A spike sorting toolbox for up to thousands of electrodes validated with ground truth recordings *in vitro* and *in vivo*

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]

March 14, 2018

[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

In recent years, multielectrode arrays and large silicon probes have been developed to record simultaneously between hundreds and thousands of electrodes packed with a high density. However, they require novel methods to extract the spiking activity of large ensembles of neurons. Here we developed a new toolbox to sort spikes from these large-scale extracellular data. To validate our method, we performed simultaneous extracellular and loose patch recordings in rodents to obtain "ground truth" data, where the solution to this sorting problem is known for one cell. The performance of our algorithm was always close to the best expected performance, over a broad range of signal to noise ratios, *in vitro* and *in vivo*. The algorithm is entirely parallelized and has been successfully tested on recordings with up to 4225 electrodes. Our toolbox thus offers a generic solution to sort accurately spikes for up to thousands of electrodes.

Introduction

As local circuits represent information using large populations of neurons throughout the brain (Buzsaki, 2010), technologies to record hundreds or thousands of them are therefore essential. One of the most powerful and widespread techniques for neuronal population recording is extracellular electrophysiology. Recently, newly developed microelectrode arrays (MEA) have allowed recording of local voltage from hundreds to thousands of extracellular sites separated only by tens of microns (Berdondini et al., 2005; Fiscella et al., 2012; Lambacher et al., 2004), giving indirect access to large neural ensembles with a high spatial resolution. Thanks to this resolution, the spikes from a single neuron will be detected on several electrodes and produce extracellular waveforms with a characteristic spatio-temporal profile across the recording sites. However, this high resolution comes at a cost: each electrode receives the activity from many neurons. To access the spiking activity of individual neurons, one needs to separate the waveform produced by each neuron and identify 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 (typically, a few tens of electrodes). However, for large-scale and dense recordings, it is still unclear how to extract the spike contributions from extracellular recordings. In particular, for thousands of electrodes, this problem is still largely unresolved.

Classical 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. Second, 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).

A few algorithms have been designed to process large-scale recordings (Marre et al., 2012; Pillow et al., 2013; Pachitariu et al., 2016; Leibig et al., 2016; Hilgen et al., 2016; Chung et al., 2017; Jun et al., 2017), but they have not been tested on real “ground truth” data.

In ground truth data, one neuron is cherry picked from among all the neurons recorded using an extracellular array using another technique, and simultaneousy recorded. Unfortunately, such data are rare. Dual loose patch and extracellular recordings have been performed for culture of neurons or in cortical slices (Anastassiou et al., 2015; Franke et al., 2015a). However, in this condition, only one or two neurons emit spikes, and this simplifies drastically the spike sorting problem. Ground truth data recorded *in vivo* are scarce (Henze et al., 2000; Neto et al., 2016) and in many cases the patch-recorded neuron is relatively far from the extracellular electrodes.

As a result, most algorithms have been tested in simulated cases where an artificial template is added at random times to an actual recording. However, it is not clear if this simulated data reproduce the conditions of actual recordings. In particular, waveforms triggered by a given neuron can vary in amplitude and shape, and most simulations do not reproduce this feature of biological data. Also, spike trains of different cells are usually correlated, and these correlations can make extracellular spikes that do overlap.

Here we present a novel toolbox for spike sorting *in vitro* and *in vivo*, validated on ground truth recordings. Our sorting algorithm is based on a combination of density-based clustering and template matching. To validate our method, we performed experiments where a large-scale extracellular recording was performed while one of the neurons was recorded with a patch electrode. We showed that the performance of our algorithm was always close to an optimal classifier, both *in vitro* and *in vivo*. We demonstrate that our sorting algorithm could process recordings from up to thousands of electrodes with similar accuracy. To handle data from thousands of electrodes, we developed a tool automating the step that is usually left to manual curation.

Our method is a fast and accurate solution for spike sorting for up to thousands of electrodes, and we provide it as a freely available software that can be run on multiple platforms and several computers in parallel. Our ground truth data are also publicly available and will be a useful resource to benchmark future improvements in spike sorting methods.

Results

Spike sorting algorithm

We developed an algorithm (called SpyKING CIRCUS) with two main steps: a clustering followed by a template matching step (see Methods for details). First, spikes are detected as threshold crossings (fig. 1A) and the algorithm isolated the extracellular waveforms for a number of randomly chosen spike times. In the following text, we will refer to the extracellular waveforms associated with a given spike time as snippets.

We divided the snippets into groups, depending on their physical positions: for every electrode we grouped together all the spikes having their maximum peak on this electrode. Thanks to this division, the ensemble of spikes was divided into as many groups as there were electrodes. The group associated with electrode k contains all the snippets with a maximum peak on electrode k. It was possible that, even among the spikes peaking on the same electrode, there could be several neurons. We thus performed a clustering separately on each group, in order to separate the different neurons present in a single group.

For each group, the snippets were first masked: we assumed that a single cell can only influence the electrodes in its vicinity, and only kept the signal on electrodes close enough to the peak (fig. 1B, see methods). Due to to this reduction, the memory needed for each clustering did no scale with the total number of electrodes. The snippets were then projected into a lower dimensional feature space using Principal Component Analysis (PCA) (usually 5 dimensions, see Methods), as is classically done in many spike sorting algorithms (Rossant et al., 2016; Einevoll et al., 2012). Note that the simple division in groups before clustering allowed us to parallelize the clustering step, making it scalable for even thousands of electrodes. Even if a spike is detected on several electrodes, it is only assigned to the electrode where the voltage peak is the largest: if a spike has its largest peak on electrode 1, but is also detected on electrode 2, it will only be assigned to electrode 1 (see fig. 1 - figure supplementary 1).

For each group, we performed a density-based clustering inspired by (Rodriguez and Laio, 2014) (see Methods). The idea of this algorithm is that the centroid of a cluster in the feature space should have many neighbours, i.e. a high density of points in their neighbourhood. The centroid should also be the point where this density is a local maximum: there should not be a point nearby with a higher density. To formalize this intuition, for each point we measured the average distance of the 100 closest points ρ (intuitively, this is inversely proportional to the local density of points), and the distance δ to the closest point of higher density (i.e. with a lower ρ). Centroids should have a low ρ and a high δ. We hypothesized that there was a maximum of ten clusters in each group (i.e. at most ten different cells peaking on the same electrode) and took the ten points with the largest δ / ρ ratio as the centroids. Each remaining point was then assigned iteratively to the nearest point with highest density, until they were all assigned to the centroids (see methods for details - note that all the numbers mentioned here are parameters that are tunable in our toolbox).

Thanks to this method we could find many clusters, corresponding to putative neurons. In many spike sorting methods, it is assumed that finding clusters is enough to solve the spike sorting problem (Chung et al., 2017). However, this neglects the specific issue of overlapping spikes (see fig. 1C). When two nearby cells spike synchronously, the extracellular waveforms evoked by each cell will superimpose (fig. 1C, see also Pillow et al. (2013)). This superimposition of two signals coming from two different cells will distort the feature estimation. As a result, these spikes will appear as points very far away from the cluster associated to each cell. An example of this phenomena is illustrated in fig. 1C. Blue and red points correspond to the spikes associated to two different cells. In green, we show the spikes of one cell when the waveform

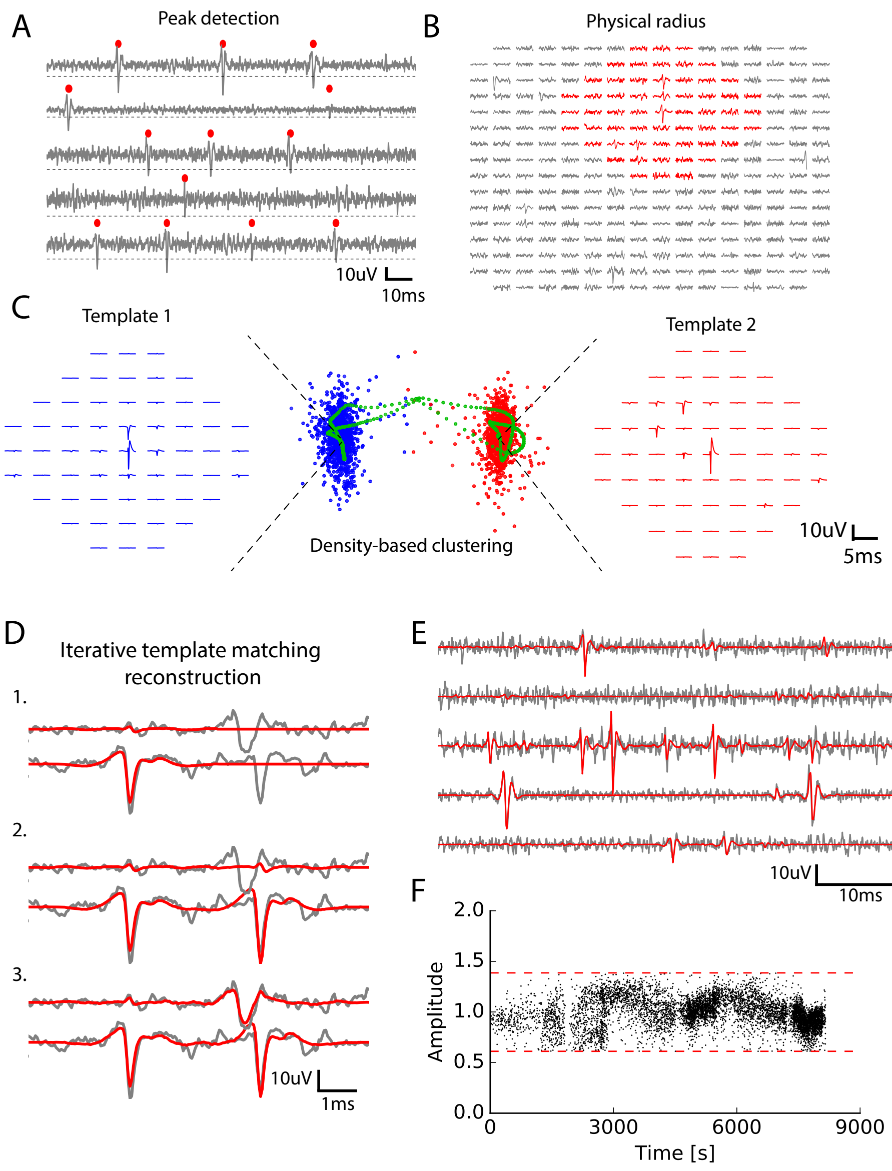

Figure 1: Main steps of the spike sorting algorithm **A.** Five randomly chosen electrodes, each of them with its own detection threshold (dash dotted line). Detected spikes, as threshold crossings, are indicated with red markers **B.** Example of a spike in the raw data. Red: electrodes that can be 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.** Example of two clusters (red and blue) with associated templates. Green points show possible combinations of two overlapping spikes from the two cells for various time delays. **D.** Graphical illustration of the template matching for *in vitro* data (see Methods). Every line is a electrode. Grey: real data. Red: sum of the templates added by the template matching algorithm; top to bottom: successive steps of the template matching algorithm. **E.** Final result of the template matching. Same legend as **D**. **F.** 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 this particular template was fitted to the data. Dashed dotted lines represent the amplitude thresholds (see methods).

of another one was added at different delays. For short delays, the presence of this additional waveform strongly distort the feature estimation. As a result, the corresponding point is far from the initial cluster, and will be missed by the clustering. To overcome this issue, we performed a template matching as the next step of our algorithm.

For this we first extracted a template from each cluster. This template is a simplified description of the cluster and is composed of two waveforms. The first one is the average extracellular waveform evoked by the putative cell (fig. 1C, left and red waveforms). The second is the direction of largest variance that is orthogonal to this average waveform (see Methods). We assumed that each waveform triggered by this cell is a linear combination of these two components. Thanks to these two components, the waveform of the spikes attributed to one cell could vary both in amplitude and in shape.

At the end of this step, we should have extracted an ensemble of templates (i.e. pairs of waveforms) that correspond to putative cells. Note that we only used the clusters to extract the templates. Our algorithm is thus tolerant to some errors in the clustering. For example, it can tolerate errors in the delineation of the cluster border. The clustering task here is therefore less demanding than in classical spike sorting algorithms where finding the correct cluster borders is essential to minimize the final error rate. By focusing on only getting the cluster centroids, we should thus have made the clustering task easier, but all the the spikes corresponding to one neuron have yet to be found. We therefore used a template matching algorithm to find all the instances where each cell has emitted a spike.

In this step we assumed that the templates of different cells spiking together sum linearly and used a greedy iterative approach inspired by the projection pursuit algorithm to match the templates to the raw data (fig. 1D, see Methods). Within a piece of raw data, we looked for the template whose first component had the highest similarity to the raw signal (here similarity is defined as the scalar product between the first component of the template and the raw data) and matched its amplitude to the signal. If this amplitude falls between pre-determined thresholds (see methods), we matched and subtracted the two components to the raw signal. These predetermined thresholds reflect the prior that the amplitude of the first component should be around 1, which corresponds to the average waveform evoked by the cell. We then reiterated this matching process until no more spike could be matched (fig. 1D, E) (see Methods). We found many examples where allowing amplitude variation was a desirable feature (see fig. 1F).

After this template matching step, the algorithm outputs putative cells, described by the templates, and associated spike trains, i.e. spike times where the template was matched to the data.

Performance on ground truth data

To test our algorithm, we performed dual recordings (fig. 2A, B) using both a multielectrode array to record many cells (see schematic on fig. 2A), and simultaneous loose patch to record the spikes of one of the cell (fig. 2B). For this cell we know what should be the output of the spike sorting. *In vitro*, we recorded 18 neurons from 14 retinas with a 252 electrode MEA where the spacing between electrodes was 30 μm (see Methods). We also generated datasets where we removed the signals of some electrodes, such that the density of the remaining electrodes was either 42 or 60 μm (by removing half or 3 quarters of the electrodes, respectively).

We then ran the spike sorting algorithm only on the extracellular data, and estimated the error rate (as the mean of false positives and false negatives, see methods) for the cell recorded in loose patch, where we know where the spikes occurred. The performance of the algorithm is not only limited by imperfections of the algorithm, but also by several factors related to the ground truth data themselves. In particular, some of the cells recorded with loose patch can evoke only a small spike on the extracellular electrode, for example if they are far from the nearest electrode or if their soma is small (Buzsáki, 2004). If a spike of the patch-recorded cell triggers a large voltage deflection, this cell should be easy to detect. However, if the triggered voltage deflection is barely detectable, even the best sorting algorithm will not perform well. Looking at fig. 2C, for *in vitro* data (see Methods), we found that there was a correlation between the error rate of our algorithm and the size of the extracellular waveform evoked by the spikes of the patch-recorded cell: the higher the waveform, the lower the error rate.

We then asked if our algorithm is close to the "best" possible performance, i.e. the only errors are due to intrinsic limitations in the ground truth data (e.g. small waveform size).

There is no method to exactly estimate this best possible performance. However, a proxy can be found by training a nonlinear classifier on the ground truth data (Harris et al., 2000; Rossant et al., 2016). We trained a nonlinear 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). 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. Note that, strictly speaking, this BEER estimate is not a lower bound of the error rate. It assumes that spikes can be all found inside a region of the feature space delineated by ellipsoidal boundaries. As we have explained above, spikes that overlap with spikes from another cell will probably be missed and this ellipsoidal assumption is also likely to be wrong in case of bursting neurons or electrode-tissue drifts . However, we used the BEER estimate because it has been used in several papers describing spike sorting methods (Harris et al., 2000; Rossant et al., 2016) and has been established as a commonly accepted benchmark. In addition, because we used rather stationary recordings (few minutes long, see Methods), we did not see strong electrode-tissue drifts.

We estimated the error made by the classifier and found that the performance of our algorithm almost always was in the same order of magnitude as the performance of this classifier, (fig. 2D, left; $r = 0.89$, $p < 10^{-5}$) over a broad range of spike sizes. For 37 neurons with large waveform sizes (above -50 μV) the average error of the classifier is 2.7 % and the one for our algorithm is 4.8 % (see fig. 2E). For 22 neurons with lower spike size (below -50 μV), the average error of the classifier is 11.1 % and the one for our algorithm is 15.2 %. This suggests that our algorithm reached an almost optimal performance on this *in vitro* data.

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 as for *in vitro* data, we also found that our algorithm achieved near optimal performance (fig. 2F, right; $r = 0.92$, $p < 10^{-5}$), although there were only two recordings where the spike size of the patch-recorded neuron was large enough to be sorted with a good accuracy. For only 2 available neurons with low optimal error rate, the average error of the classifier is 13.9 % and the one for our algorithm is 14.8 %. For 24 neurons with lower spike size, the average error of the classifier is 64.0 % and the one for our algorithm is 67.8 %. Together, these results show that our algorithm can reach a satisfying performance (i.e. comparing to the classifier error) over a broad range of spike sizes, for both *in vivo* and *in vitro* recordings.

We also compared our performance to the Kilosort algorithm (Pachitariu et al., 2016) and found similar performances (4.4 % on average over all non-decimated neurons for SpyKING CIRCUS against 4.2 % for Kilosort). Because Kilosort used a GPU, it could be run faster than our algorithm on a single machine: on a one hour recording with 252 electrodes, Kilosort can achieve a 4 times speedup on a standard desktop machine (see Methods). But without using a GPU, Kilosort was only marginally faster (1.5 speedup), and slower if we started using several cores of the machine. However, is it worth noticing that the speedup of Kilosort comes at the cost of an increased usage of memory. Kilosort used 32 GB of RAM for a maximal number of 500 neurons, while our algorithm had a much lower memory footprint, because of design choices. We have therefore found a trade off where execution speed is slightly slower, but much less memory is used. Thanks to this, we could run our algorithm to process recordings with thousands of electrodes, while Kilosort does not scale up to this number. In the next section we demonstrate that our algorithm still works accurately at that scale.

Scaling up to thousands of electrodes

A crucial condition to process recordings performed with thousands of electrodes is that every step of the algorithm should be run in parallel over different CPUs. The clustering step of our algorithm was run in parallel on different subsets of snippets as explained above. The template matching step could be run independently on different blocks of data, such that the computing time only scaled linearly with the data length. Each step of the spike sorting algorithm was therefore parallelized. The runtime of the full algorithm decreased proportionally with the numbers of CPU cores available (fig. 3A, grey area indicates where the software is "real-time" or faster). As a result, the sorting algorithm could process one hour 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 ($\geq$ 30 min) with more than 4000 electrodes, and the runtime could be be divided by simply having more CPUs available.

To test the accuracy of our algorithm on 4225 electrodes, we generated hybrid ground truth datasets where artificial spikes were added to real recordings performed on mouse retina *in vitro* (see Methods). We ran the spike sorting algorithm 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). This process, as shown in fig. 3D allowed us to generate "hybrid" datasets were we know the activity of a number of artificially injected neurons. We then ran our sorting algorithm on these datasets and measured if the algorithm was able to find at which times the artificial spikes were added. We counted a false negative error when an artificial spike was missed and a false positive error when the algorithm detected a spike when 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 spike detection threshold (normalized amplitude corresponds to the spike size divided by the spike threshold). Error rates were similar for recordings with 4255 electrodes *in vitro*, 128 electrodes *in vivo* or 252 electrodes *in vitro*. Performance did not depend on the firing rate of the injected templates (fig. 3E) and only weakly on the normalized amplitude of the templates (fig. 3F), as long as it was above the spike threshold. The accuracy of the algorithm is therefore invariant to the size of the recordings.

A crucial issue when recording thousands of densely packed electrodes is that more and more spikes overlap with each other. If an algorithm misses overlapping spikes, then the estimation of the amplitude of correlations between cells will be biased. To test if our method was able to solve the problem of overlapping spikes and thus estimate correlations properly, we generated hybrid datasets where we injected templates with a controlled amount of overlapping spikes (see Methods). We then ran the sorting algorithm and compared the injected spike trains and the spike trains recovered by SpyKING CIRCUS. We then compared the correlation between both pairs. If some overlapping spikes were missed by the algorithm, the correlation between the sorted spike trains should be lower than the correlation between the injected spike trains. We found that our method was always able to estimate the pairwise correlation between the spike trains with no underestimation (fig. 3G). Overlapping spikes were therefore correctly detected by our algorithm. The ability of our template matching algorithm to resolve overlapping spikes thus allowed an unbiased estimation of correlations between spike trains, even for thousands of electrodes.

These different tests, described above, show that SpyKING CIRCUS reached a similar performance for 4225 electrodes than for hundreds electrodes, where our ground truth recordings showed that performance was near optimal. Our algorithm is therefore also able to sort accurately recordings from thousands of electrodes.

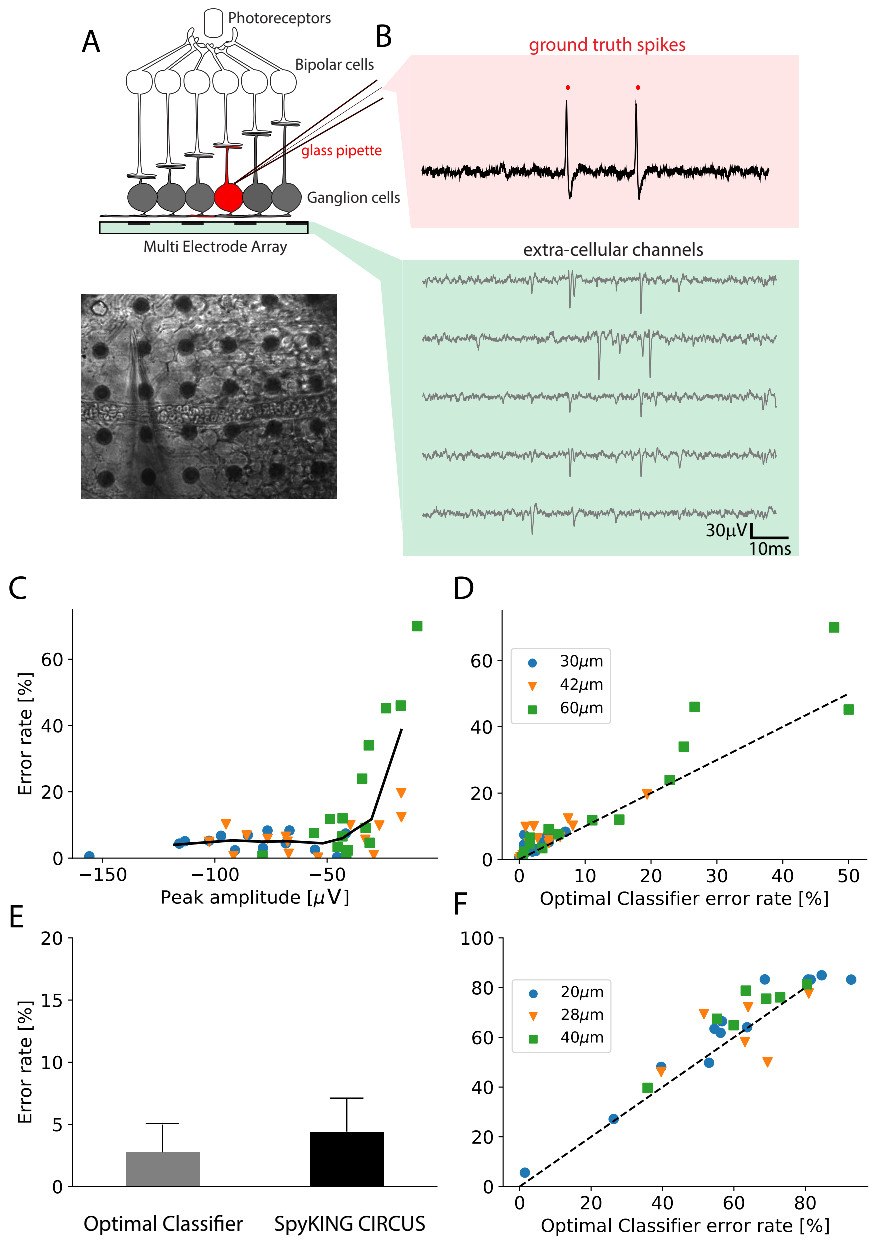

Figure 2: Performance of the algorithm on ground truth datasets **A.** Top: Schematic of the experimental protocol *in vitro*. A neuron close to the multielectrode array (MEA) recording is recorded in loose patch. Bottom: Image of the patch electrode on top of a 252 electrodes MEA, recording a ganglion cell. **B.** Top, pink box: loose patch recording showing the spikes of the recorded neuron. Bottom, green box: Extra-cellular recordings next to the loose patched soma. Each line is a different electrode **C.** Error rate of the algorithm as function of the largest peak amplitude of the ground-truth neuron, recorded extracellularly *in vitro*. **D.** Comparison between the error rates produced by the algorithm on different ground truth datasets and the error rates of nonlinear classifiers (Best Ellipsoidal Error Rate) trained to detect the spikes for *in vitro* data. **E.** Comparison of average performance for all neurons detected by the Optimal Classifier with an error less than 10% (n=37). **F.** Same as **D.** but for *in vivo* data (Neto et al., 2016) (see Methods).

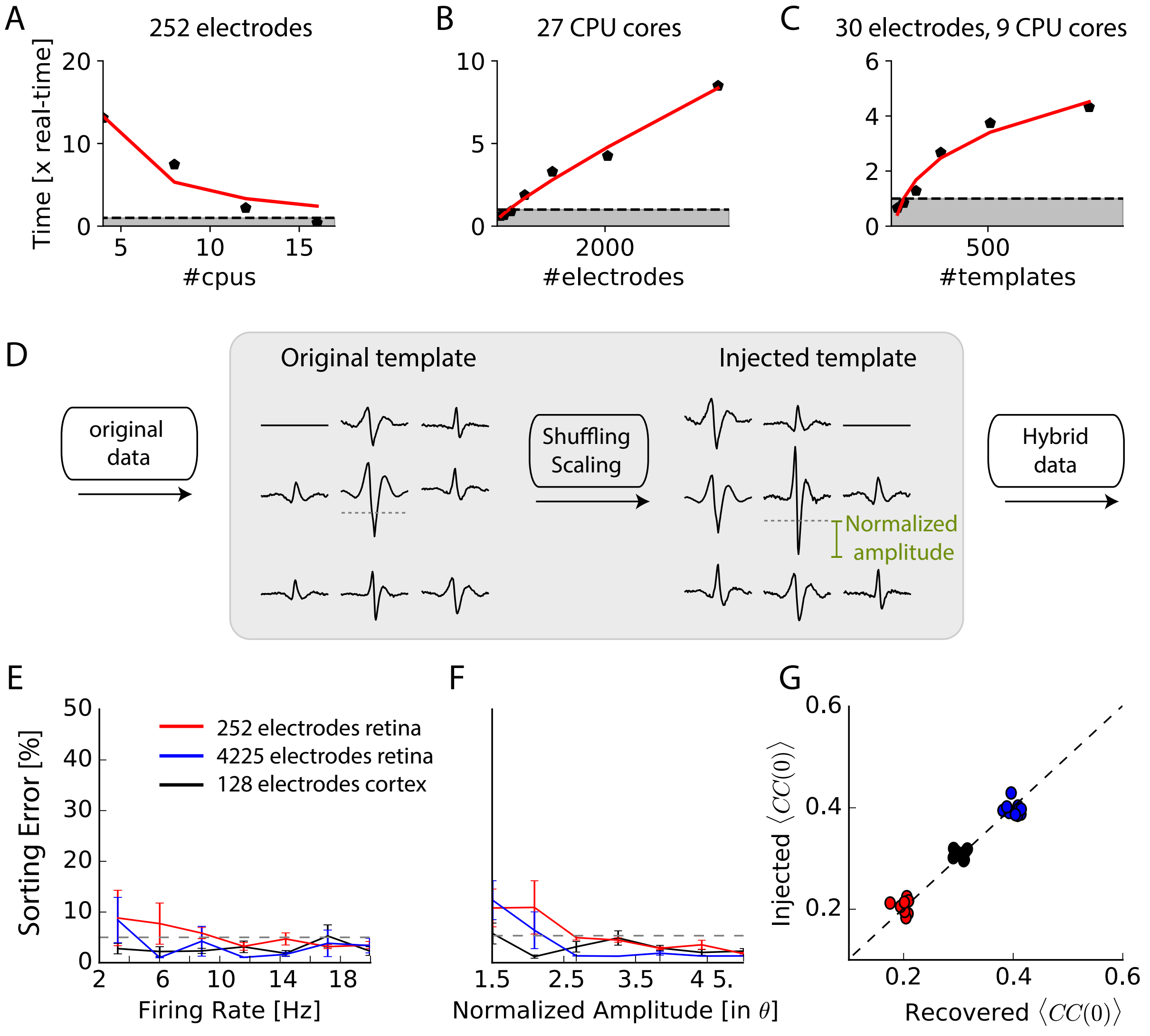

Figure 3: Scaling to thousands of electrodes. **A.** Execution time as function of the number of processors for a 90 min dataset *in vitro* with 252 electrodes, expressed as a real-time ratio, i.e. the number of hours necessary to process one hour of data. **B.** Execution time as function of the number of electrodes for a 30 min dataset recorded *in vitro* with 4225 electrodes. **C.** Execution time as function of the number of templates for a 10 min synthetic dataset with 30 electrodes. **D.** Creation of "hybrid" datasets: chosen templates are injected elsewhere in the data as new templates. Dashed-dotted lines shows the detection threshold on the main electrode for a given template, and normalized amplitude is expressed relative to this threshold (see Methods). **E.** Mean error rate as function of the firing rate of injected templates, in various datasets. Errors bars show standard error over 8 templates **F.** Error rate as function of the normalized amplitude of injected templates, in various datasets. Errors bars show standard error over 9 different templates **G.** Injected and recovered cross-correlation value between pairs of neurons for 5 templates injected at 10 Hz, with a normalized amplitude of 2 (see Methods).

Automated merging

As in most spike sorting algorithms, our algorithm may split one cell into several units. After running the entire algorithm, it is therefore necessary to merge together the units corresponding to the same cell. However, for hundreds or thousands of electrodes, going through all the pairs of units and merging them by hand would take a substantial amount of time. To overcome this problem, we designed a tool to merge automatically many units at once so that the time spent on this task does not scale with the number of electrodes and this allows us to automate this final step.

Units that likely belong to the same cell (and thus should be merged) have templates that look alike and in addition, the combined cross-correlogram between the two cell's spike trains shows a clear dip near 0 ms, indicating that the merged spike trains do not show any refractory period violation (fig. 4A, blue example). In order to automate this merging process, we formalized this intuition by estimating for each pair of units two factors that reflect if they correspond to the same cell or not. For each pair of templates, we estimated first the similarity between templates and second the dip in the center of the cross-correlogram. This dip is measured as the difference between the geometrical mean of the firing rate ϕ (i.e. the baseline of the cross-correlogram) and the value of the cross-correlogram at delay 0 ms, $\langle CC \rangle$ (see Methods and right insets in fig. 4A).

We plotted for each pair with high similarity the dip estimation against the geometrical mean of their firing rates. If there is a strong dip in the cross-correlogram (quantified by $\phi - \langle CC \rangle$), the dip quantification and the geometrical mean, ϕ, should be almost equal, and the corresponding pair should thus be close to the diagonal in the plot.

In one example, where we artificially split synthetic spike trains (fig. 4A ; see Methods), we could clearly isolate a cluster of pairs lying near this diagonal, corresponding to the pairs that needed to be merged (fig. 4A right panels). We have designed a GUI such the user can automatically select this cluster and merge all the pairs at once. Thanks to this, with a single manipulation by the user, all the pairs are merged.

We then tested this method on our *in vitro* ground truth data. In these recordings, the cell recorded with loose patch might be split by the algorithm between different spike trains. We can determine the units that correspond to the patch-recorded cell. For one particular neuron taken from our database, we can visualize all the units that need to be merged together (blue points in fig. 4B), and that should not be merged with units corresponding to other cells (green pairs in fig. 4B). For all the other cells, we do not have access to a ground truth, and thus cannot decide if the pairs should be merged or not (orange pairs in fig. 4B).

To automate the merging process entirely, we defined two simple rules to merge two units: first, their similarity should be above a threshold (similarity threshold, 0.8 in our experiments). Second, the dip estimation for this pair should be close to the geometrical mean of firing rates, i.e. their difference should be below a threshold (dip threshold). In practice, the corresponding point in fig. 4B should be above a line parallel to the diagonal. We used these rules to perform a fully automatic merging of all units. We then estimated the error rate for the ground truth cell, in the same way as the previous section. We also estimated the lowest error rate possible error rate by merging the units using the ground truth data for guidance (Best Merges, see Methods). We found that the error rate obtained with our automated method was close to this best error rate (fig. 4C). We have therefore automated the process of merging spike trains while keeping a low error rate. The performance did not vary much with the values of the two parameters (similarity threshold and dip threshold), and we used the same parameters for all the different datasets. This shows that the sorting can be fully automated while limiting the error rate to a small value. We thus have a solution to fully automate the sorting, including all the decisions that need to be taken during the manual curation step. However, because we used cross-correlograms in order to help automate the merging process, it is worth noticing that one can no longer use cross-correlograms as a validation metric.

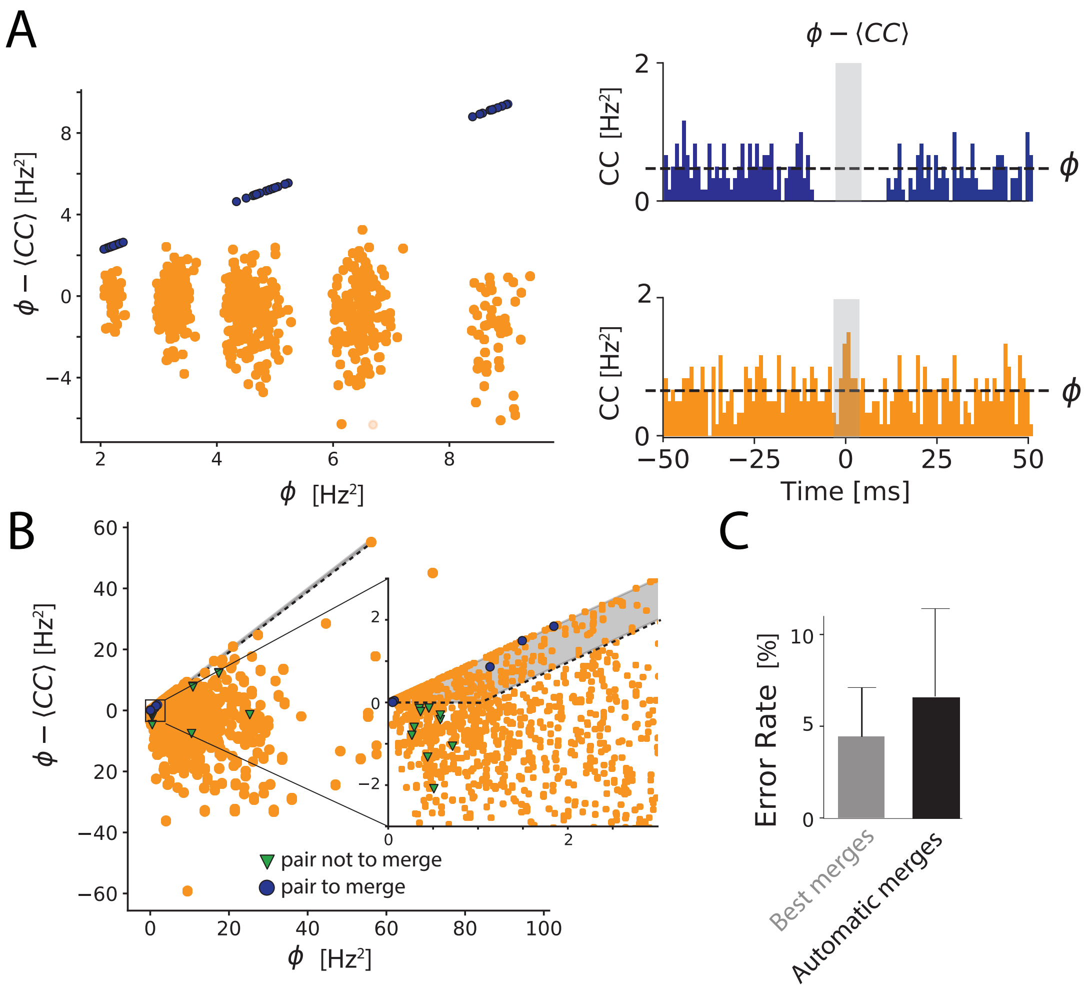

Figure 4: Automated merging **A.** Dip estimation (y-axis) compared to the geometrical mean of the firing rate (x-axis) for all pairs of units and artificially generated and split spike trains (see Methods). Blue: pairs of templates originating from the same neuron that have to be merged. Orange: pairs of templates corresponding to different cells. Panels on the right: for two chosen pairs, one that needs to be merged (in blue, top panel) and one should not be merged (orange, bottom panel) the full cross-correlogram and the geometrical mean of the firing rate (dashed line). The average correlation is estimated in the temporal window defined by the gray area. **B.** Same as A, for a ground truth dataset. Blue points: points that need to be merged. Green points: pairs that should not be merged. Orange points: pairs where our ground truth data does not allow us telling if the pair should be merged or not. The gray area corresponds to the region where pairs are merged by the algorithm. Inset: zoom on one region of the graph. **C.** Average error rate in the case where the decision of merging units was guided by the ground truth data (left) against the automated strategy designed here (right).

Discussion

We have shown that our method, based on density-based clustering and template matching, allows sorting spikes from large-scale extracellular recordings both *in vitro* and *in vivo*. 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 nonlinear classifier, 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). Note that tests were performed by different groups on our algorithm and show its high performance on various datasets (see http://spikesortingtest.com/ and http://phy.cortexlab.net/data/sortingComparison/). Our algorithm is entirely parallelized and could therefore handle long datasets recorded with thousands of electrodes. Our code has already been used by other groups (Denman et al., 2017; Mena et al., 2017; Chung et al., 2017; Wilson et al., 2017) and is available as a complete multi-platform, open source software for download (http://spyking-circus.rtfd.org) with a complete documentation. Note that all the parameters mentioned in the description of the algorithm can be modified easily to work with different kinds of data. We have made all the ground truth data available for download (see Source Code section in Methods), so that improvements in our algorithm as well as alternative solutions could be benchmarked easily in the future.

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; Chung et al., 2017) and clustering the spikes in the feature space. In this approach, the spike sorting problem is reduced to a clustering problem and this introduces several major problems. 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 task that cannot be easily automated (but see (Chung et al., 2017)). 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, using a template matching approach, so that we only needed to infer the centroid of each cluster and not their precise borders.

The template matching approach also allowed us to deconvolve overlapping spikes in a fast, efficient and automated manner. Some template matching approaches have been previously tested, mostly on *in vitro* data (Marre et al., 2012; Pillow et al., 2013; Franke et al., 2015b), but were not validated on ground truth datasets like the ones we acquired here. Also, they only had one waveform for each template, which did not allow any variation in the shape of the spike, while we have designed our template matching method to take into account not only variation in the amplitude of the spike waveform, but also in shape. Finally, several solutions did not scale up to thousands of electrodes. All GPU-based algorithms (Pachitariu et al., 2016; Lee et al., 2017; Jun et al., 2017) only scale for a few hundreds channels, and face severe memory issues for larger probes.

Finally, a common issue when sorting spikes from hundreds or thousands of electrodes is the time spent on manual curation of the data. Here we have designed a tool to automate this step by merging units corresponding to the same cell all at once, based on the cross-correlogram between cells and the similarity between their templates. Having an objective criterion for merging spike trains not only speeds up the manual curation time, it also makes the results less sensitive to human errors and variability in decisions. In some cases, it might be necessary to take into account additional variables that are specific to the experiment, but even then our tool will still significantly reduce the time spent on manual curation.

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 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. This will require adapting our method to process data "on the fly", processing new data blocks when they become available and adapting the shape of the templates 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 multielectrode array with 252 electrodes spaced 30 or 60 μm apart (Multi-Channel Systems) or with 4225 electrodes arranged in a 2D grid and spaced by 16 μm (Zeck et al., 2011; Bertotti et al., 2014) at a sampling rate of 20 kHz. 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 % O_2 and 5 % CO_2 at room temperature, on a MEA (10 μm electrodes spaced by 30 μm; Multichannel Systems, Reutlingen, Germany) with ganglion cells layer facing the electrodes. Borosilicate glass electrodes (BF100-50, Sutter instruments) 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. We used rd1 mice because going through the photoreceptor layer with the patch pipette was easier than for a wild type mouse.

For the data shown in fig. 1 and 3, we used a recording of 130 min. For the data shown in fig. 2A, 16 neurones were recorded over 14 intact retinas. Recording durations all lasted 5min. 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 (20 μm spacing) at 30 kHz performed in rat visual cortex, combined with juxta-cellular recordings. The dataset gave us a total of 13 neurons for fig. 2.C with recordings between 4 and 10 min 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. 3, we used the longest dataset recorded with 128 electrodes.

Details of the algorithm

In the following, we consider that we have N_{elec} electrodes, acquired at a sampling rate f_{rate}. Every electrode k is located at a physical position $\mathbf{p}_k = (x_k, y_k)$ in a 2D space (extension to 3D probes would be straightforward). The aims of our algorithm is to decompose the signal $\mathbf{s} = \{s_k,\ k \in 1, \dots N_{\text{elec}}\}$ as a linear sum of spatio-temporal kernels or "templates" (see equation 1).

$$\mathbf{s}(t) = \sum_{ij} a_{ij}\mathbf{w}_j(t - t_i) + b_{ij}\mathbf{v}_j(t - t_i) + \mathbf{e}(t) \quad (1)$$

where $\mathbf{s}(t)$ is the signal recorded over N_{elec} electrodes and over multiple time points. $\mathbf{w}_j(t - t_i)$ and $\mathbf{v}_j(t - t_i)$ are the two components of the template associated to each cell. They represent the waveform triggered on the electrodes by cell j. Times $\{t_i\}$ are the putative spike times over all the electrodes. a_{ij} and b_{ij} are the amplitude factors for spike time t_i for the template j, and $\mathbf{e}(t)$ is the background noise. Note that at a given spike time t_i, it is likely that only a couple of cells fire a spike. Only these cells will therefore have a_{ij} and b_{ij} different from zero. For all the other ones, these coefficients are zero simply because the cell does not fire at this time.

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 2 ms for *in vivo* and 5 ms for *in vitro* data) and only electrodes whose distance to the soma is below r_{max} (typically 100 μm for *in vivo* and 200 μm for *in vitro* data). For every electrode k centered on $\mathbf{p}_k$, we define G^k as the ensemble of nearby electrodes l such that $\|\mathbf{p}_k - \mathbf{p}_l\|_2 \leq r_{\text{max}}$.

Pre-processing

Filtering In a preprocessing stage, all the signals were individually high-pass filtered with a Butterworth filter of order three and a cutoff frequency of $f_{\text{cut}} = 100$ Hz to remove any low-frequency components of the signals. We then subtracted, for every electrode k, the median such that $\forall k$ $\text{med}(\mathbf{s}_k) = 0$.

Spike detection Once signals have been filtered, we computed a spike threshold θ_k for every electrode k: $\theta_k = \lambda \text{MAD}(\mathbf{s}_k(t))$, where MAD is the Median Absolute Deviation, and λ is a free parameter. For all the datasets shown in this paper, we set $\lambda = 6$. We detected the putative spike times t_i as times where there was at least one electrode k where $\mathbf{s}_k(t_i)$ was below the threshold $-\theta_k$ and a local minimum of the voltage $\mathbf{s}_k(t)$.

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

Basis estimation (PCA) Our first goal was to reduce the dimensionality of the temporal waveforms. We collected up to N_{p} spikes on each electrode. We thus obtained a maximum of $N_{\text{p}} \times N_{\text{elec}}$ spikes and took the waveform only on the peaking electrode for each of them. This is a collection of a large number of temporal waveforms and we then aimed at finding the best basis to project them. In order to compensate for sampling rate artifacts, we first upsampled all the collected single-electrode 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 then 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 $N_{\text{p}} = 10000$ 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.

Masking We first randomly collected a subset of many spikes t_i to perform the clustering. To minimize redundancy between collected spikes, we prevented the algorithm to have two spikes peaking on the same electrode separated by less than $N_t/2$.

Pre-clustering of the spikes Trying to cluster all the spikes from all the electrodes at once is very challenging, because they are numerous and live in a high dimensional space. We used a divide and conquer approach to parallelize this problem (Marre et al., 2012; Swindale and Spacek, 2014). Each time a spike was detected at time t_i , we searched for the electrode k where the voltage $\mathbf{s}(t_i)$ has the lowest value, i.e. such that $k = \text{argmin}_k\ \mathbf{s}_k(t_i)$. For every electrode k we collected a maximum of N_{spikes} spikes (set to 10000 by default) peaking on this electrode. Each of these spikes is represented by a spatio-temporal waveform of size $N_t \times N^k_{\text{neigh}}$, where N^k_{neigh} is the number of electrodes in the vicinity of electrode k, i.e. the number of elements in G^k. Note that, in the following we did not assume that spikes are only detected on a single electrode. We used the information available on all the neighboring electrodes.

We projected each temporal waveform on the PCA basis, estimated earlier, to reduce the dimensionality to $N_{\text{PCA}} \times N^k_{\text{neigh}}$. During this projection, the same up-sampling technique described in the Pre-processing was used. Each spike was then represented in a space with $N_{\text{PCA}} \times N^i_{\text{neigh}}$ dimensions. To reduce dimensionality even further before the clustering stage, for every electrode k 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 electrode on at max N_{spikes} described in a space of N_{PCA_2}-dimension.

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). We note the spikes $\{1, .., l\}$ associated with electrode k (and projected on the second PCA basis) as vectors $\mathbf{x}^k_{\{1,..,l\}}$ in a N_{PCA_2} dimensional space. For each of these vectors, we estimated ρ^k_l as the mean distance to the S nearest neighbors of $\mathbf{x}^k_l$. Note that $1/\rho^k_l$ can be considered as a proxy for the density of points. S is chosen such that $S = \epsilon\ N_{\text{spikes}}$, with $\epsilon = 0.01$. In our settings, since $N_{\text{spikes}} = 10000$ then $S = 100$. This density measure turned out to be more robust than the one given in the original paper and rather insensitive to changes in ϵ. To avoid a potentially inaccurate estimation of the ρ^k_l values we collected iteratively additional spikes to refine this estimate. Keeping in memory the spikes $\mathbf{x}^k_l$, we searched again in the data N^k_{spikes} different spikes and used them only to refine the estimation of ρ^k_l of our selected points $\mathbf{x}^k_l$. This refinement gave more robust results for the clustering and we performed 3 rounds of this new search. Then, for every point $\mathbf{x}^k_l$, we computed δ^k_l as the minimal distance to any other point $\mathbf{x}^k_{m,m\neq l}$ such that $\rho^k_m \leq \rho^k_l$. This corresponds to the distance to the nearest point with a higher density. 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).

Centroids and cluster definition To define the centroids we ranked the points as a function of the ratios δ/ρ and we detected the best $N^{\text{max}}_{\text{clusters}}$ points as putative centroids. By default $N^{\text{max}}_{\text{clusters}} = 10$. Intuitively, this parameter corresponds to the maximal number of cells that will peak on any given electrode. It can be seen as an upper bound of the ratio between the number of cells and the number of electrodes. Clusters were formed by assigning each point to one of the selected centroids following an iterative rule, going from the points of lowest ρ to the points of highest ρ: each point was assigned to the same cluster as the closest point with a lower ρ (Rodriguez and Laio, 2014). Thanks to this ordering we started by assigning the points of highest density to the nearest centroid, and then assigned the next points to the nearest point with higher density, which has been already assigned to a cluster. We created $N^{\text{max}}_{\text{clusters}}$ 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. The center α_m of each cluster was defined as the median of all the points composing this cluster. For each pair of cluster (m, n) we then projected all the points for each of them onto the axis joining the two centroids $\alpha_m - \alpha_n$. We defined the dispersions around the centroids α_m as $\gamma_m = \text{MAD}(\mathbf{x}_m \cdot (\alpha_m - \alpha_n))$, where $\cdot$ is the scalar product of two vectors. The normalized distance is:

$$\zeta(m,n) = \frac{\|\alpha_m - \alpha_n\|}{\sqrt{\gamma_m^2 + \gamma_n^2}} \tag{2}$$

We then iteratively merged all clusters (m, n) such that $\zeta(m, n) \leq \sigma_{\text{similar}}$. At the end of the clustering every cluster with less than $\eta\ N^i_{\text{spikes}}$ was discarded. In all the manuscript we used $\sigma_{\text{similar}} = 3$, $N^{\text{max}}_{\text{clusters}} = 10$, and $\eta = 0.005$. We chose $N^{\text{max}}_{\text{clusters}} = 10$ because we never see more than 10 neurons peaking on the same electrode, and this approximately corresponds to the ratio between density of observable cells and density of electrodes.

Template estimation At the end of the clustering phase, pooling the clusters obtained from every electrode, we obtained a list of clusters. Each cluster m is defined by a list of spike times $t^m_{\{1,..,l\}}$. During this phase we limited the number of spike times per template to a maximal value of 500 to avoid memory saturation, because we had to keep in memory these 500 snippets.

We computed the first component from the raw data as the point-wise median of all the waveforms belonging to the cluster: $\mathbf{w}_m(t) = \text{med}_l\ \mathbf{s}(t^m_l + t)$. Note that $\mathbf{w}_m$ is only different from zero on the electrodes close to its peak (see fig. 1C). This information is used internally by the algorithm to save templates as sparse structures. We set to 0 all the electrodes k where $\|\mathbf{w}^k_m(t)\| < \theta_k$, where θ_k is the detection threshold on electrode k. This allowed us to remove electrodes without discriminant information and to increase the sparsity of the templates.

We then computed the projection of all snippets in the space orthogonal to the first component: $\forall l,\ \mathbf{q}_l = \mathbf{s}(t^m_l) - \beta_l \mathbf{w}_m$, with $\beta_l = \frac{\mathbf{s}(t^m_l).\mathbf{w}_m}{\|\mathbf{w}_m\|}$. The $\mathbf{q}$ are the projections of the waveforms in a space orthogonal to $\mathbf{w}_m$. We estimated the second component of the template $\mathbf{v}_m(t)$ as the direction of largest variance in this orthogonal space (i.e. the first principal component of $\mathbf{q}_l$).

From the first components $\mathbf{w}_m$, we also computed its minimal and maximal amplitudes $a^{\text{min/max}}_m$. If $\hat{\mathbf{w}}_m$ is the normalized template, such that $\hat{\mathbf{w}}_m = \mathbf{w}_m / \|\mathbf{w}_m\|$, we computed

$$\begin{aligned} a^{\text{min}}_h &= \text{med}_l\ \mathbf{s}(t^m_l).\hat{\mathbf{w}}_m - 5\text{MAD}_l(\mathbf{s}(t^m_l).\hat{\mathbf{w}}_m) \\ a^{\text{max}}_h &= \text{med}_l\ \mathbf{s}(t^m_l).\hat{\mathbf{w}}_m + 5\text{MAD}_l(\mathbf{s}(t^m_l).\hat{\mathbf{w}}_m) \end{aligned} \tag{3}$$

Those boundaries are used during the template matching step (see below). The factor 5 allows most of the points to have their amplitude between the two limits.

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 between two electrodes would give rise to two very similar templates on two electrodes), we computed for all pairs of templates in the dictionary $CC_{\max}(m, n) = \max_t CC(\mathbf{w}_m, \mathbf{w}_n)$, where CC stands for normalized cross-correlation. If $CC_{\max}(m, n) \geq cc_{\text{similar}}$, we considered these templates to be equivalent and they were merged. In all the following, we used $cc_{\text{similar}} = 0.975$. Note that we computed 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 $\mathbf{w}_p$ could be explained as a linear combination of two templates in the dictionary. If we could find $\mathbf{w}_m$ and $\mathbf{w}_n$ such that $CC(\mathbf{w}_p, \mathbf{w}_m + \mathbf{w}_n) \geq cc_{\text{similar}}$, $\mathbf{w}_p$ 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 ($\mathbf{w}$, $\mathbf{v}$). We now need to reconstruct the signal $\mathbf{s}$ by finding the amplitudes coefficients a_{ij} and b_{ij} described in Equation 1. Because at a given spike time t_i it is likely that only a couple of cells will fire a spike, 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 $\mathbf{w}$. 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. Note that the spike times t_i were detected using the method described above and include all the threshold crossing voltages that are local minima. Each true spike can be detected over several electrodes at slightly different times such that there are many more t_i than actual spikes. With this approach we found that there was no need to shift templates before matching them to the raw data.

To match the templates to the data we used an iterative greedy approach to estimate the a_{ij} for each 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. The size of one block was typically one second, which includes a lot of spike times, and is much larger than a single snippet. The snippets were thus not fitted independently from each other. The successive blocks were always overlapping by two times the size of a snippet and we discarded the results obtained on the borders to avoid any error of the template matching that would be due to a spike split between two blocks. Such an approach allowed us to easily split the workload linearly among several processors.

Each block of raw data $\mathbf{s}$ was loaded and template-matching was performed according to the following steps:

1. Estimate the normalized scalar products $\mathbf{s}(t) \cdot \mathbf{w}_j(t - t_i)$ 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: $a_{ij} = \frac{\mathbf{s}(t).\mathbf{w}_j(t-t_i)}{\|\mathbf{w}_j(t-t_i)\|}$.
4. Check if the a_{ij} amplitude value is between $a_j^{\min}$ and $a_j^{\max}$.
5. If yes, accept this value, subtract the scaled template from the raw data: $\mathbf{s}(t) \leftarrow \mathbf{s}(t) - a_{ij}\mathbf{w}_j(t - t_i)$. Then set b_{ij} equal to the amplitude value that best fits the raw data with $\mathbf{v}_j$, and subtract it too. Then return to step 1 to re-estimate the scalar products on the residual.
6. Otherwise increase by one n_i, which counts the number of times any template has been rejected for spike time t_i.
   a If n_i reaches $n_{\text{failures}} = 3$, label this t_i as "explored". If all t_i have been explored, quit the loop.
   b Otherwise return to step 1 and iterate.

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

Automated merging

For the template similarity, we computed, for every pair of templates m and n, $CC_{\max}(m,n) = \max_t CC(\mathbf{w}_m, \mathbf{w}_n)$ (where CC is the normalized cross-correlation between the two templates - see above for the definition). To quantify the dip in the cross-correlogram, we binned the spike trains obtained for templates m and n with 2 ms bin size, and estimated the cross correlogram $r_{m,n}(\tau)$ between unit m and unit n, defined as $\langle \sigma_m(t)\ \sigma_n(t+\tau)\rangle_t$. $\sigma_m(t)$ is the number of spikes of unit m in time bin t. We then estimated the dip as the difference between the value of the cross-correlogram at time 0 ms and the geometrical mean of the firing rates, i.e. $\phi(m,n) = \langle \sigma_m(t)\rangle_t \langle \sigma_n(t)\rangle_t$. This geometrical mean would be the value of the cross-correlogram if the two spike trains were independent. The dip is therefore estimated as

$$\langle \sigma_m(t)\rangle_t \langle \sigma_n(t)\rangle_t - \langle \sigma_m(t)\ \sigma_n(t+\tau)\rangle_t \tag{4}$$

We plotted the values of the estimated dip, the template similarity and the geometrical mean of the firing rates for each pair in a Graphical User Interface (GUI). The user can quickly define at once a whole set of pairs that need to be merged. After merging a subset of the pairs, quantities $CC_{\max}$ and ϕ are re-computed, until the user decides to stop merging (see fig. 4).

If the two spike trains from templates m and n correspond to the same cell, there should be no refractory spike trains. The cross-correlogram value should be close to 0 and the dip estimation should therefore be close to the geometrical mean of the firing rates. To formalize this intuition and fully automate the merging, we decided to merge all the pairs (m,n) such that:

$$CC_{\max}(m,n) > cc_{\text{merge}} \text{ and } \langle \sigma_m(t)\ \sigma_n(t+\tau)\rangle_t \le \phi_{\text{merge}} \tag{5}$$

with $cc_{\text{merge}} = 0.8$ and $\phi_{\text{merge}} = 0.1$. This corresponds to merging all the highly similar pairs above a line parallel to the diagonal (see fig. 4A,B, gray area). With these two parameters we could automate the merging process.

Simulated ground truth tests

In order to assess the performance of the algorithm we injected new templates in real datasets (see fig. 3D). To do so, we ran the algorithm on a given dataset and obtain a list of putative templates $\mathbf{w}_{j\in\left\{1,\ldots N\right\}}$. Then, we randomly selected some of those templates $\mathbf{w}_j$ and shuffled the list of their electrodes before injecting them elsewhere in the datasets at controlled firing rates (Harris et al., 2000; Rossant et al., 2016; Kadir et al., 2014; Segev et al., 2004; Marre et al., 2012; Chung et al., 2017). This enabled us to properly quantify the performance of the algorithm. In order not to bias the clustering, when a template $\mathbf{w}_j$ was selected and shuffled as a new template $\overline{\mathbf{w}}_k$ centered on a new electrode k, we ensured that the injected template was not too similar to one that would already be in the data: $\forall h \in \left\{1,\ldots N\right\}, \max_t CC(\mathbf{w}_h, \overline{\mathbf{w}}_k) \le 0.8$. Before being injected, $\overline{\mathbf{w}}_k$ was normalized such that $\min_t \overline{\mathbf{w}}_k = \alpha_k\ \theta_k$. α_k is the relative amplitude, expressed as function of θ_k, the detection threshold on the electrode where the template is peaking. If $\alpha_k \le 1$ the template is smaller than spike threshold, and its spikes should not be detected; if $\alpha_k \ge 1$ the spikes should be detected. In fig. 3G, we injected the artificial templates into the data such that they were all firing at 10 Hz, 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.

Performance estimation

Estimation of false positives and false negatives 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 2 ms. Spikes in the ground-truth datasets that had no matches in the spike sorting results in a 2 ms window were labeled as "false negatives", while those that are not present while the algorithm detected a spike were "false positives". The false negative rate was defined as the number of false negatives divided by the number of spikes in the ground truth recording. The false positive rate was defined as the number of false positives 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. 2, 3). Note that to take into account the fact that a ground-truth neuron could be split into 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. 2) 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 $\mathbf{x}_i$ is the i-th snippet, projected in the feature space, then the optimized function $f(\mathbf{x})$ is:

$$f(\mathbf{x}) = \mathbf{x}^T A\mathbf{x} + b^T\mathbf{x} + c \tag{6}$$

We trained this function f by varying A, b and c with the objective that $f(\mathbf{x})$ 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 $f(\mathbf{x}) > 0$ were predicted as true spikes. This prediction provided an estimate 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, or "Optimal Classifier Error".

Decimation of the electrodes In order to increase the number of data points for the comparison between our sorting algorithm and the nonlinear classifiers defined by the BEER metric (see fig. 2), 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* Multielectrode Array or the 128 *in vivo* silicon probe.

Hardware specifications

The comparison between Kilosort (Pachitariu et al., 2016) and SpyKING CIRCUS was performed on a desktop machine with 32 Gb RAM and 8 cores (proc Intel® Xeon(R) CPU E5-1630 v3 @ 3.70GHz). The GPU used was a NVIDIA Quadro K4200 with 4 Gb of dedicated memory.

Implementation and Source Code

SpyKING CIRCUS 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

Variable	Explanation	Default value
Generic notations		
N_{elec}	Number of electrodes	
$\mathbf{p_k}$	Physical position of electrode k [μm]	
G_k	Ensemble of nearby electrodes for electrode k [μm]	
N^k_{neigh}	Cardinal of G_k	
θ_k	Spike detection threshold for electrode k [μV]	
$\mathbf{s(t)}$	Raw data [μV]	
$\mathbf{w}_j(t)$	First component of the template for neuron j [μV]	
$\mathbf{v}_j(t)$	Second component of the template for neuron j [μV]	
f_{rate}	Sampling frequency of the signal [Hz]	
Preprocessing of the data		
f_{cut}	Cutoff frequency for butterworth filtering	100 Hz
N_t	Temporal width for the templates	5 ms
r_{max}	Spatial radius for the templates	250 μm
λ	Gain for threshold detection for channel k (θ_k)	6
N_p	Number of waveforms collected per electrode	10000
N_{PCA}	Number PCA features kept to describe a waveform	5
Clustering and template estimation		
$\mathbf{x}^k_{1,..l}$	l spikes peaking on electrode k and projected after PCA	
ρ^k_l	Density around $\mathbf{x}^k_l$	
δ^k_l	Minimal distance from $\mathbf{x}^k_l$ to spikes with higher densities	
N_{spikes}	Number of spikes collected per electrode for clustering	10000
N_{PCA_2}	Number of PCA features kept to describe a spike	5
S	Number of neighbors for density estimation	100
$N^{\text{clusters}}_{\text{max}}$	Maximal number of clusters per electrode	10
ζ	Normalized distance between pairs of clusters	
σ_{similar}	Threshold for merging clusters on the same electrode	3
α_m	Centroid of the cluster m	
γ_m	Dispersion around the centroid α_m	
η	Minimal size of a cluster (in percent of N_{spikes})	0.005
$[a_{\text{min}}, a_{\text{max}}]$	Amplitudes allowed during fitting for a given template	
Dictionary cleaning		
$CC_{\text{max}}(m,n)$	Max over time for the Cross-correlation between $\mathbf{w}_m$ and $\mathbf{w}_n$	
cc_{similar}	Threshold above which templates are considered as similar	0.975
Template matching		
a_{ij}	Product between $\mathbf{s(t)}$ and $\mathbf{w_j}$ (normalized) at time t_i	
b_{ij}	Same as a_{ij} but for the second component $\mathbf{v_j}$	
n_{failures}	Number of fitting attempts for a given spike time	3
Automated merging		
cc_{merge}	Similarity threshold to consider neurons as a putative pair	0.8
$r_{m,n}(t)$	Cross correlogram between spikes of unit m and n	
$\phi(m,n)$	Geometrical mean of the firing rates for units m and n [Hz^2]	
ϕ_{merge}	Maximal value for the dip in the cross correlogram at time 0	0.1 [Hz^2]

Table 1: Table of all the variables and notations found in the algorithm.

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 (http://www.yger.net/software/ground-truth-recordings/).

Acknowledgments

We would like to thank Charlotte Deleuze for her help with the *in vitro* juxtacellular recordings, and Steve Baccus and Sami El Boustani for insightful discussions. We also would like to thank 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 and Foundation Figthing Blindness to SP.

References

Costas A Anastassiou, Rodrigo Perin, György Buzsáki, Henry Markram, and Christof Koch. Cell type-and activity-dependent extracellular correlates of intracellular spiking. *Journal of neurophysiology*, 114(1):608–623, 2015.

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.

György Buzsáki. Large-scale recording of neuronal ensembles. *Nature neuroscience*, 7(5):446–451, 2004.

Jason E Chung, Jeremy F Magland, Alex H Barnett, Vanessa M Tolosa, Angela C Tooker, Kye Y Lee, Kedar G Shah, Sarah H Felix, Loren M Frank, and Leslie F Greengard. A fully automated approach to spike sorting. *Neuron*, 95(6):1381–1394, 2017.

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`.

Daniel J. Denman, Joshua H. Siegle, Christof Koch, R. Clay Reid, and Timothy J. Blanche. Spatial organization of chromatic pathways in the mouse dorsal lateral geniculate nucleus. *Journal of Neuroscience*, 37(5):1102–1116, 2017.

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.

F. Franke, R. Propper, H. Alle, P. Meier, J. R. Geiger, K. Obermayer, and M. H. Munk. Spike sorting of synchronous spikes from local neuron ensembles. *J. Neurophysiol.*, 114(4):2535–2549, Oct 2015a.

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.

Felix Franke, Rodrigo Quian Quiroga, Andreas Hierlemann, and Klaus Obermayer. Bayes optimal template matching for spike sorting–combining fisher discriminant analysis with optimal filtering. *Journal of computational neuroscience*, 38(3):439–459, 2015b.

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.

Darrell A Henze, Zsolt Borhegyi, Jozsef Csicsvari, Akira Mamiya, Kenneth D Harris, and György Buzsáki. Intracellular features predicted by extracellular recordings in the hippocampus in vivo. *Journal of neurophysiology*, 84(1):390–400, 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.

James Jaeyoon Jun, Catalin Mitelut, Chongxi Lai, Sergey Gratiy, Costas Anastassiou, and Timothy D Harris. Real-time spike sorting platform for high-density extracellular probes with ground-truth validation and drift correction. *bioRxiv*, 2017. doi: 10.1101/101030.

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.

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 7.8 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`.

JinHyung Lee, David Carlson, Hooshmand Shokri, Weichi Yao, Georges Goetz, Espen Hagen, Eleanor Batty, EJ Chichilnisky, Gaute Einevoll, and Liam Paninski. Yass: Yet another spike sorter. *bioRxiv*, 2017. doi: 10.1101/151928. URL `https://www.biorxiv.org/content/early/2017/06/19/151928`.

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.

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.

Gonzalo E Mena, Lauren E Grosberg, Sasidhar Madugula, Paweł Hottowy, Alan Litke, John Cunningham, EJ Chichilnisky, and Liam Paninski. Electrical stimulus artifact cancellation and neural spike detection on large multi-electrode arrays. *PLOS Computational Biology*, 13 (11):e1005842, 2017.

J. P. Neto, G. Lopes, J. Frazao, J. Nogueira, P. Lacerda, P. Baiao, A. Aarts, A. Andrei, S. Musa, E. Fortunato, P. Barquinha, and A. R. Kampff. Validating silicon polytrodes with paired juxtacellular recordings: method and dataset. *J. Neurophysiol.*, 116(2):892–903, Aug 2016.

Marius Pachitariu, Nicholas A. Steinmetz, Shabnam N. Kadir, Matteo Carandini, and Kenneth D. Harris. Fast and accurate spike sorting of high-channel count probes with kilosort. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 4448–4456, 2016.

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.

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.

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.

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.

C. D. Wilson, G. O. Serrano, A. A. Koulakov, and D. Rinberg. A primacy code for odor identity. *Nat Commun*, 8(1):1477, Nov 2017.

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.

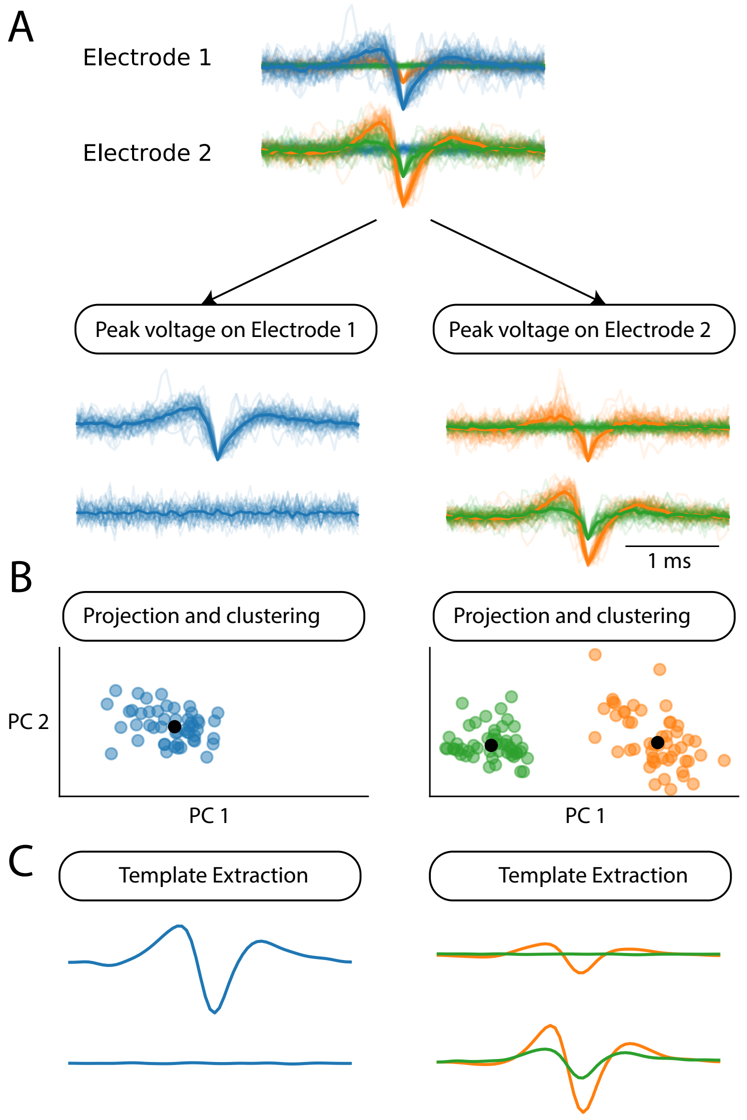
A
Electrode 1
Electrode 2
Peak voltage on Electrode 1
Peak voltage on Electrode 2
1 ms
B
Projection and clustering
Projection and clustering
PC 2
PC 1
PC 1
C
Template Extraction
Template Extraction

Fig1-fig supp1: Schematic of the parallel clustering of the spikes, in a toy example with two electrodes **A.** Pre-clustering step. The different snippets are sorted according to the electrode where they peak. This divides a set of snippets in N_{elec} groups. Each of these groups is then processed independently. **B.** Each group of snippets is projected in a low-dimension space, where clustering is performed using a density-based approach (see text and Methods). **C.** A template is extracted from each cluster and used for the template matching step.