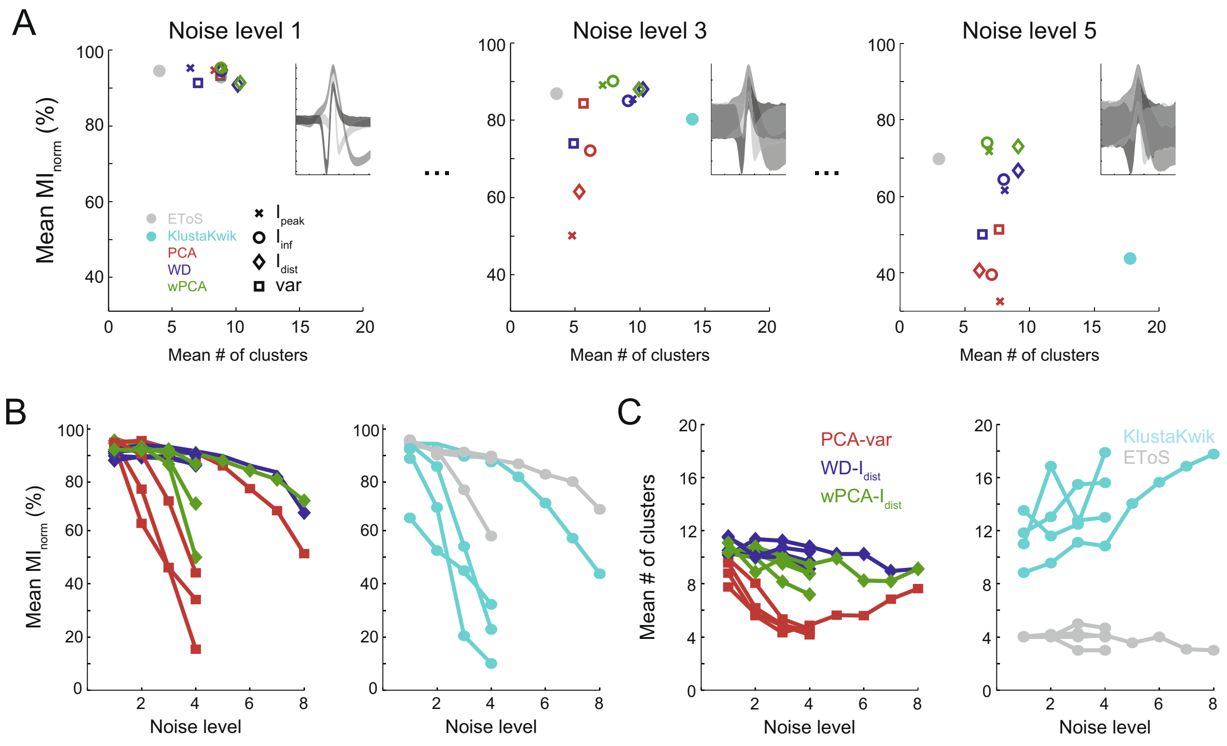

Figure 8. Spike sorting performance at different noise levels. (**A**) Average spike information (MI_{norm}) and number of detected clusters for the evaluated spike sorters in three example datasets (same set of 3 neurons differing in noise levels). The inset shows the 90% percent confidence interval (shadows) of the waveforms. (**B,C**) Mean MI_{norm} (**B**) and number of clusters (**C**) for all datasets in Dataset A plotted as a function of the noise level. The lines connect datasets formed by the same set of 3 neurons. Note that as noise increases, the classification performance decreases.

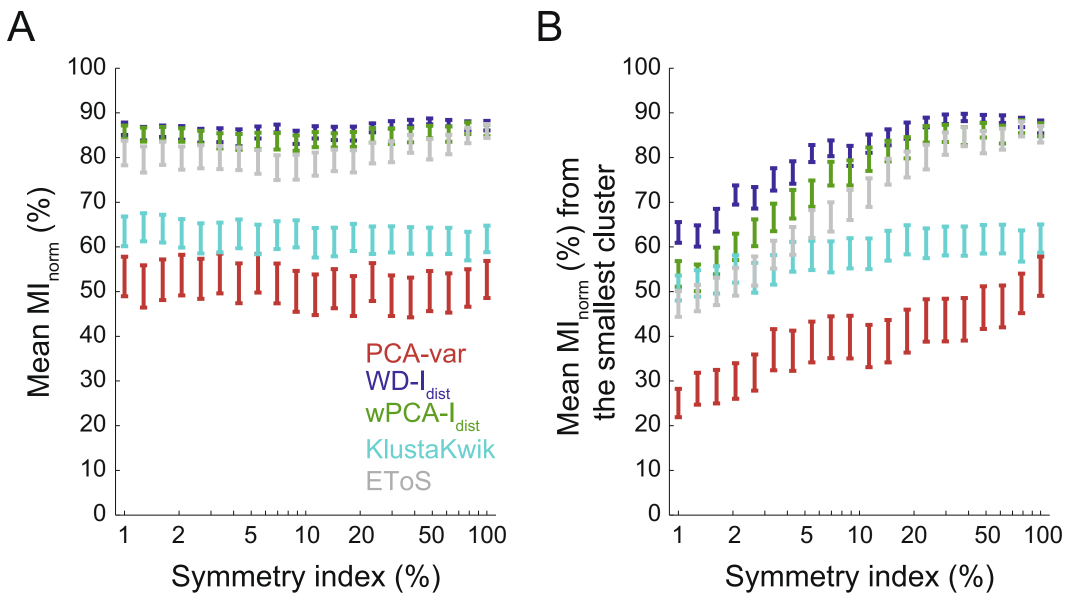

Figure 9. Sorting performance in unbalanced cluster sizes. (**A**) Average spike information (MI$_{norm}$) for sets of neurons with different cluster sizes (symmetry index). There were no apparent changes in MI$_{norm}$ across different symmetry indexes. (**B**) Similar to A, but showing only the spike information extracted from the smallest cluster. For this analysis, the other two clusters were joined prior to computing the MI$_{norm}$ (see Materials and Methods). Despite the apparent constancy of the overall information (**A**), the classification of the smallest cluster decreases with the cluster size (**B**).

The variability in EToS and KlustaKwik performance might be explained by the methods they are based on, and the characteristics of each dataset. Dataset A has, in general, a lower number of detected spikes in comparison to Dataset B (see Supplementary Table S1), which can influence the estimation of PCs for both KlustaKwik and our PCA-var method, explaining their lower performance in comparison to EToS and our other wavelet-based methods. On the other hand, EToS and our WD-I$_{dist}$ approach have a lower performance in comparison to PCA-based methods on Dataset B, which has a larger number of neurons to be sorted (5 or 10). This may suggest a limitation of using only wavelets to extract features, which does not occur when combining wavelets and PCA in the wPCA. Moreover, the PCA-var method had comparable MI$_{norm}$ to wPCA-I$_{dist}$ in the case of 5 single units (with very similar error rates; Supplementary Fig. S3), but showed a reasonable decrease in performance when sorting 10 single units, underestimating the number of clusters. This may indicate a high redundancy of the features selected in PCA, impairing the classification of all single units. This is not the case of wPCA, which does not underestimate the number of clusters and has higher performance in the 10 single unit case (Supplementary Fig. S8).

We next investigated how each algorithm performs at the different noise levels in Dataset A. The WD approach had the most robust performance, and it was closely followed by wPCA and EToS. On the other hand, we found that PCA and KlustaKwik were more sensitive, and rapidly decreased their performance as the noise level increased (Fig. 8B). These results support previous findings showing that the first PC scores capture a higher percentage of the noise energy when compared to WD[13].

Although we investigated sorting performance under different noise levels, we do not know how each method performs in the presence of outliers (e.g., superposition of different waveforms). It is possible that outliers influence the feature selection step and/or generate an extra cluster arising from a Gaussian with high standard deviation and low amplitude, due to our overclustering strategy. A possible strategy to deal with outliers is to perform spike sorting on a subset of prevalent waveforms (the "core samples"), and the remaining waveforms can then be classified (or discarded) using template matching[4]. We tested this approach by artificially adding coincident spikes in Dataset A and using the k-nearest neighborhood distance to define the core samples (Supplementary Fig. S9). As expected, outliers decreased overall sorting performance but left the classification of the core samples unaffected (Supplementary Fig. S9). Sorting performance could be improved when the identified outliers were removed prior to computing both the univariate and multivariate GMM and reinserted in the final model classification via template matching (Supplementary Fig. S9). Of note, since we use a Gaussian model for the final classification of waveforms, it is also possible to compute a confidence interval for each classification.

Previous work suggested that, because of its longer tail, a mixture model of t-Student distributions (instead of Gaussians) would be more indicated to deal with outliers[21]. However, for Dataset B our method showed better performance than EToS, which is based on t-Student's mixture model. Since they use different feature extraction methods, it is unknown whether changing Gaussians for t-distributions in our case would improve its performance. Nevertheless, our method can be adapted to other types of mixture models. In fact, the clustering and feature selection approaches proposed here can be used independently and combined with other strategies (e.g., our feature selection strategy could be combined with KlustaKwik, or our overclustering approach could be applied to other waveform features not investigated here).

For some clustering strategies, better performance was accompanied by an increased number of detected clusters (Figs 4C and 5C). Since high-performance approaches had high MI_{norm} values, this might be due to splitting the waveforms of a same neuron into more than one cluster (see Supplementary Fig. S2 and Fig. 3E). On the other hand, EToS had the most accurate number of clusters for Dataset A, while these were underestimated for Dataset B. This begs the question whether it is possible to optimize both classification performance (here defined as MI_{norm}) and the estimation of the correct number of clusters by a sorting algorithm. Spike sorters use differences in waveform shapes to separate neurons. However, the spikes of a same neuron can eventually have different waveforms shapes (e.g., the later spikes in a burst), and the clustering step will not be able to distinguish these cases from spikes arising from different neurons (see ref.[29]). Therefore, unless the classifier is able to use other variables to aid in the assignment of neuronal identity (e.g., spike timing), posterior adjustments are often required, either by a manual operator or another processing algorithm. Of note, in our GMM-based framework, merging of clusters is currently done manually using the GUI we developed (Supplementary Fig. S7). Notice that from a practical point of view it is simpler to merge clusters *a posteriori* than to separate them. In this sense, it should be noted that although EToS showed good overall performance, it often underestimated the number of clusters, which is an important limitation (see Fig. 5).

A variety of spike sorting methods have been proposed in recent years with improved capabilities of neuronal classification[9,14,21,30], but they still depend on parameter adjusting and manual curation steps[31]. Recent work has shown efforts for implementing a fully automated spike sorting[32], encapsulating these post-processing steps within the algorithm. However, this approach entirely focuses on the clustering stage, which leaves unanswered the question of how much feature extraction techniques can improve the final outcome. In our case, the adjustable parameters in our algorithm were the number of dimensions extracted (5) and the number of Gaussians in the feature extraction (8) and clustering steps (12 for Dataset A and 20 for Dataset B). In relation to the number of dimensions, most spike sorting approaches use only 2–3 features extracted by PCA. This is based on the fact that these features usually explain a large amount of the data variance and adding the other PCs does not increase classification performance. Since we also tested feature extraction methods other than PCA and which were not linked to variance, we increased the number of extracted features to five. Notably, this larger number of features does not explain the high performance of our methods, since our results show that replacing PCA by our wPCA-I_{dist} (and using only the 3 first wPCs) improves the performance of KlustaKwik (Supplementary Figs S4 and S6). However, whether there is an optimal number of extracted features is still an open question.

The number of Gaussians in the model defines the maximal number of clusters in the feature space. In other words, using 8 Gaussians in the first GMM means that this was the maximum number of neurons expected to be differentiated by a single feature. Since this initial step is only used to select features, in the case of I_{peak} and I_{inf} – which do not take into account the mean and variance of individual Gaussians – this approach could be replaced by other smoothing procedure of the empirical feature distribution or other mixing models. Similarly, in the multivariate GMM we assumed that there were at maximum 12 or 20 neurons to be separated in the single wire or tetrode recordings, respectively. But these assumptions can vary according to the type of recording and knowledge of the dataset (e.g., expected number of cells in the recorded region).

We also analyzed the performance of our approaches under different conditions of symmetry and found that the WD was the most robust algorithm, closely followed by wPCA and EToS. This is important because neurons with lower firing rates can go undetected, assimilated by larger clusters[33,34]. Notably, because each Gaussian in the mixture is fitted independently, GMMs (or other mixture models) offer a good solution to this problem as they can capture clusters of different size, assigning them to Gaussians with different standard deviations and weights. The GMM approach constitutes a free-scale smoothing whenever the data comes from multiple distributions with different scales (standard deviations), eliminating the need for defining the length (scale) of the smoothing function.

The quality of the detected waveforms can be improved by preprocessing techniques such as noise whitening or sampling jitter correction[35]. Since our method focuses on the feature extraction and clustering step of spike sorting, we did not investigate the specific effects of these techniques, which are involved in spike detection. However, it is expected that improving the quality of waveforms also improves the performance of the tested spike sorters.

Although features extracted by PCA and wPCA are not independent, they are uncorrelated, which helps to reduce the redundancy among them. This might explain why wPCA, which is also based on wavelet coefficients, is slightly better than the WD approach. In this sense, applying additional measures of independence directly on the features, such as mutual information or negentropy as used in ICA, may further ensure low levels of redundancy and improve performance. Notice that this differs from the approach of applying ICA to different channels[36–38].

Noteworthy, the current advances in recording technology have brought additional issues to the spike sorting problem[39,40]. High-density electrode arrays and their compactness now allow for recording thousands of spikes, whose waveforms are often spatially overlapped[40–43] (i.e., appear in more than one electrode). Although our algorithm provides insights into new spike sorting alternatives, the current implementation of our method is not yet optimized for parallel computing or to deal with high-density electrode arrays. This is an important limitation since analyzing the waveforms detected in the entire array increases the challenge of separating synchronous spikes and must take into account anatomical information. Recent algorithms such as MountainSort[32], SpykingCircus[44], Kilosort[45] and YASS[46] implement different solutions to solve this problem. They either perform the spike sorting in each channel individually and then merge clusters that represent the same neuron in different electrodes[47,48], or sort ensembles of spatially related waveforms/electrodes masking unrelated ones[9,30,32,44,46]. In both cases, our proposed algorithm for feature extraction could easily replace any algorithm that uses PCA prior to clustering[30,32,46,47,49], similarly to the combination of wPCA and KlustaKwik done here which led to an increase in performance (Supplementary Figs S4 and S6). We believe this gives room for important explorations in current spike sorting solutions that might help to extend the information bound through identification of better features to extract. As an example, we made available online an adaptation of YASS – which considers the spatial relation between channels – implementing our feature extraction step (Supplementary Fig. S10; https://github.com/tort-lab/GMM-spike-sorting). This sort of adaptation requires extra steps to compute the wavelet coefficients, which depend on the number of spikes, and to compute the GMMs for the features, which depend on the number of spikes and features, thus increasing computational cost. Although all the coefficients must be computed and projected on the feature space, the first operation can be efficiently done using matrix multiplication and parallel computing/GPUs. The computation of GMMs, on the other hand, can be limited to a subsample of the waveforms without significant impact on the model, so that its cost mainly depends on the number of extracted features.

Our clustering algorithm differs from other approaches using mixtures such as YASS or KlustaKwik due to the simplicity of cluster estimation. Instead of using split and merging parameters to decrease or increase the number of mixtures, the algorithm simply searches for the peaks in the probability of an overestimated model. The rationale behind this is that a model with a few more Gaussians (or other distribution) than clusters will decrease the model error globally rather than locally, without adding more peaks in the overall probability of the model. After finding the cluster number and centers, it is straightforward to determine the cluster identity of each feature sample. Any mixture-based spike sorter could be adapted to this step without much increase in the complexity of the algorithm since it only requires computing an extra model and searching for local minima.

In summary, our results bring new tools to tackle the spike sorting problem. Concerning the initial steps of feature extraction, we proposed fitting a GMM to wavelet coefficients or PC scores to estimate its clustering separability under a variety of metrics. Our results show that the combination of WD and PCA (the wPCA) is a better approach than using one of them separately as employed by most spike sorters[9,13,14,32]. Namely, the wPCA associated with a simple metric of distance between Gaussians (I_{dist}) was the best strategy of feature extraction in the two investigated datasets (Supplementary Fig. S8) and can be easily adapted to other spike sorters that deal with high-density electrode recordings, which is an essential limitation of our pipeline. Finally, in the step of unsupervised clustering, we proposed fitting an overclustered GMM, searching for local maxima to estimate cluster positions and then re-fitting the GMM with Gaussians at the estimated positions. This simple strategy can also be combined with multiples runs with different numbers of Gaussians, as done in previous studies[50,51]. We hope these findings shed new light on spike sorting and other unsupervised clustering problems.

Data Availability

The datasets analyzed in the current study are available at https://crcns.org/data-sets/hc/hc-1 and https://www2.le.ac.uk/centres/csn/research-2/spike-sorting.

References

1. Csicsvari, J., Hirase, H., Czurkó, A., Mamiya, A. & Buzsáki, G. Oscillatory coupling of hippocampal pyramidal cells and interneurons in the behaving rat. *J. Neurosci.* **19**, 274–287 (1999).
2. Ranck, J. B. Studies on single neurons in dorsal hippocampal formation and septum in unrestrained rats. *Exp. Neurol.* **41**, 462–531 (1973).
3. Barthó, P. *et al*. Characterization of neocortical principal cells and interneurons by network interactions and extracellular features. *J. Neurophysiol.* **92**, 600–608 (2004).
4. Lewicki, M. S. A review of methods for spike sorting: the detection and classification of neural action potentials. *Netw. Comput. Neural Syst.* **9**, R53–R78 (1998).
5. Brown, E. N., Kass, R. E. & Mitra, P. P. Multiple neural spike train data analysis: state-of-the-art and future challenges. *Nat. Neurosci.* **7**, 456–461 (2004).
6. Buzsáki, G. Large-scale recording of neuronal ensembles. *Nat. Neurosci.* **7**, 446–451 (2004).
7. Quiroga, R. Q. Spike sorting. *Curr. Biol.* **22**, R45–R46 (2012).
8. Harris, K. D., Henze, D. A., Csicsvari, J., Hirase, H. & Buzsáki, G. Accuracy of tetrode spike separation as determined by simultaneous intracellular and extracellular measurements. *J. Neurophysiol.* **84**, 401–414 (2000).
9. Kadir, S. N., Goodman, D. F. & Harris, K. D. High-dimensional cluster analysis with the masked EM algorithm. *Neural Comput.* **26**, 2379–2394 (2014).
10. Takekawa, T., Isomura, Y. & Fukai, T. Accurate spike sorting for multi-unit recordings. *Eur. J. Neurosci.* **31**, 263–272 (2010).
11. Takekawa, T., Isomura, Y. & Fukai, T. Spike sorting of heterogeneous neuron types by multimodality-weighted PCA and explicit robust variational Bayes. *Front. Neuroinformatics* **6** (2012).
12. Wang, H.-Y. & Wu, X.-J. Weighted PCA space and its application in face recognition. In Machine Learning and Cybernetics, 2005. Proceedings of 2005 International Conference on **7**, 4522–4527 (IEEE, 2005).
13. Hulata, E., Segev, R. & Ben-Jacob, E. A method for spike sorting and detection based on wavelet packets and Shannon's mutual information. *J. Neurosci. Methods* **117**, 1–12 (2002).
14. Quiroga, R., Nadasdy, Z. & Ben-Shaul, Y. Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering. *Neural Comput.* **16**, 1661–1687 (2004).
15. Zouridakis, G. & Tam, D. C. Multi-unit spike discrimination using wavelet transforms. *Comput. Biol. Med.* **27**, 9–18 (1997).
16. Strang, G. & Nguyen, T. Wavelets and filter banks. (SIAM, 1996).
17. Jain, A. K., Duin, R. P. W. & Mao, J. Statistical pattern recognition: A review. *IEEE Trans. Pattern Anal. Mach. Intell.* **22**, 4–37 (2000).
18. Lewicki, M. S. Bayesian modeling and classification of neural signals. *Neural Comput.* **6**, 1005–1030 (1994).
19. Sahani, M., Pezaris, J. S. & Andersen, R. A. On the separation of signals from neighboring cells in tetrode recordings. *Adv. Neural Inf. Process. Syst.* 222–228 (1998).
20. Peel, D. & McLachlan, G. J. Robust mixture modelling using the t distribution. *Stat. Comput.* **10**, 339–348 (2000).
21. Shoham, S., Fellows, M. R. & Normann, R. A. Robust, automatic spike sorting using mixtures of multivariate t-distributions. *J. Neurosci. Methods* **127**, 111–122 (2003).
22. Mallat, S. G. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE Trans. Pattern Anal. Mach. Intell.* **11**, 674–693 (1989).
23. McLachlan, G. & Peel, D. Finite mixture models. (John Wiley & Sons, 2004).
24. Lagarias, J. C., Reeds, J. A., Wright, M. H. & Wright, P. E. Convergence properties of the Nelder–Mead simplex method in low dimensions. *SIAM J. Optim.* **9**, 112–147 (1998).

25. Henze, D. A. *et al*. Simultaneous intracellular and extracellular recordings from hippocampus region CA1 of anesthetized rats. *CRCNS. org* (2009).
26. Shannon, C. E. A mathematical theory of communication, Part I, Part II. *Bell Syst Tech J* **27**, 623–656 (1948).
27. Magri, C., Whittingstall, K., Singh, V., Logothetis, N. K. & Panzeri, S. A toolbox for the fast information analysis of multiple-site LFP, EEG and spike train recordings. *BMC Neurosci.* **10**, 81 (2009).
28. Pavlov, A., Makarov, V. A., Makarova, I. & Panetsos, F. Sorting of neural spikes: When wavelet based methods outperform principal component analysis. *Nat. Comput.* **6**, 269–281 (2007).
29. Ventura, V. & Gerkin, R. C. Accurately estimating neuronal correlation requires a new spike-sorting paradigm. *Proc. Natl. Acad. Sci* (2012).
30. Rossant, C. *et al*. Spike sorting for large, dense electrode arrays. *Nat. Neurosci.* **19**, 634–641 (2016).
31. Hazan, L., Zugaro, M. & Buzsáki, G. Klusters, NeuroScope, NDManager: A free software suite for neurophysiological data processing and visualization. *J. Neurosci. Methods* **155**, 207–216 (2006).
32. Chung, J. E. *et al*. A Fully automated approach to spike sorting. *Neuron* **95**, 1381–1394.e6 (2017).
33. Quiroga, R. Q. Concept cells: the building blocks of declarative memory functions. *Nat. Rev. Neurosci.* **13**, 587–597 (2012).
34. Rey, H. G., Pedreira, C. & Quiroga, R. Q. Past, present and future of spike sorting techniques. *Brain Res. Bull.* **119**, 106–117 (2015).
35. Pouzat, C., Mazor, O. & Laurent, G. Using noise signature to optimize spike-sorting and to assess neuronal classification quality. *J. Neurosci. Methods* **122**, 43–57 (2002).
36. Takahashi, S., Anzai, Y. & Sakurai, Y. A new approach to spike sorting for multi-neuronal activities recorded with a tetrode—how ICA can be practical. *Neurosci. Res.* **46**, 265–272 (2003).
37. Mamlouk, A. M., Sharp, H., Menne, K. M., Hofmann, U. G. & Martinetz, T. Unsupervised spike sorting with ICA and its evaluation using GENESIS simulations. *Neurocomputing* **65**, 275–282 (2005).
38. Jäckel, D., Frey, U., Fiscella, M., Franke, F. & Hierlemann, A. Applicability of independent component analysis on high-density microelectrode array recordings. *J. Neurophysiol.* **108**, 334–348 (2012).
39. Einevoll, G. T., Franke, F., Hagen, E., Pouzat, C. & Harris, K. D. Towards reliable spike-train recordings from thousands of neurons with multielectrodes. *Curr. Opin. Neurobiol.* **22**, 11–17 (2012).
40. Lefebvre, B., Yger, P. & Marre, O. Recent progress in multi-electrode spike sorting methods. *J. Physiol.-Paris* **110**, 327–335 (2016).
41. Jun, J. J. *et al*. Fully integrated silicon probes for high-density recording of neural activity. *Nature* **551**, 232 (2017).
42. Neto, J. P. *et al*. Validating silicon polytrodes with paired juxtacellular recordings: method and dataset. *J. Neurophysiol.* **116**, 892–903 (2016).
43. Marques-Smith, A. *et al*. Recording from the same neuron with high-density CMOS probes and patch-clamp: a ground-truth dataset and an experiment in collaboration. *bioRxiv* 370080 (2018).
44. Yger, P. *et al*. A spike sorting toolbox for up to thousands of electrodes validated with ground truth recordings *in vitro* and *in vivo*. *eLife* **7**, e34518 (2018).
45. Pachitariu, M., Steinmetz, N. A., Kadir, S. N., Carandini, M. & Harris, K. D. Fast and accurate spike sorting of high-channel count probes with KiloSort. *In Advances in Neural Information Processing Systems* 4448–4456 (2016).
46. Lee, J. H. *et al*. Yass: Yet another spike sorter. *In Advances in Neural Information Processing Systems* 4002–4012 (2017).
47. Swindale, N. V. & Spacek, M. A. Spike sorting for polytrodes: a divide and conquer approach. *Front. Syst. Neurosci.* **8**, 6 (2014).
48. Marre, O. *et al*. Mapping a complete neural population in the retina. *J. Neurosci.* **32**, 14859–14873 (2012).
49. Hilgen, G. *et al*. Unsupervised spike sorting for large-scale, high-density multielectrode arrays. *Cell Rep.* **18**, 2521–2532 (2017).
50. Yamaguchi, Y., Aota, Y., McNaughton, B. L. & Lipa, P. Bimodality of theta phase precession in hippocampal place cells in freely running rats. *J. Neurophysiol.* **87**, 2629–2642 (2002).
51. Zheng, C., Bieri, K. W., Trettel, S. G. & Colgin, L. L. The relationship between gamma frequency and running speed differs for slow and fast gamma rhythms in freely behaving rats: slow and fast gamma correlations with speed. *Hippocampus* **25**, 924–938 (2015).

Acknowledgements
This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001 and by Conselho Nacional de Desenvolvimento Científico e Tecnológico - Brasil (CNPq). We thank Aron Miranda for critical comments on the software and Nick Steinmetz for sharing high-density electrode recording samples. We also thank the Quiroga and Buzsáki laboratories for making data publicly available.

Author Contributions
B.C.S. and V.L.d.S. designed the study under the supervision of A.B.L.T.; B.C.S. performed the analysis; B.C.S. and J.B. tested the software; B.C.S. and A.B.L.T. wrote the main manuscript text.

Additional Information
Supplementary information accompanies this paper at https://doi.org/10.1038/s41598-019-39986-6.

Competing Interests: The authors declare no competing interests.