SLAy-ing oversplitting errors in high-density electrophysiology spike sorting

Sai Koukuntla[1], Tate DeWeese[1], Alexandra Cheng[1], Robyn Mildren[1,2], Aamna Lawrence[3], Austin R. Graves[1,2,3], Kathleen E. Cullen[1,2,6,7], Jennifer Colonell[4], Timothy D. Harris[1,2,4,5] and Adam S. Charles[1,2,5*]

[1]Department of Biomedical Engineering, Johns Hopkins University School of Medicine
[2]Kavli Neurodiscovery Institute, Johns Hopkins University
[3]Department of Neuroscience, Johns Hopkins University
[4]Janelia Research Campus, Howard Hughes Medical Institute
[5]Center for Imaging Science, Johns Hopkins University
[6]Department of Otolaryngology - Head and Neck Surgery, Johns Hopkins University School of Medicine
[7]Department of Neuroscience, Johns Hopkins University School of Medicine

Abstract

The growing channel count of silicon probes has substantially increased the number of neurons recorded in electrophysiology (ephys) experiments, rendering traditional manual spike sorting impractical. Instead, modern ephys recordings are processed with automated methods that use waveform template matching to isolate putative single neurons. While scalable, automated methods are subject to assumptions that often fail to account for biophysical changes in action potential waveforms, leading to systematic errors. Consequently, manual curation of these errors, which is both time-consuming and lacks reproducibility, remains necessary. To improve efficiency and reproducibility in the spike-sorting pipeline, we introduce here the Spike-sorting Lapse Amelioration System (SLAy), an algorithm that automatically merges oversplit spike clusters. SLAy employs two novel metrics: (1) a waveform similarity metric that uses a neural network to obtain spatially informed, time-shift invariant low-dimensional waveform representations, and (2) a cross-correlogram significance metric based on the earth-mover's distance between the observed and null cross-correlograms. On a diverse set of datasets with realistic simulated oversplitting, SLAy achieves high recall and near-perfect precision in identifying ground truth merges. We also demonstrate that SLAy achieves $\sim 85\%$ with human curators across a diverse set of animal models, brain regions, and probe geometries. To illustrate the impact of spike sorting errors on downstream analyses, we develop a new burst-detection algorithm and show that SLAy fixes spike sorting errors that preclude the accurate detection of bursts in neural data. SLAy leverages GPU parallelization and multithreading for computational efficiency, and is compatible with Phy and NeuroData Without Borders, making it a practical and flexible solution for large-scale ephys data analysis.

1 Introduction

Modern neuroscience strives to understand how concerted activity across distributed neural populations guides higher-order phenomena, including sensory perception and behavior. High-density electrophysiological probes such as Neuropixels [1, 2, 3] have revolutionized these efforts by enabling chronic recordings of hundreds of neurons with single-neuron precision, opening up unprecedented possibilities in modeling brain-wide population coding and dynamics [4, 5, 6]. Because ephys probes can reach deep brain regions and record at sub-millisecond resolution, they are particularly well-suited for studying neural processes that

*Corresponding author: adamsc@jhu.edu

involve rapid, coordinated activity across multiple brain regions – capabilities that comparable techniques, such as two-photon calcium imaging, lack due to limited imaging depth and slow calcium indicator kinetics. Indeed, Neuropixels probes have advanced our understanding of a wide variety of neural processes, including perception [7, 8], decision making [9, 10], and navigation [11, 12]. Extracting these valuable insights from ephys recordings, which capture the joint electrical activity of many neurons, relies on spike sorting: identifying of the timing and identity of individual neurons' action potentials from the raw electrical traces. Although ephys recordings have historically been spike sorted by hand [13], the increasing channel count of high-density electrophysiology datasets – especially those recorded using multiple probes – renders manual spike sorting impractical and necessitates automated solutions.

Automated methods assume a probe records a distinct stereotyped voltage waveform for each neuron, determined by the neuron's intrinsic biophysical properties and location relative to the probe. Early semi-automated methods involve three steps [14] (Fig. 1a, left): (1) thresholding to detect spikes, (2) dimensionality reduction, often using principal component analysis (PCA), to represent spike waveforms in a low-dimensional feature space, and (3) manual or automatic clustering in this feature space to identify putative neurons [15, 16]. While faster than fully manual clustering, these methods struggle with superimposed spikes and typically require some manual intervention.

Modern algorithms such as Kilosort [2, 17] and SpyKING CIRCUS [18] among others [19, 20], employ template matching (Fig. 1a, right). They initialize a set of waveform templates that correspond to putative neurons, then alternate between detecting spike times with a greedy algorithm (e.g. matching pursuit [21]) and refining the templates. While template matching improves on earlier methods, it remains imperfect. Since template matchers assume a static waveform for each putative neuron (with exception [22]), they fail to account for waveform changes caused by probe movement or biophysical mechanisms (e.g. short-term facilitation during bursting), both of which are commonly observed in chronic recordings from behaving animals. As a result, template matchers often 'oversplit'—or mistakenly assign a (presumed) single neuron to multiple putative single units. Overmerging errors, while also present, are much less common, since template matchers are intentionally biased towards oversplitting [17]. Although some template-matchers include post-sort merging steps [17, 18], they rely heavily on waveform similarity to suggest merges and thus are unable to account for changes in waveform shape. Therefore, even state-of-the-art spike sorting algorithms require additional merging by a manual curator. Spike sorting pipelines must overcome these common sources of waveform variance to enable the accurate study of neural activity.

Correcting template matching errors involves merging oversplit clusters and excluding low quality units that correspond to multiple ground truth neurons or noise. Failing to address these errors can result in flawed scientific conclusions; analyses of single-neuron responses depend directly on the accurate assignment of spikes to neurons, and biased splitting or merging may compromise the validity of dimensionality reduction and subsequent population-level analyses. Oversplitting in spike sorting has been shown to lead to underestimates of pairwise correlations between neurons [23], and similar errors in calcium transient attribution have been quantitatively shown to generate false correlations and misrepresent inferred coding properties [24, 25].

Although automated procedures for cluster quality labeling have been developed [26, 27], no reliable automated method exists for merging oversplit clusters. The required manual curation is both time-consuming and lacks reproducibility; curation decisions vary widely between curators due to factors including subjective judgment, fatigue, and differences in curation order [28, 16]. Growing interest in simultaneous multiprobe recordings [10, 29, 30] further exacerbates the limitations of manual curation, increasing both the time required and the potential for fatigue-related errors. Thus, automated and quantitative methods for spike sorting error correction are essential for robust and reproducible pipelines.

To address the lack of a scalable and reproducible method for correcting template matching errors, we propose the Spike-sorting Lapse Amelioration System (SLAy). We first use heuristic metrics to upper bound the frequency of oversplitting for popular spike sorters, and then apply SLAy's merging procedure and novel metrics. To obtain an objective measure of SLAy's performance, we artificially introduce oversplitting errors into real recordings and assess SLAy's ability to correct them. We demonstrate that SLAy achieves high recall and near-perfect precision in identifying ground truth merges across a variety of datasets and oversplitting types. We also demonstrate that SLAy achieves $\sim 85\%$ agreement with human curators on

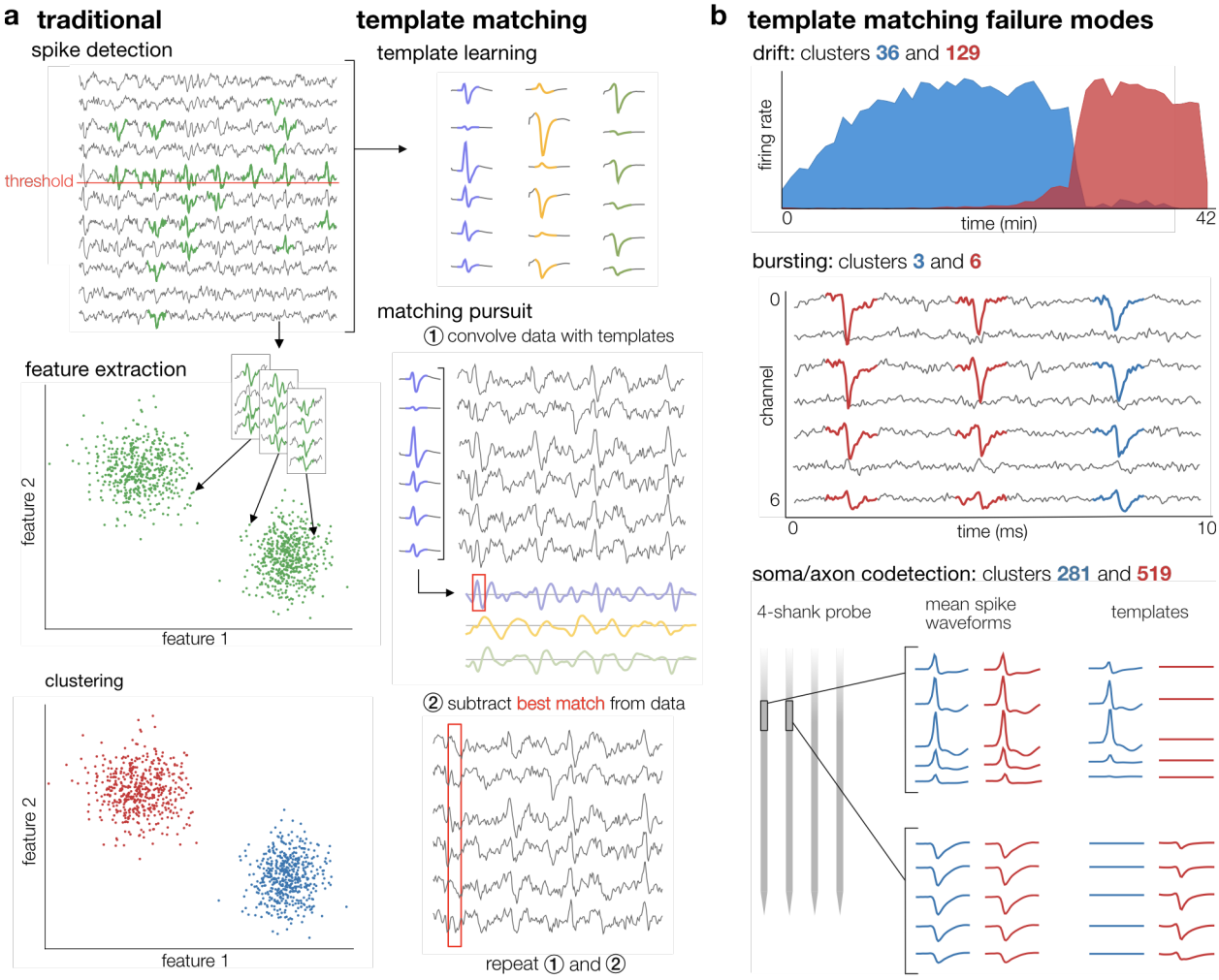

Figure 1: Spike sorting errors in high-density electrophysiology data. (a) Left: Early automatic spike sorters involve three steps – spike detection, feature extraction, and clustering. Right: Newer template matching algorithms start with a template for each putative unit and alternate between detecting spike times with matching pursuit, and refining templates. **(b)** Common failure modes for template matching algorithms include waveform changes due to drift (top), biophysical waveform changes due to bursting (middle), and co-detection of multiple parts of a single neuron (bottom).

unaltered recordings and corrects common failure modes of template matching algorithms. Separately, we develop and apply a novel burst detection algorithm before and after merging to highlight the importance of merging oversplit clusters prior to downstream analyses. By providing an automated method to correct template matching errors, SLAy stands to improve the efficiency of ephys data processing while maintaining the validity of downstream scientific conclusions.

2 Results

2.1 Spike sorting merge errors occur across brain regions, animal models, and probe geometries

We first sought to obtain an upper bound on the amount of oversplitting by using two heuristic metrics to identify potential errors across a variety of recordings. These two metrics measure complementary signatures of potential oversplitting, with the first focused on the patterns of temporal activity (spike times) and the latter focused on physical similarity (neural location and waveform similarity).

The first metric, the *presence ratio*, quantifies the fraction of coarse time bins in which a cluster is active. Active neurons rarely remain silent for tens of seconds, so a low presence ratio likely indicates missing spikes. These spikes might have been assigned to another cluster—perhaps due to waveform changes caused by drift (Fig. 1b, top)—or missed entirely. However, the presence ratio does not accurately quantify oversplitting related to bursting (Fig. 1b, middle), as burst-induced waveform changes occur on much shorter time scales than the coarse bins used to calculate presence ratio.

The second metric, the *template spatial mismatch*, aims to measure distance between centroids estimated from a cluster's template and mean waveforms. A high spatial mismatch means that the template does not capture the true amplitude distribution of the cluster's spikes across channels, suggesting improper template learning. Neurons that span many channels, due to either proximity to the probe or codetection of multiple parts of the neuron, may be split into two clusters with spatially localized templates (Fig. 1b, bottom). The example in Fig. 1b (bottom) shows the oversplitting of single neuron into two units with highly coincident spike times. Based on the waveform shapes, the two units likely correspond to dendritic spikes (blue) and axonal spikes (orange) from the same neuron [31].

We used our metrics to quantify potential oversplitting by two of the most widely used spike-sorting algorithms, Kilosort 2.5 (KS2.5) [2] and Kilosort 4 (KS4)[17], to sort four chronic recordings:

1. a macaque motor and premotor cortex recording from a *10 mm Neuropixels 1.0 probe (NP 1.0)* (data from [32]),
2. a macaque midbrain recording from a *prototype 128-channel Neuropixels NHP probe (NP NHP 128-channel)*,
3. a mouse hippocampus recording from a *Neuropixels 2.0 probe (NP 2.0)*, and
4. a rat insula recording from a *Neuropixels 2.0 probe*.

In most of the recordings analyzed, the majority of clusters exhibited high presence ratios, with only a few potentially oversplit clusters falling below 0.7 (Fig. 2, left). However, in the monkey midbrain recording, a substantial fraction ($> 50\%$) of clusters showed low presence ratios. This outcome is not surprising given the narrow action potentials of midbrain neurons, especially in primates, make it more difficult to distinguish spikes from noise.

All recordings analyzed contained a few clusters with template spatial mismatch up to 100 μm (5$\times$ the vertical site spacing on Neuropixels 1.0 probes), but a substantial fraction of clusters ($> 5\%$) in monkey motor cortex and rat insula had high spatial mismatches of up to 1000 μm. Such large mismatches are likely due to the co-detection an action potential within different compartments of the same neuron, such as the cell body, dendritic tuft, or axon terminal, at different locations on the probe. Nonetheless, this is a sign of oversplitting; these spikes belong to the same ground truth neuron and must be highly coincident in time to co-occur in the mean waveform, but the template matching algorithms split them into separate clusters.

Notably, Kilosort 2.5 and Kilosort 4 produced similar distributions for both the presence ratio and spatial mismatch metrics across all recordings. This suggests the algorithmic changes in Kilosort 4 do not substantially reduce errors related to waveform changes or co-detection of different parts of the same neuron.

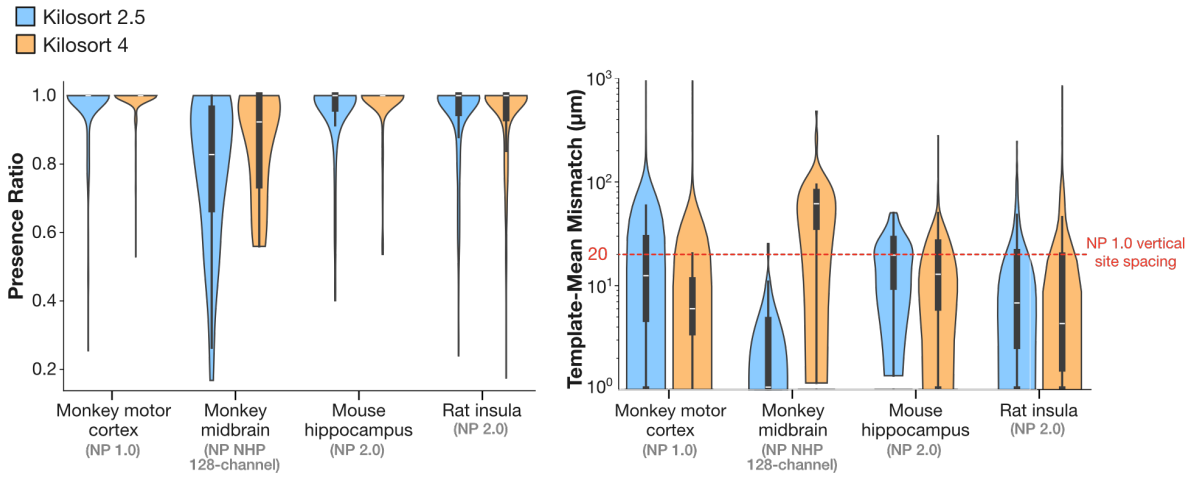

Figure 2: Heuristics quantify potential oversplitting for a variety of recordings. We used two heuristics, *presence ratio* and *template-mean* spatial mismatch to estimate an upper bound on oversplitting errors for 4 chronic Neuropixels recordings. Recordings were obtained from a variety of animal models, brain regions, and probe geometries. **(left)** A low presence ratio, the fraction of time a cluster is active, suggests undetected spikes or oversplitting due to drift. The monkey midbrain recording we analyzed appears to have a substantial number of these errors. **(right)** A high spatial mismatch between the estimated centroid of a cluster's template and mean waveform suggests improper template learning, which may signify oversplitting. All recordings we analyzed had some degree of template-mean mismatch, but the monkey motor cortex and rat insula recordings had the most, presumably due to codetection of soma and axons at different locations on the probe. Heuristics are somewhat consistent between Kilosort 2.5 and Kilosort 4, suggesting that Kilosort 4 suffers from similar errors as Kilosort 2.5.

2.2 Quantifying waveform similarity, spike time dependence, and refractory period violations

The presence ratio and template mismatch metrics serve as diagnostics, estimating an upper bound on the amount of oversplitting errors. Correcting oversplitting requires an additional step to identify clusters that should be merged. Currently, merging decisions are made by human curators using graphical interfaces [33, 18] to qualitatively assess the waveform similarity and cross-correlogram shape for each pair of clusters [34]. Our algorithm, SLAy, automates this time-consuming and subjective manual curation process through a combination of metrics that quantitatively capture the same criteria human curators use to make merging decisions: waveform similarity, cross-correlogram shapes, and refractory period violations.

2.2.1 Waveform Similarity

One of the primary identifiers of neural identity is the unit's action potential waveform. When deciding whether to merge two units, two measures of cluster waveform similarity are typically considered. The first metric is the correlation, or, equivalently, the normalized inner product between template waveforms. Correlation provides a first-order measure of shape similarity and is used by spike sorting algorithms during post-sorting cluster-merging steps [17], and by human curators to prioritize cluster pairs to inspect manually. However, template similarity is only reliable when the learned templates accurately capture the spatiotemporal shape of spike waveforms. As the cluster assessment above demonstrates, popular spike sorters have nontrivial rates of inaccurate templates (Fig. 2, right). Furthermore, Phy, the most widely used GUI for manual curation, uses templates to determine which channels to display for each cluster [33]. Thus, improperly learned templates and the resulting shape correlations can cause even diligent human curators to miss potential merges.

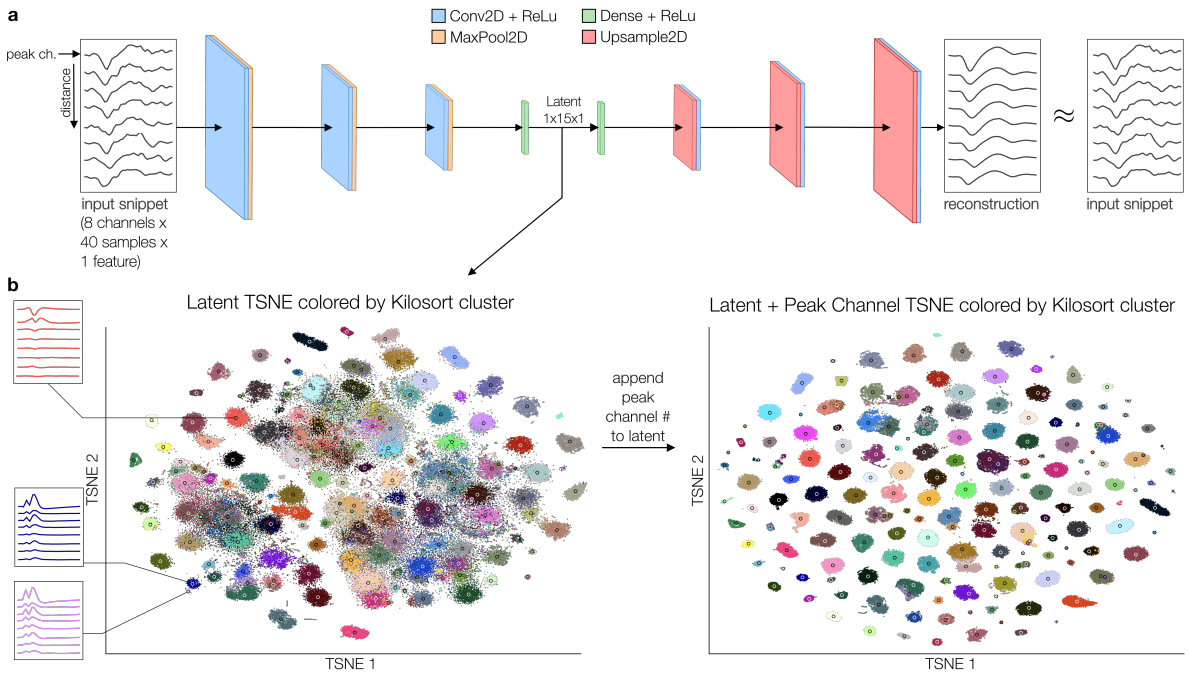

Figure 3: Autoencoder-based low-dimensional representations of spike waveforms (a) We use a convolutional autoencoder architecture to obtain a low-dimensional, time-shift invariant feature space in which to assess waveform similarity. **(b)** Left: The autoencoder latent space shows three large groups of clusters when using only waveform shape information, indicating relatively low waveform shape diversity. Right: Including information about the location of clusters on the probe improves cluster isolation. Clusters correlate roughly with Kilosort cluster labels, though oversplitting is apparent in that some clusters contain data-points with multiple colors, representing multiple kilosort units.

The second method involves projecting individual spikes from each cluster into a low-dimensional feature space, typically through principal component analysis (PCA), and visually determining the degree of overlap between the distributions for pairs of clusters. However, as principal components are computed independently for each channel, the resulting low-dimensional representations often fail to capture information about the spatial decay of waveforms. Additionally, PCA projections are highly sensitive to temporal shifts of waveform peaks, and aligning individual spikes in the presence of noise is computationally expensive. Lastly, since PCA representations span a low-dimensional linear subspace of the waveform space, they tend to smooth waveforms and remove clear distinguishing features that vary nonlinearly between clusters.

To address the shortcomings of these existing measures of waveform similarity, SLAy computes waveform similarity in an expressive feature space obtained using a convolutional autoencoder (AE) (Fig. 3a). AEs can learn highly nonlinear features that represent the details of observed spike waveforms more faithfully than linear PCA approaches, thus preserving distinguishing features that can improve the accuracy of merge decisions. AEs consist of two neural networks: an encoder that generates a low-dimensional representation of its input and a decoder that reconstructs the input as accurately as possible from this low-dimensional representation. The inclusion of convolutional layers allows the AE features to capture the spatial decay of waveforms across channels and remain invariant to small time shifts in the input waveform, avoiding the shortcomings of PCA features.

To train our AE, we extract localized snippets of spike waveforms (8 channels $\times$ 40 samples) with the 8 highest amplitude channels sorted by proximity to the cluster's peak channel. Cropping in this way removes positional information about the spike, allowing the autoencoder to learn only features that are related to

the waveform shape. Limiting the snippet size rather than feeding in all probe channels also reduces the number of network parameters, thus reducing the amount of data and computation time needed for training.

Once the AE is trained, we pass each spike waveform through the encoder to obtain 15-dimensional representations and calculate the centroids of each cluster in this latent feature space. Visualizations of the latent space reveal visual waveform similarity decreases as the distance between clusters increases (Fig. 3b, left). Using shape features alone results in latent distributions that are not well isolated despite high reconstruction accuracy, suggesting low waveform diversity across neurons. Reincorporating spatial information by appending the peak channel position to the 15-dimensional latent (only for visualization purposes) results in much cleaner separation between clusters. Even in this 2D visualization of the AE latent space, oversplit or contaminated clusters are apparent (Fig. 3b, right). Furthermore, our AE achieves more accurate snippet reconstructions than PCA at the same dimensionality, highlighting the importance of nonlinear features for faithfully representing spike waveforms.

To calculate the waveform similarity between two clusters from these features, we compute a similarity metric, σ, based on the Euclidean distance between the cluster centroids in the feature space. To constrain the values of the metric to be between 0 and 1, we normalize the metric by the median nearest-neighbor distance dist_{nn}. Because clusters with similar waveforms shapes on different parts of the probe cannot correspond to the same ground truth neuron, we also apply a multiplicative penalty based on the ratios of waveform amplitude at the two peak channels (Fig. 4, top left). The overall similarity metric is computed as

$$\sigma_{ij} = \exp\left(\frac{-\|\mu_i - \mu_j\|}{\text{dist}_{nn}}\right) \times \sqrt{\text{decay}_{ij} \times \text{decay}_{ji}},$$

where i, j are the cluster identities, μ_i and μ_j are cluster centroids in the autoencoder latent space, $\text{decay}_{ij} = \left(\frac{\text{amp}_{i,\text{peak}_j}}{\text{amp}_{i,\text{peak}_i}}\right)$, and $\text{decay}_{ji} = \left(\frac{\text{amp}_{j,\text{peak}_i}}{\text{amp}_{j,\text{peak}_j}}\right)$. The resulting waveform similarity metric accounts for spatial waveform decay and nonlinearly varying waveform features while being robust to improper template learning and imprecise spike time identification.

2.2.2 Cross-correlogram Shape Metrics

Although waveform shape is the primary criterion for merging clusters, curators must also consider the characteristics of the cross-correlogram (CCG). For example, spikes over the course of a burst can have substantial waveform variability due to the complex biophysics underlying action potential generation. Template matchers often assign these spikes to separate clusters, and waveform shape variability can make waveform similarity alone insufficient for merging these clusters. Therefore, merge decisions must also account for significant peaks and dips in the CCG, which correspond to the bursting interspike interval (ISI) and refractory period, respectively. Conversely, a cluster pair with very high waveform similarity is very unlikely to belong to the same ground truth neuron if their CCG contains many spike collisions during the refractory period. To account for CCG characteristics when determining which clusters to merge, SLAy uses two metrics based on CCG shape: one that encourages merges between clusters whose CCG exhibits large peaks or troughs, and another that penalizes merges that would result in excessive refactory period violations.

No widely accepted method exists to determine whether a peak in a CCG is statistically significant, as human curators usually inspect CCGs visually to make qualititative judgments about their shapes. Human curators are advised to merge clusters only if the CCG resembles the autocorrelograms (ACGs) of the individual clusters [34]. However, this approach can overlook potential merges, as biased splitting due to bursting activity or drift-related changes in waveform shape can divide a single ground truth neuron into units with distinct ACGs (which differ from the CCG).

One less prevalent approach is to perform bin-wise significance tests between the observed CCG and a "null" CCG, with a correction for multiple comparisons [35]. While this method does not assume that template matchers oversplit in a unbiased way, the statistical dependency between adjacent bins violates the independence assumption of many correction methods, which can result in underestimates of CCG significance. Additionally, bin-wise comparisons are highly sensitive to sampling noise and the choice of bin width.

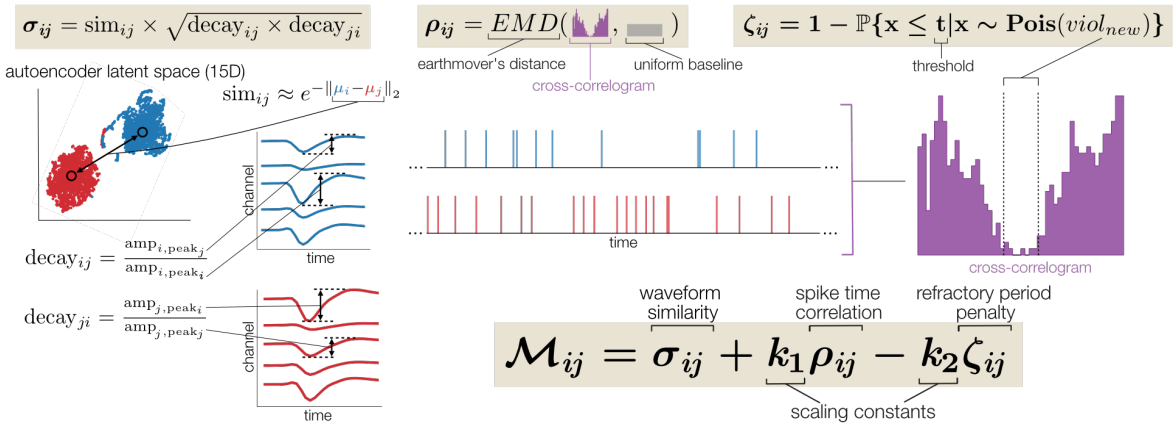

Figure 4: Metric for automatic cluster merging. We threshold a heuristic metric to determine which pairs of clusters to merge. The metric $\mathcal{M}_{ij}$ between clusters i and j consists of three terms: inter-cluster similarity (left), spike-time correlation (middle), and refractory period violation penalty (right). **Left:** Cluster similarity is defined as the distance between clusters in autoencoder latent space, scaled by the geometric mean of the waveform decay. **Middle:** Spike-time correlation is defined as the earth-mover's distance (EMD) between the spike-train cross-correlogram and a uniform distribution. **Right:** Refractory period penalty is defined as the minimum contamination probability across a range of possible refractory periods. Bottom: parameters k_1 and k_2 (user-defined) control the relative influence of these measures on the merging decisions.

To preserve CCG significance, we instead use the earthmover's distance (EMD) between the observed CCG and a uniform distribution as a measure of significance (Fig. 4, top middle). CCGs with pronounced peaks or troughs have large probability mass displacements relative to a uniform distribution and thus produce large EMDs. CCGs with random chance deviations from a uniform distribution instead yield low EMDs even if an individual bin passed a binwise significance test. However, this metric is only valid if the observed CCG resembles the "true" CCG given infinite observations. Because low-event CCGs are both unreliable estimates and far from uniform, we apply a multiplicative penalty to the metric for cluster pairs that have CCGs with low spike counts. We also considered using the distance from the observed CCG to a per-cluster-pair null CCG computed by repeatedly shuffling spike times. However, computing the null CCGs required many shuffling iterations to obtain a good estimate, increasing the computational cost without improving the CCG significance estimate. (Fig. S1).

To quantify the level of refractory period contamination, we use a modified version of the sliding refractory period (RP) violation metric, a standard metric for autocorrelogram refractory period contamination [27] (Methods). Given 1) the observed CCG, 2) a user-defined threshold of allowable contamination as a proportion of the baseline CCG event rate (see Methods), and 3) a range of possible refractory periods, the sliding RP metric calculates the probability that the "true" CCG has an unacceptable level of contamination, assuming Poisson-distributed CCG events (Fig. 4, top right). This is repeated for each potential refractory period width, with the refractory period penalty defined as the minimum contamination probability across the set of refractory periods. One drawback of this metric is that it may overestimate contamination when the absolute count of allowable contamination is low. For example, if the allowable refractory period contamination for a given CCG is 2 events and the observed CCG has 0 events in its refractory period, the sliding RP violation metric estimates that the CCG still has a ∼14% chance of being contaminated. This behavior is desired, however, since clusters should only be merged if there is statistical confidence that their CCG has a low rate of refractory period violations.

2.2.3 Merge Decisions

To determine which cluster pairs to merge together, SLAy calculates a weighted sum of the three metrics described above, summing together the waveform similarity and CCG significance metric and subtracting the refractory period penalty (Fig. 4, bottom). The scaling constants $k_1, k_2 \leq 1$ allow users to adjust the relative importance of each term. For example, in particularly bursty areas like hippocampus, a higher k_1 could correct for burst-related oversplitting, while low-burst areas like cortex could have k_1 set close to 0 to avoid false positive merges. k_2 can be increased if unit isolation is especially important to downstream analysis. By default, we set k_1 to 0.25 and k_2 to 1 since we want waveform similarity to the primary driver of merge decisions, but we want refactory period violations to strongly discourage merging.

2.3 Burst Detection Algorithm

To illustrate the importance of cluster merging, we adapted a Hidden Markov model (HMM)-based algorithm [36] to detect bursts in spike trains (Fig. 5a). The algorithm assumes that a bursting system transitions between several intrinsic states during which the system has different firing rates. Each HMM state i_k is parameterized by a Poisson rate $a_k \geq a_0 s^k$ that governs the distribution of ISIs, where a_0 is the neuron's overall average firing rate and $s > 1$ is a multiplicative scaling factor that controls how much the firing rate increases between HMM states. The "states" here are simply a modeling choice to adequately capture the statistics of the observed ISIs, and do not necessarily correspond to actual biophysical states of a neuron. We use a method from the original algorithm [36] to pre-calculate the number of HMM states based on s, the length of the spike train, and minimum observed ISI.

A period of intensity k is defined as an interval where the neuron's state is $\geq i_k$. Transitioning to less active states (i.e. lower intraburst firing rate) has no cost, while transitioning to a more active state has a cost proportional to the number of skipped states. The state sequence is inferred by minimizing a cost function that includes the transition cost and the negative log-likelihood of the observed ISI given the underlying state. The bursts are then identified by designating states above a user-defined firing rate as bursting states. We find that initalizing with $s = 3$ and designating firing rates higher than $50Hz$ work well for neuronal spike trains. The original algorithm fixes the Poisson rates at $a_k = a_0 s^k$, while we initialize the rates using s but then estimate the rates during optimization using expectation-maximization, with the constraint $a_k \geq a_0 s^k$.

To assess the performance of our burst detection algorithm, we generated simulated spike trains with a wide variety of firing statistics [37] (Fig. 5c) and compared our algorithm's performance against those of three existing methods: MaxInterval [38], logISI [39], and the original HMM algorithm [36]. On spike trains consisting of only bursts, our algorithm achieves near-perfect classification accuracy (burst vs. non-burst) regardless of the burst length or intra-burst frequency (Fig. 5b). The MaxInterval and logISI methods achieve near-perfect classification on two out of three of the simulated spike trains, while the the original Kleinberg HMM algorithm achieves near-perfect accuracy on one simulation and moderate accuracy on the other two. Our algorithm also outperforms the others on simulated spike trains containing both bursts and single spikes.

2.4 SLAy performance on simulated oversplitting

We first sought to assess the performance of SLAy on a dataset with known ground truth. However, simulated high-density ephys recordings are not biologically realistic – spike sorters regularly achieve near-perfect performance on them. Therefore, there are few errors to correct and results on these datasets are not representative of real-world performance. On the other hand, most real recordings do not have ground truth for merging decisions, since manual curation is highly subjective. To address this issue, we artificially introduced oversplitting errors into real recordings and assessed SLAy's ability to correct them. The four types of oversplitting errors we introduced were: 1) random, 2) burst-induced, 3) drift-induced, and 4) amplitude variance-induced (see Methods). We did not attempt to simulate spatial mismatch oversplitting, as Kilosort clusters are already spatially localized, and thus cannot be split into spatially separated clusters.

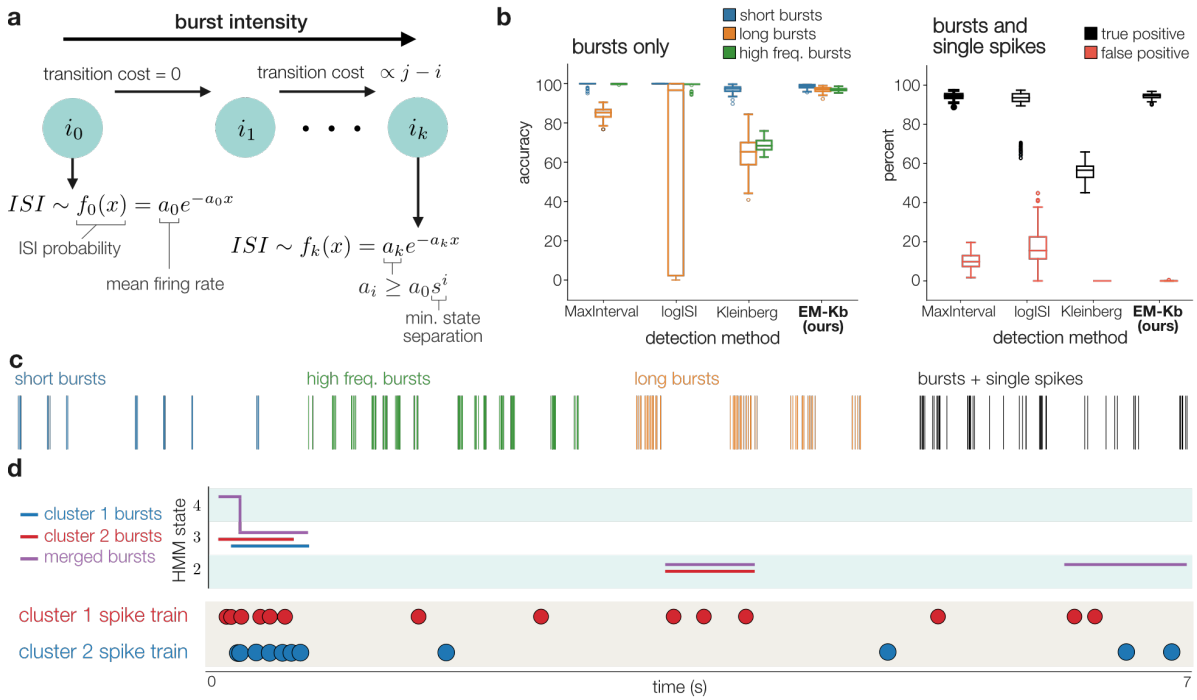

Figure 5: Robust burst detection. (a) Our burst detection algorithm relies on a Hidden Markov model where each state corresponds to a different firing rate, and adjacent states have firing rates separated by at least a scaling factor of s. Over the course of a spike train, neurons transition to higher states when bursting and lower states when firing normally. We learn both the state sequence and per-state firing rates using an expectation-maximization algorithm. **(b)** Our burst detection algorithm outperforms three existing methods on simulated spike trains with a variety of burst statistics. **(c)** Example simulated spike trains with different bursting properties. **(d)** Applying our burst detection algorithm before and after using SLAy shows that correcting oversplitting errors is necessary for accurate downstream analysis. After merging, our bursting algorithm detects additional bursts and estimates a higher intensity for the already detected bursts.

For each recording and Kilosort version in Section 2.2, we applied SLAy to the four types of simulated errors and quantified the performance of our merging algorithm by computing precision and recall. Recall is defined as the fraction of ground truth merges that were suggested by SLAy, while precision is defined as the fraction of suggested merges that were correct. Thus, if SLAy suggested many false positive merges, precision would be low, while if SLAy missed many ground truth merges, recall would be low. Across a balanced mixture of the error types, SLAy achieved a generally high recall rate and near-perfect precision across all datasets (Fig. 6a-b). Recall was highest for random and amplitude variance-induced oversplitting, while performance on the other two types was still high but more variable across datasets (Fig. S5). The majority of missed merges were due to high rates of refractory period violations in the CCGs of the clusters. This is not surprising, as SLAy's merging metric is designed to merge conservatively, sacrificing recall for precision.

2.5 SLAy performance on real neural recordings

We applied SLAy, our automatic merging algorithm, to four high-density Neuropixels recordings from different animal models, brain regions, and probe geometries to assess the validity of the suggested merges (Fig. 6). Each recording was sorted with both Kilosort 2.5 and Kilosort 4, and the mouse hippocampus recording was additionally sorted with SPYKINGCircus [18]. Using the same parameters across all four

recordings and all sorters, SLAy identified that on average ~ 10 % of the putative clusters were oversplit and needed to be merged. Although we could not reliably quantify the frequency of false negatives due to the lack of objective ground truth data, manual curators agreed with $\sim 85\%$ of our suggested merges on average. Notably, this percentage was fairly consistent between animals, brain regions, probe geometries, and spike sorters without additional tuning of thresholds or parameters for each recording.

To determine whether SLAy effectively addressed the identified subclasses of oversplitting errors, we recomputed oversplitting metrics for clusters involved in SLAy's suggested merges. Across all recordings and spike sorters, clusters involved in a merge were generally more likely to have a low presence ratio before merging than clusters not involved in a merge. Additionally, clusters involved in suggested merges were more likely to have significant spatial mismatch compared to clusters not involved in a merge (Fig. 6). We qualitatively confirmed the validity of our merges by examining t-SNE plots of the autoencoder latent space before and after merging, revealing most merges occurred between clusters that were close together in the feature space.

Macaque Cortex We first applied SLAy to premotor cortical activity recorded from an awake, behaving macaque using a Neuropixels 1.0 probe [32]. After removing noise and multi-unit activity, KS2.5 identified 130 single units, and SLAy suggested 3 merges (7 units, 5.38% of 130). All of these merges were deemed correct by an expert manual curator (100% agreement). KS4 identified 200 single units, and SLAy suggested 4 merges (8 units, 4.00% of 200), 3 of which were deemed correct (75% agreement).

To determine whether the suggested merges indeed fixed identified subclasses of oversplitting errors, we compared oversplitting metrics for clusters involved in SLAy's suggested merges to unmerged clusters. To assess whether SLAy fixed the oversplitting of bursting neurons, we used our novel burst detection algorithm to compare the bursting activity before and after merging.

On this recording, SLAy was systematically biased towards the identified subclasses of oversplits; clusters involved in suggested merges were substantially more likely to have a lower presence ratio and higher template mismatch compared to unmerged clusters. Importantly, only one merge suggested by SLAy resulted in a significant increase in number of bursts and burst intensity (average firing rate within a burst), consistent with the rarity of bursting neurons in macaque premotor cortex. We qualitatively confirmed the validity of our merges by examining TSNE plots of the autoencoder latent space before and after merging, and observed that most merges occurred between clusters that were close together in the feature space (Supp. Fig. S8).

Macaque Midbrain Next, we applied SLAy to activity recorded from macaque midbrain using a prototype 128-channel Neuropixels NHP probe chronically implanted in an awake, behaving macaque. After removing noise and multi-unit activity, KS2.5 identified 49 single units, and SLAy suggested 9 merges (18 units, 36.7% of 49). 8 of these merges were deemed correct by a manual curator (88.9% agreement). KS4 identified 36 single units, and SLAy suggested no merges.

As in the previous recording, SLAy's suggested merges for KS2.5 were biased towards low presence ratio and high template mismatch clusters. A few of the suggested merges also resulted in a substantial increase in burst number and intensity. Although neurons in macaque midbrain do not burst in the traditional sense, some neurons in midbrain have very high firing rates. This can result in high variance in waveform amplitude and/or shape due to the biophysics of synaptic transmission (e.g. vesicle depletion or short-term facilitation presynaptically, or changes in intrinsic conductances postsynaptically).

Due to changes in spike amplitude over the course of the spike train, Kilosort places the highest amplitude spikes into a separate cluster, though inspecting the spike train visually clearly suggests that the two Kilosort clusters correspond to the same ground truth neuron.

Mouse Hippocampus We next applied SLAy to activity recorded from mouse hippocampus using a Neuropixels 2.0 probe. After removing noise and multi-unit activity, KS2.5 identified 116 single units, and SLAy suggested 5 merges (10 units, 8.6% of 116). 4 of these merges were deemed correct by a manual curator (80% agreement). KS4 identified 196 single units, and our algorithm suggested 21 merges (43 units, 21.9% of 196). 19 of these merges were deemed correct by a manual curator (90.48% agreement). Similar to our

results in macaque, merges suggested by SLAy were more likely to involve clusters with low presence ratio and template mismatch for both Kilosort versions. Unlike the macaque recordings, however, more than half of our algorithm's suggested merges resulted in a substantial increase in burst intensity (Supp. Fig. S7). This is consistent with the abundance of bursting neurons in hippocampus.

Importantly, the bursting use case justifies the inclusion of cross-correlation significance in our merging metric. An oversplit bursting neuron merged by SLAy was split by KS2.5 into two clusters that have qualitatively different waveforms but a CCG with an asymmetrical peak structure characteristic of burst-associated oversplitting (Fig. S2). The two cluster waveforms are different enough the waveform similarity alone would be insufficient to result in a merge. However, this pair of clusters has a very high CCG significance that, when combined with the moderate waveform similarity value, pushes the merge decision metric over the threshold.

Rat Insula Next we applied SLAy to activity recorded from rat insula using a Neuropixels 2.0 probe during free alcohol drinking. After removal of noise clusters, KS2.5 identified 406 single-units. SLAy suggested 21 merges (42 units, 10.3 % of 406), 17 of which were deemed correct by a manual curator (81.0% agreement). KS4 identified 649 single units and SLAy suggested 43 merges (87 units, 13.4% of 649), 35 of which were deemed correct by a manual curator (81.39 % agreement). As in the hippocampus recording, SLAy's suggested merges for both Kilosort versions involved clusters more likely to have high template-mean mismatch compared to unmerged clusters. Our suggested merges resulted in a marked increase in number of detected bursts and burst intensity, with the number of bursts more than doubling post-merge for several clusters S7. This is consistent with the fact that insula, like hippocampus, contains many bursting neurons.

Merging beyond Kilosort: SpyKING CIRCUS Although Kilosort is the most widely used spike sorting algorithm, we sought to assess SLAy's performance on recordings sorted with other spike sorters. We sorted the mouse hippocampus recording from above with SpyKING CIRCUS and applied SLAy. After removal of noise clusters, the recording had 40 putative clusters.

SLAy suggested 10 merges (22 oversplit clusters, 7.14% of 308), all of which were deemed correct by a manual curator (100% agreement). Interestingly, the suggested merges for SpyKING CIRCUS were more likely to be multi-way merges (> 2 clusters) than for Kilosort. Although the suggested merges here were not systematically biased towards low presence ratio or high template mismatch clusters, this may be due to the fact the SpyKING CIRCUS clusters were very unlikely to have low presence ratio or high template mismatch to begin with. Regardless, SLAy still correctly identified merges at a similar rate to the recordings sorted with Kilosort.

3 Discussion

Automated spike sorting using template matching has become the dominant method for spike sorting high-density ephys recordings. However, despite their inclusion of post-sorting merge steps, template matchers introduce oversplitting errors that can compromise the validity of downstream analyses, for example by biasing estimates of correlation between pairs of neurons [23]. Fixing these errors is especially important when analyzing recordings that are more susceptible to oversplitting (e.g. regions with bursting neurons or recordings with probe motion). Though these errors typically require manual correction, we introduce SLAy to automatically correct these errors in a quantitative and reproducible fashion. SLAy suggests valid merges across a variety of animal models, brain regions, probe geometries, and spike sorters. Performing these merges improves the validity of downstream analyses; SLAy systematically merged oversplit bursting neurons in CA1 and insula, accurately assigning all spikes within a burst to the same neuron, resulting in increased number and intensity of detected bursts.

Although no postprocessing tools exist to automatically correct spike sorting output, simular procedures have been devised for other goals. For example, [40] uses mean waveform similarity and ACG-CCG similarity shapes by subtracting aligned versions of the waveforms and correlograms. The resulting metrics are used to merge the outputs of multiple spike sorters on a single recording with an algorithm specifically geared

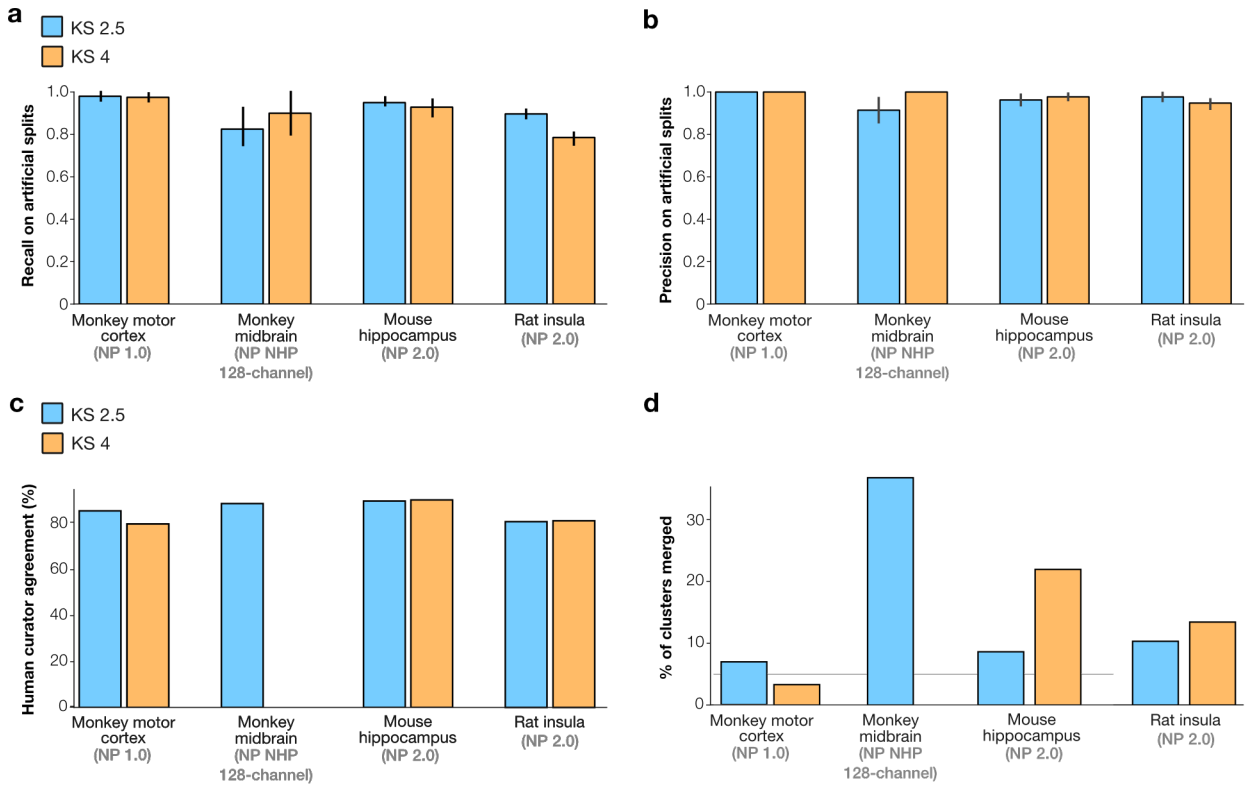

Figure 6: SLAy correctly identifies merges on simulated and real datasets. (a) On real datasets that we artificially oversplit, SLAy achieves high recall across a variety of datasets, meaninng it correctly identifies most ground truth merges. **(b)** SLAy achieves near-perfect precision across a variety of artifically split datasets, meaning it rarely suggests false positive merges. **(c)** Human curators agree with an average of $\sim$ 85% of SLAy's suggested merges across two Kilosort versions on a variety of animal models, brain regions, and probe geometries. **(d)** SLAy merges $\sim$ 10% of clusters on average, but the proportion can vary significantly between recordings.

towards detecting Purkinje cells in cerebellum. Importantly, they perform a merge only if both waveform similarity and ACG-CCG similarity are above user-defined thresholds; thus, this procedure cannot accountfor biophysical waveform variability as SLAy does. Similarly, [41] and [42] use mean waveform metrics and spatial localization of neurons among other metrics to track individual neurons across recordings spanning multiple days. These approaches, although not designed for post-sorting correction of a single spike sorter's output, share conceptual similarities with SLAy's methodology.

To our knowledge, SLAy employs the first use of a neural network to obtain a feature space to calculate spike waveform similarity. Beau et al. do use the latent space of an autoencoder to predict neuronal cell type, but only use activity on the peak channel of each waveform [43]. While not useful for identifying cell type, information about the spatial decay of recorded spikes is helpful in determining whether two putative units have similar spatiotemporal waveforms and, thus, may correspond to the same ground truth neuron. Other studies have used autoencoders to denoise ephys recordings, but these approaches are either biased or do not use the low-dimensional latents as a feature space. For example, Yet Another Spike Sorter (YASS) [19] uses an autoencoder to denoise spikewaveforms before clustering. However, because it is trained on synthetic data without realistic noise, YASS hallucinates spikes if given a snippet where no spike is present, and does not use the encoding vectors as a feature space. DeepInterpolation [44] uses a convolutional autoencoder to denoise spike recordings, but uses residual connections that allow information to bypass the low-dimensional

encoding, making it unsuitable for feature extraction. Many modern spike sorters still use per-channel principal components as spike features [17, 18]; given the gain in explained variance of autoencoder-based features over PC features, it may be worthwhile to investigate whether clustering in an autoencoder-based feature space confers any advantage.

While we show that oversplitting can bias single-neuron analyses of bursting neurons, it may be argued that population analysis techniques involving dimensionality reduction are robust to oversplitting. In this case, one might assume that split units would be sufficiently correlated to collapse together when performing dimensionality reduction. However, this assumption holds only if oversplitting is completely unbiased, which is generally not true. While our method does suggest some merges that human curators disagree with, their incidence is fairly low, and previous studies have shown that overmerging biases downstream analysis only if the merged clusters have significantly anticorrelated firing [45].

SLAy could be improved in several ways. First, we have not developed a systematic procedure to set SLAy's parameters; though we achieve consistent results across five recordings with a single set of parameters, parameter tuning may improve the validity of suggested merges further. To make the selection of the merge threshold more flexible and interpretable by humans, it may be beneficial to incorporate a semi-supervised threshold-setting scheme where the metric coefficients and merge threshold are adjusted based on a small subset of human-approved merge suggestions.

Despite these potential improvements, SLAy represents a significant step towards fully automating spike sorting curation – an critical unmet need given the growing scale of electrophysiological recordings. Our method can easily be used alongside existing postprocessing tools to yield a largely automated data processing pipeline that eliminates the significant time burden associated with manual curation.
Code: https://github.com/saikoukunt/SLAy

4 Methods

4.1 Recordings

We applied SLAy to 4 Neuropixels recordings:

- Mouse hippocampus, Neuropixels 2.0: This recording was collected by Dohoung Kim in Albert Lee's lab at the Janelia Research Campus. It was about 60 minutes long across 384 channels.
- Male Long-Evans insula, Neuropixels 1.0 during a free drinking task: This recording was 60 minutes long across 384 channels, and was collected by Aamna Lawrence in Patricia Janak's lab at Johns Hopkins University.
- Female macaque midbrain, 128-channel prototype Neuropixels NHP during vestibular stimulation (using an Intan recording system): This recording was 10 minutes long across 128 channels, and was collected by Robyn Mildren in Kathleen Cullen's lab at Johns Hopkins University.
- Macaque motor and premotor cortex, Neuropixels 1.0: This recording is a publicly available recording from Bijan Pesaran's lab [32]. It was 10 minutes long across 384 channels.

4.2 Processing and spike sorting

For the mouse hippocampus, rat insula, and macaque motor cortex recordings, preprocessing and spike sorting were performed using a custom fork of Jennifer Colonell's fork of the Allen Institute ecephys pipeline. Common average referencing, bandpass filtering, and artifact removal were performed using the CatGT module of the pipeline. Preprocessing for the macaque midbrain data was done using a custom Python script. Kilosort 2.5, Kilosort 4, and SPYKINGCircus (via SpikeInterface [46]) were all run with default parameters.

4.3 Cluster quality labeling

After sorting, we used a custom automated pipeline to automatically label clusters as single-unit, multi-unit, or noise. Clusters were marked as noise if they failed any of the following quality metric tests:

- firing rate > 0.1 Hz
- signal-to-noise ratio > 2,

The remaining clusters were marked as multi-unit if they had sliding refractory violations < 0.1, and single-unit otherwise.

4.4 Autoencoder architecture and training

Architecture The convolutional encoder contains 7 layers – 3 pairs of 3x3 2D convolutional layers + 2D 2x2 max pooling layers, and a final dense layer to produce a 15-dimensional feature vector. The number of convolutional filters doubles between each layer. The convolutional decoder also contained 7 layers, in reverse order – an initial dense layer followed by 3 pairs of 2x2 2D upsampling and 3x3 2D convolutional layers. A Rectified Linear Unit (ReLU) activation function was applied to the output of each layer except the output layer.
Training We trained a separate network for each spike sorting run (5 total). Spike snippets for autoencoder training were extracted from the preprocessed recordings, and consisted of the peak + 7 nearest channels, using a window from 10 samples before to 30 samples after the inferred spike time. We extracted a maximum of 1000 snippets per cluster, and clusters with less than 100 total spikes were ignored. Snippets were split into training and testing sets using an 80-20 split, and only the training set was used for model training. The autoencoder training procedure uses an L2 loss function, a batch size of 128, a learning rate of 0.001, and 25 epochs of training for all networks. All networks were implemented and trained using PyTorch and CUDA on an NVIDIA 3080 RTX GPU.

4.5 Waveform similarity calculation

To calculate the waveform similarities between clusters, we first calculate a spike feature vector for each spike snippet and take the mean feature vector of each Kilosort cluster as the cluster centroids. For each pair of clusters, we then calculate the unnormalized Euclidean distance between cluster centroids: $\mathbf{dist_{ae}} = ||\mu_1 - \mu_2||$ where μ_1 and μ_2 are the cluster centroids. We calibrate these distances to the traditional inner product-based waveform similarity by finding the average distance $\mathbf{dist_{ref}}$ in autoencoder space between 10 unit pairs with 0.6 inner product similarity. The waveform similarity is then calculated as

$$\mathbf{sim_{ij}} = \mathbf{e}^{-\mathbf{0.5}\,\frac{\mathbf{dist_{ae}}}{\mathbf{dist_{ref}}}}.$$

Exponentiating the negative distance yields a similarity metric between 0 and 1 rather than a distance metric, and scaling by 0.5 and $dist_{ref}$ scales the metric so that a cluster pair with $dist_{ae} = dist_{ref}$ yields a similarity of 0.6. This metric measures the shape similarity between two waveforms but does not include spatial information. To re-introduce spatial information and penalize similarity between waveforms that are similar but on different parts of the probe, we calculate a spatial decay between peak channels for each cluster:

$$\mathbf{decay_{ij}} = \frac{\mathbf{amp_{i,peak_j}}}{\mathbf{amp_{i,peak_i}}}.$$

If the peak channels are far apart, $amp_{i,peak_j}$ and $decay_{ij}$ will be close to 0. If the peak channels are the same ($amp_{i,peak_j} = amp_{i,peak_i}$), the decay metric will be 1. To calculate the final metric, we penalize the distance-based similarity using the calculated decay metrics:

$$\mathbf{waveform\ similarity} = \mathbf{sim_{ij}} \times \sqrt{\mathbf{decay_{ij}} \times \mathbf{decay_{ji}}}.$$

After calculating the waveform similarities, only cluster pairs with a similarity greater than a threshold (0.4 by default) are considered for merging, and the rest are assumed to not be oversplit. This filtering step considerably improves the runtime of SLAy by avoiding computationally expensive CCG calculations for dissimilar clusters.

4.6 Cross-correlogram significance metric

Cross-correlograms were computed using a multithreaded Python function with a window size of 2 seconds and a minimum bin width of 0.5 ms. To determine CCG significance, we first selected a window of interest by applying a trough finding algorithm to a low-pass filtered version of the second derivative of the CCG. The identified troughs in the second derivative correspond to the locations of sharp changes in the CCG, so the window of interest was taken as 1.5 times the distance from the CCG center to the most central set of troughs. If the CCG did not have a high enough event rate in the window of interest (defined by a user-set threshold), the window was expanded to the 1.5 times the subsequent trough location until the CCG had an acceptable event rate in the window of interest. We then calculated the CCG significance metric as the earthmover's distance between the windowed CCG and a uniform distribution divided by a scaling factor of 0.25. 0.25 corresponds to the unnormalized EMD between a single peak at the center of the CCG and a uniform distribution, which provides a reasonable upper bound on the EMD for a cluster pair. We thus divided by 0.25 to normalize the metric to be between 0 and 1. If the window based on the furthest set of troughs still had an event rate that was too low, we penalized the metric by the squared ratio between the observed event rate and the minimum allowed event rate. Window sizes and bin widths specified above are the default values, but all values are user-adjustable.

4.7 Refractory period violation calculation

Refractory period violations are computed using 2-second window CCGs calculated in the same manner as above. Candidate refractory periods ranged from 0.5 ms to 10 ms (user-adjustable). We first calculated the maximum allowed refractory period contamination. We estimated the baseline CCG event rate as the maximum of the low-pass filtered CCG, and took the maximum allowed event rate to be a fraction (0.25 by default) of the baseline. For each candidate refractory period, we calculated the observed number of refractory period violations and the maximum allowed number of refractory period violations. We then took the refractory period penalty to be the cumulative probability of observing up to the observed number of violations under a Poisson distribution with a mean equal to the maximum allowed violations. If the observed number of violations is small compared to the allowed violations, the penalty will be small. If the observed number of violations is large compared to the allowed violations, the penalty will be large.

4.8 Merging process

Our final merge decision metric, $\mathcal{M}_{ij}$, integrates the above metrics to determine whether to merge clusters:

$$\mathcal{M}_{\mathbf{ij}} = \sigma_{ij} + k_1\rho_{ij} - k_2\zeta_{ij},$$

where σ_{ij}, ρ_{ij}, and ζ_{ij} are the waveform similarity metric, cross-correlation significance metric, and refractory period penalty, respectively. k_1 and k_2 are coefficients ≤ 1 that control the relative importance of the metrics. To perform merges, we sort the cluster pairs by $\mathcal{M}_{ij}$ in descending order and iterate over them. If $\mathcal{M}_{ij}$ is greater than a threshold and both clusters in the pair still exist, we merge them. If $\mathcal{M}_{ij}$ is greater than the threshold but one of the clusters has already been merged into a new cluster, we have a potential multi-way merge. We perform the multi-way merge only if all pairwise merges between constituent clusters have an $\mathcal{M}_{ij}$ greater than the threshold.

Algorithm 1 SLAy merging algorithm

```
procedure RunMerge(params)
    Load spike times, cluster assignments, quality labels, and raw data
    Compute spike counts and mean waveforms for each cluster if necessary

    Generate spike snippets for training the autoencoder
    if a pretrained autoencoder model does not exist then
        Train a new convolutional autoencoder on the spike snippets
        Save the trained model
    else
        Load the existing autoencoder model
    end if
    Calculate pairwise cluster similarity (sim) using the autoencoder's latent space: sim_ij = e^(-0.5 dist_ae/dist_ref) × √(decay_ij × decay_ji).
    Identify candidate pairs with sim > sim_thresh
    Calculate cross-correlation significance (xcorr_sig) for candidate pairs
    Calculate refractory period penalty (ref_pen) for candidate pairs

    Calculate the final_metric for each pair: final_metric ← sim + k_1 × ×xcorr_sig − (k_2 × ref_pen)
    Sort the candidate pairs with final_metric above a threshold in descending order.
    for pair in list do
        if both clusters in the pair still exist then
            Merge the clusters in the pair
        else if one or more of the clusters in the pair has been merged then
            Identify the resulting merged cluster(s) and constituent clusters
            if all constituent cluster pairwise merges have final_metric above the threshold then
                Merge the resulting merged cluster(s)
            end if
        end if
    end for
end procedure
```

4.9 Burst detection

To detect bursts in spike trains, we developed the EM-Kleinberg model, an extension of a model used to identify bursts in text streams [36]. The original model is a generative hidden Markov model that assumes the system is in one of a few hidden states, each of which is associated with Poisson emissions at a certain firing rate. To avoid overfitting that usually affects HMM models of bursting, each state's firing rate is fixed at a multiple s of the previous state's. Transitioning to higher firing rate states is associated with a transition cost proportional to the number of states skipped, while transitioning to lower firing rate states has no cost. The optimal state sequence is inferred by finding the sequence that minimizes the sum of the transition costs and the negative log-likelihood of the observed interspike intervals. Periods with high firing states are designated as bursts.

While the original algorithm uses states with fixed firing rates and a single iteration to infer the optimal state sequence, we instead infer the firing rates from data using an iterative expectation-maximization (EM) process that jointly infers the firing rates and optimal state sequence. We allow each state's firing rate to be $\geq s$ times the previous state's firing rate. The first step of the EM process is similar to the optimization in the original model; it uses the forward-backward algorithm to estimate the optimal state sequence given the firing rates of each state and the observed ISIs. This step uses a maximum likelihood estimator to update the estimated firing rates of each state given the inferred state sequence. Although it is typical to run EM-algorithms until the likelihood converges, we instead minimize runtime by stopping inference after 5 iterations.

We compared the performance of our burst detection algorithm to the existing logISI and MaxInterval burst detection algorithms and the original Kleinberg model. We simulated 100 instances of each of 4 types of spike trains using the parameters and procedure specified in Cotterill et. al. The logISI and MaxInterval performance numbers were taken directly from Cotterill et. al. The Kleinberg and EM-Kleinberg algorithms were implemented in Python using parameters $s = 5, \gamma = .3$, designating states ≥ 2 as bursty states.

4.10 Simulated oversplitting experiment

For each of the four error types described below, we first identified a pool of suitable candidate units and then randomly selected a number of these candidates (*num_splits*) to be artificially split. The number of each type of split was equal, and the total number of splits was set to 15% of the total number of single units in the recording. To generate error bars for SLAy'S recall and precision, this entire procedure was repeated 5 times for each dataset.

Burst splitting To simulate oversplitting due to bursting, we first identified candidate units where at least 30% of their inter-spike intervals (ISIs) were less than 20 ms. Within these units, we defined a burst as a sequence of 3 or more spikes with ISIs under 20 ms. To ensure the resulting clusters were sufficiently large for analysis, only units where the total number of bursting spikes was greater than 600 were included as candidates. For each selected unit, we split it into two new units: the first contained all non-bursting spikes and the spikes from the first half of each burst, while the second contained only the spikes from the second half of each burst.

Amplitude splitting Earlier versions of Kilosort modeled the amplitude distribution of each unit as a Gaussian, and thus oversplit units with highly variable spike amplitudes. To model errors from high amplitude variance, we identified candidate units whose amplitude variance was between the 75th and 95th percentile of the distribution for that recording. This allowed us to target high-variance units while excluding extreme outliers that may have resulted from over-merging errors. For each selected unit, we chose a random *split_ratio* between 30% and 50% and assigned the *split_ratio* of spikes with the highest amplitudes to a new unit.

Drift splitting To simulate oversplitting due to probe drift, we constructed a time-dependent splitting probability defined by a three-part piecewise linear function. The probability of being assigned to the new cluster ramped from 0 to 0.1 over the first 40% of the recording, from 0.1 to 0.9 over the middle 20%, and from 0.9 to 1.0 over the final 40%. As candidates for this split, we selected units that were active throughout the recording, defined as having at least 30% of their spikes in each half of the recording. For each selected

unit, we iterated through its spikes and assigned a spike to the new unit if a randomly sampled number was less than the splitting probability at that spike's time.

Random splitting To simulate random oversplitting, we marked all units that passed the noise and multi-unit criteria as candidates. For each unit selected to be oversplit, we selected a random *split_ratio* of spikes between 30% and 50% and randomly assigned that percentage of spikes to a new unit.

4.11 Accuracy and Validation Metrics

Merge suggestions for each recording were validated by either S.J., A.L., or S.K. based on the guidelines in the Phy practical guide to manual curation [34]. The reported agreement percentage was calculated as the percentage of merges that were deemed valid. Because Phy does not allow comparison of clusters whose templates are not spatially overlapping, a custom plotting script was used for manual curation.

To quantify the firing rate uniformity of clusters, we coarsely bin spike trains into 20-second bins and measure the fraction of bins with greater than 1 spike.

To quantify the level of spatial template mismatch for a cluster, we calculated the distance between the centroids of the mean and template waveforms. Centroids were calculated as an amplitude-weighted average of the positions of channels where the waveform amplitude was at least 20% the peak amplitude.

To quantify the change in burst intensity and number of bursts for a suggested merge, we run the EM-Kleinberg burst detection algorithm on the constituent cluster spike trains before merging, and the merged spike train after merging. The change in burst intensity was calculated as the percentage change in average HMM state during bursts.

Code Availability

Code and documentation for SLAy is available at https://github.com/saikoukunt/SLAy

Acknowledgements

We thank Dohoung Kim, Albert Lee, Kathleen Cullen, and Patricia Janak for allowing us to use their Neuropixels recordings to test and validate SLAy, as well as helpful discussions and feedback.

SK, AC, AG, and TH were supported by NIH grant NIH U01NS115587. ASC was partially supported by NSF CAREER award 2340338. AL was supported by NIH R01AA031609 and NIH R01AA027213. RM was supported by the Kavli Neuroscience Discovery Institute Distinguished Postdoctoral Fellowship and the Natural Sciences and Engineering Postdoctoral Fellowship. This work was supported by the National Institutes of Health/National Institute on Deafness and Other Communication Disorders (R01-DC002390 and R01-DC018061) to K.E.C.

References

[1] James J. Jun et al. "Fully integrated silicon probes for high-density recording of neural activity". In: *Nature* 551.7679 (2017). Number: 7679 Publisher: Nature Publishing Group, pp. 232–236. ISSN: 1476-4687. DOI: 10.1038/nature24636. URL: https://www.nature.com/articles/nature24636/.

[2] Nicholas A. Steinmetz et al. "Neuropixels 2.0: A miniaturized high-density probe for stable, long-term brain recordings". In: *Science* 372.6539 (2021). Publisher: American Association for the Advancement of Science. DOI: 10.1126/science.abf4588. URL: https://www.science.org/doi/10.1126/science.abf4588.

[3] Eric M. Trautmann et al. *Large-scale high-density brain-wide neural recording in nonhuman primates.* May 4, 2023. DOI: 10.1101/2023.02.01.526664. URL: https://www.biorxiv.org/content/10.1101/2023.02.01.526664v3 (visited on 09/01/2023).

[4] Noga Mudrik, Gal Mishne, and Adam S Charles. "SiBBlInGS: similarity-driven building-block inference using graphs across states". In: *Proceedings of the 41st International Conference on Machine Learning*. 2024, pp. 36504–36530.

[5] Noga Mudrik et al. "CREIMBO: Cross-Regional Ensemble Interactions in Multi-view Brain Observations". In: *The Thirteenth International Conference on Learning Representations*. 2025.

[6] Noga Mudrik et al. "Decomposed linear dynamical systems (dlds) for learning the latent components of neural dynamics". In: *Journal of Machine Learning Research* 25.59 (2024), pp. 1–44.

[7] Carsen Stringer et al. "Spontaneous behaviors drive multidimensional, brainwide activity". In: *Science* 364.6437 (2019). Publisher: American Association for the Advancement of Science, eaav7893. DOI: 10.1126/science.aav7893. URL: https://www.science.org/doi/full/10.1126/science.aav7893.

[8] Matthew K. Leonard et al. "Large-scale single-neuron speech sound encoding across the depth of human cortex". In: *Nature* (Dec. 13, 2023). Publisher: Nature Publishing Group, pp. 1–10. ISSN: 1476-4687. DOI: 10.1038/s41586-023-06839-2. URL: https://www.nature.com/articles/s41586-023-06839-2 (visited on 12/13/2023).

[9] Nicholas A. Steinmetz et al. "Distributed coding of choice, action and engagement across the mouse brain". In: *Nature* 576.7786 (2019). Number: 7786 Publisher: Nature Publishing Group, pp. 266–273. ISSN: 1476-4687. DOI: 10.1038/s41586-019-1787-x. URL: https://www.nature.com/articles/s41586-019-1787-x.

[10] Adrian G. Bondy et al. *Coordinated cross-brain activity during accumulation of sensory evidence and decision commitment*. Pages: 2024.08.21.609044 Section: New Results. Aug. 25, 2024. DOI: 10.1101/2024.08.21.609044. URL: https://www.biorxiv.org/content/10.1101/2024.08.21.609044v3 (visited on 01/13/2025).

[11] Richard J. Gardner et al. "Toroidal topology of population activity in grid cells". In: *Nature* 602.7895 (2022). Number: 7895 Publisher: Nature Publishing Group, pp. 123–128. ISSN: 1476-4687. DOI: 10.1038/s41586-021-04268-7. URL: https://www.nature.com/articles/s41586-021-04268-7.

[12] Isabel I. C. Low et al. "Dynamic and reversible remapping of network representations in an unchanging environment". In: *Neuron* 109.18 (Sept. 15, 2021). Publisher: Elsevier, 2967–2980.e11. ISSN: 0896-6273. DOI: 10.1016/j.neuron.2021.07.005. URL: https://www.cell.com/neuron/abstract/S0896-6273(21)00504-3 (visited on 01/13/2025).

[13] Charles M. Gray et al. "Tetrodes markedly improve the reliability and yield of multiple single-unit isolation from multi-unit recordings in cat striate cortex". In: *Journal of Neuroscience Methods* 63.1 (Dec. 1, 1995), pp. 43–54. ISSN: 0165-0270. DOI: 10.1016/0165-0270(95)00085-2. URL: https://www.sciencedirect.com/science/article/pii/0165027095000852 (visited on 01/13/2025).

[14] Kenneth D. Harris et al. "Accuracy of Tetrode Spike Separation as Determined by Simultaneous Intracellular and Extracellular Measurements". In: *Journal of Neurophysiology* 84.1 (2000). Publisher: American Physiological Society, pp. 401–414. ISSN: 0022-3077. DOI: 10.1152/jn.2000.84.1.401. URL: https://journals.physiology.org/doi/full/10.1152/jn.2000.84.1.401.

[15] Alex Rodriguez and Alessandro Laio. "Machine learning. Clustering by fast search and find of density peaks". In: *Science (New York, N.Y.)* 344.6191 (June 27, 2014), pp. 1492–1496. ISSN: 1095-9203. DOI: 10.1126/science.1242072.

[16] Cyrille Rossant et al. "Spike sorting for large, dense electrode arrays". In: *Nature Neuroscience* 19.4 (2016). Number: 4 Publisher: Nature Publishing Group, pp. 634–641. ISSN: 1546-1726. DOI: 10.1038/nn.4268. URL: https://www.nature.com/articles/nn.4268.

[17] Marius Pachitariu, Shashwat Sridhar, and Carsen Stringer. *Solving the spike sorting problem with Kilosort*. Pages: 2023.01.07.523036 Section: New Results. Jan. 7, 2023. DOI: 10.1101/2023.01.07.523036. URL: https://www.biorxiv.org/content/10.1101/2023.01.07.523036v1 (visited on 09/01/2023).

[18] Pierre Yger et al. "A spike sorting toolbox for up to thousands of electrodes validated with ground truth recordings in vitro and in vivo". In: *eLife* 7 (2018). Ed. by David Kleinfeld. Publisher: eLife Sciences Publications, Ltd, e34518. ISSN: 2050-084X. DOI: 10.7554/eLife.34518. URL: https://doi.org/10.7554/eLife.34518.

[19] JinHyung Lee et al. *YASS: Yet Another Spike Sorter applied to large-scale multi-electrode array recordings in primate retina*. Mar. 20, 2020. DOI: 10.1101/2020.03.18.997924. URL: https://www.biorxiv.org/content/10.1101/2020.03.18.997924v1 (visited on 01/13/2025).

[20] *tridesclous/tridesclous*. original-date: 2015-11-10T14:06:13Z. Nov. 18, 2024. URL: https://github.com/tridesclous/tridesclous (visited on 01/13/2025).

[21] S.G. Mallat and Zhifeng Zhang. "Matching pursuits with time-frequency dictionaries". In: *IEEE Transactions on Signal Processing* 41.12 (Dec. 1993). Conference Name: IEEE Transactions on Signal Processing, pp. 3397–3415. ISSN: 1941-0476. DOI: 10.1109/78.258082. URL: https://ieeexplore.ieee.org/document/258082 (visited on 01/13/2025).

[22] Julien Boussard et al. *DARTsort: A modular drift tracking spike sorter for high-density multi-electrode probes*. Aug. 14, 2023. DOI: 10.1101/2023.08.11.553023. URL: https://www.biorxiv.org/content/10.1101/2023.08.11.553023v1 (visited on 01/13/2025).

[23] Marlene R. Cohen and Adam Kohn. "Measuring and interpreting neuronal correlations". In: *Nature neuroscience* 14.7 (June 27, 2011), pp. 811–819. ISSN: 1097-6256. DOI: 10.1038/nn.2842. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3586814/ (visited on 03/18/2025).

[24] Jeffrey L. Gauthier et al. "Detecting and correcting false transients in calcium imaging". In: *Nature Methods* 19.4 (2022). Number: 4 Publisher: Nature Publishing Group, pp. 470–478. ISSN: 1548-7105. DOI: 10.1038/s41592-022-01422-5. URL: https://www.nature.com/articles/s41592-022-01422-5.

[25] Hakan Inan, Murat A Erdogdu, and Mark Schnitzer. "Robust Estimation of Neural Signals in Calcium Imaging". In: *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc., 2017. URL: https://papers.nips.cc/paper/2017/hash/e449b9317dad920c0dd5ad0a2a2d5e49-Abstract.html (visited on 01/13/2025).

[26] Julie M. J. Fabre et al. *Bombcell: automated curation and cell classification of spike-sorted electrophysiology data*. Version 1.0.0. July 21, 2023. DOI: 10.5281/zenodo.8172822. URL: https://zenodo.org/records/8172822 (visited on 01/13/2025).

[27] *Spike sorting pipeline for the International Brain Laboratory*. figshare. May 4, 2022. DOI: 10.6084/m9.figshare.19705522.v4. URL: https://figshare.com/articles/online_resource/Spike_sorting_pipeline_for_the_International_Brain_Laboratory/19705522/4 (visited on 01/13/2025).

[28] F. Wood et al. "On the variability of manual spike sorting". In: *IEEE Transactions on Biomedical Engineering* 51.6 (2004). Conference Name: IEEE Transactions on Biomedical Engineering, pp. 912–918. ISSN: 1558-2531. DOI: 10.1109/TBME.2004.826677.

[29] Susu Chen et al. *Brain-wide neural activity underlying memory-guided movement*. 2023. DOI: 10.1101/2023.03.01.530520. URL: https://www.biorxiv.org/content/10.1101/2023.03.01.530520v1.

[30] Maxwell D. Melin et al. *Large scale, simultaneous, chronic neural recordings from multiple brain areas*. June 25, 2024. DOI: 10.1101/2023.12.22.572441. URL: https://www.biorxiv.org/content/10.1101/2023.12.22.572441v2 (visited on 01/13/2025).

[31] Shirly Someck et al. "Positive and biphasic extracellular waveforms correspond to return currents and axonal spikes". In: *Communications Biology* 6.1 (Sept. 18, 2023). Publisher: Nature Publishing Group, pp. 1–15. ISSN: 2399-3642. DOI: 10.1038/s42003-023-05328-6. URL: https://www.nature.com/articles/s42003-023-05328-6 (visited on 06/09/2025).

[32] John Choi et al. "Optimal Adaptive Electrode Selection to Maximize Simultaneously Recorded Neuron Yield". In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., 2020, pp. 6160–6171. URL: https://proceedings.neurips.cc/paper/2020/hash/445e1050156c6ae8c082a8422bb7dfc0-Abstract.html (visited on 02/17/2025).

[33] *cortex-lab/phy*. original-date: 2014-07-30T12:26:51Z. Jan. 7, 2025. URL: https://github.com/cortex-lab/phy (visited on 01/14/2025).

[34] Stephen Lenzi and Nicholas Steinmetz. *Manual Clustering Practical Guide (by S. Lenzi and N. Steinmetz) - phy*. URL: https://phy.readthedocs.io/en/latest/sorting_user_guide/#analysis (visited on 01/14/2025).

[35] Kyle Kirkland. *Crosscorrelation*. URL: https://www.med.upenn.edu/mulab/crosscorrelation.html (visited on 02/21/2025).

[36] Jon Kleinberg. "Bursty and Hierarchical Structure in Streams". In: (2004), p. 25.

[37] Ellese Cotterill et al. "A comparison of computational methods for detecting bursts in neuronal spike trains and their application to human stem cell-derived neuronal networks". In: *Journal of Neurophysiology* 116.2 (Aug. 1, 2016), pp. 306–321. ISSN: 0022-3077. DOI: 10.1152/jn.00093.2016. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4969396/ (visited on 02/27/2023).

[38] Alexander Kirillov. "NeuroExplorer Manual". In: ().

[39] Valentina Pasquale, Sergio Martinoia, and Michela Chiappalone. "A self-adapting approach for the detection of bursts and network bursts in neuronal cultures". In: *Journal of Computational Neuroscience* 29.1 (Aug. 2010), pp. 213–229. ISSN: 1573-6873. DOI: 10.1007/s10827-009-0175-1.

[40] Victor Llobet, Aurélien Wyngaard, and Boris Barbour. *Automatic post-processing and merging of multiple spike-sorting analyses with Lussac*. Pages: 2022.02.08.479192 Section: New Results. Feb. 10, 2022. DOI: 10.1101/2022.02.08.479192. URL: https://www.biorxiv.org/content/10.1101/2022.02.08.479192v1 (visited on 01/13/2025).

[41] Augustine Xiaoran Yuan et al. "Multi-day neuron tracking in high-density electrophysiology recordings using earth mover's distance". In: *eLife* 12 (July 10, 2024). Ed. by Adrien Peyrache and Panayiota Poirazi. Publisher: eLife Sciences Publications, Ltd, RP92495. ISSN: 2050-084X. DOI: 10.7554/eLife.92495. URL: https://doi.org/10.7554/eLife.92495 (visited on 01/14/2025).

[42] Enny H. van Beest et al. "Tracking neurons across days with high-density probes". In: *Nature Methods* (Sept. 27, 2024). Publisher: Nature Publishing Group, pp. 1–10. ISSN: 1548-7105. DOI: 10.1038/s41592-024-02440-1. URL: https://www.nature.com/articles/s41592-024-02440-1 (visited on 01/14/2025).

[43] Maxime Beau et al. "A deep learning strategy to identify cell types across species from high-density extracellular recordings". In: *Cell* (Feb. 28, 2025). ISSN: 0092-8674. DOI: 10.1016/j.cell.2025.01.041. URL: https://www.sciencedirect.com/science/article/pii/S0092867425001102 (visited on 03/07/2025).

[44] Jérôme Lecoq et al. "Removing independent noise in systems neuroscience data using DeepInterpolation". In: *Nature Methods* 18.11 (Nov. 2021). Publisher: Nature Publishing Group, pp. 1401–1408. ISSN: 1548-7105. DOI: 10.1038/s41592-021-01285-2. URL: https://www.nature.com/articles/s41592-021-01285-2 (visited on 01/13/2025).

[45] "Accurate Estimation of Neural Population Dynamics without Spike Sorting". In: *Neuron* 103.2 (July 17, 2019). Publisher: Elsevier, 292–308.e4. ISSN: 0896-6273. DOI: 10.1016/j.neuron.2019.05.003. URL: https://www.cell.com/neuron/abstract/S0896-6273(19)30428-3 (visited on 01/14/2025).

[46] Alessio P. Buccino et al. "SpikeInterface, a unified framework for spike sorting". In: *eLife* 9 (Nov. 10, 2020), e61834. ISSN: 2050-084X. DOI: 10.7554/eLife.61834.

Supplement

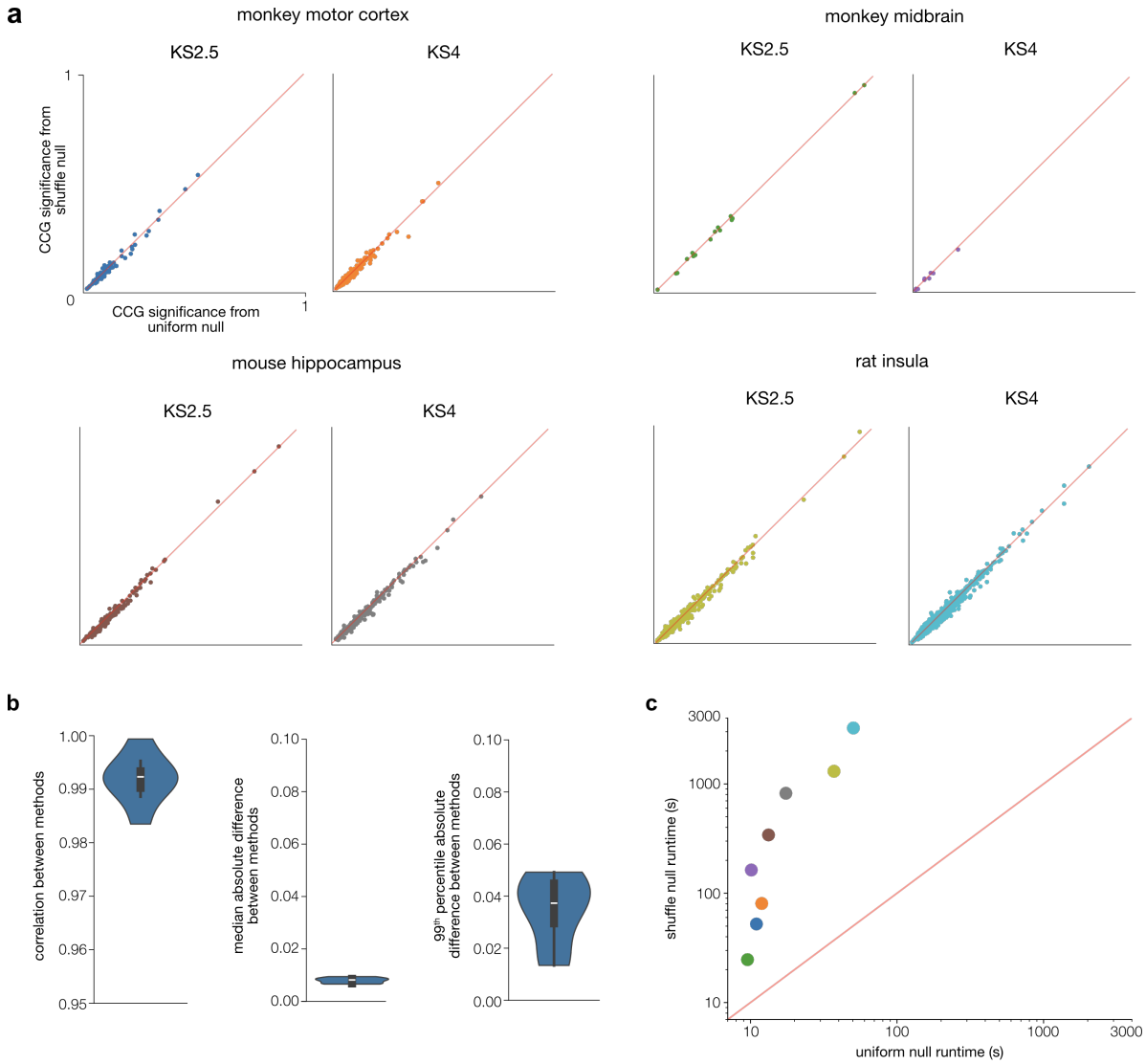

Figure S1: Uniform null vs. per-cluster shuffle null for CCG significance metric. Using a uniform distribution as the null produces highly similar metric values while being much faster than a shuffle distribution. **(a)** Both methods of calculating null CCGs produce similar metrics across all recordings. Each colored dot represents a candidate cluster pair for merging, and the red line represents $y = x$. **(b)** Mean correlation across recordings between null methods is high (0.992) and mean absolute differences are low (~ 0.01 for median abs. difference, ~ 0.04 for 99th percentile.) Note that the CCG significance metric is further multipled by a constant < 1, (0.25 by default) when calculating the final merging metric, so differences at this scale will have little impact on merging decisions. **(c)** The uniform null is about an order of magnitude faster to calculate than the shuffle null. For long recordings (~ 1 hour) or recordings with many clusters (~ 1000), the uniform null takes ~ 100 s while the shuffle null takes ~ 3000 s. Each dot corresponds to one recording, and colors correspond to (a).

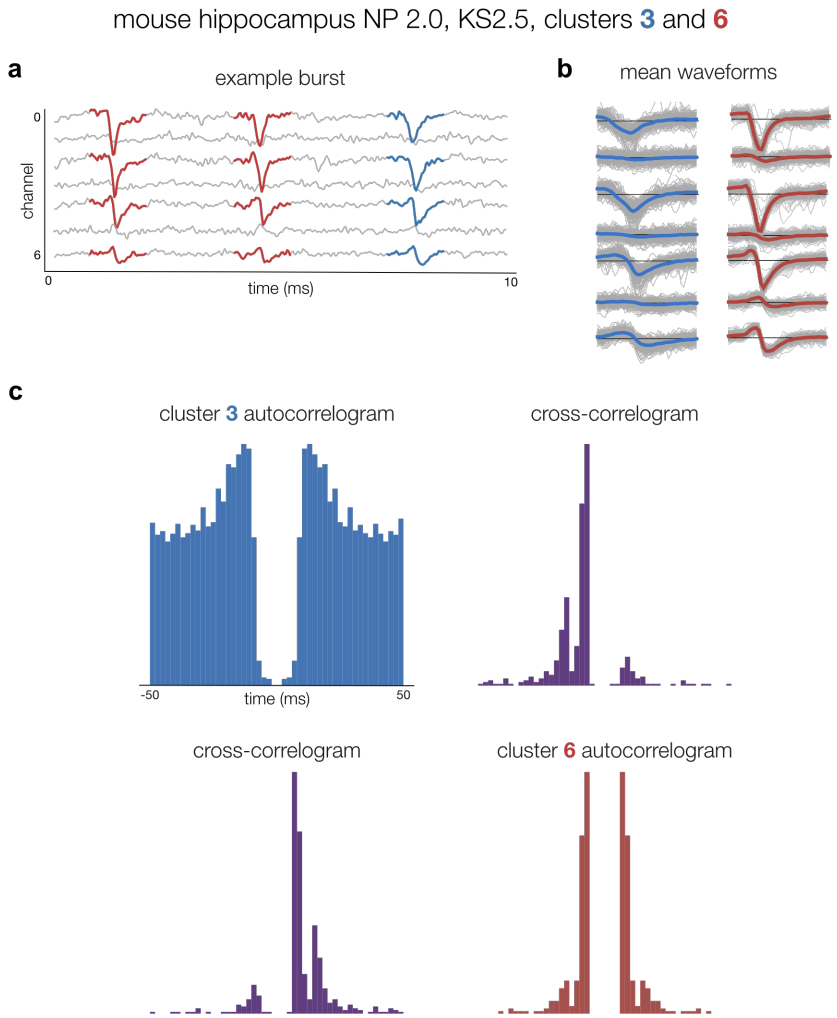

Figure S2: SLAy accounts for biophysical waveform changes. SLAy merges two clusters in the mouse hippocampus recording that were oversplit by Kilosort 2.5 due to bursting-related waveform changes. **(a)** A salient burst in the recording was split into two different clusters. The first two spikes from the recording were assigned by Kilosort to cluster 6 (red), and the last spike in the burst had a wider waveform shape, causing it to be assigned to a cluster 3 instead (blue). **(b)** The mean waveforms for the two clusters show significant differences (colored traces), though the distributions of individual spikes (thin grey traces) overlap between clusters. **(c)** The cross-correlogram has a highly peaked and asymmetric shape, characteristic of an oversplit bursting neuron.

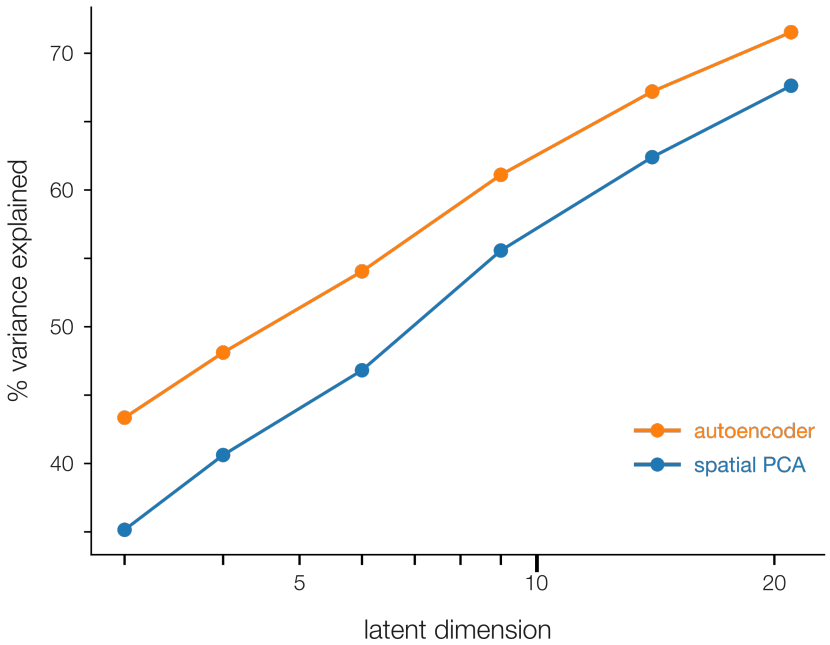

Figure S3: Autoencoder-based features explain more variance in waveforms than PCA- based features. For the same dimensionality, autoencoder-based features explain 5-10 % more variance in waveform shape than PCA-based features.

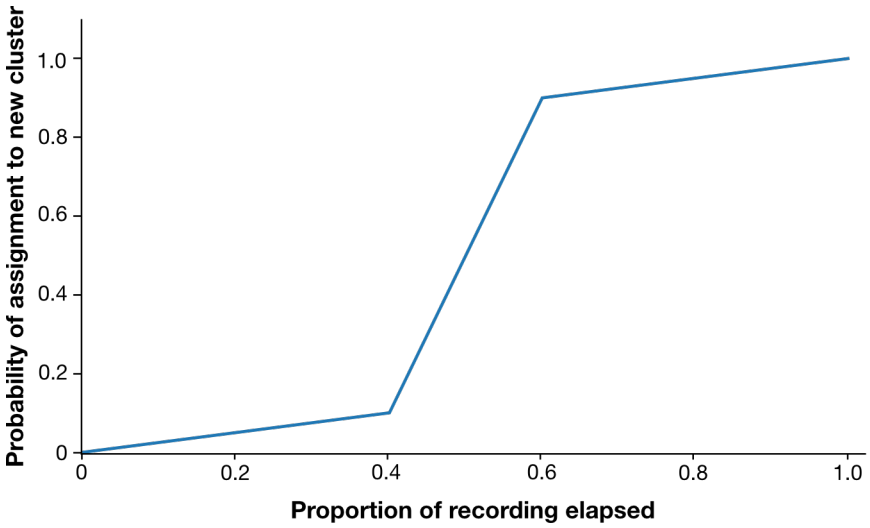

Figure S4: Split probability function for artifical drift splits. Spikes in the first 40% of the recording have a 0-10% chance of being assigned to the new cluster, spikes in the middle 20% have a 10-90% chance, and spikes in the last 40% have a 90-100% chance. This more realistically models drift-related errors, which are a gradual process rather than a sudden change in spike assignment.

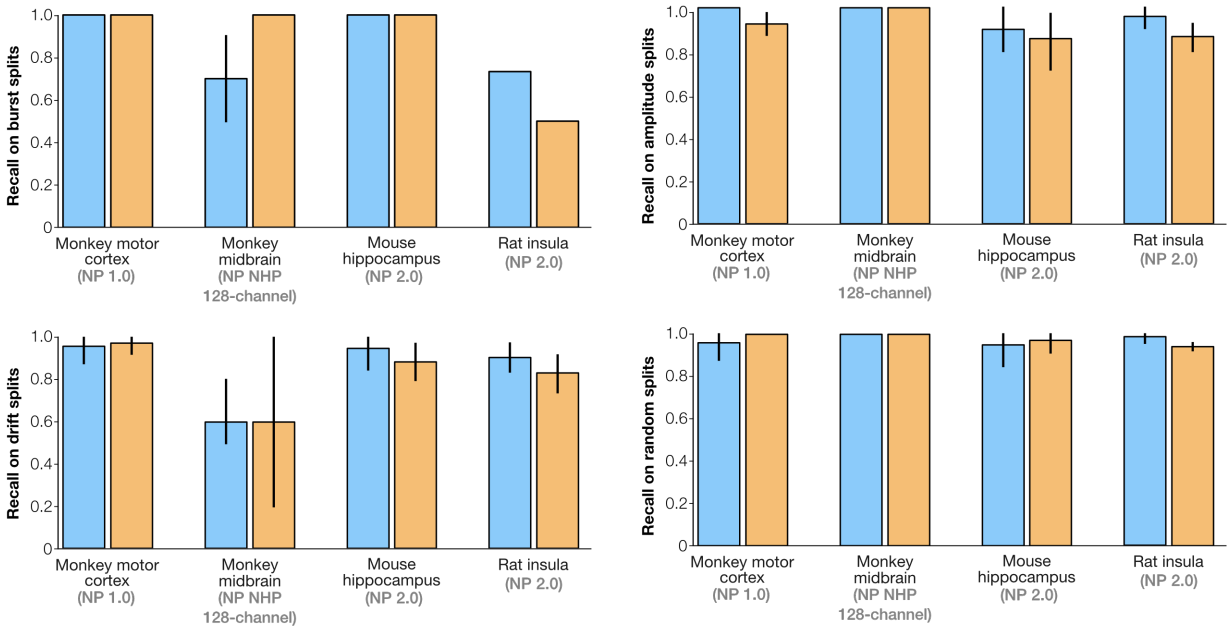

Figure S5: SLAy performance by artificial split type. SLAy generally achieves high recall for all types of artificial splits, but performance on burst splits and drift splits varies more across recording types. Note that the error bars for monkey midbrain are large since only 2 splits of each type were created for this recording due to low total unit count.

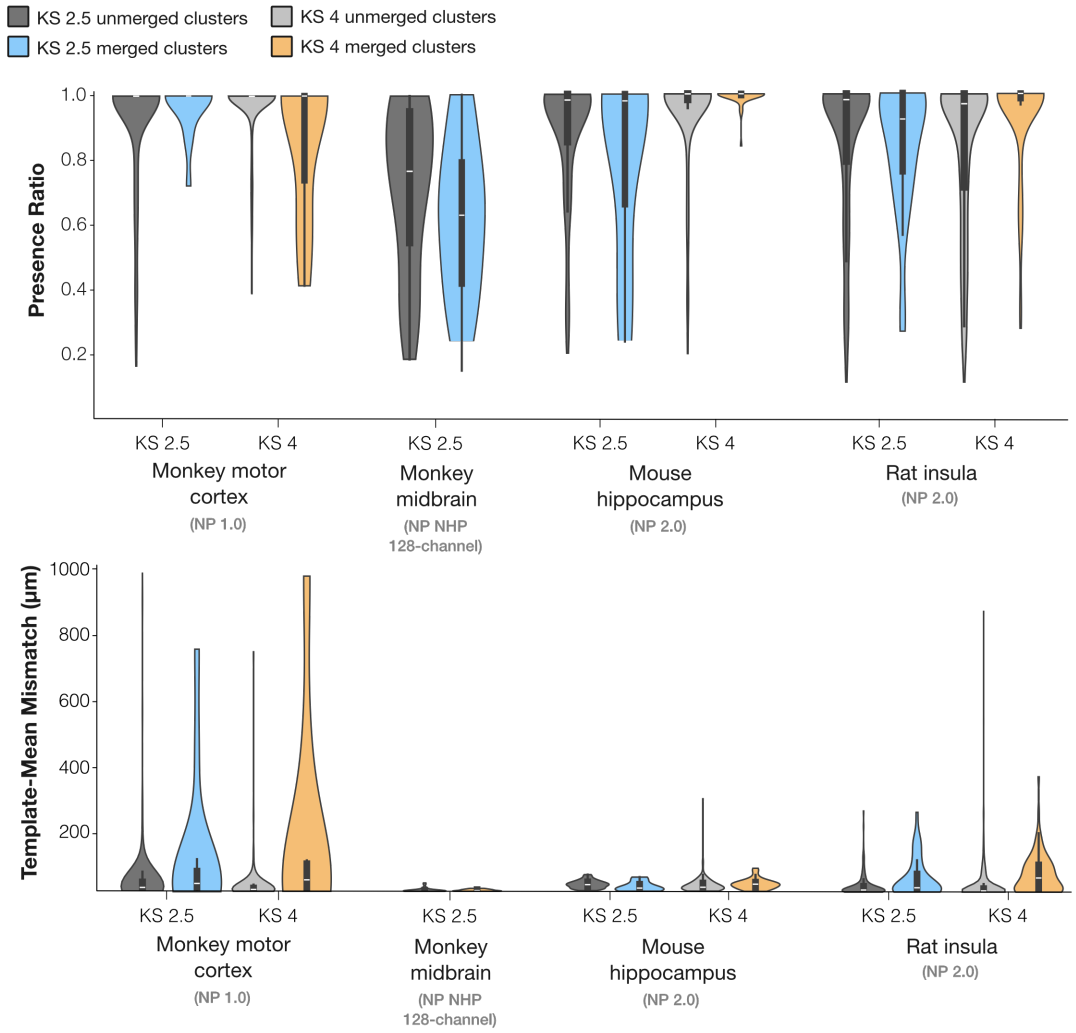

Figure S6: Units merged by SLAy are generally more likely to have low presence ratio (top) and high template-mean mismatch (bottom) than unmerged clusters, suggesting that SLAy systematically merges oversplit units.

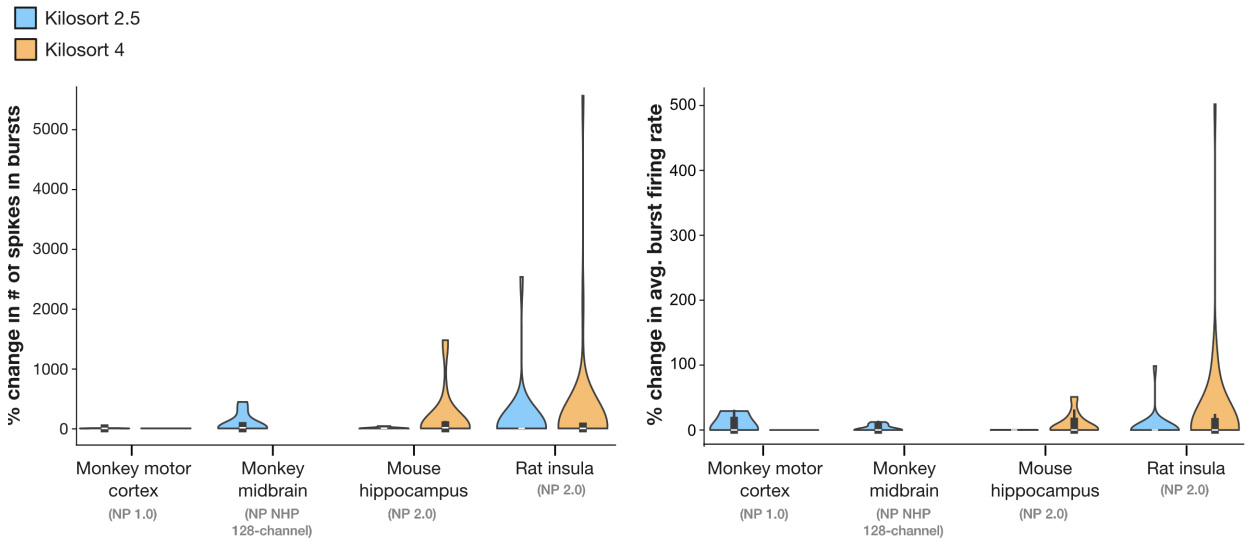

Figure S7: SLAy merges result in an increase in the number of detected bursts and the intensity (average firing rate) within the bursts.

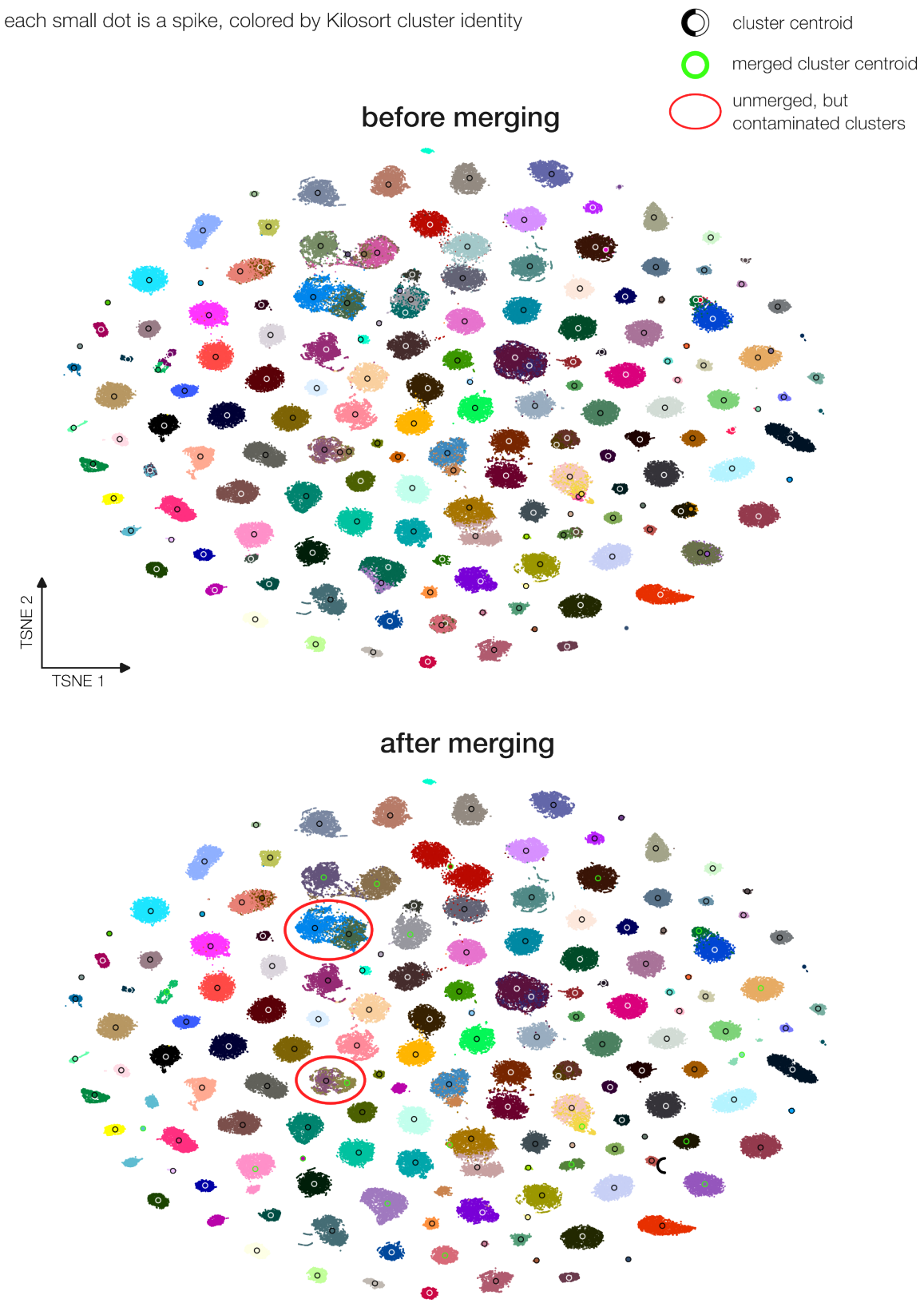

Figure S8: SLAy merges clusters that are close together in waveform space. TSNE plots of autoencoder latent space colored by unit ID, before (top) and after (bottom) merging. The green cluster centroids show that only clusters that are close together in latent space were merged together. Red circles show units that are close together in waveform space, but were not merged together, presumably because of refractory period violations.