bioRxiv preprint doi: https://doi.org/10.1101/067843; this version posted August 4, 2016. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

Fast and accurate spike sorting *in vitro* and *in vivo* for up to thousands of electrodes

Pierre Yger[1,2], Giulia L.B. Spampinato[1,2], Elric Esposito[1,2]
Baptiste Lefebvre[1], Stéphane Deny[1], Christophe Gardella[1,3], Marcel Stimberg[1]
Florian Jetter[4], Guenther Zeck[4], Serge Picaud[1], Jens Duebel[1] and Olivier Marre[1]

August 4, 2016

[1]Institut de la Vision, INSERM UMRS 968, UPMC UM 80, CNRS UMR 7210, Paris. [2]These authors contributed equally to this work. [3] Laboratoire de Physique Statistique, CNRS, ENS, UPMC, 75005 Paris. [4]NMI, Neurophysics Group, Reutlingen, Germany.

Abstract

Understanding how assemblies of neurons encode information requires recording large populations of cells in the brain. In recent years, multi-electrode arrays and large silicon probes have been developed to record simultaneously from hundreds or thousands of electrodes packed with a high density. However, these new devices challenge the classical way to do spike sorting. Here we developed a new method to solve these issues, based on a highly automated algorithm to extract spikes from extracellular data, and show that this algorithm reached near optimal performance both *in vitro* and *in vivo*. The algorithm is composed of two main steps: 1) a "template-finding" phase to extract the cell templates, i.e. the pattern of activity evoked over many electrodes when one neuron fires an action potential; 2) a "template-matching" phase where the templates were matched to the raw data to find the location of the spikes. The manual intervention by the user was reduced to the minimal, and the time spent on manual curation did not scale with the number of electrodes. We tested our algorithm with large-scale data from *in vitro* and *in vivo* recordings, from 32 to 4225 electrodes. We performed simultaneous extracellular and patch recordings to obtain "ground truth" data, i.e. cases where the solution to the sorting problem is at least partially known. The performance of our algorithm was always close to the best expected performance. We thus provide a general solution to sort spikes from large-scale extracellular recordings.

bioRxiv preprint doi: https://doi.org/10.1101/067843; this version posted August 4, 2016. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

Introduction

Throughout the brain, local circuits represent information using large populations of neurons [Buzsaki, 2010], and technologies to record hundreds or thousands of neurons simultaneously in the brain are therefore essential. One of the most powerful and widespread techniques for neuronal population recording is extracellular electrophysiology. Recently, newly developed microelectrode arrays have allowed recording the local voltage from hundreds to thousands of extracellular sites separated only by tenth of microns [Berdondini et al., 2005, Fiscella et al., 2012, Lambacher et al., 2004], giving an indirect access to large neural ensembles with a high spatial resolution. In these recordings, the spikes from each recorded neuron produce extracellular waveforms with a characteristic spatio-temporal profile across the recording sites. To access the spiking activity of individual neurons, we need to reconstruct the waveform produced by each neuron and tell when it appears in the recording. This process, called spike sorting, has received a lot of attention for recordings with a small number of electrodes. However, for large-scale and dense recordings, the problem of reliably extracting the spike contributions from extracellular recordings is still largely unresolved.

Current spike sorting algorithms cannot process this new type of data for several reasons. First, many algorithms do not take into account that the spikes of a single neuron will evoke a voltage deflection on many electrodes. Then, most algorithms do not scale up to hundreds or thousands of electrodes *in vitro* and *in vivo*, because their computation time would increase exponentially with the number of electrodes [Rossant et al., 2016]. Finally, all the solutions proposed so far usually require a significant amount of manual curation. Datasets from thousands of electrodes necessitate a spike sorting algorithm that is extensively automated. Furthermore, the few algorithms that have been designed to process large-scale recordings have not been tested on data where one neuron is recorded by the large-scale recordings and simultaneously by another technique, so that the success rate of the spike sorting algorithm can be measured [Pachitariu et al., 2016, Leibig et al., 2016, Hilgen et al., 2016].

Here we present a novel method for spike sorting that can scale up to recordings from thousands of electrodes. Based on a combination of density-based clustering and template matching, the method is fully automated, and only includes a final step of manual curation whose duration does not scale with the number of recorded cells. We tested the performance of the algorithm in two ways. First, we created several synthetic or "hybrid" datasets to properly quantify the performance of the algorithm. Then, we performed experiments where a large-scale extracellular recording was performed while one of the neurons was recorded with a patch electrode, and show that the performance of our algorithm was always close to an optimal classifier, both *in vitro* and *in vivo*. Therefore, our method appears to be a general, fast and scalable solution for spike sorting.

Results

We sorted several datasets taken from large-scale *in vitro* and *in vivo* recordings. In all cases, electrodes were densely packed such that a spike from a single cell would affect the voltage of several electrodes. In vitro, we analyzed recordings from retinal ganglion cells recorded with planar multi-electrode arrays: rodent retina were recorded either by 252 electrodes spaced by either 30 or 60 microns, or by 4255 electrodes spaced by 16 microns. In vivo, we analyzed recordings of the cortex taken with 32 and 128 electrodes spaced by 20 microns (see Methods).

Identifying templates for each cell

We developed an algorithm with two main steps, a clustering followed by a template matching step (see Methods for details). First, we detected the spikes as threshold crossings (fig. 1A),

bioRxiv preprint doi: https://doi.org/10.1101/067843; this version posted August 4, 2016. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

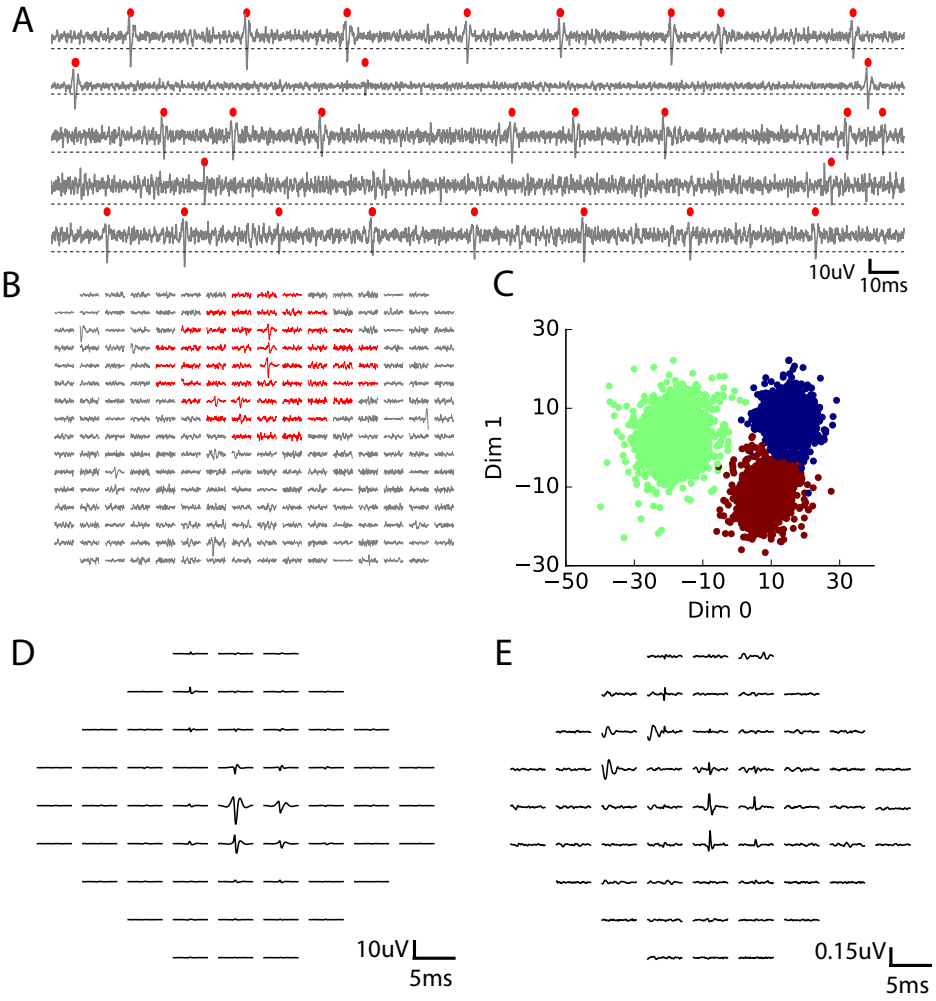

Figure 1: Local clustering of the snippets **A.** Five randomly chosen channels, each of them with its own detection threshold (dash dotted line). Detected spikes, as threshold crossings, are indicated with a red marker **B.** Example of a spike in the raw data. Red: electrodes that can be affected by the spike, i.e. the ones close enough to the electrode where the voltage peak is the highest ; gray: other electrodes that should not be affected. **C.** For snippets collected in a given electrode, a robust density-based clustering detected three non-overlapping color-coded clusters (see Methods) **D.** The first component of the template $\boldsymbol{T}(t)$, corresponding to the green cluster shown in **C. E.** The second component of the template $\boldsymbol{U}(t)$ associated to the same green cluster (see Methods).

bioRxiv preprint doi: https://doi.org/10.1101/067843; this version posted August 4, 2016. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

and isolated the extracellular waveforms for a number of randomly chosen spike times, that we call snippets in the following. Snippets were then clustered in different groups that should correspond to putative cells. Compared to classical sorting methods, here our purpose was not to obtain a full clustering of all the snippets, but rather to get the centroid of each cluster, which is way less demanding. We also made several natural assumptions so that this clustering step could be divided in parallel tasks that could be run independently.

We divided the snippets in subsets according to their physical positions: we grouped together the snippets peaking on the same electrode. Following this we had as many groups of snippets as electrodes. Each group was then clustered independently. To efficiently cluster the snippets inside each of these groups, we had to reduce the dimensionality of the snippets. We assumed that a single cell can only influence the electrodes in its vicinity, i.e. close enough to its physical location (fig. 1B). We discarded all the electrodes whose distance to the peaking electrode was above a pre-defined radius (see fig. 1B). We also projected each snippet on stereotyped temporal waveforms to reduce the dimensionality (see Methods).

This subdivision in groups allows clustering each group in parallel, such that the computation time of this step only scales linearly with the number of electrodes, and can be divided by the number of computing cores available. We then performed a density-based clustering inspired by [Rodriguez and Laio, 2014] (see Methods) on each group. This density-based clustering is very efficient at finding the centroid of each cluster, which meets exactly our needs (see fig. 1C for three identified clusters on a given electrode). We adapted the algorithm to increase its robustness, and to avoid fine tuning parameters for a given dataset (see Methods). The resulting clusters were only used to define a "template" for each cell.

This template is a simplified description of the cluster and is composed of two waveforms. The first one is the average extracellular waveform inside the cluster (fig. 1D). The second is the direction of largest variance that is orthogonal to this average waveform (fig 1E ; see Methods). We assumed that each snippet (i.e. voltage deflection) triggered by this cell is a linear combination of these two components. Finding these components for each cell does not require to have all the spikes of these cells clustered together: only a subset is good enough to estimate them. Therefore, this template extraction step works even if the clustering step only assigned a subset of spikes, which allowed processing very large datasets.

At the end of this first step, we have extracted an ensemble of templates (i.e. pairs of waveforms) that correspond to putative cells. By focusing on only getting the cluster centroids, we have made the clustering task easier. However, we have not assigned all the spikes to a cell. Therefore, we used a template matching algorithm to find all the instances where each cell has emitted a spike.

Matching templates to the raw data

We assumed that the templates of different cells spiking together sum linearly and used a greedy iterative approach inspired by the projection pursuit algorithm (fig. 2A, see Methods). Within a piece of raw data, we looked for the template whose first component had the highest similarity to the raw signal and matched its amplitude to the signal. If this amplitude falls between pre-determined thresholds (fig. 2A, B, C), we matched and subtracted the two components to the raw signal. We then re-iterated this matching process until no spike could be matched anymore (fig. 2D) (see Methods). In most cases, the spike times of a given template showed a nice refractory period (fig. 2E).

This template matching algorithm could be run independently on different blocks of data, such that the computing time only scaled linearly with the data length. We also took advantage of GPU computing resources to accelerate the computations. Each step of the spike sorting algorithm was parallelized, so the runtime of the full algorithm decreased proportionally with the numbers of CPU cores available (fig. 3A). As a result, the sorting could process one hour

bioRxiv preprint doi: https://doi.org/10.1101/067843; this version posted August 4, 2016. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

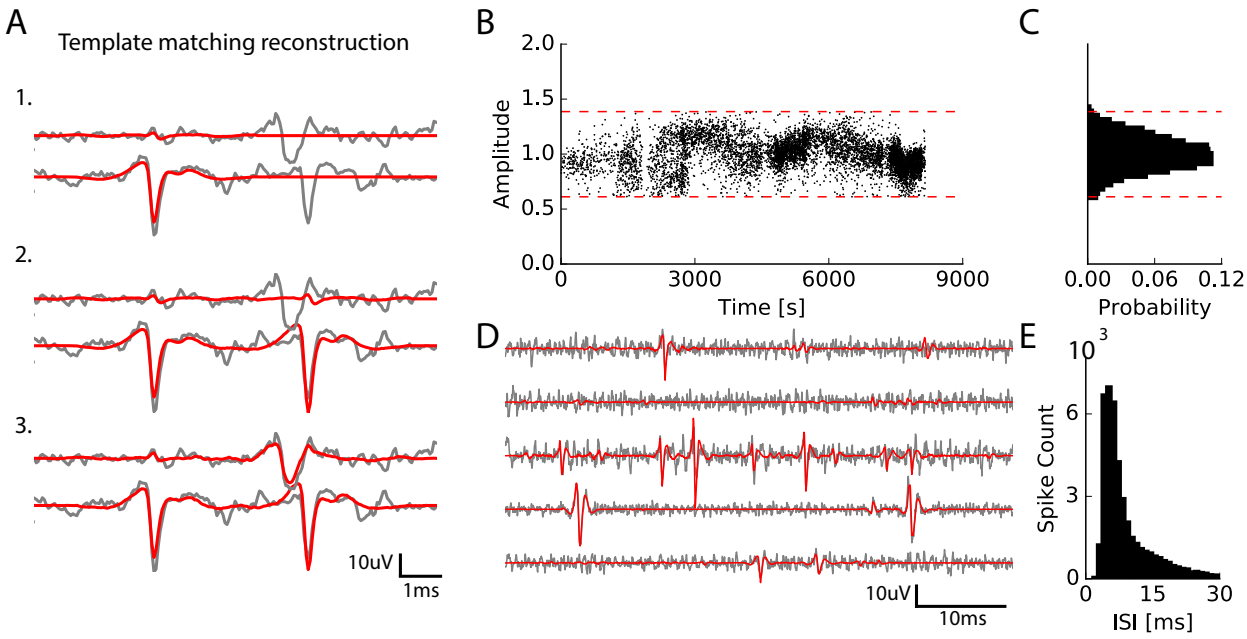

Figure 2: Illustration of the template matching phase **A.** Graphical illustration of the template matching with *in vitro* data (see Methods). Every line is a channel. Grey: the real data. Red: sum of the templates added by the template matching algorithm ; top to bottom: successive steps of the template matching algorithm (red). **B.** Examples of the fitted amplitudes for the first component of a given template, as a function of time. Each dot correspond to a spike time at which that particular template has been fitted to the data. Dashed dotted lines represent the boundaries allowed during the matching procedure. **C.** Distribution of the amplitudes for the same template, centered around 1. **D.** Final result of the template matching. Gray: extracellular signals for five channels. Red: sum of the matched templates. **E.** A typical Inter-Spike Interval (ISI) for a given template, showing a clear refractory period.

bioRxiv preprint doi: https://doi.org/10.1101/067843; this version posted August 4, 2016. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

of data recorded with 252 electrodes in one hour with 9 CPU cores (spread over 3 computers) (fig. 3A, B). It also scaled up linearly with the number of electrodes (fig. 3B), and with the number of templates (fig. 3C). It was therefore possible to run it on long recordings ($\geq$ 30min) with more than 4000 electrodes.

Merging similar templates

After running our algorithm, despite a cleaning step where we removed obvious duplicates or sums of templates (see Methods), it is very likely that a single cell can be represented by two templates or more. Therefore, it is necessary to merge the templates and the corresponding spike trains that represent the same cell. Two criteria can be used to merge templates corresponding to the same cell. First, their templates should be similar. Second, there should be a dip around 0 in the cross-correlogram between the two spike trains (fig. 3D). This is due to the refractory period of the cell: two consecutive spikes from the same cell have to be separated by at least a couple of milliseconds.

To detect which pairs of spike trains had a dip in their correlogram, we computed the difference between the cross-correlogram of the two spike trains and a control, time-reversed correlogram over a given time window (fig. 3D; see Methods). This difference was plotted against the control value over the same time window for all the pairs with a high template similarity (fig. 3D). Almost all the pairs that needed to be merged appeared as an isolated group of points close to the equality line, which corresponds to a very strong dip in the cross-correlogram. To demonstrate this, we first simulated an ensemble of spike trains belonging to fake neurons and we randomly split them into several subsets (see Methods). This simulated data gave us a case where we know exactly which pairs need to be merged, and which ones correspond to different cells. We observed that all the pairs that had to be merged were grouped along the equality line, as expected (fig. 3D, left).

In a more realistic example, we took spike trains extracted from a large-scale recording where the merging was done manually (fig. 3E), and observed a similar grouping. In this representation, all the pairs that needed to be merged could therefore be selected all at once. We developed a graphical user interface to automatically plot these pairs, visualize individual cross-correlograms, and select the group of points of pairs that needed to be merged. The user could then quickly isolate and select the pairs of templates to be merged. By deciding which pairs needed to be merged from a single plot, this method avoids going through all the pairs to decide which ones need to be selected and drastically reduces the time of manual curation. This saves the time of the user, but also avoids having a criterion that is purely subjective and could change over the time spent at doing manual curation.

Performance validation

We first validated our method with simulated ground truth datasets where artificial spikes were added in the recordings and retrieved by the algorithm. We ran the spike sorting on different datasets, picked some templates and used them to create new artificial templates, that we added at random places to the real recordings (see Methods and fig. 4A). We then run our sorting algorithm on this novel or "hybrid" datasets with these novel artificial cells randomly added, and measured if the algorithm was able to find at which times the artificial spikes were added. We counted a false negative error when such a spike was missed, and a false positive error when the algorithm detected a spike while there was not any (see Methods). Summing these two types of errors, the total error rate remained below 5% for all the spikes whose size was significantly above threshold. Fig. 4B shows the exhaustive results for 100 templates injected into a 252 electrodes recording in the retina, but the size or the type of the recording did not affect the error rates: they were similar for recordings with 4255 electrodes *in vitro* or 128 electrodes *in vivo*. Performance did not depend on the firing rate of the injected templates (fig. 4C), and

bioRxiv preprint doi: https://doi.org/10.1101/067843; this version posted August 4, 2016. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

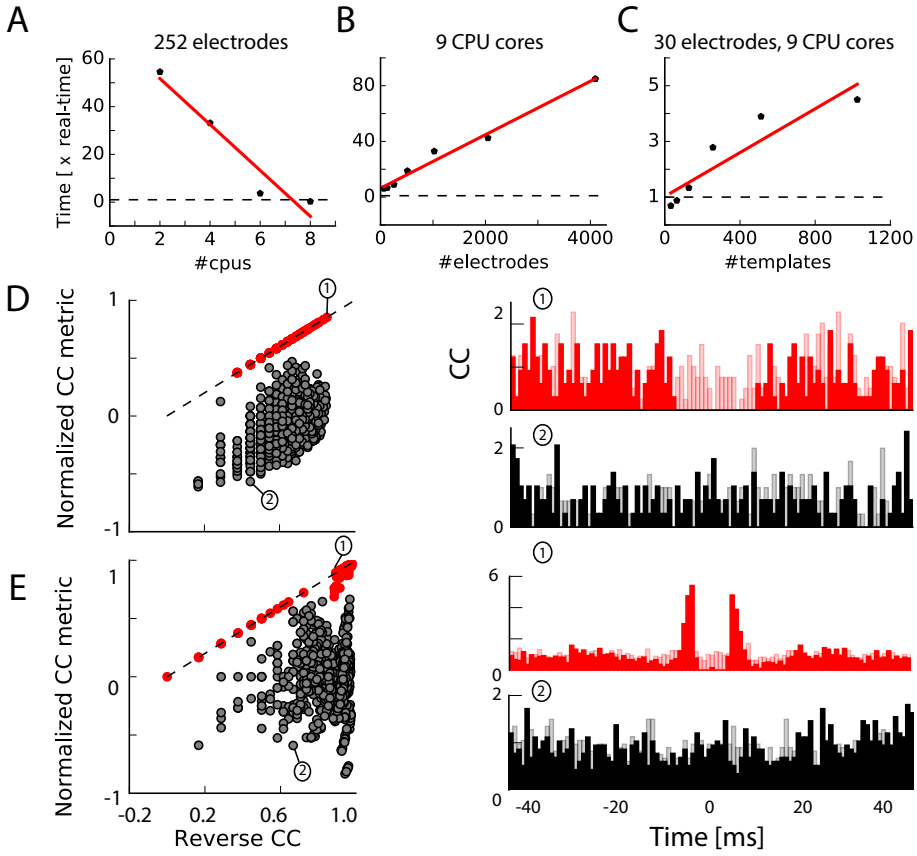

Figure 3: Management of large datasets **A.** Execution time as function of the number of processors, for a 90min dataset *in vitro* with 252 channels, expressed as a real-time ratio, i.e. the number of hours necessary to process one hour of data. **B.** Execution time as function of the number of channels, for a 30min dataset recorded *in vitro* with 4225 channels. **C.** Execution time as function of the number of templates, for a 10min synthetic dataset with 30 channels. **D.** Normalized Cross Correlation Metric compared to the Reverse Correlation for artificially generated and split spike trains (see Methods). Red: pairs of templates originating from the same neuron, that have to be merged. Black: pairs of templates corresponding to different cells. Insets on the right: for two chosen pairs (see numbers) the full cross-Correlogram (plain color) and the reverse correlogram (shaded color) **E.** Same as **D.**, but with 10 real neurons selected after sorting a dataset recorded with 252 channels on a rat retina (see Methods).

bioRxiv preprint doi: https://doi.org/10.1101/067843; this version posted August 4, 2016. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

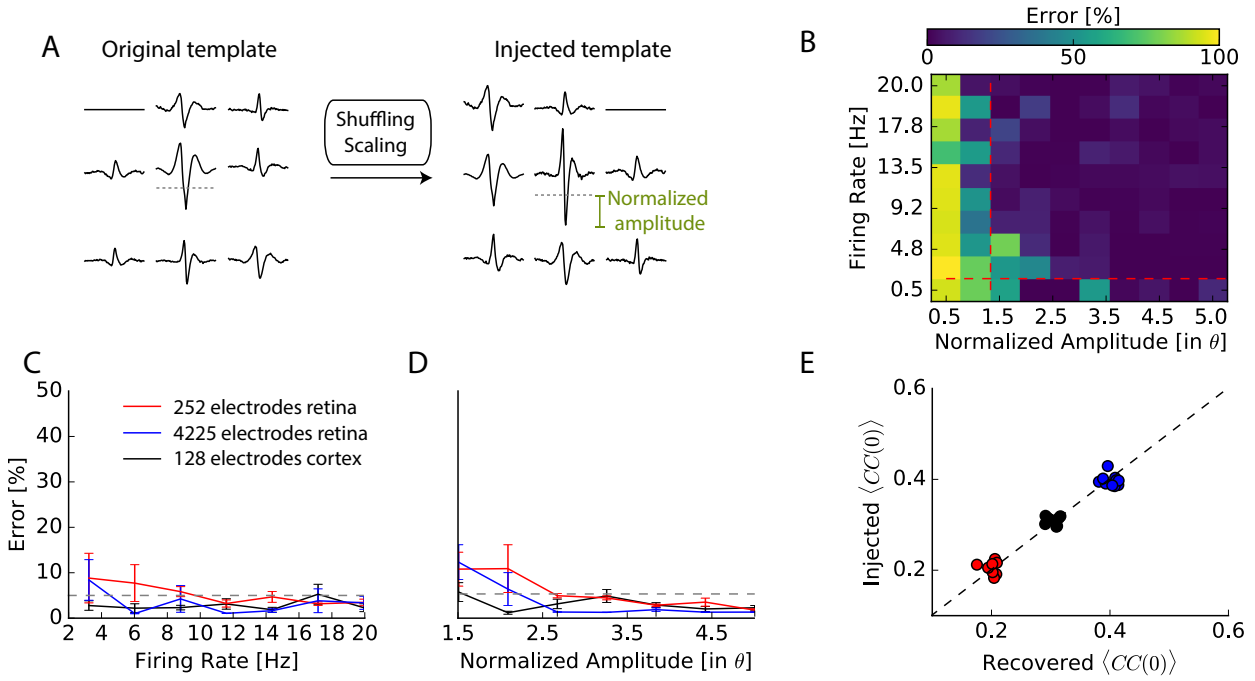

Figure 4: Performance of the algorithm on hybrid ground truth data **A.** A chosen template is injected elsewhere in the data as a new template. Dashed-dotted lines shows the detection threshold on the main electrode of the template, and normalized amplitude is expressed relatively to this threshold (see Methods). **B.** Error rates as function of the amplitude and the rate of injected templates, in a 252 electrode recording performed *in vitro* in the retina (see Methods). **C.** Mean error rate as function of the firing rate of injected templates, in various datasets. Errors bars show the standard error over 8 templates **D.** Error rate as function of the normalized amplitude of injected templates, in various datasets. Errors bars show standard error over 9 different templates **E.** Injected and recovered cross-correlation value between pairs of neurons for 5 templates injected at 10Hz, with a normalized amplitude of 2 (see Methods).

only weakly on the normalized amplitude of the templates (fig. 4D), as long as it was above the spike threshold. Performance was not only satisfying for individual cells, but also to estimate correlations properly. We injected templates with a controlled amount of overlapping spikes (see Methods). The algorithm was always able to estimate the pairwise correlation between the spike trains (fig. 4E). The ability of our template matching algorithm to resolve overlapping spikes thus allowed an unbiased estimation of correlations between spike trains.

Performance on ground truth data

The previous tests were done using "hybrid" data mixing real recordings and artificial templates. To test our algorithm in a more realistic situation, we performed dual recordings (fig. 5A, B). While we recorded many cells using a multi-electrode array (see schematic on fig. 5A), we simultaneously performed loose-patch recording of one of the cells, and recorded its spikes (fig. 5B). For one of the cells we know what should be the output of the spike sorting. In vitro, we recorded 26 neurons from 14 retinas with a 252 electrode MEA where the spacing between electrodes was 30 microns (see Methods). To test our algorithm in cases where the electrode

bioRxiv preprint doi: https://doi.org/10.1101/067843; this version posted August 4, 2016. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

density is lower, we generated datasets where we removed the signals of some electrodes, such that the density of the remaining electrodes was either 42 or 60 microns (by removing half or 3 quarters of the electrodes, respectively).

We then ran the spike sorting algorithm on the extracellular data, and estimated the error rates for the cell recorded in loose-patch, where we know where the spikes occurred. The performance of the algorithm will depend on the waveform triggered by the cell on the extracellular electrodes, as this is an intrinsic limit to the problem of spike sorting. If a spike of the patch-recorded cell triggers a large voltage deflection, this cell should be easy to detect. On the other hand, if the triggered voltage deflection was barely detectable, we expect that our algorithm should not perform well. To quantify this, we estimated a theoretical "best" performance for each recording. This best performance was found by training a non-linear classifier on the extracellular waveforms triggered by the spikes of the recorded cell, similar to [Harris et al., 2000, Rossant et al., 2016] (referred to as the Best Ellipsoidal Error Rate (BEER), see Methods). Note that this classifier "knows" where the true spikes are and simply quantifies how well they can be separated from the other spikes based on the extracellular recording. On the contrary, our spike sorting algorithm does not have access to any information about when the patch-recorded cell spiked.

We estimated the error made by the classifier and found that the performance of our algorithm almost always matched the performance of this classifier (fig. 5C, left), and this over a broad range of spike sizes. This suggests that our algorithm reached an almost optimal performance on this *in vitro* data. As expected, larger deflections lead to better performances (see inset of fig. 5C) for both the classifier and our algorithm, such that their respective performances were always close. Note also that because the BEER estimate is not designed to deal with overlapping spikes, it was possible for our template-matching based algorithm to obtain better performances in some cases.

We also used similar ground truth datasets recorded *in vivo* in rat cortex using dense silicon probes with either 32 or 128 recording sites [Neto et al., 2016]. With the same approach than for *in vitro* data, we also found that our algorithm reached a near optimal performance (fig. 5C, right). Together, these results show that our algorithm can reach an almost optimal performance (i.e. comparing to the BEER error) in various realistic cases, for both *in vivo* and *in vitro* recordings.

Discussion

We have shown that a method based on density-based clustering and template matching allows to sort spikes from large-scale extracellular recordings both *in vitro* and *in vivo*. Our algorithm is entirely parallelized and could therefore easily handle long datasets recorded with thousands of electrodes. The only step that required manual intervention was the final step of merging spike trains corresponding to a same cell, that has been split in several templates by the algorithm. We presented a novel approach to merge all the spike trains at once during manual curation. This new procedure minimizes the time spent on manual curation. It also guarantees that all pairs will be merged according to similar criteria. We tested the performance of our algorithm on "ground truth" datasets, where one neuron is recorded both with extracellular recordings and with a patch electrode. We showed that our performance was close to an optimal non-linear classifier that was trained using the true spike trains. Our algorithm has also been tested on purely synthetic datasets [Hagen et al., 2015], and similar results were obtained (data not shown).

Classical approaches to the spike sorting problem involve extracting some features from each detected spike [Hubel, 2015, Meister et al., 1994, Lewicki, 1994, Einevoll et al., 2012, Quiroga et al., 2004, Hill et al., 2011, Pouzat et al., 2002, Litke et al., 2004] and clustering the spikes in the feature space. In this approach, the spike sorting problem is reduced to a clustering

bioRxiv preprint doi: https://doi.org/10.1101/067843; this version posted August 4, 2016. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

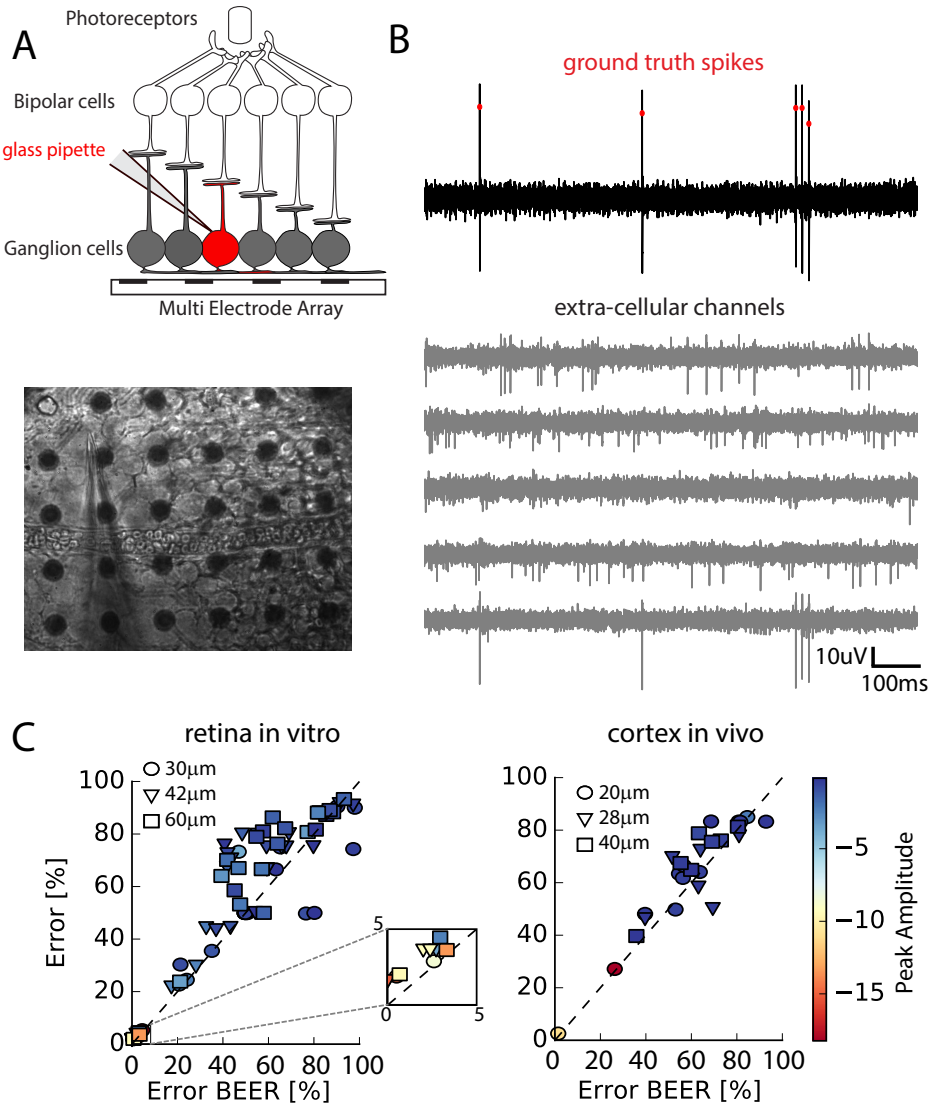

Figure 5: Performance of the algorithm on ground truth datasets **A.** Top: Schematic of the experimental protocol *in vitro*. A neuron close to the multi-electrode array (MEA) recording is recorded juxta-cellularly. Bottom: Image of the patch electrode on top of a 252 electrodes MEA, recording a ganglion cell. **B.** Top: Juxta-cellular recording showing the spikes of the recorded neuron. Bottom: Extra-cellular recordings next to the soma. **C.** Comparison between the error rates produced by the algorithm on different ground truth datasets and by the error rates of non-linear classifiers (Best Ellipsoidal Error Rate) trained to detect the spikes, either *in vitro* (Left), or *in vivo* (Right) (see Methods). Recorded neurons are color-coded as function of the maximal normalized amplitude recorded extra-cellularly.

bioRxiv preprint doi: https://doi.org/10.1101/067843; this version posted August 4, 2016. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

problem, and this induces several major issues. First, to assign the spikes to the correct cell, the different cells must be separated in the feature space. Finding the exact borders of each cell in the feature space is a hard problem, that cannot be easily automated. As a result, most methods require a lot of manual curation, and therefore work only for a small number of electrodes. Second, running a clustering algorithm on data with thousands of electrodes is very challenging. Finally, overlapping spikes will appear as strong deviations in the feature space, and will therefore be missed in this approach. These three issues preclude the use of this approach for large-scale recordings with dense arrays of electrodes. In comparison, here we have parallelized the clustering step efficiently, and we have used a template matching approach, so that we only needed to get the centroid of each cluster, and not their precise borders. The template matching approach also allowed to deconvolve overlapping spikes in a fast and efficient manner.

Few template matching approaches have been tested, but only on *in vitro* data [Marre et al., 2012, Pillow et al., 2013]. Also, they only had one component for template matching, and did not allow any variation in the shape of the spike. Moreover, to get the templates, some used a manual approach [Segev et al., 2004, Prentice et al., 2011] that cannot be scaled up, or a k-means algorithm that would also be challenged by large-scale data. In all cases, they required a significant amount of manual curation, with a time that scaled linearly with the number of electrodes. Finally, they were not tested on *in vivo* data. *in vivo* recordings seem to be prone to more variation in the spike waveform than *in vitro* recordings, and we have designed our template matching method to take into account not only variation in the amplitude of the template, but also in shape. So our approach improves previous template matching algorithms by decomposing each spike as a sum of two waveforms.

Our method is almost fully automated. This paves the way towards online spike sorting for large-scale recordings. Several applications, like brain machine interfaces, or closed-loop experiments [Franke et al., 2014, Hamilton et al., 2015, Benda et al., 2007], will require an accurate online spike sorting. Our method is entirely parallel and can therefore be run in "real time" (i.e. one hour of recording processed in one hour) with enough computer power. This will require adapting our method to process data "on the fly", processing new data blocks when they come, and probably adapting the shape of the template over time.

Methods

Experimental recordings

***in vitro* recordings with 252 or 4225 electrodes** Retinal tissue was obtained from adult (8 weeks old) male Long-Evans rat (Rattus norvegicus) or mouse (mus musculus, 4-9 weeks old) and continuously perfused with Ames Solution (Sigma-Aldrich) and maintained at 32°C. Ganglion cell spikes were recorded extracellularly from a multi-electrode array with 252 electrodes spaced 30 or 60μm apart (Multi-Channel Systems), or with 4225 channels arranged in a 2D grid and spaced by 16μm [Zeck et al., 2011, Bertotti et al., 2014], at a sampling rate of 20kHz. Experiments were performed in accordance with institutional animal care standards.

For the ground truth recordings, electrophysiological recordings were obtained from *ex-vivo* isolated retinae of rd1 mice (4/5 weeks old). The retinal tissue was placed in AMES medium (Sigma-Aldrich, St Louis, MO; A1420) bubbled with 95% O2 and 5% CO2 at room temperature, on a MEA (10μm electrodes spaced by 30μm; Multichannel Systemps, Reutlingen, Germany) with ganglion cells layer facing the electrodes. Borosilicate glass (BF100-50, Sutter instruments) electrodes were filled with AMES with a final impedance of 6-9 MΩ. Cells were imaged with a customized inverted DIC microscope (Olympus BX 71) mounted with a high sensitivity CCD Camera (Hamamatsu ORCA -03G) and recorded with an Axon Multiclamp 700B patch clamp amplifier set in current zero mode.

bioRxiv preprint doi: https://doi.org/10.1101/067843; this version posted August 4, 2016. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

For the data shown in fig. 2 and 4, we used a recording of 130min. For the data shown in fig. 5A, 16 neurones were recorded over 14 intact retinas. Recording durations ranged from 2min to 12min. The thresholds for the detection of juxta-cellular spikes were manually adjusted for all the recordings.

***in vivo* recordings with 128 electrodes** We use the freely available datasets provided by [Neto et al., 2016]. Those are 32 or 128 dense silicon probes recordings (20μm spacing) at 30kHz performed in rat visual cortex, combined with juxta-cellular recordings. The dataset gave us a total of 13 neurons for fig 5. C, with recordings between 4 and 10min each. Similarly to the *in vitro* case, the detection thresholds for the juxta-cellular spikes were manually adjusted based on the data provided by [Neto et al., 2016] and on spike-triggered waveforms. For the validation with "hybrid" dataset, shown in fig. 4, we used the longest dataset recorded with 128 electrodes.

Details of the algorithm

In the following, we consider that we have N_{elec} channels, acquired at a sampling rate f_{rate}. Every channel i is located at a physical position $\boldsymbol{p_i} = (x_i, y_i)$ in a 2D space (extension to 3D probes would be straightforward). The aims of our algorithm is to decompose the signal $\boldsymbol{s}$ over all channels $1, \dots N_{\text{elec}}$ as a linear sum of spatio-temporal kernels or "templates" (see equation 1).

$$\boldsymbol{s}(t) = \sum_{ij} a_{ij}\boldsymbol{T}_j(t - t_i) + b_{ij}\boldsymbol{U}_j(t - t_i)) + \boldsymbol{e}(t) \tag{1}$$

where $\boldsymbol{s}(t)$ is the signal recorded over N_{elec} electrodes and over multiple time points. $\boldsymbol{T}_j(t - t_i)$ and $\boldsymbol{U}_j(t - t_i)$ are the templates associated to each cell, which represents the waveform triggered on the electrodes by cell j. Times t_i are all the putative spike times over all the electrodes. a_{ij} and b_{ij} are the amplitude factors for spike time t_i for cluster j, and $\boldsymbol{e}(t)$ is the background noise.

The algorithm can be divided into two main steps, described below. After a preprocessing stage, we first run a clustering algorithm to extract a dictionary of "templates" from the recording. Second, we use these templates to decompose the signal with a template-matching algorithm. We assume that a spike will only influence the extracellular signal over a time window of size N_t (typically 2ms for *in vivo* and 5ms for *in vitro* data), and only electrodes whose distance to the soma is below r (typically, 100μm for *in vivo* and 200μm for *in vitro* data). For every channel i centered on $\boldsymbol{p_i}$, we define N^i_{neigh} as the ensemble of nearby channels such that $\|\boldsymbol{p_i} - \boldsymbol{p_j}\|_2 \leq r$. In the following, we note $\boldsymbol{s}_i[t_{\min}, t_{\max}]_i$ the temporal slice of all time samples on channel i such that $t \in [t_{\min}, t_{\max}]$.

Pre-processing

Filtering In a preprocessing stage, all the channels were individually high-pass filtered with a Butterworth filter of order three and a cutoff frequency of 500Hz, to remove any low-frequency components of the signals. We then substracted, for every channel i, the median such that $\forall i, \text{med}(\boldsymbol{s_i}) = 0$.

Spike detection Once signals have been filtered, we computed a spike threshold θ_i for every channel $\boldsymbol{s}_i(t)$: $\theta_i = k\text{MAD}(\boldsymbol{s}_i(t))$, where MAD is the Median Absolute Deviation, and k is a free parameter. For all the datasets shown in this paper, we set $k = 6$. For every channel $\boldsymbol{s}_i$ we detected the putative spike times t^i_k as all the local minima below θ_i.

bioRxiv preprint doi: https://doi.org/10.1101/067843; this version posted August 4, 2016. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

Whitening To remove spurious spatial correlations between nearby recordings channels, we performed a spatial whitening on the data. To do so, we searched for a maximum of 20s of recordings where there are no spikes (i.e no threshold crossings). We then computed the Covariance Matrix of the noise $\boldsymbol{C}_{\text{spatial}}$, and estimated its eigenvalues $\{d_i\}$ and associated eigenvectors $\{\boldsymbol{V}\}$. From the diagonal matrix $\boldsymbol{D} = \text{diag}(\frac{1}{\sqrt{d+\epsilon}})$, where $\epsilon = 10^{-18}$ is a regularization factor to ensure stability, we computed the whitening matrix $\boldsymbol{W}$ as $\boldsymbol{VDV^T}$. In the following, each time blocks of data are loaded, they are multiplied by $\boldsymbol{W}$. After whitening, we recomputed the spike detection threshold θ_i of each channel i in the whitened space.

Basis estimation (PCA) To identify the spatio-temporal waveforms embedded in the data, we need to reduce their dimensionality. We collected up to N_{w} spikes on each channel, at times t_k^i. In order to compensate for sampling rate artifacts, we upsampled all the collected waveforms by bicubic spline interpolation to 5 times the sampling rate f_{rate}, aligned on their local minima, and then re-sampled at f_{rate}. We performed a Principal Component Analysis (PCA) on these centered and aligned waveforms and kept only the first N_{PCA} principal components. In all the calculations, we used default values of $N_{\text{w}} = 10000$ and $N_{\text{PCA}} = 5$. These Principal components were used during the clustering step.

Clustering

The goal of the clustering step is to construct a dictionary of templates. As opposed to former clustering approaches of spike sorting [Quiroga et al., 2004, Harris et al., 2000, Kadir et al., 2014], because this clustering step is followed by a template matching, we do not need to perform the clustering on all the spikes: the templates are extracted from the centroids of the clusters.

Masking We first collected many spikes to perform the clustering. To minimize redundancy between collected spikes, each time a local minimum t_k^i was selected on a given channel i, we prevented the algorithm to select any other minima in a spatio-temporal area around t_k^i, to avoid the selection of multiple threshold crossings originating from the same event. We excluded all the peaks on the neighboring channels $j \in N_{\text{neigh}}^i$ during a time window centered on t_k^i: $[t_k^i - N_t/2, t_k^i + N_t/2]$.

Pooling of the spikes In order to parallelize the problem, we used a divide and conquer approach [Marre et al., 2012, Swindale and Spacek, 2014]. Each time a spike was detected at time t_k^i on electrode i, we searched for electrode ϕ where the voltage has the lowest value, i.e. such that $\phi = \text{argmin}_j \boldsymbol{s}_j\left[t_k^i - N_t/2, t_k^i + N_t/2\right]$. So for every electrode i we collected spikes peaking on this electrode. Each of these spikes is represented by a spatio-temporal waveform of size $N_t \times N_{\text{neigh}}^i$. We projected each waveform on the PCA basis estimated earlier to reduce the dimensionality to $N_{\text{PCA}} \times N_{\text{neigh}}^i$. Note that during this projection, the same up-sampling technique described in the Pre-processing was used. For every channel i we collected N_{spikes}^i spikes, and each of them is a vector of size $N_{\text{PCA}} \times N_{\text{neigh}}^i$. The maximal number of spikes collected is defined by the user as at N_{spikes}, and we used a default value of $N_{\text{spikes}} = 10000$. To reduce dimensionality even further before the clustering stage, for every channel i we performed a PCA on the collected spikes, and kept only the first N_{PCA_2} principal components (in all the paper, $N_{\text{PCA}_2} = 5$). Therefore, we performed a clustering in parallel for every channel, on at max N_{spikes} described in a space of N_{PCA_2}-dimension. Note that this pre-grouping does not assume that the spikes are only detected on a single electrode. This clustering performed on each spike ensemble used the information available on all the neighboring electrodes.

bioRxiv preprint doi: https://doi.org/10.1101/067843; this version posted August 4, 2016. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

Clustering by search of local density peaks To perform the clustering, we used a modified version of the algorithm published in [Rodriguez and Laio, 2014]. For N data points $\boldsymbol{x_i}$, $i \in \{1, \dots N\}$ in a M dimensional space $\mathbb{R}^M$ (in our case $\boldsymbol{x_i}$ are spikes assigned to channel i and $M = N_{\text{PCA}_2}$), we computed the "density" ρ as the mean distance to the S nearest neighbors of $\boldsymbol{x_i}$. S is chosen such that $S = \epsilon N$, with $\epsilon = 0.01$. This density measure turned out to be more robust than the one given in the original paper, and rather insensitive to changes in ϵ. Then, for every point $\boldsymbol{x_i}$, we computed δ_i as the minimal distance to any other point $\boldsymbol{x_{j \neq i}}$ such that $\rho_j < \rho_i$. The intuition of the algorithm is that the centroids should be points with a high density (i.e. low ρ), and far apart from each others (high δ).

Accurate estimation of the density landscape We performed the clustering only on a subset of N^i_{spikes} for each channel. To avoid potential inaccurate estimation of the densities ρ_i, we collected iteratively additional spikes to refine this estimate. Keeping in memory the spikes $\boldsymbol{x}_i$, we searched again in the data N^i_{spikes} different spikes and used them to refine the estimation of ρ_i of our selected points $\boldsymbol{x}_i$. When doing m passes, the complexity only scales as $\mathcal{O}(mN^{i\ 2}_{\text{spikes}})$. In all the following we used $m = 3$, and automatically stop updates if not enough spikes can be found in the dataset.

Centroids and cluster definition To define the centroids we ranked the points as function of the products $\rho\delta$, we detected the best $N^{\max}_{\text{clusters}}$ points as putative centroids. Clusters were formed by an iterative rule, going from the points of lowest rho to the points of highest ρ: each point was assigned to the same cluster than the closest point with a lower ρ [Rodriguez and Laio, 2014]. We created at max $N^{\max}_{\text{clusters}}$ clusters. Once this is done, we iteratively merged pairs of clusters that were too similar to each others.

Merging similar clusters We computed a normalized distance ζ between each pair of clusters y_1 and y_2. The center $\boldsymbol{\alpha_i}$ of cluster Y_i was defined as $\boldsymbol{\alpha_i} = \text{med}(\text{x}_\text{i} \in \text{Y}_\text{i})$, and we can therefore project all the points from clusters Y_1 and Y_2 onto the axis joining the two centroids $\boldsymbol{\gamma_{1,2}} = \boldsymbol{\alpha_1} - \boldsymbol{\alpha_2}$. We defined the dispersions around the centroids $\boldsymbol{\alpha_1}$ and $\boldsymbol{\alpha_2}$ as $\beta_{1/2} = \text{MAD}((\boldsymbol{x}_{i \in Y_{1/2}}.\boldsymbol{\gamma_{1,2}})^2)$, where . is scalar product of the two vectors, and the normalized distance is:

$$\zeta(Y_1, Y_2) = \frac{\|\text{med}(\boldsymbol{x}_{i \in Y_1}.\boldsymbol{\gamma_{1,2}}) - \text{med}(\boldsymbol{x}_{i \in Y_2}.\boldsymbol{\gamma_{1,2}})\|}{\sqrt{\beta_1^2 + \beta_2^2}} \tag{2}$$

We then iteratively merged all clusters (Y_i, Y_j) such that $\zeta(Y_i, Y_j) \leq \sigma_{\text{similar}}$. At the end of the clustering, every cluster with less than ηN^i_{spikes} was discarded. In all the manuscript, we used $\sigma_{\text{similar}} = 3$, $N^{\max}_{\text{clusters}} = 10$, and $\eta = 0.005$.

Template estimation At the end of the clustering phase, pooling the clusters obtained from every electrode, we obtained for every cluster Y_k a list of spike times $\{t^i_k\}_i$. We computed the first component from the raw data as the point-wise median of all the waveforms belonging to the cluster: $\boldsymbol{T_k}(\tau) = \text{med}_i(\boldsymbol{s}(t^i_k + \tau))$.Note that $\boldsymbol{T_k}$ is only different from zero on the electrodes close to its peak (see fig. 1D). This information is used internally by the algorithm to save templates as sparse structures. Moreover, during template estimation, we limited the number of spike times per template to a maximal value of 500 to avoid memory saturation. To enhance the compression level of the template $\boldsymbol{T_k}$, we set to 0 all the channels j where $|\boldsymbol{T_k}(t)| < \theta_j$, if θ_j is the detection threshold on channel j. This allowed us to remove channels without discriminant information, and to increase the sparsity of the templates. Once the template first component $\boldsymbol{T_k}$ had been extracted, we computed its minimal and maximal amplitudes $a^{\min/\max}_k$ based on data $\boldsymbol{x}_{i \in Y_k}$ used during the clustering. If $\hat{\boldsymbol{T}}_\boldsymbol{k}$ is the normalized template, such that $\hat{\boldsymbol{T}}_\boldsymbol{k} = \boldsymbol{T_k}/\|\boldsymbol{T_k}\|$,

bioRxiv preprint doi: https://doi.org/10.1101/067843; this version posted August 4, 2016. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

we computed

$$a_k^{\min} = \text{med}(\boldsymbol{x}_{j\in y_k}.\hat{\boldsymbol{T}}_{\boldsymbol{k}}) - 5\text{MAD}(\boldsymbol{x}_{j\in y_k}.\hat{\boldsymbol{T}}_{\boldsymbol{k}})$$
$$a_k^{\max} = \text{med}(\boldsymbol{x}_{j\in y_k}.\hat{\boldsymbol{T}}_{\boldsymbol{k}}) + 5\text{MAD}(\boldsymbol{x}_{j\in y_k}.\hat{\boldsymbol{T}}_{\boldsymbol{k}})$$

Those boundaries are used during the template matching phase (see below). Finally, we computed the projection of all snippets in the space orthogonal to the first component: $\forall i \in y_k, x'_i = \boldsymbol{x}_i - \alpha_i \boldsymbol{T_k}$, with $\alpha_i = \frac{\boldsymbol{x}_i.\boldsymbol{T_k}}{\|\boldsymbol{T_k}\|_2}$. The x' are the projections, and from that we computed the second component of the template, $\boldsymbol{U_k}(t)$ as the direction of largest variance that is orthogonal to the average waveform $\boldsymbol{T_k}(t)$ (i.e. the first principal component).

Removing redundant templates To remove redundant templates that may be present in the dictionary because of the divide and conquer approach (for example a neuron in between two electrodes would give rise to two very similar templates on two electrodes), we computed for all pairs of templates in the dictionary $CC_{\max}(i,j) = \max_t CC(\boldsymbol{T_i}, \boldsymbol{T_j})$, where CC stands for normalized Cross-Correlation. If $CC_{\max}(i,j) \geq cc_{\text{similar}}$, we considered those templates to be equivalent and they were merged. In all the following, we used $cc_{\text{similar}} = 0.975$. Note that we are computing the Cross-Correlations between normalized templates, such that two templates that have the same shape but different amplitudes are merged. Similarly, we searched if any template $\boldsymbol{T_k}$ could be explained as a linear combination of two templates in the dictionary. If we could find $\boldsymbol{T_i}$ and $\boldsymbol{T_j}$ such that $CC(\boldsymbol{T_k}, \boldsymbol{T_i} + \boldsymbol{T_j}) \geq cc_{\text{similar}}$, $\boldsymbol{T_k}$ was considered to be a mixture of two cells, and was removed from the dictionary.

Template matching

At the end of this "template-finding" phase, we have found a dictionary of templates $(\boldsymbol{T}, \boldsymbol{U})$. We now need to reconstruct the signal $\boldsymbol{s}$ by finding the amplitudes coefficients a_{ij} and b_{ij} described in Equation 1. Note that most a_{ij} and b_{ij} in this equation are equal to 0. For the other ones, most a_{ij} values are around 1, because a spike usually appears on electrodes with an amplitude close to the average first component T. In this template matching step, all the other parameters have been determined by template extraction and spike detection, so the purpose is only to find the values of these amplitudes. To do so, we used an iterative greedy approach to estimate the a_{ij} for each subgroup t_i, which bears some similarity to the matching pursuit algorithm [Mallat and Zhang, 1993]. The fitting was performed in blocks of putative spike times, $\{t_i\}$, that were successively loaded in memory. Such an approach allowed us to easily split the load linearly among several processors. Each block of raw data $\boldsymbol{s}$ was loaded and processed according to the following steps during the template-matching phase:

1. Estimate the normalized scalar products $\boldsymbol{s}(t) \cdot \boldsymbol{T}_j(t - t_i)$ for each template j and putative spike time t_i, for all the i and j in the block of raw data.
2. Choose the (i,j) pair with the highest scalar product, excluding the pairs (i,j) which have already been tried and the t_i's already explored (see below).
3. Set a_{ij} equal to the amplitude value that best fits the raw data: $a_{ij} = \frac{\boldsymbol{s}(t).\boldsymbol{T}_j(t-t_i)}{\|\boldsymbol{T}_j(t-t_i)\|}$.
4. Check if the a_{ij} amplitude value is between $a_j^{\min}$ and $a_j^{\max}$.
5. If so, then accept this value, subtract the scaled template from the raw data: $\boldsymbol{s}(t) \rightarrow \boldsymbol{s}(t) - a_{ij}\boldsymbol{T}_j(t - t_i)$. Then set b_{ij} equal to the amplitude value that best fits the raw data with $\boldsymbol{U_j}$, and subtract it too.
6. Return to step 1 to re-estimate the scalar products on the residual.
7. Otherwise, increase by one n_i, which counts the number of times any template has been rejected for spike time t_i. If n_i reaches $n_{\text{failures}} = 3$, label this t_i as "explored". If all t_i have been explored, quit the loop. Otherwise return to step 1 and iterate.

bioRxiv preprint doi: https://doi.org/10.1101/067843; this version posted August 4, 2016. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

The parameters of the algorithm were the amplitude thresholds $a_j^{\min}$ and $a_j^{\max}$, computed as described in the Template Estimation section.

Semi-automated merging

To speed up the manual stage where a human operator has to review all the templates (and thus putative neurons) and decide which one are duplicated or truncated, we developed a dedicated Graphical User Interface (GUI). Such a GUI is especially useful when the number of detected neurons is important, which is the case with the new generation of dense probes. Templates likely to be merged are templates that look alike, and such that the combined cross-correlogram between the two cell's spike trains show a clear dip near 0ms, indicating that the merged spike trains do not show any refractory period violation (fig 3D, E). To speed up considerably the time spent in processing those datasets, and reducing the inherent variabilities of a human interaction on the sorting process, we provided, for all pairs of templates, a mathematical quantification of the similarity between templates, and of the dip in the cross-correlogram. In all the following $CC_{i,j}(t)$ represents the Cross Correlogram between the spike times of $\boldsymbol{T_i}$ and those of $\boldsymbol{T_j}$. For the template similarity, we computed, for every pair of templates $(\boldsymbol{T_i}, \boldsymbol{T_j})$, $\alpha(\boldsymbol{T_i}, \boldsymbol{T_j}) = \max_t CC_{i,j}(t)$. To quantify the dip in the cross-correlogram, we computed the cross-correlogram $CC_{i,j}(t)$ between the spikes of $\boldsymbol{T_i}$ and those of $\boldsymbol{T_j}$ and a control version $\widehat{CC}_{i,j}(t)$ obtained by keeping the spike times from $\boldsymbol{T_i}$ but reversing in time those from $\boldsymbol{T_j}$. We then computed the average value over time of $\langle CC_{i,j}(t)\rangle$ and $\langle \widehat{CC}_{i,j}(t)\rangle$ for $t \leq \tau_0$. In the following we used $\tau_0 = 2$ms. This value will depend on the dataset and can be chosen by the user in our Graphical User Interface. From that, we computed the normalized value:

$$\beta(\boldsymbol{T_i}, \boldsymbol{T_j}) = \frac{\langle \widehat{CC}_{i,j}(t)\rangle - \langle CC_{i,j}(t)\rangle}{\langle \widehat{CC}_{i,j}(t)\rangle + \langle CC_{i,j}(t)\rangle} \tag{3}$$

The dip measure $\beta(\boldsymbol{T_i}, \boldsymbol{T_j})$ is termed normalized CC metric in fig. 3D,E. It is compared to $\langle \hat{CC}_{i,j}(t)\rangle$, termed Reverse CC in the same figure. For a better visualization, in the insets of fig. 3D,E, the cross-correlograms are such that the mean over time is equal to 1.

For two templates, $\alpha(\boldsymbol{T_i}, \boldsymbol{T_j})$ allowed us to quantify how similar they are, while $\beta(\boldsymbol{T_i}, \boldsymbol{T_j})$ gave us an insight about how strong is the dip in the cross-correlogram between their spike times. To guide the human into the process of merging pairs, we then showed in 2D plots all pairs of templates as function of $\alpha(\boldsymbol{T_i}, \boldsymbol{T_j})$ and $\beta(\boldsymbol{T_i}, \boldsymbol{T_j})$. In such a space, the user can quickly define at once a whole set of pairs that need to be merged. After merging, quantities α and β are re-computed, and the workflow can keep going until the user decides to stop merging (see fig. 3).

Simulated ground truth tests

Injection of artificial templates In order to assess the performance of the algorithm, we injected new templates in real datasets (see fig. 4 A-D). To do so, we ran the algorithm on a given dataset, and obtain a list of putative templates $\boldsymbol{T}_{i \in \{1,\dots N\}}$. Then, we randomly selected some of those templates $\boldsymbol{T_i}$ and shuffled the list of their channels, before injecting them elsewhere in the datasets at controlled firing rates [Harris et al., 2000, Rossant et al., 2016, Kadir et al., 2014]. This allowed us to properly quantify the performance of the algorithm. In order not to bias the clustering, when a template $\boldsymbol{T_i}$ was selected and shuffled as a new template $\overline{\boldsymbol{T}}_{\boldsymbol{j}}$ centered on a new channel j, we ensured that the injected template was not too similar to one that would already be in the data: $\forall i \in \{1, \dots N\}, \max_t CC(\boldsymbol{T_i}, \overline{\boldsymbol{T_j}}) \leq 0.8$. Before being injected, $\overline{\boldsymbol{T}}_{\boldsymbol{j}}$ was normalized it such that $\min_t \overline{\boldsymbol{T}}_{\boldsymbol{j}} = \alpha_j \theta_j$. Therefore, the normalized amplitude α_j of $\overline{\boldsymbol{T}}_{\boldsymbol{j}}$, was the relative amplitude, expressed as function of θ_j, the detection threshold on the electrode where

bioRxiv preprint doi: https://doi.org/10.1101/067843; this version posted August 4, 2016. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

the template is peaking. If $\alpha_j \leq 1$, it means that the template has the same height than the noise level, and its spikes should not be detected; if $\alpha \geq 1$ the spikes should be detected.

Injection of correlated templates In fig. 4E, we selected a particular template $\boldsymbol{T}$, and kept constant the part on the electrode i where it was peaking the most, while shuffling all its neighboring channels. By doing so, we obtained 5 distinct copies of the template $\overline{\boldsymbol{T}}_{\mathbf{1...5}}$, representing 5 neurons whose soma would be located next to the same channel i. The normalized amplitudes of those templates were equal to 2. To demonstrate how the code could resolve the problem of overlapping spikes, we injected them into the data such that they were all firing at 10Hz, but with a controlled correlation coefficient c that could be varied (using a Multiple Interaction Process [Kuhn et al., 2003]). This parameter c allowed us to quantify the percentage of pairwise correlations recovered by the algorithm for overlapping spatio-temporal templates.

Generation of artificial spike trains To illustrate the meta-merging GUI, we generated artificial spike trains where we can have a ground truth for which pairs should be merged. We generated 10 spikes trains ω_i at 10Hz with a correlation coefficient $c = 0.01$ (using a Multiple Interaction Process [Kuhn et al., 2003]), and a refractory period of $\tau_{\text{ref}} = 5$ms. Then, for every spike trains ω_i, we split it in N_k^i sub spike trains, with N_k^i uniformly drawn between [1, 5]. This gave us a number of cells between 10 and 50 (see fig. 3D). The same procedure is repeated in fig. 3E, but starting from 10 neurons obtained after a real sorting session from the rat retina.

Estimation of false positive and false negative To quantify the performance of the algorithm, we matched the spikes recovered by the algorithm to the real ground-truth spikes (either synthetic or obtained with juxta-cellular recordings). A spike was considered to be a match if it had a corresponding spike in the ground truth at less than 2ms. Spikes in the ground-truth datasets that had no matches in the spike sorting results in a 2 ms window were labeled as "False Negative", while those that are not present while the algorithm detected a spike were "False Positive". False negative rate was defined as the number of false negative divided by the number of spikes in the ground truth recording. False positive rate was defined as the number of false positive divided by the number of spikes in the spike train extracted by the algorithm. In the paper, the error is defined as mean of the False negative and the False positive rates (see fig. 4, 5). Note that to take into account the fact that a ground-truth neuron could be split in several templates at the end of the algorithm, we always compared the ground-truth cells with the combination of templates that minimized the error.

Theoretical estimate To quantify the performance of the software with real ground-truth recordings (see fig. 5), we computed the Best Ellipsoidal Error Rate (BEER), as described in [Harris et al., 2000]. This BEER estimate gave an upper bound on the performance of any clustering-based spike sorting method using elliptical cluster boundaries. After thresholding and feature extraction, snippets were labeled according to whether or not they contained a true spike. Half of this labeled data set was then used to train a perceptron whose decision rule is a linear combination of all pairwise products of the features of each snippet. If x_i is the i-th snippet, projected in the feature space, then the optimized function $f(x)$ is:

$$f(x) = x^T A x + b^T x + c \tag{4}$$

We trained this function f by varying A, b and c with the objective that $f(x)$ should be 1 for the ground truth spikes, and -1 otherwise. These parameters were optimized by a stochastic gradient descent with a regularization constraint. The resulting classifier was then used to predict the occurrence of spikes in the snippets in the remaining half of the labeled data. Only the snippets where $f(x) > 0$ were predicted as true spikes. This prediction provided an estimate

bioRxiv preprint doi: https://doi.org/10.1101/067843; this version posted August 4, 2016. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

of the false negative and false positive rates for the BEER estimate. The mean between the two was considered to be the BEER error rate.

Decimation of the electrodes In order to increase the number of data points for the comparison between our sorting algorithm and the non-linear classifiers defined by the BEER metric (see fig. 5), we ran the analysis several times on the same neurons, but removing some electrodes, to create recordings at a lower electrode density. We divided by a factor 2 or 4 the number of electrodes in the 252 *in vitro* Multi-Electrode Array or the 128 *in vivo* silicon probe.

Implementation and Source Code

The code is a pure python package, based on the python wrapper for the Message Passing Interface (MPI) library [Dalcin et al., 2011] to allow parallelization over distributed computers, and is available with its full documentation at http://spyking-circus.rtfd.org. Results can easily be exported to the kwik or phy format [Rossant and Harris, 2013]. All the datasets used in this manuscript will also be available on-line, for testing and comparison with other algorithms.

Acknowledgments

We would like to thank Steve Baccus and Sami El Boustani for insightful discussions. We also would like to thanks Kenneth Harris, Cyrille Rossant and Nick Steimetz for feedbacks and the help with the interface to the phy software.

Funding This work was supported by ANR-14-CE13-0003 to P.Y., ANR TRAJECTORY, ANR OPTIMA, the French State program Investissements d'Avenir managed by the Agence Nationale de la Recherche [LIFESENSES: ANR-10-LABX-65], a EC grant from the Human Brain Project (FP7-604102)), and NIH grant U01NS090501 to OM, ERC Starting Grant (309776) to JD.

References

J. Benda, T. Gollisch, C. K. Machens, and A. V. Herz. From response to stimulus: adaptive sampling in sensory physiology. *Curr. Opin. Neurobiol.*, 17(4):430–436, Aug 2007.

Luca Berdondini, PD Van Der Wal, Olivier Guenat, Nicolaas F de Rooij, Milena Koudelka-Hep, P Seitz, R Kaufmann, P Metzler, N Blanc, and S Rohr. High-density electrode array for imaging in vitro electrophysiological activity. *Biosensors and bioelectronics*, 21(1):167–174, 2005.

G. Bertotti, D. Velychko, N. Dodel, St. Keil, D. Wolansky, B. Tillak, M. Schreiter, A. Grall, P. Jesinger, R. Rohler, M. Eickenscheidt, A. Stett, A. Moller, K.H. Boven, G. Zeck, and R. Thewes. A cmos-based sensor array for in-vitro neural tissue interfacing with 4225 recording sites and 1024 stimulation sites. *Biomedical Circuits and Systems Conference (BioCAS), 2014 IEEE*, pages 304–307, 22-24 Oct. 2014.

G. Buzsaki. Neural syntax: cell assemblies, synapsembles, and readers. *Neuron*, 68(3):362–385, Nov 2010.

bioRxiv preprint doi: https://doi.org/10.1101/067843; this version posted August 4, 2016. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

Lisandro D. Dalcin, Rodrigo R. Paz, Pablo A. Kler, and Alejandro Cosimo. Parallel distributed computing using Python. *Advances in Water Resources*, 34(9):1124–1139, September 2011. ISSN 03091708. doi: 10.1016/j.advwatres.2011.04.013. URL http://dx.doi.org/10.1016/j.advwatres.2011.04.013.

Gaute T Einevoll, Felix Franke, Espen Hagen, Christophe Pouzat, and Kenneth D Harris. Towards reliable spike-train recordings from thousands of neurons with multielectrodes. *Current opinion in neurobiology*, 22(1):11–17, 2012.

Michele Fiscella, Karl Farrow, Ian L Jones, David Jäckel, Jan Müller, Urs Frey, Douglas J Bakkum, Péter Hantz, Botond Roska, and Andreas Hierlemann. Recording from defined populations of retinal ganglion cells using a high-density cmos-integrated microelectrode array with real-time switchable electrode selection. *Journal of neuroscience methods*, 211(1):103–113, 2012.

Felix Franke, David Jäckel, Jelena Dragas, Jan Müller, Milos Radivojevic, Douglas Bakkum, and Andreas Hierlemann. High-density microelectrode array recordings and real-time spike sorting for closed-loop experiments: an emerging technology to study neural plasticity. *Closing the Loop Around Neural Systems*, page 31, 2014.

Espen Hagen, Torbjørn V. Ness, Amir Khosrowshahi, Christina Sørensen, Marianne Fyhn, Torkel Hafting, Felix Franke, and Gaute T. Einevoll. ViSAPy: a Python tool for biophysics-based generation of virtual spiking activity for evaluation of spike-sorting algorithms. *Journal of neuroscience methods*, 245:182–204, April 2015. ISSN 1872-678X. URL http://view.ncbi.nlm.nih.gov/pubmed/25662445.

Lei Hamilton, Marc McConley, Kai Angermueller, David Goldberg, Massimiliano Corba, Louis Kim, James Moran, Philip D Parks, Sang Chin, Alik S Widge, et al. Neural signal processing and closed-loop control algorithm design for an implanted neural recording and stimulation system. In *Engineering in Medicine and Biology Society (EMBC), 2015 37th Annual International Conference of the IEEE*, pages 7831–7836. IEEE, 2015.

Kenneth D Harris, Darrell A Henze, Jozsef Csicsvari, Hajime Hirase, and György Buzsáki. Accuracy of tetrode spike separation as determined by simultaneous intracellular and extracellular measurements. *Journal of neurophysiology*, 84(1):401–414, 2000.

Gerrit Hilgen, Martino Sorbaro, Sahar Pirmoradian, Jens-Oliver Muthmann, Ibolya Kepiro, Simona Ullo, Cesar Juarez Ramirez, Alessandro Maccione, Luca Berdondini, Vittorio Murino, Diego Sona, Francesca Cella Zanacchi, Upinder Bhalla, Evelyne Sernagor, and Matthias H Hennig. Unsupervised spike sorting for large scale, high density multielectrode arrays. *bioRxiv*, 2016. doi: 10.1101/048645. URL http://biorxiv.org/content/early/2016/04/13/048645.

Daniel N Hill, Samar B Mehta, and David Kleinfeld. Quality metrics to accompany spike sorting of extracellular signals. *The Journal of Neuroscience*, 31(24):8699–8705, 2011.

David Hubel. Tungsten microelectrode for recording from single units. *Science*, 125(3247):549-50, 2015.

Shabnam N Kadir, Dan FM Goodman, and Kenneth D Harris. High-dimensional cluster analysis with the masked em algorithm. *Neural computation*, 2014.

Alexandre Kuhn, Ad Aertsen, and Stefan Rotter. Higher-Order Statistics of Input Ensembles and the Response of Simple Model Neurons. *Neural Computation*, 15(1): 67–101, January 2003. ISSN 0899-7667. doi: 10.1162/089976603321043702. URL http://dx.doi.org/10.1162/089976603321043702.

bioRxiv preprint doi: https://doi.org/10.1101/067843; this version posted August 4, 2016. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

A. Lambacher, M. Jenkner, M. Merz, B. Eversmann, R.A. Kaul, F. Hofmann, R. Thewes, and P. Fromherz. Electrical imaging of neuronal activity by multi-transistor-array (mta) recording at 7.8Å μm resolution. *Applied Physics A*, 79(7):1607–1611, 2004. ISSN 1432-0630. doi: 10.1007/s00339-004-2991-5. URL `http://dx.doi.org/10.1007/s00339-004-2991-5`.

C. Leibig, T. Wachtler, and G. Zeck. Unsupervised neural spike sorting for high-density microelectrode arrays with convolutive independent component analysis. *J. Neurosci. Methods*, 271:1–13, Jun 2016.

Michael S Lewicki. Bayesian modeling and classification of neural signals. *Neural computation*, 6(5):1005–1030, 1994.

AM Litke, Bezayiff N, Chichilnisky EJ, Cunningham W, Dabrowski W, Grillo AA, Grivich M, Grybos P, Hottowy P, Kachiguine S, Kalmar RS, Mathieson K, Petrusca D, Rahman M, and Sher A. What does the eye tell the brain? development of a system for the large scale recording of retinal output activity. *IEEE Transactions on Nuclear Science*, 51(4):1434–1440, 2004.

S. G. Mallat and Zhifeng Zhang. Matching pursuits with time-frequency dictionaries. *Signal Processing, IEEE Transactions on*, 41(12):3397–3415, December 1993. ISSN 1053-587X. doi: 10.1109/78.258082. URL `http://dx.doi.org/10.1109/78.258082`.

Olivier Marre, Dario Amodei, Nikhil Deshmukh, Kolia Sadeghi, Frederick Soo, Timothy E Holy, and Michael J Berry. Mapping a complete neural population in the retina. *The Journal of Neuroscience*, 32(43):14859–14873, 2012.

Markus Meister, Jerome Pine, and Denis A Baylor. Multi-neuronal signals from the retina: acquisition and analysis. *Journal of neuroscience methods*, 51(1):95–106, 1994.

Joana P. Neto, GonÃ§alo Lopes, Joao Frazao, Joana Nogueira, Pedro Lacerda, Pedro Baiao, Arno Aarts, Alexandru Andrei, Silke Musa, Elvira Fortunato, Pedro Barquinha, and Adam R. Kampff. Validating silicon polytrodes with paired juxtacellular recordings: method and dataset. *Biorxiv*, 2016.

Marius Pachitariu, Nicholas Steinmetz, Shabnam Kadir, Matteo Carandini, and Kenneth D Harris. Kilosort: realtime spike-sorting for extracellular electrophysiology with hundreds of channels. *bioRxiv*, 2016. doi: 10.1101/061481. URL `http://biorxiv.org/content/early/2016/06/30/061481`.

Jonathan W Pillow, Jonathon Shlens, EJ Chichilnisky, and Eero P Simoncelli. A model-based spike sorting algorithm for removing correlation artifacts in multi-neuron recordings. *PloS one*, 8(5):e62123, 2013.

C. Pouzat, O. Mazor, and G. Laurent. Using noise signature to optimize spike-sorting and to assess neuronal classification quality. *J Neurosci Methods*, 122(1):43–57, 2002.

Jason S Prentice, Jan Homann, Kristina D Simmons, Gašper Tkačik, Vijay Balasubramanian, and Philip C Nelson. Fast, scalable, bayesian spike identification for multi-electrode arrays. *PloS one*, 6(7):e19884, 2011.

R Quian Quiroga, Zoltan Nadasdy, and Yoram Ben-Shaul. Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering. *Neural computation*, 16(8):1661–1687, 2004.

bioRxiv preprint doi: https://doi.org/10.1101/067843; this version posted August 4, 2016. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

A. Rodriguez and A. Laio. Clustering by fast search and find of density peaks. *Science*, 344(6191):1492–1496, June 2014. ISSN 0036-8075. doi: 10.1126/science.1242072. URL `http://dx.doi.org/10.1126/science.1242072`.

Cyrille Rossant and Kenneth D Harris. Hardware-accelerated interactive data visualization for neuroscience in python. *Frontiers in neuroinformatics*, 7, 2013.

Cyrille Rossant, Shabnam N Kadir, Dan FM Goodman, John Schulman, Mariano Belluscio, Gyorgy Buzsaki, and Kenneth D Harris. Spike sorting for large, dense electrode arrays. *Nature Neuroscience*, pages 19, 634–641, 2016.

Ronen Segev, Joe Goodhouse, Jason Puchalla, and Michael J Berry. Recording spikes from a large fraction of the ganglion cells in a retinal patch. *Nature neuroscience*, 7(10):1154–61, Oct 2004.

Nicholas V. Swindale and Martin A. Spacek. Spike sorting for polytrodes: a divide and conquer approach. *Frontiers in systems neuroscience*, 8, 2014. ISSN 1662-5137. doi: 10.3389/fnsys.2014.00006. URL `http://dx.doi.org/10.3389/fnsys.2014.00006`.

G. Zeck, A. Lambacher, and P. Fromherz. Axonal transmission in the retina introduces a small dispersion of relative timing in the ganglion cell population response. *PLoS ONE*, 6(6):e20810, 2011.