Fast and accurate spike sorting of high-channel count probes with KiloSort

Marius Pachitariu[1], **Nick Steinmetz**[1], **Shabnam Kadir**[1]
Matteo Carandini[1] **and Kenneth Harris**[1]
[1] UCL, UK {ucgtmpa, }@ucl.ac.uk

Abstract

New silicon technology is enabling large-scale electrophysiological recordings in vivo from hundreds to thousands of channels. Interpreting these recordings requires scalable and accurate automated methods for spike sorting, which should minimize the time required for manual curation of the results. Here we introduce KiloSort, a new integrated spike sorting framework that uses template matching both during spike detection and during spike clustering. KiloSort models the electrical voltage as a sum of template waveforms triggered on the spike times, which allows overlapping spikes to be identified and resolved. Unlike previous algorithms that compress the data with PCA, KiloSort operates on the raw data which allows it to construct a more accurate model of the waveforms. Processing times are faster than in previous algorithms thanks to batch-based optimization on GPUs. We compare KiloSort to an established algorithm and show favorable performance, at much reduced processing times. A novel post-clustering merging step based on the continuity of the templates further reduced substantially the number of manual operations required on this data, for the neurons with near-zero error rates, paving the way for fully automated spike sorting of multichannel electrode recordings.

1 Introduction

The oldest and most reliable method for recording neural activity involves lowering an electrode into the brain and recording the local electrical activity around the electrode tip. Action potentials of single neurons can then be observed as a stereotypical temporal deflection of the voltage, called a spike waveform. When multiple neurons close to the electrode fire action potentials, their spikes must be identified and assigned to the correct cell, based on the features of the recorded waveforms, a process known as spike sorting [1, 2, 3, 4, 5, 6, 7]. Spike sorting is substantially helped by the ability to simultaneously measure the voltage at multiple closely-space sites in the extracellular medium. In this case, the recorded waveforms can be seen to have characteristic spatial shapes, determined by each cell's location and physiological characteristics. Together, the spatial and temporal shape of the waveform provides all the information that can be used to assign a given spike to a cell.

New high-density electrodes, currently being tested, can record from several hundred closely-spaced recording sites. Fast algorithms are necessary to quickly and accurately spike sort tens of millions of spikes coming from 100 to 1,000 cells, from recordings performed with such next-generation electrodes in awake, behaving animals. Here we present a new algorithm which provides accurate spike sorting results, with run times that scale near-linearly with the number of recording channels. The algorithm takes advantage of the computing capabilities of low-cost commercially available graphics processing units (GPUs) to enable approximately realtime spike sorting from 384-channel probes.

30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

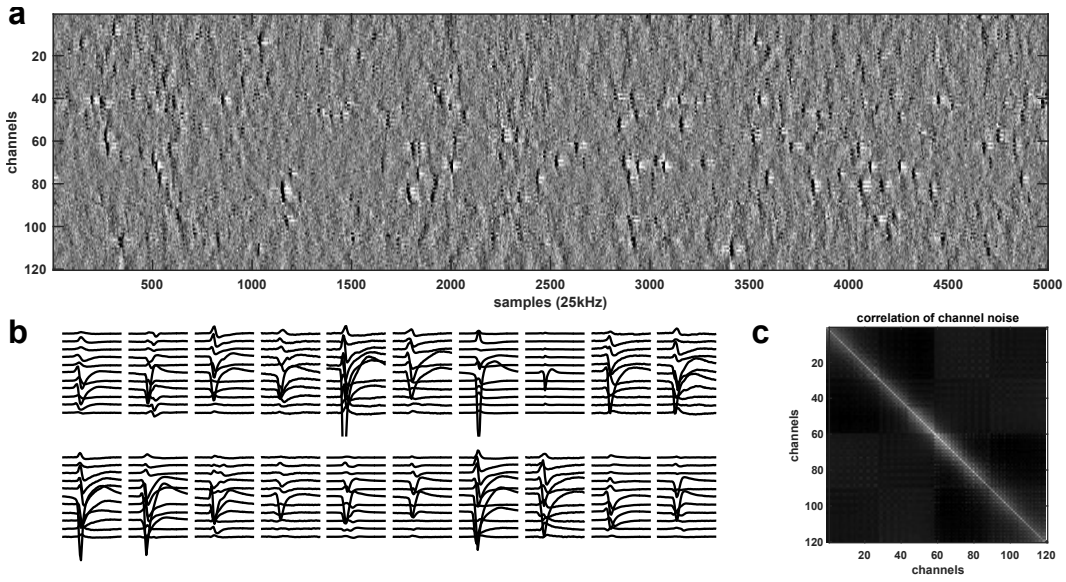

Figure 1: **Data from high-channel count recordings. a,** High-pass filtered and channel-whitened data. Negative peaks are action potentials. **b,** Example mean waveforms, centered on their peaks. **c,** Example cross-correlation matrix across channels (before whitening).

1.1 High-density electrophysiology and structured sources of noise

Next-generation high-density neural probes allow the spikes of most neurons to be recorded on 5 to 50 channels simultaneously (Fig. 1b). This provides a substantial amount of information per spike, but because other neurons also fire on the same channels, a clustering algorithm is still required to demix the signals and assign spikes to the correct cluster. Although the dense spacing of channels provides a large amount of information for each spike, structured sources of noise can still negatively impact the spike sorting problem. For example, the superimposed waveforms of neurons distant from the electrode (non-sortable units) add up and constitute a continuous random background (Fig. 1a) against which the features of sortable spikes (Fig. 1b) must be distinguished. In behaving animals, another major confound is given by the movement of the electrode relative to the tissue, which creates an apparent inverse movement of the waveform along the channels of the probe.

1.2 Previous work

A traditional approach to spike sorting divides the problem into several stages. In the first stage, spikes are detected that have maximum amplitudes above a pre-defined threshold and these spikes are projected into a common low-dimensional space, typically obtained by PCA. In the second stage, the spikes are clustered in this low-dimensional space using a variety of approaches, such as mixtures of Gaussians [8] or peak-density based approaches [9]. Some newer algorithms also include a third stage of template matching in which overlapping spikes are found in the raw data, that may have been missed in the first detection phase. Finally, a manual stage in a GUI is required for awake recordings, to manually perform merge and split operations on the imperfect automated results.

Here instead we combine these steps into a single model with a cost function based on the error of reconstructing the entire raw voltage dataset with the templates of a set of candidate neurons. We derive approximate inference and learning algorithms that can be successfully applied to very large channel count data. This approach is related to a previous study [6],

but whereas the previous work scales is impractically slow for recordings with large numbers of channels, our further modelling and algorithmic innovations have enabled the approach to be used quickly and accurately on real datasets. We improve the generative model of [] from a spiking process with continuous L1-penalized traces, to a model of spikes as discrete temporal events. The approach of [6] does not scale well to high channel count probes, as it requires the solution of a generic convex optimization problem in high dimensions.

2 Model formulation

We start with a generative model of the raw electrical voltage. Unlike previous approaches, we do not pre-commit to the times of the spikes, nor do we project the waveforms of the spikes to a lower-dimensional PCA space. Both of these steps discard potentially useful information, as we show below.

2.1 Pre-processing: common average referencing, temporal filtering and spatial whitening

To remove low-frequency fluctuations, such as the local field potential, we high-pass filter each channel of the raw data at 300 Hz. To diminish the effect of artifacts shared across all channels, we subtract at each timepoint the median of the signal across all recording sites, an operation known as common average referencing. This step is best performed after high-pass filtering, because the LFP magnitude is variable across channels but can be comparable in size to the artifacts.

Finally, we whiten the data in space to remove noise that is correlated across channels (Fig. 1c). The correlated noise is mostly due to far neurons with small spikes [10], which have a large spatial spread over the surface of the probe. Since there are very many such neurons at all recording sites, their noise averages out to have normal statistics with a stereotypical cross-correlation pattern across channels (Fig. 1c). We distinguish the noise covariance from the covariance of the large, sortable spikes, by removing the times of putative spikes (detected with a threshold criterion) from the calculation of the covariance matrix. We use a symmetrical whitening matrix that maintains the spatial structure of the data, known as ZCA, defined as $W_{\text{ZCA}} = \Sigma^{-1/2} = ED^{-1/2}E^T$, where E, D are the singular vectors and singular values of the estimated covariance matrix Σ. To regularize D, we add a small value to its diagonal. For very large channel counts, estimation of the full covariance matrix Σ is noisy, and we therefore compute the columns of the whitening matrix W_{ZCA} independently for each channel, based on its nearest 32 channels.

2.2 Modelling mean spike waveforms with SVD

When single spike waveforms are recorded across a large number of channels, most channels will have no signal and only noise. To prevent these channels from biasing the spike sorting problem, previous approaches estimate a mask over those channels with sufficient SNR to be included in a given spike. To further reduce noise and lower the dimensionality of the data for computational reasons, the spikes are usually projected into a small number of temporal principal components per channel, typically three. Here we suggest a different method for simultaneous spatial denoising/masking and for lowering the dimensionality of spikes, which is based on the observation that mean spike waveforms are very well explained by an SVD decomposition of their spatiotemporal waveform, with as few as three components (Fig. 2ab). However the spatial and temporal components of the SVD vary substantially from neuron to neuron, hence the same set of temporal basis functions per channel cannot be used to model all neurons (Fig. 2ab), as typically done in standard approaches. We analyzed the ability of the classical and proposed methods for dimensionality reduction, and found that the proposed decomposition can reconstruct waveforms with $\sim$5 times less residual variance than the classical approach. This allows it to capture small but very distinguishable features of the spikes, which ultimately can help distinguish between neurons with very similar waveforms.

2.3 Integrated template matching framework

To define a generative model of the electrical recorded voltage, we take advantage of the approximately linear additivity of electrical potentials from different sources in the extracellular medium. We combine the spike times of all neurons into a N_{spikes}-dimensional vector $\mathbf{s}$, such that the waveforms start at time samples $\mathbf{s}+1$. We define the cluster identity of spike k as $\sigma(k)$, taking values into the set $\{1, 2, 3, ..., N\}$, where N is the total number of neurons. We define the unit-norm waveform of neuron n as the matrix $K_n = U_n W_n$, of size number of channels by number of sample timepoints t_s (typically 61). The matrix K_n is defined by its low-dimensional deconstruction into three pairs of spatial and temporal basis functions, U_n and W_n, such that the norm of $U_n W_n$ is 1. The value of

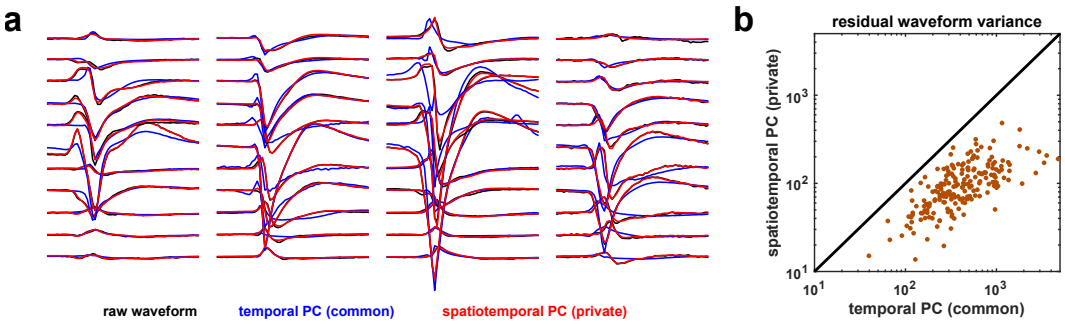

Figure 2: **Spike reconstruction from three private PCs. a,** Four example average waveforms (black) with their respective reconstruction with three common temporal PCs/channel (blue) and with reconstruction based on three spatiotemporal PCs (red), private to each spike. The red traces mostly overlap the black traces. **b,** Summary of residual waveform variance for all neurons in one dataset.

the electrical voltage at time t on channel i is defined by

$$\begin{aligned} V(i,t) &= V_0(i,t) + \mathcal{N}(0,\epsilon) \\ V_0(i,t) &= \sum_{k, s(k)<t}^{s(k)\geq t-t_s} x_k K_{\sigma(k)}\left(i, t-s(k)\right) \\ x_k &\sim \mathcal{N}\left(\mu_{\sigma(k)}, \lambda\mu^2_{\sigma(k)}\right), \end{aligned} \tag{1}$$

where $x_k > 0$ is the amplitude of spike k. Spike amplitudes in the data can vary significantly even for spikes from the same neuron, due to factors like burst adaptation and drift. We modelled the mean and variance of the amplitude variability, with the variance of the distribution scaling with the square of the mean. λ and ϵ are hyperparameters that control the relative scaling with respect to each other of the reconstruction error and the prior on the amplitude. In practice we set these constant for all recordings.

This model formulation leads to the following cost function, which we minimize with respect to the spike times, cluster assignments, amplitudes and templates

$$\mathcal{L}(\mathbf{s}, \mathbf{x}, K, \sigma) = \|V - V_0\|^2 + \frac{\epsilon}{\lambda}\sum_k \left(\frac{x_k}{\mu_{\sigma k}} - 1\right)^2 \tag{2}$$

3 Learning and inference in the model

To optimize the cost function, we alternate between finding the best spike times $\mathbf{s}$, cluster assignments σ and amplitudes $\mathbf{x}$ (template matching) and optimizing the template K parametrization with respect to $\mathbf{s}, \sigma, \mathbf{x}$ (template optimization). We initialize the templates using a simple scaled K-means clustering model, which we in turn initialize with prototypical spikes determined from the data. After the final spike times and amplitudes have been extracted, we run a final post-optimization merging algorithm which finds pairs of clusters whose spikes form a single continuous density. These steps are separately described in detail below.

3.1 Stacked initializations with scaled K-means and prototypical spikes

The density of spikes can vary substantially across the probe, depending on the location of each recording site in the brain. Initialization of the optimization in a density-dependent way can thus assign more clusters to regions that require more, relieving the main optimization from the local-minima prone problem of moving templates from one part of the probe to another. For the initialization, we thus start by detecting spikes using a threshold rule, and as we load more of the recording we keep a running subset of prototypical spikes that are sufficiently different from each other by an L2 norm criterion. We avoid overlapping spikes to be counted as prototypical spikes by enforcing

a minimum spatiotemporal peak isolation criterion on the detected spikes. Out of the prototypical spikes thus detected, we consider a fixed number N which had most matches to other spikes in the recording.

We then used this initial set of spikes to initialize a scaled K-means algorithm. This algorithm uses the same cost function described in equation 2, with spike times $\mathbf{s}$ fixed to those found by a threshold criterion. Unlike standard K-means, each spike is allowed to have variable amplitude [11].

3.2 Learning the templates via stochastic batch optimization

The main optimization re-estimates the spike times $\mathbf{s}$ at each iteration. The "online" nature of the optimization helps to accelerate the algorithm and to avoid local minima. For template optimization we use a simple running average update rule

$$A_n^{\text{new}}(i,t_0) \leftarrow (1-p)^{j_n} A_n^{\text{old}}(i,t_0) + (1-(1-p)^{j_n}) \sum_{k \in \text{batch}}^{\sigma(k)=n} V(i, s(k)+t_0), \tag{3}$$

where A_n is the running average waveform for cluster n, j_n represents the number of spikes from cluster n identified in the current batch, and the running average weighs past samples exponentially with a forgetting constant p. Thus A_n approximately represents the average of the past p samples assigned to cluster n. Note that different clusters will therefore update their mean waveforms at different rates, depending on their number of spikes per batch. Since firing rates vary over two orders of magnitude in typical recordings (from < 0.5 to 50 spikes/s), the adaptive running average procedure allows clusters with rare spikes to nonetheless average enough of their spikes to generate a smooth average template.

Like most clustering algorithms, the model we developed here is prone to non-optimal local minima. We used several techniques to ameliorate this problem. First, we annealed several parameters during learning, to encourage exploration of the parameter space, which stems from the randomness induced by the stochastic batches. We annealed the forgetting constant p from a small value (typically 20) at the beginning of the optimization to a large value at the end (typically several hundred). We also anneal from small to large the ratio ϵ/λ, which controls the relative impact of the reconstruction term and amplitude bias term in equation 2. Therefore, at the beginning of the optimization, spikes assigned to the same cluster are allowed to have more variable amplitudes. Finally, we anneal the threshold for spike detection (see below), to allow a greater mismatch between spikes and the available templates at the beginning of the optimization. As optimization progresses, the templates become more precise, and spikes increase their projections onto their preferred template, thus allowing higher thresholds to separate them from the noise.

3.3 Inferring spike times and amplitudes via template matching

The inference step of the proposed model attempts to find the best spike times, cluster assignments and amplitudes, given a set of templates $\{K_n\}_n$ with low rank-decompositions $K_n = U_n W_n$ and mean amplitudes μ_n. The templates are obtained from the running average waveform A_n, after an SVD decomposition to give $A_n \sim \mu_n K_n = \mu_n U_n W_n$, with $\|U_n W_n\| = 1$, with U_n orthonormal and W_n orthogonal. The primary roles of the low-rank representation are to guarantee fast inferences and to regularize the waveform model.

We adopt a parallelized matching pursuit algorithm to iteratively estimate the best fitting templates and subtract them off from the raw data. In standard matching pursuit, the best fitting template is identified over the entire batch, its best reconstruction is subtracted from the raw data, and then the next best fitting template is identified, iteratively until the amount of explained variance falls below a threshold, which constitutes the stopping criterion. To find the best fitting template, we estimate for each time t and each template n, the decrease in the cost function obtained by introducing template n at location t, with the best-fitting amplitude x. This is equivalent to minimizing a standard quadratic function of the form $ax^2 - 2bx + c$ over the scalar variable x, with a, $-2b$ and c derived as the coefficients of x^2, x and 1 from equation 2

$$a = 1 + \frac{\epsilon}{\lambda \mu_n^2}; \;\; b = (K_n \star V)(t) + \frac{\epsilon}{\lambda \mu_n}; \;\; c = \lambda \mu_n^2, \tag{4}$$

where $\star$ represents the operation of temporal filtering (convolution with the time-reversed filter). Here the filtering is understood as channel-wise filtering followed by a summation of all filtered traces, which computes the dot product between the template and the voltage snippet starting at each timepoint t. The decrease in cost $dC(n,t)$ that would occur if a spike of neuron n were added at time t, and the best x are given by

$$\begin{aligned} x_{\text{best}} &= \frac{b}{a} \\ dC(n,t) &= \frac{b^2}{a} - c \end{aligned} \tag{5}$$

Computing b requires filtering the data V with all the templates K_n, which amounts to a very large number of operations, particularly when the data has many channels. However, our low-rank decomposition allows us to reduce the number of operations by a factor of $N_{\text{chan}}/N_{\text{rank}}$, where N_{chan} is the number of channels (typically > 100) and N_{rank} is the rank of the decomposed template (typically 3). This follows from the observation that

$$\begin{aligned} V \star K_n &= V \star (U_n W_n) \\ &= \sum_j (U_n(:,j)^T \cdot V) \star W_n(j,:), \end{aligned} \tag{6}$$

where $U_n(:,j)$ is understood as the j-th column of matrix U_n and similarly $W_n(j,:)$ is the j-th row of W_n. We have thus replaced the matrix convolution $V \star K_n$ with a matrix product $U_n^T V$ and N_{rank} one-dimensional convolutions. We implemented the matrix products and filtering operations efficiently using consumer GPU hardware. Iterative updates of dC after template subtraction can be obtained quickly using pre-computed cross-template products, as typically done in matching pursuit []. The iterative optimization stops when a pre-defined threshold criterion on dC is larger than all elements of dC.

Due to its greedy nature, matching pursuit can have bad performance at reducing the cost function in certain problems. It is, however, appropriate to our problem, because spikes are very rare events, and overlaps are typically small, particularly in high-dimensions over the entire probe. Furthermore, typical datasets contain millions of spikes and only the simple form of matching pursuit can be efficiently employed. We implemented the simple matching pursuit formulation efficiently on consumer GPU hardware. Consider the cost improvement matrix $dC(n,t)$. When the largest element of this matrix is found and the template subtracted, no values of dC need to change except those very close in time to the fitted template (t_s samples away). Thus, instead of finding the global maximum of dC, we can find local maxima above the threshold criterion, and impose a minimal distance (t_s) between such local maxima. The identified spikes can then be processed in parallel without affecting each other's representations.

We found it unnecessary to iterate the (relatively expensive) parallel matching pursuit algorithm during the optimization of the templates. We obtained similar templates when we aborted the parallel matching pursuit after the first parallel detection step, without detecting any further overlapping spikes. To improve the efficiency of the optimization we therefore only apply the full parallel template matching algorithm on the final pass, thus obtaining the overlapping spikes.

4 Benchmarks

First, we timed the algorithm on several large scale datasets. The average run times for 32, 128 and 384 channel recordings were 10, 29 and 140 minutes respectively, on a single GPU-equipped workstation. These were significant improvements over an established framework called KlustaKwik [8], which needed approximately 480 and 10-20 thousand minutes when ran on 32 and 128 channel datasets on a standard CPU cluster (we did not attempt to run KlustaKwik on 384 channel recordings).

The significant improvements in speed could have come at the expense of accuracy losses. We compared Kilosort and Klustakwik on 32 and 128 channel recordings, using a technique known as "hybrid ground truth" [8]. To create this data, we first selected all the clusters from a recording that had been previously analysed with KlustaKwik, and curated by a human expert. For each

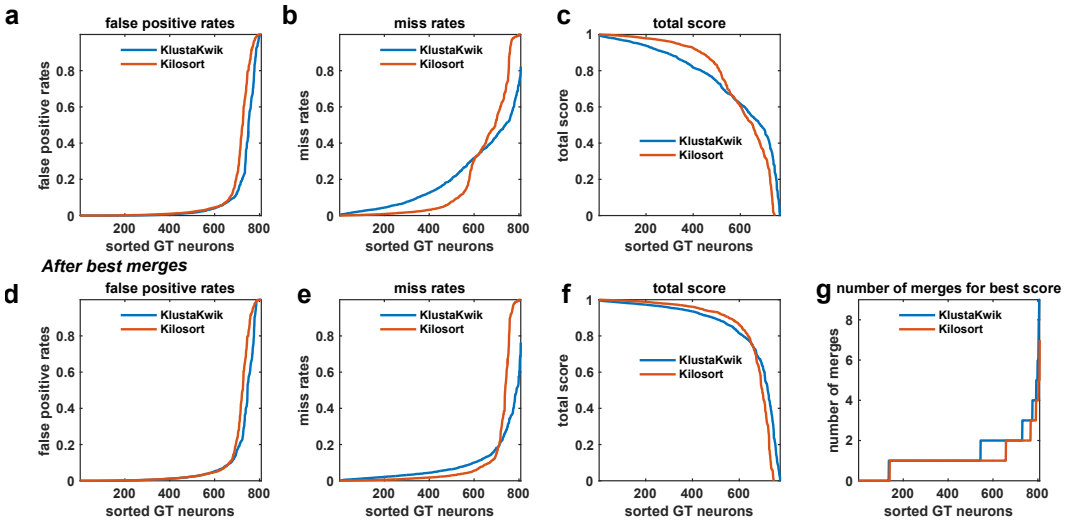

Figure 3: **Hybrid ground truth performance of proposed (KiloSort) versus established (KlustaKwik) algorithm. a,** Distribution of false positive rates. **b,** Distribution of misses. **c,** Total score. **def,** Same as (abc) after greedy best possible merges. **g,** Number of merges required to reach best score.

cluster, we extracted its raw waveform and denoised it with an SVD decomposition (keeping the top 7 dimensions of variability). We then addded the de-noised waveforms at a different but nearby spatial location on the probe with a constant channel shift, randomly chosen for each neuron. To avoid increasing the spike density at any location on the probe, we also subtracted off the denoised waveform from its original location.

Finally, we ran both KiloSort and KlustaKik on 16 instantiations of the hybrid ground truth. We matched ground truth cells with clusters identified by the algorithms to find the maximizer of the score $= 1 -$ false positive rate $-$ miss rate, where the false positive rate was normalized by the number of spikes in the test cluster, and the miss rate was normalized by the number of spikes in the ground truth cluster. Values close to 1 indicate well-sorted units. Both KiloSort and KlustaKwik performed well, with KiloSort producing significantly more cells with well-isolated clusters (53% vs 35% units with scores above 0.9).

We also estimated the best achievable score following manual sorting of the automated results. To minimize human operator work, algorithms are typically biased towards producing more clusters than can be expected in the recording, because manually merging an over-split cluster is easier, less time-consuming and, less error-prone than splitting an over-merged cluster (the latter requires choosing a carefully defined separation surface). Both KiloSort and KlustaKwik had such a bias, producing between two and four times more clusters than the expected number of neurons.

To estimate the best achievable score after operator merges, we took advantage of the ground truth data, and automatically merged together candidate clusters so as to greedily maximize their score. Final best results as well as the required number of matches are shown in Figure 3defg (KiloSort vs KlustaKwik 69% vs 60% units with scores above 0.9). The relative performance improvement of KiloSort is clearly driven by fewer misses (Fig 3e), which are likely due to its ability to detect overlapping spikes.

5 Extension: post-hoc template merging

We found that we can further reduce human operator work by performing most of the merges in an automated way. The most common oversplit clusters show remarkable continuity of their spike densities (Fig. 4). In other words, no discrimination boundary can be identified orthogonal to which the oversplit cluster appears bimodal. Instead, these clusters arise as a consequence of the algorithm partitioning clusters with large variance into multiple templates, so as to better explain their total variance. In KiloSort, we can exploit the fact that the decision boundaries between any two clusters

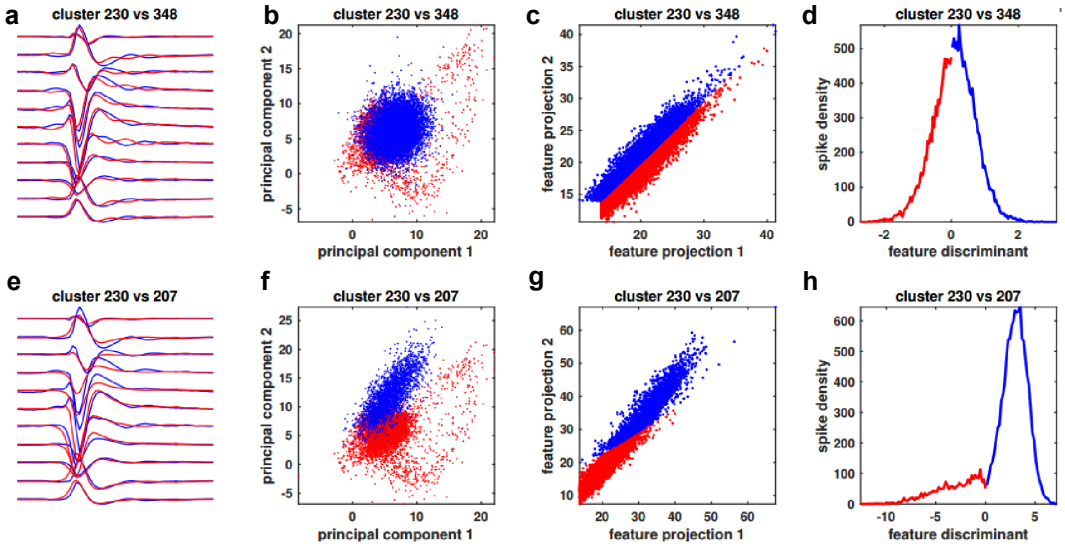

Figure 4: **PC and feature-space projections of two pairs of clusters that should be merged. ae,** Mean waveforms of merge candidates. **bf,** Spike projections into the top PCs of each candidate cluster. **cg,** Template feature projections for the templates corresponding to the candidate clusters. **dh,** Discriminant of the feature projections from (cg) (see main text for exact formula).

are in fact planes (which we show below). If two clusters belong to the same neuron, their one-dimensional projections in the space orthogonal to the decision boundary will show a continuous distribution (Fig. 4cd and 4gh), and the clusters can be merged. We use this idea to sequentially merge any two clusters with continuous distributions in their 2D feature spaces. Note that the best principal components for each cluster's main channel are much less indicative of a potential merge (Fig 4b and 4f).

To see why the decision boundaries in KiloSort are linear, consider two templates K_i and K_j and consider that we have arrived at the instance of template matching where a spike k needs to be assigned to one of these two templates. Their respective cost function improvements are $dC(i,t) = \frac{a_i^2}{b_i}$, and $dC(j,t) = \frac{a_j^2}{b_j}$, using the convention from equations 4. The decision of assigning spike k to one or the other of these templates is then equivalent to determining the sign of $dC(i,t) - dC(j,t)$, which is a linear discriminant of the feature projections

$$\text{sign}(dC(i,t) - dC(j,t)) = \text{sign}(a_i/b_i^{\frac{1}{2}} - a_j/b_j^{\frac{1}{2}}) \tag{7}$$

where b_i and b_j do not depend on the data and $a_{i,j}$ are linear functions of the raw voltage, hence the decision boundary between any two templates is linear (Fig. 4).

6 Discussion

We have demonstrated here a new framework for spike sorting of high-channel count electrophysiology data, which offers substantial accuracy and speed improvements over previous frameworks, while also reducing the amount of manual work required to isolate single units. KiloSort is currently enabling spike sorting of up to 1,000 neurons recorded simultaneously in awake animals and will help to enable the next generation of large-scale neuroscience. The code is available online at https://github.com/cortex-lab/KiloSort.

References

[1] Rodrigo Quian Quiroga. Spike sorting. *Current Biology*, 22(2):R45–R46, 2012.

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] Chaitanya Ekanadham, Daniel Tranchina, and Eero P Simoncelli. A unified framework and method for automatic neural spike identification. *Journal of neuroscience methods*, 222:47–55, 2014.

[7] Felix Franke, Robert Pröpper, Henrik Alle, Philipp Meier, Jörg RP Geiger, Klaus Obermayer, and Matthias HJ Munk. Spike sorting of synchronous spikes from local neuron ensembles. *Journal of neurophysiology*, 114(4):2535–2549, 2015.

[8] C Rossant, SN Kadir, DFM Goodman, J Schulman, MLD Hunter, AB Saleem, A Grosmark, M Belluscio, GH Denfield, AS Ecker, AS Tolias, S Solomon, G Buzsaki, M Carandini, and KD Harris. Spike sorting for large, dense electrode arrays. *Nature Neuroscience*, 19:634–641, 2016.

[9] Alex Rodriguez and Alessandro Laio. Clustering by fast search and find of density peaks. *Science*, 344(6191):1492–1496, 2014.

[10] Joana P Neto, Gonçalo Lopes, João Frazão, Joana Nogueira, Pedro Lacerda, Pedro Baião, Arno Aarts, Alexandru Andrei, Silke Musa, Elvira Fortunato, et al. Validating silicon polytrodes with paired juxtacellular recordings: method and dataset. *bioRxiv*, page 037937, 2016.

[11] Adam Coates, Andrew Y Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *International conference on artificial intelligence and statistics*, pages 215–223, 2011.