Journal of Neuroscience Methods 288 (2017) 82–98

Contents lists available at ScienceDirect

Journal of Neuroscience Methods

journal homepage: www.elsevier.com/locate/jneumeth

Model-based spike sorting with a mixture of drifting *t*-distributions

Kevin Q. Shan[a,b], Evgueniy V. Lubenov[a,b], Athanassios G. Siapas[a,b,*]

[a] Division of Biology and Biological Engineering, California Institute of Technology, Pasadena, United States
[b] Division of Engineering and Applied Science, California Institute of Technology, Pasadena, United States

HIGHLIGHTS

- A nonstationary generative model for spike sorting is proposed.
- This model tracks unit drift in chronic recordings and is robust to outliers.
- It offers improved estimates of single unit isolation in empirical data.
- An efficient software implementation is provided for fitting the model.

ARTICLE INFO

Article history:
Received 27 January 2017
Received in revised form 16 June 2017
Accepted 20 June 2017
Available online 23 June 2017

Keywords:
Clustering
Heavy tails
Chronic recording
Spike overlap
Cluster drift
Unit isolation metrics

ABSTRACT

Background: Chronic extracellular recordings are a powerful tool for systems neuroscience, but spike sorting remains a challenge. A common approach is to fit a generative model, such as a mixture of Gaussians, to the observed spike data. Even if non-parametric methods are used for spike sorting, such generative models provide a quantitative measure of unit isolation quality, which is crucial for subsequent interpretation of the sorted spike trains.
New method: We present a spike sorting strategy that models the data as a mixture of drifting *t*-distributions. This model captures two important features of chronic extracellular recordings—cluster drift over time and heavy tails in the distribution of spikes—and offers improved robustness to outliers.
Results: We evaluate this model on several thousand hours of chronic tetrode recordings and show that it fits the empirical data substantially better than a mixture of Gaussians. We also provide a software implementation that can re-fit long datasets in a few seconds, enabling interactive clustering of chronic recordings.
Comparison with existing methods: We identify three common failure modes of spike sorting methods that assume stationarity and evaluate their impact given the empirically-observed cluster drift in chronic recordings. Using hybrid ground truth datasets, we also demonstrate that our model-based estimate of misclassification error is more accurate than previous unit isolation metrics.
Conclusions: The mixture of drifting *t*-distributions model enables efficient spike sorting of long datasets and provides an accurate measure of unit isolation quality over a wide range of conditions.

1. Introduction

Chronic extracellular recordings offer access to the spiking activity of neurons over the course of days or even months. However, the analysis of extracellular data requires a process known as spike sorting, in which extracellular spikes are detected and assigned to putative sources. Despite many decades of development, there is no universally-applicable spike sorting algorithm that performs best in all situations.

Approaches to spike sorting can be divided into two categories: model-based and non-model-based (or non-parametric). In the model-based approach, one constructs a generative model (e.g. a mixture of Gaussian distributions) that describes the probability distribution of spikes from each putative source. This model may be used for spike sorting by comparing the posterior probability that a spike was generated by each source. Fitting of such models may be partially or fully automated using maximum likelihood or Bayesian methods, and the model also provides an estimate of the misclassification error.

* Corresponding author at: Caltech MC 139–74, 1200 E California Blvd, Pasadena, CA 91125, United States.
E-mail address: thanos@caltech.edu (A.G. Siapas).

http://dx.doi.org/10.1016/j.jneumeth.2017.06.017

In the non-parametric approach, spike sorting is treated solely as a classification problem. These classification methods may range from manual cluster cutting to a variety of unsupervised learning algorithms. Regardless of the method used, scientific interpretation of the sorted spike train still requires reliable, quantitative measures of unit isolation quality. Often, these heuristics either explicitly (Hill et al., 2011) or implicitly (Schmitzer-Torbert et al., 2005) assume that the spike distribution follows a mixture of Gaussian distributions.

However, a mixture of Gaussians does not adequately model the cluster drift and heavy tails that are observed in experimental data (Fig. 1). Cluster drift is a slow change in the shape and amplitude of recorded waveforms (Fig. 1C), usually ascribed to motion of the recording electrodes relative to the neurons (Snider and Bonds, 1998; Lewicki, 1998). This effect may be small for short recordings (<1 h), but can produce substantial errors if not addressed in longer recordings (Fig. 7). Even in the absence of drift, spike residuals have heavier tails than expected from a Gaussian distribution, and may be better fit using a multivariate *t*-distribution (Figs. 1D and 6 ; see also Shoham et al., 2003; Pouzat et al., 2004).

To address these issues, we model the spike data as a mixture of drifting *t*-distributions (MoDT). This model builds upon previous work that separately addressed the issues of cluster drift (Calabrese and Paninski, 2011) and heavy tails (Shoham et al., 2003), and we have found the combination to be extremely powerful for modeling and analyzing experimental data. We also discuss the model's robustness to outliers, provide a software implementation of the fitting algorithm, and discuss some methods for reducing errors due to spike overlap.

We used the MoDT model to perform spike sorting on 34,850 tetrode-hours of chronic tetrode recordings (4.3 billion spikes) from the rat hippocampus, cortex, and cerebellum. Using these experimental data, we evaluate the assumptions of our model and provide recommended values for the model's user-defined parameters. We also analyze how the observed cluster drift may impact the performance of spike sorting methods that assume stationarity. Finally, we evaluate the accuracy of MoDT-based estimates of misclassification error and compare this to the performance of other popular unit isolation metrics in the presence of empirically-observed differences in firing rate and spike variability.

2. Methods

2.1. Mixture of drifting t-distributions (MoDT) model

Spike sorting begins with spike detection and feature extraction. During these preprocessing steps, spikes are detected as discrete events in the extracellular voltage trace and represented as points $\boldsymbol{y}_n$ in some D-dimensional feature space.

The standard mixture of Gaussians (MoG) model treats this spike data $\boldsymbol{y}_n$ as samples drawn from a mixture distribution with PDF given by

$$f_{\mathrm{MoG}}\left(\boldsymbol{y}_n;\phi\right) = \sum_{k=1}^{K}\alpha_k f_{\mathrm{mvG}}\left(\boldsymbol{y}_n;\boldsymbol{\mu}_k,\boldsymbol{C}_k\right),$$

where $\phi = \{\ldots, \alpha_k, \boldsymbol{\mu}_k, \boldsymbol{C}_k, \ldots\}$ is the set of fitted parameters, K is the number of mixture components, α_k are the mixing proportions, and $f_{\mathrm{mvG}}(\boldsymbol{y};\boldsymbol{\mu},\boldsymbol{C})$ is the PDF of the multivariate Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{C}$:

$$f_{\mathrm{mvG}}(\boldsymbol{y};\boldsymbol{\mu},\boldsymbol{C}) = \frac{1}{(2\pi)^{D/2}|\boldsymbol{C}|^{1/2}}\exp\left[-\frac{1}{2}\delta^2(\boldsymbol{y};\boldsymbol{\mu},\boldsymbol{C})\right].$$

Table 1
Mathematical notation. Lowercase bold letters ($\boldsymbol{y}_n$, $\boldsymbol{\mu}_{kt}$) denote D-dimensional vectors, and uppercase bold letters ($\boldsymbol{C}_k$, $\boldsymbol{Q}$) denote $D \times D$ symmetric positive definite matrices.

Dimensions	
D	Number of feature space dimensions
N	Number of spikes
K	Number of clusters
T	Number of time frames
Given data	
$\boldsymbol{y}_n$	Observed spike n
t_n	Time frame in which spike n occurred
w_n	Weighting of spike n (multiplier applied to log-likelihood)
User-defined constants	
ν	t-distribution degrees-of-freedom parameter
$\boldsymbol{Q}$	Drift regularization parameter
Fitted model parameters	
α_k	Mixing proportion for cluster k
$\boldsymbol{\mu}_{kt}$	Location parameter for cluster k in time frame t
$\boldsymbol{C}_k$	Scale parameter for cluster k
Latent variables introduced by EM procedure	
z_{nk}	Posterior probability that spike n belongs to cluster k
u_{nk}	Scaling variable introduced in formulating the t-distribution as a Gaussian-Gamma compound distribution

For notational convenience, let δ^2 denote the squared Mahalanobis distance

$$\delta^2(\boldsymbol{y};\boldsymbol{\mu},\boldsymbol{C}) = (\boldsymbol{y}-\boldsymbol{\mu})^{\top}\boldsymbol{C}^{-1}(\boldsymbol{y}-\boldsymbol{\mu}).$$

We make two changes to this model. First, we replace the multivariate Gaussian distribution with the multivariate *t*-distribution. The PDF for this distribution, parameterized by location $\boldsymbol{\mu}$, scale $\boldsymbol{C}$, and degrees-of-freedom ν, is given by

$$f_{\mathrm{mvt}}(\boldsymbol{y};\boldsymbol{\mu},\boldsymbol{C},\nu) = \frac{1}{(\nu\pi)^{D/2}|\boldsymbol{C}|^{1/2}}\frac{\Gamma((\nu+D)/2)}{\Gamma(\nu/2)}\left[1+\frac{1}{\nu}\delta^2(\boldsymbol{y};\boldsymbol{\mu},\boldsymbol{C})\right]^{-(\nu+D)/2}$$

Second, we break up the dataset into T time frames (we used a frame duration of 1 min) and allow the cluster location $\boldsymbol{\mu}$ to change over time. The mixture distribution becomes

$$f_{\mathrm{MoDT}}\left(\boldsymbol{y}_n;\phi\right) = \sum_{k=1}^{K}\alpha_k f_{\mathrm{mvt}}\left(\boldsymbol{y}_n;\boldsymbol{\mu}_{kt_n},\boldsymbol{C}_k,\nu\right),$$

where $t_n \in \{1,\ldots,T\}$ denotes the time frame for spike n. We use a common ν parameter for all components and have chosen to treat it as a user-defined constant. The fitted parameter set is thus $\phi = \{\ldots, \alpha_k, \boldsymbol{\mu}_{k1},\ldots,\boldsymbol{\mu}_{kT},\boldsymbol{C}_k,\ldots\}$.

In order to enforce consistency of the component locations across time, we introduce a prior on the location parameter that penalizes large changes over consecutive time steps. This prior has a joint PDF proportional to

$$f_{\mathrm{prior}}(\boldsymbol{\mu}_{k1},\ldots,\boldsymbol{\mu}_{kT}) = \prod_{t=2}^{T} f_{\mathrm{mvG}}(\boldsymbol{\mu}_{kt}-\boldsymbol{\mu}_{k(t-1)};\boldsymbol{0},\boldsymbol{Q}), \quad (1)$$

where $\boldsymbol{Q}$ is a user-defined covariance matrix that controls how much the clusters are expected to drift.

2.2. EM algorithm for model fitting

Assuming independent spikes and a uniform prior on the other model parameters, we can obtain the maximum *a posteri-*

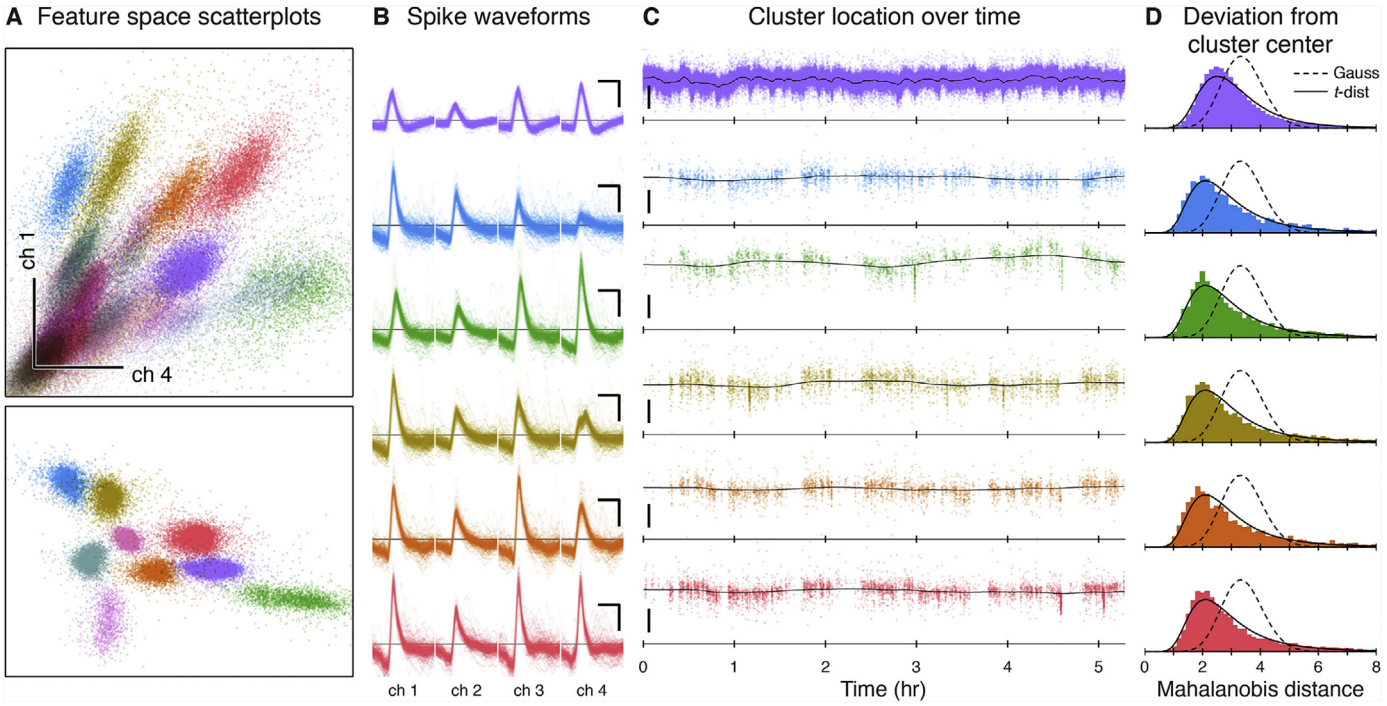

Fig. 1. Extracellular recordings contain drifting, heavy-tailed clusters. (A) Scatterplots of spikes in feature space, color-coded by putative identity. Spike waveforms recorded on 4 tetrode channels were projected onto a 12-dimensional feature space using 3 principal components from each channel. Top: scatterplot of the first principal component from channels 1 and 4. Bottom: a different projection of the data, showing only the best-isolated single units. Scale bar: 50 μV RMS (see Section 2.5). (B) Spike waveforms (inverted polarity) for six example units. Scale bar: 200 μV, 0.5 ms. (C) Cluster drift in feature space. *y*-axis shows one of the 12 feature space dimensions. Black line indicates the cluster center fitted using the MoDT model. Scale bar: 50 μV RMS. (D) Distribution of the non-squared Mahalanobis distance (δ) from the fitted cluster center to the observed spikes. Lines indicate the theoretical distributions for Gaussian and *t*-distributed spikes; see Appendix E.1 for derivation.

ori (MAP) estimate of the fitted parameters ϕ by maximizing the log-posterior, which is equivalent (up to an additive constant) to the following:

$$L(\phi) = \sum_{n=1}^{N} w_n \log f_{\text{MoDT}}(\boldsymbol{y}_n; \phi) + \sum_{k=1}^{K} \log f_{\text{prior}}(\boldsymbol{\mu}_{k1}, \ldots, \boldsymbol{\mu}_{KT}).$$

Note that we have introduced a weight w_n for each spike. This allows us to fit the model to a weighted subset of the data while remaining consistent with the full dataset (Feldman et al., 2011).

As with most mixture distributions, it is intractable to optimize $L(\phi)$ directly. However, by introducing additional latent random variables, we obtain a "complete-data" log-posterior $L_c(\phi, Z, U)$ that allows us to decompose the problem and optimize it using an expectation-maximization (EM) algorithm (McLachlan and Peel, 2000).

In the E-step, we compute the expected value of L_c assuming that these latent variables follow their conditional distribution given the observed data and the fitted parameters $\hat{\phi}$ from the previous EM iteration. The conditional expectations of these latent variables are given by:

$$z_{nk} = \frac{\hat{\alpha}_k f_{\text{mvt}}\left(\boldsymbol{y}_n; \hat{\boldsymbol{\mu}}_{kt_n}, \hat{\boldsymbol{C}}_k, \nu\right)}{\sum_\kappa \hat{\alpha}_\kappa f_{\text{mvt}}\left(\boldsymbol{y}_n; \hat{\boldsymbol{\mu}}_{\kappa t_n}, \hat{\boldsymbol{C}}_\kappa, \nu\right)}, \tag{2}$$

$$u_{nk} = \frac{\nu + D}{\nu + \delta^2\left(\boldsymbol{y}_n; \hat{\boldsymbol{\mu}}_{kt_n}, \hat{\boldsymbol{C}}_k\right)}. \tag{3}$$

The z_{nk} correspond to the posterior probability that spike n was produced by component k, and may thus be used for spike sorting. The u_{nk} arises from the formulation of the *t*-distribution as a Gaussian-Gamma compound distribution and may be interpreted as a scaling variable that "Gaussianizes" the multivariate *t*-distribution. In the Gaussian case (the limit of a *t*-distribution as $\nu \to \infty$), we have $u_{nk} = 1$ for all spikes. For finite ν, note that u_{nk} decreases as the Mahalanobis distance δ increases.

Next we can compute the conditional expectation of $L_c(\phi, Z, U)$ over these latent variables. Following Peel and McLachlan (2000), we find that this is equivalent (up to an additive constant) to

$$\begin{aligned} J(\phi; \hat{\phi}) = &\sum_{n=1}^{N} w_n \sum_{k=1}^{K} z_{nk}\left[\log \alpha_k - \frac{1}{2}\log|\boldsymbol{C}_k| \right. \\ &\left. - \frac{1}{2} u_{nk} \delta^2(\boldsymbol{y}_n; \boldsymbol{\mu}_{kt_n}, \boldsymbol{C}_k)\right] \\ &+ \sum_{t=1}^{T} -\frac{1}{2}\delta^2(\boldsymbol{\mu}_{kt} - \boldsymbol{\mu}_{k(t-1)}; 0, \boldsymbol{Q}). \end{aligned}$$

In the M-step, we maximize $J(\phi, \hat{\phi})$ with respect to the fitted parameters. The optimal value for the mixing proportions α is simply a weighted version of the mixture of Gaussians (MoG) M-step update:

$$\underset{\alpha_k}{\text{argmax}} J(\phi; \hat{\phi}) = \frac{\sum_n w_n z_{nk}}{\sum_n w_n}. \tag{4}$$

The optimal value for the cluster scale parameter $\boldsymbol{C}$ is also similar to the MoG case, but each spike is additionally scaled by u_{nk}:

$$\begin{aligned} &\underset{\boldsymbol{C}_k}{\text{argmax}} J\left(\phi; \hat{\phi}\right) \\ &= \frac{\sum_n w_n z_{nk} u_{nk}\left(\boldsymbol{y}_n - \boldsymbol{\mu}_{kt_n}\right)\left(\boldsymbol{y}_n - \boldsymbol{\mu}_{kt_n}\right)^T}{\sum_n w_n z_{nk}}. \end{aligned} \tag{5}$$

For the cluster location parameters $\boldsymbol{\mu}$, note that $J(\phi, \hat{\phi})$ is quadratic with respect to $\boldsymbol{\mu}$ and its maximum occurs where the

Table 2
Computational runtime for model fitting. Time required to perform 20 EM iterations on the sample dataset shown in Fig. 1 (D=12, K = 26, N = 1.9 million). `fitgmdist` is a mixture of Gaussians fitting routine that is part of the MATLAB Statistics and Machine Learning Toolbox. `modt` is a MATLAB implementation of our EM algorithm that may be downloaded from https://github.com/kqshan/MoDT. In addition to fitting the richer MoDT model, it supports two additional features (data weights and GPU computing) that can dramatically reduce fitting times.

Model type	Algorithm description	Runtime (s)
MoG	`fitgmdist`	123.43
MoG	`modt` in Gaussian mode	103.53
MoDT	`modt`	104.56
MoDT	`modt` on GPU	7.06
MoDT	`modt`, 5% subset	5.74
MoDT	`modt`, 5% subset, GPU	1.24

gradient $\nabla_{\boldsymbol{\mu}} J(\phi, \hat{\phi}) = 0$. Hence we can find the optimal $\boldsymbol{\mu}$ by solving the following linear system of equations:

$$\nabla_{\boldsymbol{\mu}_k} J(\phi, \hat{\phi}) = \begin{bmatrix} \boldsymbol{b}_{k1} \\ \boldsymbol{b}_{k2} \\ \vdots \\ \boldsymbol{b}_{kT} \end{bmatrix} - \boldsymbol{A} \begin{bmatrix} \boldsymbol{\mu}_{k1} \\ \boldsymbol{\mu}_{k2} \\ \vdots \\ \boldsymbol{\mu}_{kT} \end{bmatrix} = 0, \tag{6}$$

where

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{M}_{k1} + \boldsymbol{Q}^{-1} & -\boldsymbol{Q}^{-1} & & \\ -\boldsymbol{Q}^{-1} & \boldsymbol{M}_{k2} + 2\boldsymbol{Q}^{-1} & -\boldsymbol{Q}^{-1} & \\ & -\boldsymbol{Q}^{-1} & \ddots & \ddots \\ & & \ddots & \boldsymbol{M}_{kT} + \boldsymbol{Q}^{-1} \end{bmatrix}$$

and

$$\boldsymbol{M}_{kt} = \boldsymbol{C}_k^{-1} \sum_{n\,:\,t_n = t} w_n z_{nk} u_{nk}$$
$$\boldsymbol{b}_{kt} = \boldsymbol{C}_k^{-1} \sum_{n\,:\,t_n = t} w_n z_{nk} u_{nk} \boldsymbol{y}_n.$$

Although this is a $DT \times DT$ linear system, its sparsity structure allows us to solve for $\boldsymbol{\mu}$ with a complexity that scales linearly with T (Paninski et al., 2010). Typically, $DT \ll N$ and solving Eq. (6) accounts for a negligible fraction of the overall computational runtime. Appendix A describes some alternative methods for the M-step update of $\boldsymbol{\mu}$.

Note that the scaling variable u_{nk} acts as an additional weighting term in the optimization of $\boldsymbol{\mu}$ and $\boldsymbol{C}$. Since u_{nk} decreases as spike n gets far away from cluster k, any outliers are automatically discounted during the fitting process. As a result, the fitted parameters are considerably more robust to the presence of outliers than in the Gaussian case (Fig. 2). This property makes the t-distribution a useful model even when the underlying data are Gaussian, but unmodeled noise or artifacts may be present.

2.3. Software implementation

We provide a MATLAB implementation of this EM algorithm at https://github.com/kqshan/MoDT. In this section, we measure the runtime on a desktop workstation with an Intel Core i5-7500 CPU, 32 GB of memory, and an NVidia GeForce GTX 1080 graphics card, running MATLAB R2017a (64-bit) on Ubuntu 16.04.2 with CUDA toolkit 8.0, using double-precision arithmetic.

Our implementation offers a mild speedup over the MATLAB built-in mixture-of-Gaussians fitting routine, despite fitting a more complex model (Table 2). In addition, it supports the use of a weighted training subset, which offers a proportional reduction in runtime at the expense of model accuracy, and supports the use of GPU computing using the NVidia CUDA computing platform.

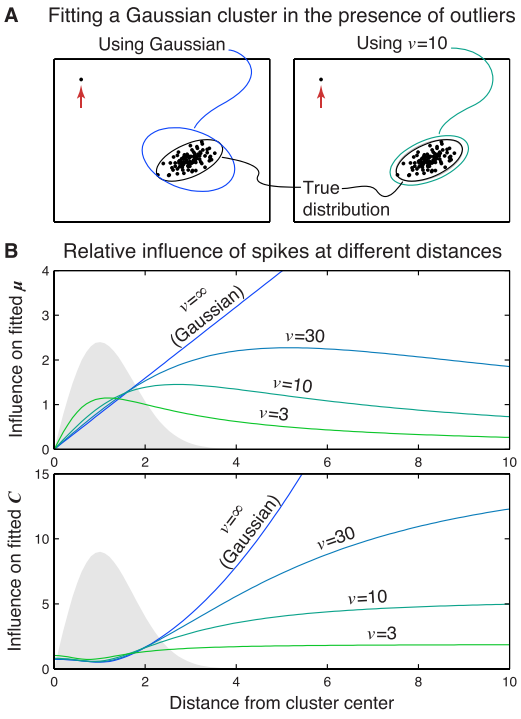

Fig. 2. Fitted parameters of the t-distribution are robust to outliers. (A) In this example we generated 100 points from a Gaussian distribution and added a single outlier (red arrow). This has stretched out the estimated covariance when using a Gaussian fit (left). In contrast, fitting with a t-distribution (right) comes much closer to the true parameters. (B) For a Gaussian model, the relative influence of a single spike grows unbounded with increasing distance from the cluster center, allowing outliers to exert an undue influence on the fitted model parameters. When fitting a t-distribution, the scaling variable u_{nk} effectively discounts any spikes far away from the cluster center, thereby limiting the effect of outliers. The grey histogram in the panel background shows the theoretical distance distribution for a Gaussian cluster. See Appendix E.3 for more detail.

How does this runtime scale with the model dimensions D, K, N, and T? The most computationally intensive operations are computing the Mahalanobis distance (D^2KN), updating the cluster location $\boldsymbol{\mu}$ ($DKN + D^3KT$), and updating the scale parameter $\boldsymbol{C}$ (D^2KN). Since the number of spikes is typically much larger than the number of time frames or dimensions ($N \gg DT$), we expect the fitting time to scale as D^2KN overall. To test these scaling laws, we measured the runtime while varying each model dimension (Fig. 3).

Surprisingly, we found that the CPU runtime scaled almost linearly with D. This is because the CPU's memory throughput (17 GB/s), rather than its computing power (218 GFLOPS), is the limiting factor when $D < 500$, and memory access scales linearly with D.

The GPU's higher memory throughput (320 GB/s) affords a substantial speedup on small D. Like many consumer-grade GPUs, this device's single-precision computing power (8.2 TFLOPS) is substantially higher than its double-precision capacity (257 GFLOPS), and switching to single-precision arithmetic dramatically increases performance on compute-limited tasks.

As expected, we found the runtime scales linearly with K and N, and the effect of T is negligible. The GPU shows similar trends, but with reduced efficiency at small N and T due to poor utilization of the hardware resources.

Finally, we measured the peak memory usage to be approximately 5 $N \times K$ matrices and 4 $D \times N$ matrices, for a total of $8(5K + 4D)N$ bytes (double-precision). Fitting larger datasets may require using a weighted subset of spikes and/or performing the optimization in batches.

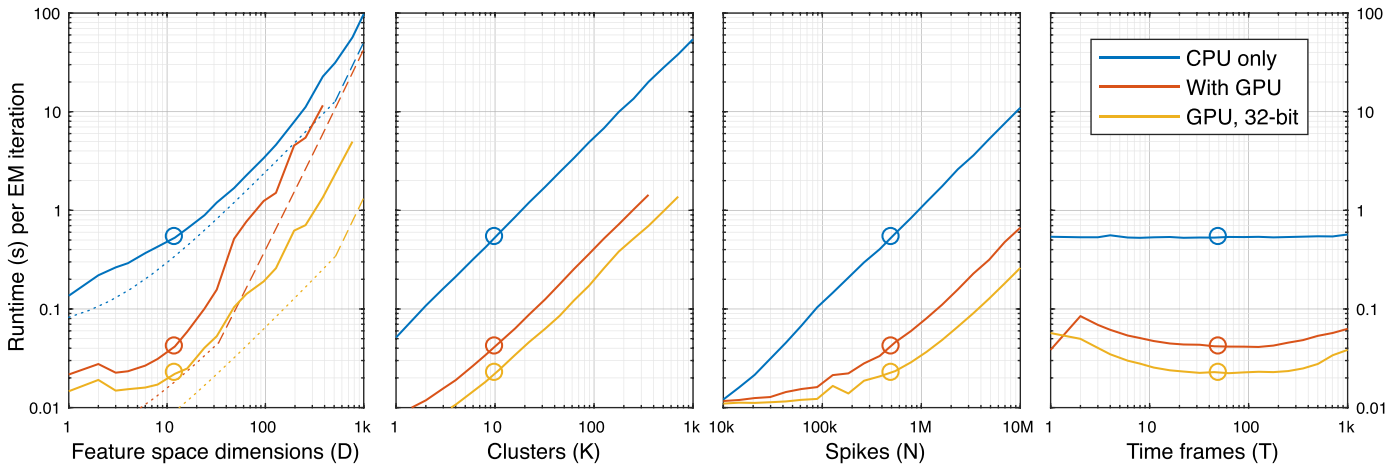

Fig. 3. Scaling of computational runtime with model dimensions. Starting from a baseline of $D = 12$, $K = 10$, $N = 500{,}000$, $T = 50$, we varied each dimension and measured the runtime on CPU (blue) and GPU (red). We also measured GPU runtime using single-precision (32-bit) arithmetic (yellow). For D, we show the theoretical limits imposed by the hardware's computing power (dashed line) and memory throughput (dotted line). Peak memory usage is $(5K + 4D)N$ elements, and the GPU line ends when we run out of GPU memory (8 GB).

2.4. Interactive clustering

We relied on human operators to guide the model fitting in an interactive clustering step. The user can change the number of clusters (K) by choosing clusters to split or merge, and we re-fit the model after each operation. To ensure an adequate user experience, we used weighted training subsets as necessary to ensure that we could re-fit the model and update the user interface in a matter of seconds.

The quality of the initial fit is an important factor in determining the user workload. When the dataset comes from a sequence of recordings with stable electrodes, we initialize the model using the fitted parameters from the previous dataset. Using this initialization, we have found that most datasets require minimal user interaction.

When no prior dataset exists, we use a split-and-merge technique (Ueda et al., 2000) for parameter initialization and the Bayes information criterion (BIC) for model selection, similar to the method described by Tolias et al. (2007). This initialization works well in brain areas with a low density of neurons (e.g. cortex), but typically requires manual intervention in brain areas with greater multi-unit activity, especially if the units of interest fire very sparsely (e.g. hippocampal area CA1).

2.5. Data preprocessing

To evaluate the MoDT model for spike sorting, we collected 34,850 tetrode-hours (34 terabytes) of chronic tetrode data by implanting 10 Long-Evans rats with 24-tetrode arrays targeting areas of the hippocampus, cortex, and cerebellum. Extracellular signals were digitized at 25 kHz and recorded continuously, but these recordings were broken into datasets ranging from 1 to 25 h in duration, depending on experimental needs. All animal procedures were in accordance with National Institutes of Health (NIH) guide for the care and use of laboratory animals, and approved by the Caltech Institutional Animal Care and Use Committee.

To detect spikes, we first bandpass filtered the data using an FIR filter with a 600–6000 Hz passband and then upsampled to 50 kHz (Blanche and Swindale, 2006). As a spike detection metric, we computed the nonlinear energy operator (Mukhopadhyay and Ray, 1998) for each channel, took the maximum value across channels, smoothed this using a Gaussian window ($\sigma = 0.4$ ms), and took the square root so that the detection metric would scale linearly with spike amplitude. This last step is irrelevant for spike detection, but produces better fits when estimating false negatives in Section 2.9. We detected spikes by identifying peaks in this detection metric that exceed a given threshold and then saved a short window around each spike as the spike waveform.

We then performed feature extraction to represent each spike as a D-dimensional feature vector $\boldsymbol{y}_n$. We used principal component analysis (PCA), which approximates each spike as a linear combination of orthogonal basis waveforms (also known as PCA axes). Traditionally, these PCA axes are chosen to minimize the total squared approximation error; we used an alternative objective function (Huber loss; see e.g. Udell et al., 2016) to ensure that the chosen basis would not be unduly influenced by high-amplitude artifacts. We selected 3 PCA axes from each of the 4 tetrode channels (12 dimensions in total).

This dimensionality reduction preserves the 2-norm of the spike waveforms; we further scale these by $1/\sqrt{P}$, where P is the number of samples in the waveform, so that the units of the feature space can be interpreted as the root-mean-square (RMS) amplitude of the spike waveforms along the PCA axes. Although these RMS amplitudes have units of microvolts, these values are smaller than the waveform peak amplitudes that are commonly reported in spike sorting. For example, the blue unit in Fig. 1 (second row in panels B–D) has an RMS amplitude of 100 μV along the first PCA feature dimension on channel 1, but its spike peak amplitude on that channel is 384 μV.

2.6. Overlapping spikes

If two neurons fire near-simultaneously, their spikes will overlap and produce a waveform that is the sum of both waveforms. Properly resolving these overlapping spikes is an outstanding challenge in spike sorting, and is not directly addressed by the MoDT model. In this section, we describe some techniques we have used to mitigate this issue.

First, we can deconvolve mildly-overlapping spikes during feature extraction. Let us denote the extracted spike waveform (Fig. 4A) as the P-dimensional vector $\boldsymbol{s}_n$.

Given a set of D basis waveforms (denoted as the $P \times D$ matrix $\boldsymbol{W}$), traditional PCA-based feature extraction corresponds to the optimization problem

$$\underset{\boldsymbol{y}_n}{\text{minimize}} \; \|\boldsymbol{s}_n - \boldsymbol{W}\boldsymbol{y}_n\|^2.$$

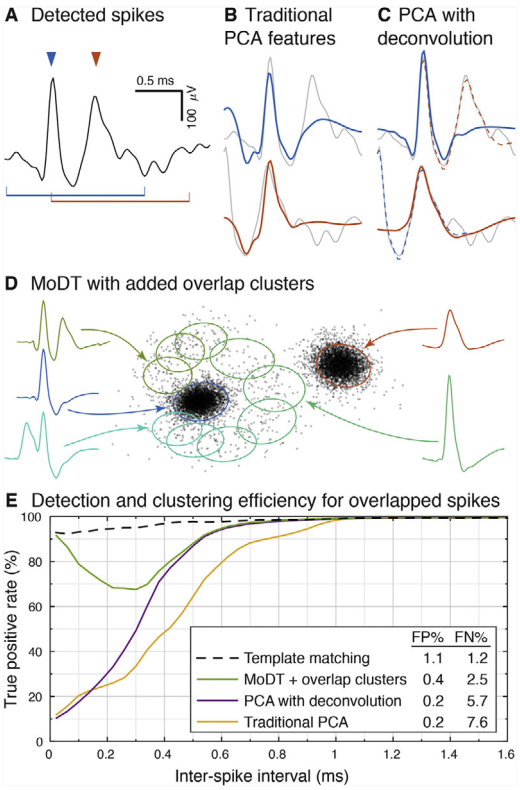

Fig. 4. Handling overlapping spikes. (A) Example voltage trace in which two neurons fire in close succession. Two spikes were detected (blue + red triangles) and a 1.5 ms window around each spike (blue + red brackets) was saved as a spike waveform. Panels B and C compare two alternative feature extraction methods applied to these spikes. (B) In PCA-based feature extraction, each detected spike (grey) is approximated as a linear combination of basis waveforms. Traditionally, this is performed for each spike independently, and these approximations (blue + red waveforms) can become distorted by the presence of nearby spikes. (C) If we account for this temporal overlap, we can deconvolve nearby spikes during feature extraction. Dashed lines show the contribution of the other spike. (D) Feature space scatterplot containing overlapping spikes that could not be deconvolved. Blue and red ellipses show the model clusters for single spikes; overlapping spikes appear as outliers. By augmenting the model with overlap clusters (green ellipses), we can reassign these overlapping spikes to the appropriate units. (E) Performance on a hybrid ground truth dataset with 12 donor units (a combined firing rate of 76.4 Hz). Legend shows the overall false positives and negatives as a percentage of the total spikes.

Since we have chosen an orthonormal basis, the optimal feature vector is simply $\boldsymbol{y}_n = \boldsymbol{W}^{\mathrm{T}}\boldsymbol{s}_n$.

Unfortunately, this approach treats each spike independently of the rest, which means that nearby spikes can distort the result; note the unusual shapes of the PCA approximations in Fig. 4B. This distortion impacts spike sorting performance for spikes separated by as much as 0.9 ms (Fig. 4E, yellow line).

However, we can determine from the detected spike times whether two waveforms should overlap. If we let $\boldsymbol{T}_\tau$ denote the $P \times P$ matrix corresponding to a temporal shift by τ samples, then the joint reconstruction problem for two spikes separated by τ samples is

$$\underset{\boldsymbol{y}_n, \boldsymbol{y}_{n+1}}{\text{minimize}} \left\| \begin{bmatrix} \boldsymbol{s}_n \\ \boldsymbol{s}_{n+1} \end{bmatrix} - \begin{bmatrix} \boldsymbol{W} & \boldsymbol{T}_\tau \boldsymbol{W} \\ \boldsymbol{T}_{-\tau} \boldsymbol{W} & \boldsymbol{W} \end{bmatrix} \begin{bmatrix} \boldsymbol{y}_n \\ \boldsymbol{y}_{n+1} \end{bmatrix} \right\|^2. \tag{7}$$

By acknowledging this overlap during the feature extraction process, we can reduce the PCA approximation error (Fig. 4C) and more accurately assign these mildly-overlapping spikes (Fig. 4E, purple line).

However, this PCA-based deconvolution requires that both spikes are detected. If two closely-overlapping spikes are detected as a single event, it cannot be deconvolved this way and will show up as an outlier during the clustering process. Thanks to the t-distribution's robustness to outliers, these outliers do not substantially affect the model fitting (Fig. 2) and can be ignored during the interactive clustering process.

Afterwards, a number of options are available. If the estimated number of false negatives due to spike overlap ("censored events" in Section 2.9) is sufficiently low, then it may be adequate to leave the spike assignments as-is. Unless otherwise specified, this is the method used for the analyses in this report.

Alternatively, we can augment the fitted model with additional components corresponding to these overlapped waveforms (Fig. 4D). Appendix B.1 describes how we generated these overlap clusters. Any spikes assigned to these overlap clusters are then reassigned to their source units. In our test dataset, this correctly reassigned 70–90% of overlapping spikes (Fig. 4E, green line), yielding an overall false negative rate of 2.5%.

A third option is to transform the fitted cluster centers $\boldsymbol{\mu}_{kt}$ back into waveform space as $\boldsymbol{W}\mu_{kt}$ for use in a template matching algorithm (Lewicki, 1998; Segev et al., 2004; Prentice et al., 2011; Pillow et al., 2013). We tested the hybrid Bayes-optimal template matching algorithm (Franke et al., 2015). This method maintains a high detection rate regardless of inter-spike interval (Fig. 4E, dashed line), but required careful tuning of the detection threshold to achieve an acceptable level of false positives. Appendix B.2 discusses some additional considerations for template matching.

2.7. Computational infrastructure

All analysis was implemented in MATLAB and performed on a small computer cluster consisting of 4 compute nodes (a total of 64 cores) and 11 storage nodes (serving data using the NFS network file system). For non-interactive tasks, we used the MATLAB Parallel Computing Toolbox to submit batch jobs to a Sun Grid Engine job scheduler. We used the DataJoint MATLAB toolbox (Yatsenko et al., 2015) with a MySQL relational database to (1) keep track of which datasets were waiting in the compute queue and which were ready for interactive clustering, (2) automate the computation of derived quantities and (3) store all metadata, ranging from recording parameters to unit quality metrics, in an efficiently-queryable manner.

The interactive clustering step was the bottleneck of the spike sorting process due to the small number of human users (4 users vs. 64 compute cores) and their low availability (only a few hours per day, compared to the 99% uptime of the computer cluster). We did not explicitly track the amount of time spent clustering, but based on file modification timestamps, we estimate that it took 500 user-hours to cluster these 34,850 tetrode-hours of data. Average throughput ranged from 13 to 27 datasets per user-hour, independent of the dataset's recording duration.

2.8. Measuring unit isolation using the MoDT model

The MoDT model may be fitted to previously spike-sorted data by using the given spike assignments, rather than the z_{nk} computed in the E-step, during the M-step update. If the spike sorting algorithm can provide soft assignments (i.e. each spike has a probability of belonging to each cluster rather than being fully assigned to a single cluster), then these probabilities can be substituted directly as $\hat{z}_{nk}$. For hard assignments, the equivalent posterior is

$$\hat{z}_{nk} = \begin{cases} 1 & \text{if spike } n \text{ was assigned to cluster } k \\ 0 & \text{otherwise.} \end{cases}$$

Model fitting still requires iterative evaluation of Eqs. (3)–(6), but typically converges in fewer than 10 iterations since $\hat{z}_{nk}$ is fixed.

After fitting the model parameters (α_k, $\boldsymbol{\mu}_{kt}$, $\boldsymbol{C}_k$), the z_{nk} defined by Eq. (2) provides a model-based estimate of the probability that spike $\boldsymbol{y}_n$ was produced by each of the source clusters. Summing these z_{nk} provides the expected number of misclassified spikes. Following Hill et al. (2011), we define the false positive (FP) fraction and the false negative (FN) ratio for cluster k as

$$\begin{aligned} \text{FP\%} &= \frac{1}{|\mathcal{N}_k|} \sum_{n \in \mathcal{N}_k} \sum_{\kappa \neq k} z_{n\kappa} \\ \text{FN\%} &= \frac{1}{|\mathcal{N}_k|} \sum_{n \notin \mathcal{N}_k} z_{nk}, \end{aligned}$$

where $\mathcal{N}_k$ is the set of spikes assigned to cluster k. In the hypothesis testing literature, the FP fraction is also known as the false discovery rate. The FN ratio does not have a similar analogue, and it may be greater than one.

The MoDT model also provides a natural generalization of Gaussian-based unit isolation metrics to the drifting case. By setting $\nu = \infty$, the fitted $\boldsymbol{\mu}_{kt}$ and $\boldsymbol{C}_k$ correspond to the time-varying cluster mean and the cluster covariance, respectively. These were used to compute Mahalanobis distances for the comparative analysis of unit isolation quality metrics (Fig. 9C and D).

2.9. Unit quality criteria

Overall, we detected 4.3 billion spikes and spike-sorted these into 48,620 clusters. However, not all of these clusters correspond to single units, which are typically interpreted as the spiking activity of individual neurons. Spike detection and sorting are not perfect, and a given cluster may contain spurious spikes (false positives) or may not capture all of the spikes from a given neuron (false negatives). Depending on the scientific question being addressed, our subsequent analysis may be more or less sensitive to the presence of such errors. Reliable, quantitative measures of unit quality are therefore critically important for the proper interpretation of the spike sorting output.

To evaluate the quality of these clusters, we performed the inspection steps described by Hill et al. (2011) and computed their recommended quality metrics: false positives and negatives due to misclassification (Section 2.8), false negatives from the spike detection threshold (by fitting a truncated Gaussian distribution to the detection metric) and false negatives from censored events (by considering all spikes with a higher detection metric as potential censoring candidates).

We typically require that putative single units have false positives <10% and combined false negatives <10%. Hill et al. also describe an estimate of false positives based on refractory period violations, but we did not use this as part of the unit selection criteria because it would have excluded too many low-firing units. For example, a unit with a firing rate of 0.1 Hz and 1 refractory period violation in 24 h would have an estimated 50% false positives using this metric. Instead, we only require that the fraction of spikes that violate the refractory period is <1%. These criteria identified 20,630 putative single units, accounting for 852 million spikes over 89,127 unit-hours.

However, in our subsequent analysis we are faced with the following challenge: we wish to characterize the empirical properties of single units so that we can decide what assumptions to make during spike sorting, but we need to perform spike sorting in order to obtain single units to characterize. To break this circular dependency, we focused our analysis on the best-isolated units because these are the least sensitive to the spike sorting procedure used. This also ensures that we are studying the natural tails of the spike distribution rather than artificially truncating them via the spike classifier boundaries.

Therefore we applied a more conservative set of selection criteria than usual. We tightened the false positive/negative thresholds to 2% and further required that units have a mean spike peak amplitude >200 μV and come from a recording >2 h in duration. Using these criteria, we identified 4432 high-amplitude, well-isolated single units, accounting for 338 million spikes over 32,890 unit-hours.

2.10. Hybrid ground truth datasets

To validate the performance of our spike sorting toolchain, we generated "hybrid ground truth" datasets by injecting known spikes into an acceptor dataset (Rossant et al., 2016). In order to more realistically capture the waveform variability, we used the actual spike waveforms from the original data (with overlapping spikes identified and removed using the "overlap clusters" method described in Section 2.6) rather than synthesizing them from the mean waveform, but otherwise followed the procedure described by Rossant et al.

We selected 45 well-isolated units to form our base set of donor units. However, this selection is unavoidably biased towards units that are easy to cluster using our current method. In order to more fully characterize the space of possible units, we generated additional units by modifying the spike amplitude (by scaling the spike waveforms), firing rate (by dropping a subset of spikes), and drift rate (by temporally compressing the spike train) of these base units.

We thus obtained 450 donor units that ranged in spike peak amplitude from 50 to 700 μV, firing rate from 0.003 to 30 Hz, and drift rate from 0.06 to 5000 μV^2/h. For each donor, we selected an acceptor dataset from the same tetrode but several days earlier or later.

3. Results

In this section, we evaluate the MoDT model using the data we have collected. First, we use the 4432 best-isolated single units to characterize the cluster drift and the distribution of spike residuals in our chronic tetrode recordings, and thus provide recommendations for the user-defined parameters in the MoDT model. We also use these units—which we have tracked over several hours at a time—to consider the consequences of using a stationary model for spike sorting. Next, we evaluate the MoDT spike sorting performance on our 450 hybrid ground truth datasets, which represent a wide range of amplitudes, firing rates, and drift rates. Finally, we compare the MoDT-based estimate of misclassification error to several other commonly-used metrics of unit isolation quality.

3.1. Cluster drift in empirical data

We quantified the cluster drift by measuring the distance from each unit's current location (determined using a 40-min moving average) to its location at the start of the recording. Individual clusters may move closer or farther away from where they started (Fig. 5A), but over all units, the average distance increases and the distribution spreads out (Fig. 5B). These distances are measured in feature space units (μV RMS, see Section 2.5).

Note that the cluster location prior in Eq. (1) corresponds to a Gaussian random walk with a constant rate of drift. However, the observed distribution of drift distances is much broader than expected from such a process, and so this aspect of the MoDT model should be treated as a regularizer rather than an attempt to accurately model the underlying phenomena.

The MoDT model parameter $\boldsymbol{Q}$ is a user-defined constant that controls this regularization. Fig. 5C shows the effect of changing this parameter over a wide range of values. The log-likelihood ratio (LLR) is a measure of the MoDT model's quality of fit compared to

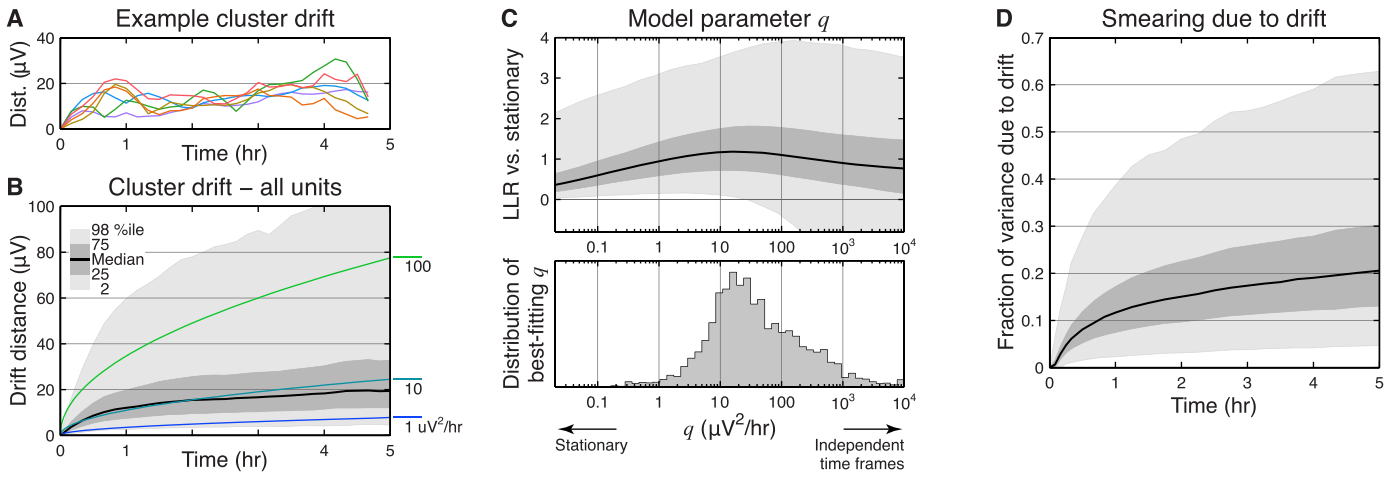

Fig. 5. Cluster drift in well-isolated units. (A) Cluster drift of the 6 example units from Fig. 1. Distances are measured from the unit's current location (determined using a 40-min moving average) to its location at the start of the dataset. (B) Cluster drift of all well-isolated units in our analysis. Shaded regions indicate quantiles across units. Colored lines show expected drift distances for 3 different drift rates. (C) Effect of changing the model's drift regularization parameter $\boldsymbol{Q}$. We only considered isotropic matrices $\boldsymbol{Q} = q\boldsymbol{I}$. Top: Test-set log-likelihood ratio (LLR per spike) comparing the drifting vs. stationary model for all units. Bottom: Overall distribution of the best-fitting q for each unit. (D) If drift is not accounted for, it produces an apparent "smearing" of the spike distribution in feature space. This panel shows the fraction of spike variability that can be accounted for by cluster drift.

a stationary alternative; values greater than zero indicate that the MoDT model provided a better fit.

In this analysis we considered only isotropic matrices $\boldsymbol{Q} = q\boldsymbol{I}$, where q is a positive scalar and $\boldsymbol{I}$ is the identity matrix. When $q = 0$, the MoDT model is equivalent to a stationary (non-drifting) mixture model. As we increase q, we allow more drift in the model, and initially we find that this improves the quality of fit for all units. If we further increase q, we find that the quality of fit eventually diminishes due to overfitting (see Appendix E.4 for more detail).

The optimal value of q varies across units (Fig. 5C, bottom) and depends on the stability of the tetrode and the firing rate of the unit. Since we use the same value of q for all units, we chose a relatively low value ($2\ \mu V^2/h$), which is lower than optimal for many units but still outperforms a stationary model for the vast majority of units. This produces a smoothed estimate that may not follow all of the fluctuations in cluster location, but is still able to capture slower trends (see e.g. Fig. 1C). Despite this excessive smoothing, we still find that cluster drift accounts for 12–30% of the spike variability observed in longer recordings (Fig. 5D).

3.2. Heavy-tailed residuals in empirical data

We also quantified the heavy-tailed distributions of the spike clusters. First, we note that these heavy tails are present even in the extracellular background noise when no spikes are detected (Fig. 6A). This is consistent with the data shown by previous spike sorting studies, including those that have considered the Gaussian distribution to be an adequate approximation (Fee et al., 1996; Pouzat et al., 2004; Prentice et al., 2011).

However, modeling the spike residuals as a Gaussian distribution dramatically underestimates the fraction of spikes that are located away from the cluster center (Fig. 6B and C). Again, the observed distribution is more consistent with a t-distribution than a Gaussian.

In the MoDT model, the parameter ν is a user-defined constant that controls the heavy-tailedness of the assumed spike distribution. At $\nu = 1$, it corresponds to a Cauchy distribution, which has infinite variance. As $\nu \to \infty$, it approaches a Gaussian distribution. The Gaussian version of the MoDT model is equivalent to the "Mixture of Kalman filters" (Calabrese and Paninski, 2011).

We found that most units were best fit with ν in the range 3–20 (Fig. 6D), with some differences between brain areas and cell types. For comparison, Shoham et al. (2003) reported a range of 7–15 for single-electrode recordings in macaque motor cortex. We performed spike sorting using $\nu = 7$ as this provided a good approximation to both limits of the observed range.

3.3. Consequences of using a stationary model

Cluster drift is a well-known feature of chronic recordings, and many techniques have been proposed to address this phenomenon. A common approach is to break the recording into chunks, perform spike sorting on each chunk independently, and finally link the clusters across time (Bar-Hillel et al., 2006; Tolias et al., 2007; Wolf and Burdick, 2009; Shalchyan and Farina, 2014; Dhawale et al., 2015).

This approach comes with a tradeoff: short chunks may not contain enough spikes from low-firing neurons, but long chunks suffer more from the effects of drifting clusters. We characterized this tradeoff by breaking our recordings into chunks of varying duration, re-fitting each chunk with a stationary model, and analyzing the result. We identified three common failure modes of this approach (Fig. 7): (A) fragmented units due to non-clusterable chunks, (B) loss of isolation between units, and (C) splitting of single units.

Unit fragmentation occurs when a unit cannot be linked across chunks. The proposed linking algorithms do not link units over a gap in activity, so a single non-clusterable chunk will break the chain of linked units. We evaluated this by counting how many spikes a given unit fired within each chunk, and we considered any chunk with fewer than 25 spikes to be non-clusterable for that unit (Fig. 7A). Fig. 7D shows the overall fraction of non-clusterable chunks (dashed green line) and the fraction of units that are thus fragmented (solid green line).

Longer chunks are therefore needed to ensure that each chunk contains enough spikes to prevent unit fragmentation. However, longer chunks expose us to more cluster drift, which can cause a loss of isolation and splitting of single units.

Loss of isolation occurs when two drifting clusters occupy the same region of feature space at different times, which appears as cluster overlap under a stationary analysis (Fig. 7B). Fig. 7D (line B) shows the fraction of our well-isolated units would have failed to meet our quality threshold if had instead used a stationary model to evaluate the unit isolation.

Cluster drift may also produce a multi-modal distribution that leads to a single unit being split into two clusters (Fig. 7C). We quan-

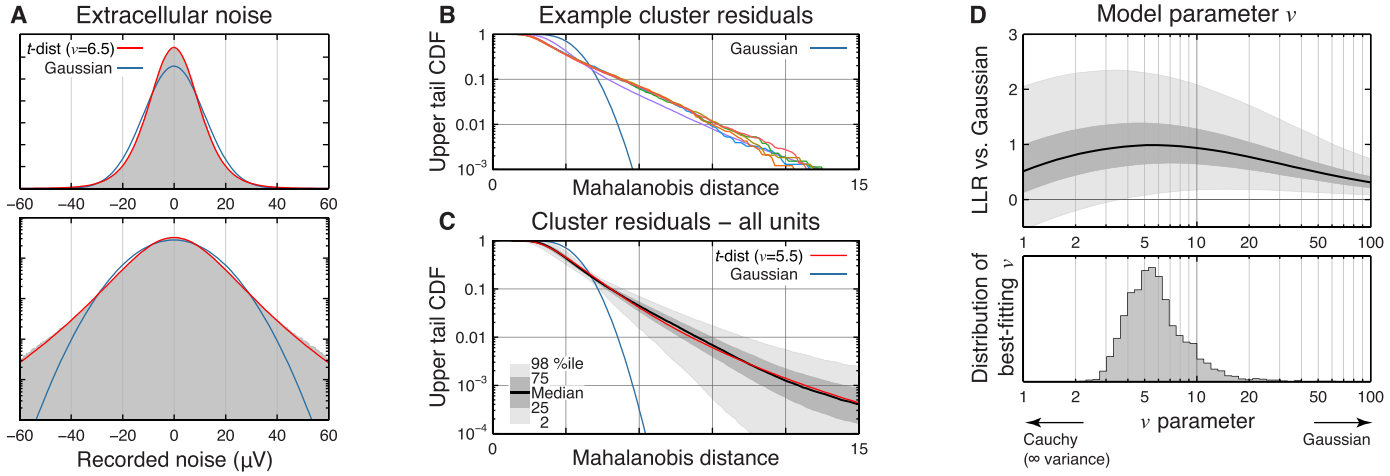

Fig. 6. Heavy-tailed residuals in extracellular noise and well-isolated units. (A) Distribution of extracellular noise for an example tetrode channel during periods where no spikes were detected. Red and blue lines show a t-distribution and Gaussian fit, respectively. Bottom panel shows the same histogram with a logarithmic y-scale. (B) Upper tail CDF (fraction of a unit's spikes that lie beyond a given Mahalanobis distance from the cluster center) for the 6 example units from Fig. 1. (C) Upper tail CDF for all well-isolated units. Shaded areas indicate quantiles across units. Theoretical distributions for a t-distribution and Gaussian are shown for reference. (D) Effect of changing the model's ν parameter, which controls the heavy-tailedness of the distribution. Top: Log-likelihood ratio (LLR per spike) comparing the t-distribution vs. Gaussian model for all units. Bottom: Overall distribution of best-fitting ν for each unit.

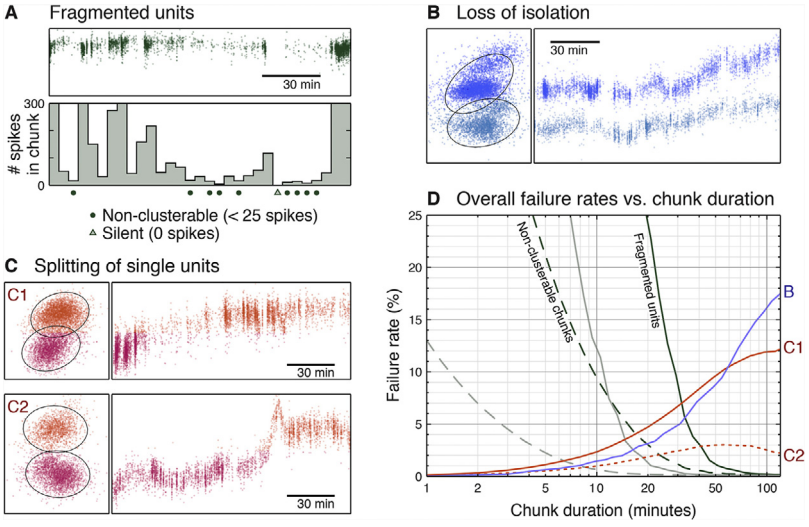

Fig. 7. Failure modes of a stationary approach. An alternative approach to handling cluster drift is to break the recording into chunks, perform spike sorting on each chunk independently, and finally link the spike clusters over time. We identified three common failure modes of this approach. (A) In order to link a unit over time, each chunk must contain enough of that unit's spikes to form a cluster (we used a threshold of 25 spikes). If any chunk is non-clusterable, then the unit cannot be successfully linked and will become fragmented. Panel D shows the overall prevalence of these failures for varying chunk durations (dashed and solid green lines). The light grey lines (dashed and solid) repeat this analysis with the clusterability threshold set at 1 spike. (B) Drift causes clusters to become smeared out over time. If we analyze these irregularly-shaped clusters as stationary distributions, then some units will appear to overlap even though they remain well-isolated over time. This loss of isolation artificially reduces the yield of good units. (C) Drift may produce a multi-modal density distribution. As a result, the clustering algorithm may split a single unit into two clusters (C1). In some cases, these two clusters may be quite well-isolated from each other (C2). (D) Overall prevalence of these failure modes for varying chunk durations.

tified this effect by identifying cases where the Bayes information criterion (BIC) would justify splitting a cluster into two (Fig. 7D, line C1). In some cases, the resulting clusters are well-isolated from one another (less than 5% overlap; Fig. 7D, line C2) and would likely require timing information to identify them as a spuriously split unit.

These tradeoffs are faced by any approach, whether model-based or not, that performs spike sorting on each chunk independently. Although the MoDT model also uses discrete time frames, it avoids this tradeoff by aggregating data across frames: it uses the same cluster scale matrix $\boldsymbol{C}$ for all time frames and incorporates a drift regularizer that effectively smoothes the estimated cluster location $\boldsymbol{\mu}$ over time. As a result, it is able to track units regardless of how few spikes it may fire in a given time frame, which enables us to use sufficiently short time frames (1 min) that the effects of drift are negligible.

Note that some other approaches also avoid this tradeoff. These also use models that allow for gradual parameter variation and disallow splitting or merging of clusters over time (Pouzat et al., 2004; Franke et al., 2010; Calabrese and Paninski, 2011; Carlson et al., 2013).

3.4. Spike sorting performance on hybrid ground truth datasets

To quantify the performance of our spike sorting toolchain, we generated 450 hybrid ground truth datasets by injecting known spikes into existing datasets. We then detected the spikes, assigned them to putative single units, and measured the resulting error

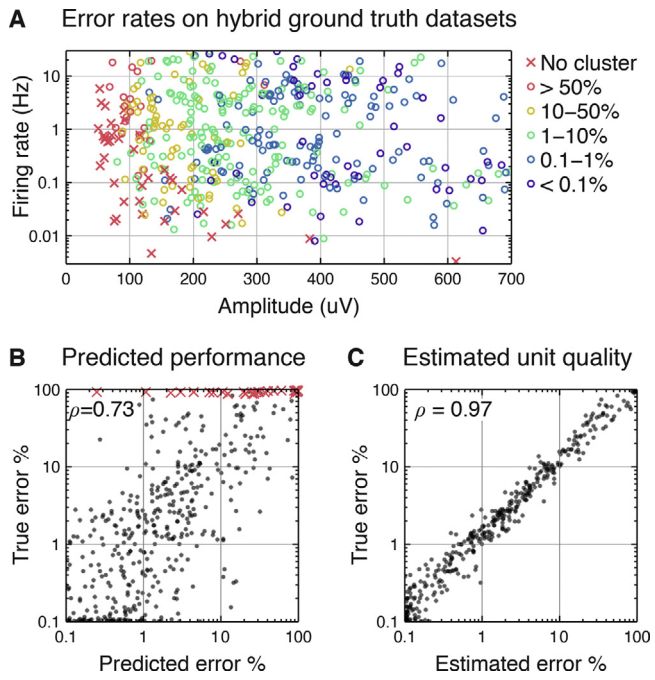

Fig. 8. Spike sorting performance. (A) Total error rates (false positives + false negatives) on 450 hybrid ground truth datasets. The location of each dot indicates the amplitude and firing rate of the injected unit, and its color indicates the total error rate for that unit. Red X's indicate cases where no single cluster corresponds to the injected unit. (B) Although performance is correlated to many attributes of the injected unit, it is difficult to predict the spike sorting performance based on these attributes alone. ρ is Spearman's rank correlation. (C) The unit quality metrics described in Section 2.9 provide an accurate estimate of the true error rate.

rates. Unsurprisingly, the results varied widely and depended on the amplitude and firing rate of the injected units (Fig. 8A).

To see how the performance correlates with these attributes of the injected units, we performed a linear regression on the log-transformed attributes, which yielded the predictor

$$E = 3.5 \times 10^{8}\, A^{-3.5}\, FR^{-0.20}\, DR^{0.17},$$

where E is the predicted error rate (%), A is spike peak amplitude (μV), FR is firing rate (Hz), and DR is drift rate (μV^2/h). Each of these regression coefficients is significant at a $P < 0.01$ level. Although this regression describes the overall trends, it still fails to predict spike sorting performance on a case-by-case basis (Fig. 8B).

These results underscore the difficulty of making general claims about spike sorting performance. Given the variety of experimental conditions—brain area, cell type, probe geometry, electrode impedance, presence of artifacts, etc.—it is difficult if not impossible to guarantee that a particular spike sorting algorithm or parameter set will achieve a given performance specification in all circumstances.

Instead, performance must be evaluated on a case-by-case basis, and in the absence of ground truth, we must rely on quantitative estimates of unit quality. The metrics described in Section 2.9 accurately estimate the true error on these hybrid ground truth datasets (Fig. 8C).

3.5. Comparative analysis of unit isolation metrics

The quality of unit isolation is a major component of this estimated error rate, and a number of unit isolation metrics have previously been proposed. Isolation distance and L-ratio (Schmitzer-Torbert et al., 2005) are two such metrics that have found widespread use. Below we consider how well these measures compare to estimates of misclassification error derived from a Gaussian model (Hill et al., 2011), K-means consensus (Fournier et al., 2016), and the MoDT model we propose.

Isolation distance and L-ratio are based on the Mahalanobis distance δ_{nk} from cluster k to spike n. If there are N_k spikes in cluster k, then its isolation distance is the N_kth smallest value of δ_{nk}^2 among the spikes not assigned to that cluster. L-ratio is defined as L/N_k, where L is the sum, over all spikes n not assigned to cluster k, of the complementary CDF of a χ_D^2 distribution evaluated at δ_{nk}^2. This summand can be interpreted as the P-value, using the Mahalanobis distance as the test statistic, under the null hypothesis that the given spike came from a Gaussian distribution fitted to the spikes assigned to cluster k.

Misclassification errors are spikes assigned to a cluster that should have been assigned to a different cluster. These may be estimated from a generative model as we describe in Section 2.8; we tested both a Gaussian distribution and a t-distribution with $\nu = 7$. K-means consensus is a non-model-based approach in which K-means is used to partition the data based on their Euclidean distance in feature space. This is repeated multiple times from random initializations, and the estimated misclassification error is computed from the fraction of a given cluster's spikes that have been co-partitioned with other clusters' spikes.

We compared these metrics in three ways. First, we noted that the spike clusters in our empirical data varied in size (number of spikes N_k) and scale (overall waveform variability trace($\boldsymbol{C}_k$)), and investigated whether the metrics would be sensitive to this aspect of the data (Fig. 9A). We synthesized clusters using a t-distribution ($\nu = 5.5$) in a 12-dimensional feature space and used this to compare the output of the unit isolation metrics with the true misclassification error. By repeating this simulation using clusters of different sizes and scales, we can analyze the sensitivity of the metrics to these extraneous factors (Fig. 9B).

Second, we evaluated each of the five metrics on our 450 hybrid ground truth datasets (Fig. 9C). In contrast to Fig. 8, we are considering only misclassification errors and not false negatives due to spike detection or censoring.

Finally, we compared the outputs of the first 4 metrics to that of the t-distribution model using our empirical data (Fig. 9D). Since the ground truth is not known, this comparison serves only to illustrate whether the t-distribution model agrees with these other isolation metrics.

This analysis shows that isolation distance suffers from one important flaw: the contaminating cluster is completely ignored if it contains fewer spikes than the cluster being measured. In such cases, the isolation distance is determined by the location of the second-nearest cluster, and may be arbitrarily large. As a result, a large isolation distance does not imply a low misclassification error, particularly for units with many spikes.

L-ratio is a more informative metric, but its value can be difficult to interpret. The relationship between the L-ratio and the true error rate is highly dependent on the dimensionality of the feature space and the heavy-tailedness of the distribution (Fig. C.1). This makes it difficult to compare L-ratio thresholds across experimental settings unless the underlying noise statistics are known.

The remaining metrics—estimated misclassification error based on a Gaussian model, K-means consensus, or a $\nu = 7$ t-distribution model—are easier to interpret and offer fairly similar performance. The Gaussian model's estimates are less accurate for very well-isolated units or when estimating FP and FN separately (Fig. C.2), but such inaccuracies may be irrelevant if such units already meet the necessary criteria on unit isolation quality.

K-means consensus worked very well on the synthetic clusters and hybrid ground truth datasets. However, we encountered a few issues when applying it to empirical data. First, we had to break long datasets into chunks in order to account for drift, and were thus faced with the tradeoff analyzed in Fig. 7. During periods when

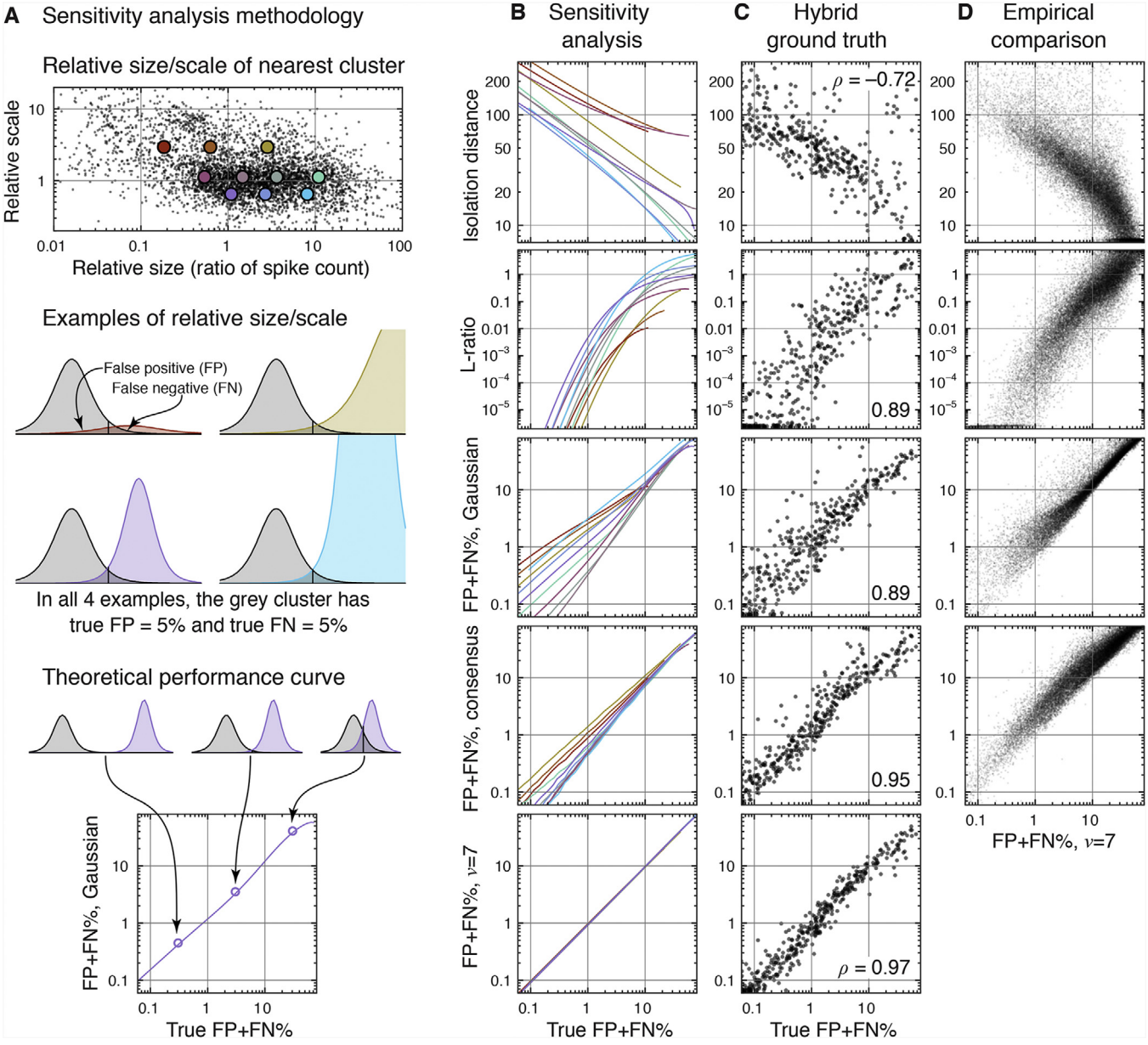

Fig. 9. Comparative analysis of unit isolation metrics. (A) Sensitivity analysis methodology. *Top*: Potential sources of contamination (i.e. neighboring clusters) come in a variety of sizes and scales. We selected 10 representative cases (colored dots) for our sensitivity analysis. *Middle*: We performed our sensitivity analysis by generating synthetic clusters and comparing our unit isolation metrics to the true misclassification error. These four examples have the same true error, but use a different size/scale for the contaminating cluster. *Bottom*: As we move the contaminating cluster closer or farther away, we trace out a curve relating the unit isolation metric to the true error. We repeat this process for each of the 10 cases, producing the colored curves in panel B. (B) Sensitivity analysis results. Ideally, each of the 10 colored curves should lie on top of one another, indicating that the metric is not sensitive to these theoretical variations in cluster size and scale. (C) Validation of isolation metrics using hybrid ground truth datasets. Ideally, the metric should be monotonic with the true misclassification error. ρ is Spearman's rank correlation. (D) Comparison of isolation metrics on experimental data. These panels show how the t-distribution compares to the other unit isolation metrics over all 48,620 fitted clusters.

a unit is silent, it is assigned zero false negatives and it cannot contribute to other units' false positive counts. Second, we encountered unreliable estimates for units that contributed a small fraction of the overall spikes. In such cases, the lack of a consistent K-means partitioning was due to the small number of spikes in the cluster rather than overlap with neighboring clusters. Finally, this metric took much longer to compute than the other metrics tested.

Finally, the $\nu = 7$ t-distribution model performs nearly perfectly in the sensitivity analysis, but this is not surprising since these data were synthesized using a similar distribution. However, we also found that it continues to provide accurate estimates over a wide range of ν, including the Gaussian case (Fig. C.1), indicating that it is relatively insensitive to the underlying spike distribution. This was also the most accurate metric on the hybrid ground truth datasets.

4. Discussion

4.1. Spike sorting for chronic recordings

Continuous recordings over the course of days or weeks can be a powerful tool for studying the long-term dynamics of neural firing properties (Harris et al., 2016). This requires us to track single units over long periods of time, and model-based clustering using the MoDT model is well-suited for this task.

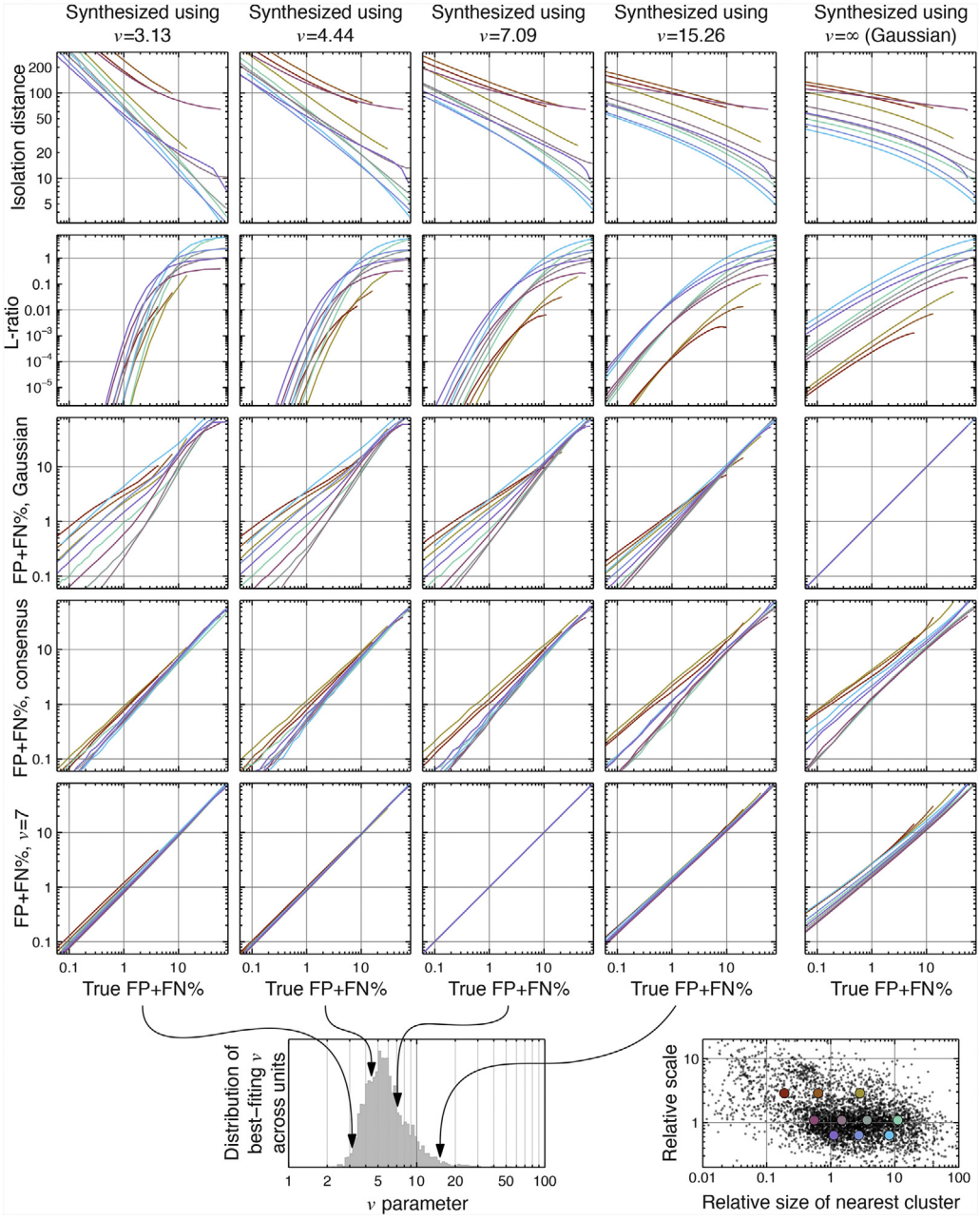

Fig. C.1. Sensitivity analysis of unit quality metrics with varying tail distribution. We repeated our sensitivity analysis (Fig. 9B) using a range of ν for the synthetic clusters. The ν parameter controls the heavy-tailedness of the t-distribution, with smaller values corresponding to heavier tails. We chose four values of ν based on the distribution we observed in our well-isolated units (bottom left), and also included a Gaussian case for reference. Note that the values of isolation distance and L-ratio vary dramatically over this range of ν. The estimated FP + FN using a Gaussian model provides exact results when the data are Gaussian, but becomes sensitive to the relative size/scale of the contaminating cluster when the data come from a heavy-tailed distribution. In contrast, the estimated FP+FN using a t-distribution model remains insensitive to cluster size/scale and provides a relatively accurate estimate across the range of tested parameters.

In particular, establishing when a neuron is silent is often just as important as knowing when it is firing. However, this involves demonstrating an absence of spikes in the region of feature space where we expected to see them, and this fundamentally requires a model-based approach.

This is why we did not consider the possibility of linking over non-clusterable chunks in our earlier analysis (Section 3.3). Although the proposed linking algorithms could be modified to link over a non-clusterable chunk, doing so poses a problem when using the sorted spike trains to draw conclusions about neural activity. Linking a unit over a non-clusterable chunk would imply that it was silent during this period. However, one's inability to cluster a unit in a given chunk does not certify that it was silent; it could have fired insufficient spikes to warrant its own cluster or it could have been spuriously merged into another cluster.

In contrast, the MoDT model effectively interpolates the cluster's expected location between consecutive "sightings" of the unit, giving us a reasonable guarantee that the lack of spikes assigned to this unit in the intervening period is indeed due to its silence. Although linking is still necessary for continuous recordings, the MoDT model simplifies the linking process by allowing us to perform spike sorting in segments up to 10 h in duration. The use of longer segments ensures that all units will fire enough spikes to be clustered, reduces the number of segments that need to be linked, and enables the use of overlapping segments. For example, a week-long recording can be broken into 21 ten-hour segments with an

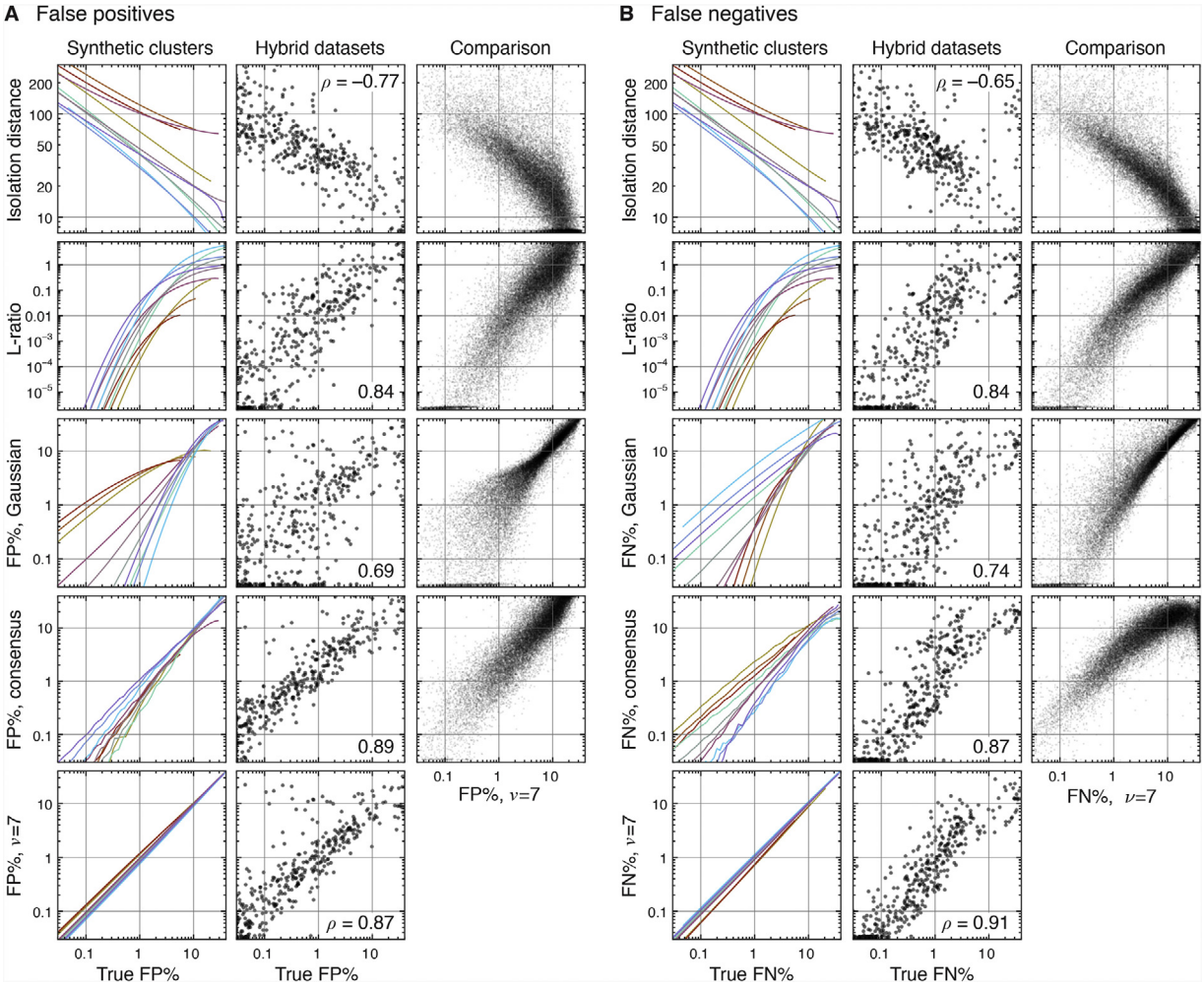

Fig. C.2. Separate analysis of false positive and false negative estimates. We performed the analysis in Fig. 9 separately for the false positive (FP) and false negative (FN) estimates. Note that the Gaussian model and K-means consensus both show increased sensitivity to the size/scale of the contaminating cluster. However, the effects on the FP and FN estimates tended to be opposite in sign for a given case, partially canceling when we consider the sum FP + FN, as in Fig. 9.

overlap of 2 h each. We can then establish cluster correspondences based on the spike assignments of the overlapping data.

4.2. Measuring unit isolation quality in chronic recordings

The analysis of long recordings also requires unit quality metrics that can handle drift. The MoDT model accomplishes this by explicitly tracking the clusters over time. The use of a *t*-distribution also provides a natural robustness to outliers (Fig. 2) and produces accurate estimates of misclassification error over a wide range of conditions (Fig. 9).

However, the MoDT model is still a highly structured model. Each cluster is elliptically symmetric with a predetermined tail distribution, and the drift regularization discourages sudden changes in the cluster's location. It is only through slow drift over time that we can trace out an irregularly-shaped cluster in feature space (e.g. Fig. 7B). In contrast, non-parametric approaches allow clusters to take on arbitrary shapes, which may require additional review to ensure that they correspond to biophysically plausible spike distributions.

Furthermore, it is important to acknowledge that unit isolation quality is a time-varying quantity. Drift may cause two clusters to be well-separated at one point in time, but begin to overlap later. If subsequent analyses are restricted to a particular subset of the overall recording, then the unit isolation measures should be based on those epochs as well. Model-based approaches accommodate this requirement by providing a continuous estimate of misclassification error, which may then be integrated over the appropriate epochs.

Finally, we would like to caution that isolation quality is only one aspect of unit quality overall. Section 2.9 (see also Hill et al., 2011) describes a number of additional quality measures. For example, estimating false negatives due to spike detection is an equally important yet frequently overlooked metric. This is especially important in the presence of cluster drift, as fluctuations in spike amplitude may affect detection efficiency, which could manifest as apparent changes in firing rate.

4.3. Model extensions

The MoDT model we have presented consists of three components: a spike distribution model, a drift regularizer, and an EM fitting algorithm. These components may be modified or extended in several ways.

For example, we modeled the spike distribution using a *t*-distribution, which is elliptically symmetric. However, some neurons fire bursts in which subsequent spikes exhibit a reduced amplitude, producing a skewed distribution that has a longer tail in one direction. This one-dimensional skew may be modeled using

a restricted multivariate skew t-distribution, which can be fitted using an EM algorithm (Lee and McLachlan, 2014).

We also used a very simple form of drift regularization, but this could be replaced with a more sophisticated model. High-density probes may benefit from a model that explicitly accounts for correlated changes in cluster location due to physical motion of a rigid multi-site probe. This would improve tracking of neurons with a low firing rate.

Finally, the EM algorithm has been widely studied and improved upon in many ways. The basic algorithm is well-suited for large-scale data processing and is amenable to parallel computing on GPU hardware (Fig. 3) or distributed computing using high-level data flow engines (Meng et al., 2016). As we consider applying this model to data collected from high-density probes, a variety of algorithmic approximations may also be useful to consider (Appendix D).

The MoDT model thus offers a modular framework that may be readily adapted to experimental needs.

5. Conclusion

In this paper we have described a mixture of drifting t-distributions (MoDT) model, which captures two important features of experimental data—cluster drift and heavy tails—and is robust to outliers. When used for spike sorting, the MoDT model can increase unit yield by separating clusters that appear to overlap and decrease user workload by reducing the incidence of clusters that are spuriously split due to drift. As a unit isolation metric, this model provides accurate estimates of misclassification error over a wide range of conditions. These features, along with a computationally efficient EM algorithm, make this well-suited for analysis of long datasets.

Acknowledgements

We thank all the lab members who contributed user-hours towards data collection and spike sorting (C. Wierzynski, A. Hoenselaar, M. Papadopoulou, B. Sauerbrei). Special thanks to Alex Ecker for development of the `moksm` MATLAB package, and Andreas Hoenselaar for development of the clustering GUI.

This work was supported by the Mathers Foundation, the Moore Foundation, NSF grants 1546280, 1146871, NIH grants 1DP1OD008255/5DP1MH099907, 1R01MH113016, and iARPA contract D16PC00003.

Appendix A. Additional comments on the M-step μ update

In this appendix we discuss the optimization of $\boldsymbol{\mu}$ performed by Eq. (6) and compare it to other techniques.

First, note that the optimization of $\boldsymbol{\mu}$ depends on the value of $\boldsymbol{C}$ and vice-versa. Although the standard EM algorithm calls for maximizing $J(\phi, \hat{\phi})$ over all ϕ, the convergence of a generalized EM algorithm requires only that we improve upon the previous $\hat{\phi}$ (Dempster et al., 1977). Therefore, we need not simultaneously optimize $\boldsymbol{\mu}$ and $\boldsymbol{C}$, but may instead optimize them one at a time.

The optimization of $\boldsymbol{\mu}$ could also be performed using a Kalman filter with a Rauch–Tung–Striebel backwards pass (Calabrese and Paninski, 2011). Since we have multiple observations per time frame, the forward pass can be performed more efficiently using an alternative parameterization of the Kalman filter known as the *information filter* (see e.g. Anderson and Moore, 1979). In fact, our $\boldsymbol{M}_{kt}$ and $\boldsymbol{b}_{kt}$ correspond to the information matrix and vector, respectively, of the information filter's observation update step. Nonetheless, we have found that it is faster to solve Eq. (6) directly using standard numerical linear algebra routines (i.e. LAPACK `dpbsv`).

For more insight into this optimization, consider the unregularized case ($\boldsymbol{Q} \to \infty$ and hence $\boldsymbol{Q}^{-1} = \boldsymbol{0}$). In this scenario, the time frames are independent of one another (the $\boldsymbol{A}$ matrix is block diagonal) and the solution to Eq. (6) is simply

$$\boldsymbol{\mu}_{kt} = \bar{\boldsymbol{y}}_{kt} = \frac{\sum_{n:t_n=t} w_n z_{nk} u_{nk} \boldsymbol{y}_n}{\sum_{n:t_n=t} w_n z_{nk} u_{nk}}, \tag{A.1}$$

i.e. the weighted sample mean of spikes assigned to cluster k in time frame t.

As we add regularization, the optimal $\boldsymbol{\mu}$ becomes a temporally smoothed version of this weighted average. Consider for example the case where $\boldsymbol{Q}$ and $\boldsymbol{C}_k$ are isotropic, i.e. $\boldsymbol{Q} = q\boldsymbol{I}$ and $\boldsymbol{C}_k = c\boldsymbol{I}$, where $\boldsymbol{I}$ is the identity matrix. In this scenario, we can rewrite equation (6) as

$$\boldsymbol{\mu}_{kt} = \frac{N_{kt}\bar{\boldsymbol{y}}_{kt} + (c/q)\,\boldsymbol{\mu}_{k(t-1)} + (c/q)\,\boldsymbol{\mu}_{k(t+1)}}{N_{kt} + (c/q) + (c/q)}, \tag{A.2}$$

where $N_{kt} = \sum_{n:t_n=t} w_n z_{nk} u_{nk}$ is the weighted number of spikes assigned to this cluster in this time frame. Eq. (A.2) is analogous to a bi-directional exponentially-weighted moving average. Each $\boldsymbol{\mu}_{kt}$ is a weighted average of that time frame's sample mean $\bar{\boldsymbol{y}}_{kt}$ and its neighbors $\boldsymbol{\mu}_{k(t-1)}$ and $\boldsymbol{\mu}_{k(t+1)}$.

In time frames with no spikes ($N_{kt} = 0$), $\boldsymbol{\mu}_{kt}$ simply takes the midpoint between its neighbors. As we increase the number of spikes ($N_{kt}\uparrow$), $\boldsymbol{\mu}_{kt}$ places more weight on its own sample mean rather than interpolating between its neighbors. This is evident when comparing the high- and low-firing units in Fig. 1C. Likewise, tightening the cluster variance ($c\downarrow$) or increasing the expected drift ($q\uparrow$) will also shift the average in Eq. (A.2) in favor of the sample mean.

Finally, note that the equations in Section 2 and in this appendix use units of [feature space units]^2/[time frame] for $\boldsymbol{Q} = q\boldsymbol{I}$. In contrast, Fig. 5 and Section 3.1 report q in units of [feature space units]^2/h for ease of interpretation.

Appendix B. Supplementary methods for overlapping spikes

This appendix contains additional comments on our methods for handling overlapping spikes (Section 2.6).

First, note that for the sake of clarity, Eq. (7) shows only the single-channel case and omits the weighting factor that corrects for double-counting the error in the overlap window.

Also note that the hybrid ground truth dataset analyzed in Fig. 4E was created slightly differently from the datasets described in Section 2.10. Instead of choosing a donor unit and acceptor dataset from the same tetrode on different days, we selected 12 simultaneously-recorded donor units from different tetrodes so that we could preserve the temporal structure of the spikes. The average firing rates of the donor units ranged from 0.15 to 26.4 Hz, with a combined firing rate of 76.4 Hz. The acceptor dataset was relatively quiet, with a spike detection rate of 1.5 Hz.

B.1 Overlap clusters

In this section, we describe how we augmented the model with overlap clusters. To generate the overlap cluster corresponding to units k_1 and k_2 firing with a particular temporal offset, we first need to determine their temporal offsets τ_1, τ_2 relative to the detected spike time. To do this, we construct an overlap waveform by overlapping their individual mean waveforms $\boldsymbol{W}\boldsymbol{\mu}_{k_1}$ and $\boldsymbol{W}\boldsymbol{\mu}_{k_2}$. We then pass this through the spike detection algorithm to determine the center of the detected spike. For example, all of the overlap waveforms in Fig. 4D ended up aligned to the blue spike.

This gives us the transformation matrices

$$\begin{aligned}\boldsymbol{U}_1 &= \boldsymbol{W}^\top \boldsymbol{T}_{\tau_1} \boldsymbol{W} \\ \boldsymbol{U}_2 &= \boldsymbol{W}^\top \boldsymbol{T}_{\tau_2} \boldsymbol{W},\end{aligned}$$

where $\boldsymbol{T}_\tau$ is the $P \times P$ matrix corresponding to a temporal shift by τ samples. These can be used to derive the model parameters of the overlap cluster:

$$\begin{aligned}\boldsymbol{\mu} &= \boldsymbol{U}_1 \boldsymbol{\mu}_{k_1} + \boldsymbol{U}_2 \boldsymbol{\mu}_{k_2} \\ \boldsymbol{C} &= \boldsymbol{U}_1 \boldsymbol{C}_{k_1} \boldsymbol{U}_1^\top + \boldsymbol{U}_2 \boldsymbol{C}_{k_2} \boldsymbol{U}_2^\top \\ \alpha &= \alpha_{k_1} \alpha_{k_2} \beta.\end{aligned}$$

β is the probability of any two units firing with the given temporal offset, and depends on the overall spike detection rate.

However, the number of possible cluster combinations is quite large (on the order of K^2) and is compounded by the number of temporal offsets that we need to consider (we used 21 offsets from −0.4 to +0.4 ms). We pruned the number of overlap clusters by evaluating the posterior probability (z_{nk}) that a spike located at the overlap cluster's center belongs to that cluster vs. one of the base clusters, and keeping only the overlap clusters with the largest posterior. On our benchmarking computer (Section 2.3), it took 2.9 s to generate and prune the overlap clusters (from a base model with $K = 17$) and 68.8 s to process the test dataset (4.2 million spikes) with 750 overlap clusters.

This method works best when the overlapping spikes occur simultaneously (inter-spike interval = 0 in Fig. 4E), because that provides the best conditions for feature extraction. When the spikes are slightly offset (∼0.3 ms), the contribution of the second spike may be nearly orthogonal to the PCA axes used in feature extraction, resulting in poor discrimination.

B.2 Template matching

For the template-matching approach analyzed in Fig. 4E, we used the hybrid Bayes-optimal template matching algorithm (Franke et al., 2015). This template matching algorithm assumes homoscedasticity, which allows for efficient computation of the relative posterior likelihood, and uses a "hybrid" approach combining a partial enumeration of the potential overlaps with iterative greedy refinement.

This method achieved a high detection rate regardless of inter-spike interval, but required careful tuning of the detection threshold. The reported performance should be interpreted as a best-case scenario, as it involved manual titration of the cluster-specific "prior" parameter to minimize the error rates on the test dataset. Performing this sort of parameter optimization on experimental data—for which the ground truth is not known—would require the development of quantitative unit quality metrics that are suitable for use with template-based spike detection.

This method also sacrificed performance on non-overlapping spikes (0.7% false negatives vs. 0.2% using traditional clustering), possibly because the bursting neurons and high signal-to-noise ratio violate the algorithm's assumption of homoscedasticity. Other template matching approaches allow for limited heteroscedasticity in the form of amplitude variability (Prentice et al., 2011; Pachitariu et al., 2016) or a second axis of variability (Yger et al., 2016).

Appendix C. Additional analysis of unit isolation metrics

In this appendix we perform additional analysis of unit isolation metrics to supplement Fig. 9.

Fig. C.1 repeats the sensitivity analysis of Fig. 9B using a range of distributions for the synthetic clusters. As we vary the heaviness of the distribution tails, we find that the relationship between the true error rate and the isolation distance or L-ratio can vary dramatically. For example, an L-ratio of 0.01 corresponds to an FP+FN between 1.5–6% for a heavy-tailed distribution ($\nu = 3.1$), but could range anywhere from 0.1 to 10% for a Gaussian distribution. The values of these metrics also depend strongly on the dimensionality of the feature space. In contrast, the $\nu = 7$ model-based estimate (and the consensus-based estimate, to a lesser degree) remains insensitive to the cluster size/scale and provides an accurate estimate of misclassification error across the range of distributions.

Fig. C.2 repeats the analyses from Fig. 9 for the FP and FN separately. Both the Gaussian model and K-means consensus show increased sensitivity to the size/scale of the contaminating cluster. Performance on the hybrid ground truth datasets was also affected.

Appendix D. EM algorithm for high-density probes

As we consider applying this model to data collected from high-density probes, we are faced with potential increases in all 3 dimensions D, K, and N. In order to mitigate this increase in complexity, a number of algorithmic modifications may be useful to consider.

First, we can take advantage of the fact that spikes have a limited spatial extent. Masked EM (Kadir et al., 2014) reduces the effective D in these situations and can prevent the fitted scale matrix $\boldsymbol{C}$ from becoming ill-conditioned.

Next, we found that the first 5 eigenvectors of $\boldsymbol{C}$ typically explained 95% or more of the cluster's variability, and the remaining eigenvectors were dominated by homoscedastic noise. This suggests that after whitening, the matrix $\boldsymbol{C}$ could be approximated as the sum of a low-rank and an isotropic component (Magdon-Ismail and Purnell, 2010). Using this approximation reduces the E-step complexity from D^2KN to DKN and further reduces the risk of ill-conditioned scale matrices.

We also found that most of the z_{nk} are very close to zero. By ignoring those spikes during the M-step, we can reduce the complexity of the M-step from D^2KN to D^2N. We found that applying a threshold on z_{nk} produced more accurate results than "hard EM" (in which each spike is assigned to only one cluster), which tends to underestimate the covariance of highly-overlapping clusters.

Taken together, these modifications would reduce the computational complexity of model fitting from D^2KN to $(d^2 + dK)N$, where d is the number of non-masked dimensions.

Finally, we note that EM requires multiple iterations to converge, but some clusters—particularly the well-isolated ones—converge faster than others. Excluding such clusters from subsequent E- and M-steps can reduce the overall fitting time while maintaining the general convergence properties of the EM algorithm (Neal and Hinton, 1998).

Appendix E. Analysis methods

This appendix contains additional details on the analyses performed to generate the figures in this paper.

E.1 Theoretical distribution of Mahalanobis distance

Fig. 1D shows the distribution of the non-squared Mahalanobis distance x computed using the fitted $\boldsymbol{\mu}$ and the cluster-specific sample covariance $\boldsymbol{\Sigma}$

$$x = \sqrt{\delta^2(\boldsymbol{y}; \boldsymbol{\mu}, \boldsymbol{\Sigma})}.$$

If $\boldsymbol{y}$ come from a multivariate Gaussian distribution and we assume that $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ equal the true mean and covariance, then x^2 will be distributed according to a chi-squared distribution with D degrees of freedom. We can apply a change of variables to obtain

the theoretical probability distribution function (PDF) of the non-squared x:

$$f_{\text{Gauss}}(x) = 2x f_{\chi^2}\left(x^2; D\right),$$

where $f_{\chi^2}(x; k)$ denotes the PDF of a chi-squared distribution with k degrees of freedom. This $f_{\text{Gauss}}(x)$ is the dashed line in Fig. 1D, and its corresponding cumulative distribution function (CDF) is shown in Fig. 6B and C.

If $\boldsymbol{y}$ come from a multivariate t-distribution with ν degrees of freedom, then the quantity $(1/D)\,\delta^2(\boldsymbol{y};\ \boldsymbol{\mu},\ \boldsymbol{C})$ will be distributed according to an F distribution with $(D,\ \nu)$ degrees of freedom (Box and Tiao, 1973). However, our Mahalanobis distance x is computed using the sample covariance $\boldsymbol{\Sigma}$ rather than the t-distribution scale parameter $\boldsymbol{C}$. If we assume that the sample covariance $\boldsymbol{\Sigma}$ equals the expected covariance $(\nu/(\nu-2))\boldsymbol{C}$ (this requires $\nu > 2$, as the expected covariance is undefined when $\nu \le 2$), then $\delta^2(\boldsymbol{y};\ \boldsymbol{\mu},\ \boldsymbol{C}) = \nu x^2/(\nu-2)$, and applying this change of variables gives us

$$f_{t-\text{dist}}(x; \nu) = \frac{2\nu x}{(\nu-2)D} f_F\left(\frac{\nu x^2}{(\nu-2)D}; D, \nu\right),$$

where $f_F(x; k_1, k_2)$ denotes the PDF of an F distribution with (k_1, k_2) degrees of freedom. This $f_{t-\text{dist}}(x;\ \nu)$ is the solid line in Fig. 1D, and its corresponding CDF is shown in Fig. 6C.

E.2 Robust covariance estimation

In this paper we have focused on data with heavy tails (Fig. 6), but the t-distribution's robustness to outliers suggests that it may be a useful model even when the data are Gaussian (Fig. 2).

However, fitting a t-distribution to Gaussian data causes us to overestimate the distribution tails. In Fig. 2A, note how the fitted t-distribution's 99% confidence ellipse (green) is inflated relative to the true ellipse (black). In this situation, we would like to use the fitted t-distribution to derive robust estimates of the Gaussian parameters.

As the number of samples $N \to \infty$, the fitted $\boldsymbol{\mu}$ converges to the true mean of the Gaussian distribution. However, the fitted $\boldsymbol{C}$ is a biased estimator of the true covariance $\boldsymbol{\Sigma}$. If we assume the data are Gaussian, we can compute a correction factor by solving for β in the following:

$$\int_0^\infty \frac{\nu + D}{\beta\nu + x} x^{D/2} e^{-x/2}\, dx = 2^{D/2}\Gamma(\frac{D}{2})D.$$

We can then use $\beta\boldsymbol{C}$ as a robust estimate for the Gaussian covariance. The corresponding 99% confidence ellipse is actually shown in Fig. 2A (light grey ellipse in right panel), but it is visually obscured by the 99% confidence ellipse of the true distribution (black).

E.3 Relative influence of single spikes on fitted parameters

In Fig. 2B we analyze the relative influence of a single spike on the fitted parameters. This appendix describes how this "relative influence" is computed.

The top panel shows the relative influence on the fitted $\boldsymbol{\mu}$ as a function of the spike's distance from the cluster center. To compute this quantity, let $\hat{\boldsymbol{\mu}}$ denote the fitted cluster location with N spikes, and consider the effect of adding a spike $\boldsymbol{y}_{N+1}$. In the case of a single cluster and a single time frame, the M-step update (Eq. (6)) gives us

$$\boldsymbol{\mu} = \frac{\left(\sum_n u_n \boldsymbol{y}_n\right) + u_{N+1}\boldsymbol{y}_{N+1}}{\left(\sum_n u_n\right) + u_{N+1}}.$$

If the new spike is located at a distance $\|\boldsymbol{y}_{N+1} - \hat{\boldsymbol{\mu}}\| = d$ from the cluster center, then the change in the fitted $\boldsymbol{\mu}$ is

$$\|\boldsymbol{\mu} - \hat{\boldsymbol{\mu}}\| = \frac{u_{N+1}\|\boldsymbol{y}_{N+1} - \hat{\boldsymbol{\mu}}\|}{\left(\sum_n u_n\right) + u_{N+1}} = \frac{u_{N+1}\, d}{\left(\sum_n u_n\right) + u_{N+1}}.$$

We will denote this quantity $I_{\boldsymbol{\mu}}(d)$, the influence of a single spike at a distance d. For simplicity, let us assume that N is large (and hence $\sum_n u_n \gg u_{N+1}$) and that the cluster scale $\boldsymbol{C} = c\boldsymbol{I}$. Under these assumptions,

$$I_{\boldsymbol{\mu}}(d) = \|\boldsymbol{\mu} - \hat{\boldsymbol{\mu}}\| \approx \frac{1}{\sum_n u_n} \frac{(\nu + D)\, d}{\nu + d^2/c}.$$

Note that $I_{\boldsymbol{\mu}}(d) \to 0$ as $d \to \infty$.

In Fig. 2B we consider the case where the true data are distributed according to a standard multivariate normal distribution ($\boldsymbol{\Sigma} = \boldsymbol{I}$). For a given ν, the fitted cluster scale can be determined using the procedure in Appendix E.2. The top panel of Fig. 2B shows the relative influence $I_{\boldsymbol{\mu}}(d)/\mathbb{E}[I_{\boldsymbol{\mu}}(d)]$, where $\mathbb{E}[I_{\boldsymbol{\mu}}(d)]$ is the expected value of $I_{\boldsymbol{\mu}}$ over the spike distribution.

The bottom panel of Fig. 2B repeats this analysis for the scale parameter $\boldsymbol{C}$. Again assuming that $\boldsymbol{C} = c\boldsymbol{I}$,

$$I_{\boldsymbol{C}}(d) = \frac{1}{N+1}\sqrt{\left(\frac{(\nu + D)\, d^2}{\nu + d^2/c} - c\right)^2 + (D-1)c^2}.$$

E.4 Model assessment using the likelihood ratio

Figs. 5C and 6 D use the log likelihood ratio (LLR) to evaluate the quality of fit of a MoDT model under various choices for the user-defined model parameters.

The LLR is the logarithm of the likelihood ratio comparing the MoDT model to some alternative (a stationary t-distribution in Fig. 5 and a drifting Gaussian distribution in Fig. 6). Values greater than zero indicate that the MoDT model produced a better fit. We report the LLR divided by the number of spikes so that we may compare across datasets of different sizes.

When evaluating the effect of varying the drift regularization parameter $\boldsymbol{Q}$, it is important to distinguish between fitting the underlying cluster drift and capturing the random noise of the observed spikes. Therefore we performed cross-validation using a holdout, and Fig. 5C reports the LLR evaluated on the validation test set. This was performed by randomly partitioning the spikes into two equal-sized subsets (a training set and a test set) and fitting the models to the training set only. Note that relaxing the drift regularization (i.e. increasing $\boldsymbol{Q}$) will always improve the model's ability to fit the training data. However, the likelihood of the test set increases initially (due to the model's improved ability to track the cluster drift) but eventually decreases due to overfitting to the training set.

References

Anderson, B., Moore, J.B., 1979. Optimal Filtering. Prentice-Hall, Englewood Cliffs, NJ.

Bar-Hillel, A., Spiro, A., Stark, E., 2006. Spike sorting: Bayesian clustering of non-stationary data. J. Neurosci. Methods 157, 303–316.

Blanche, T.J., Swindale, N.V., 2006. Nyquist interpolation improves neuron yield in multiunit recordings. J. Neurosci. Methods 155, 81–91.

Box, G.E.P., Tiao, G.C.C., 1973. Bayesian Inference in Statistical Analysis. Addison-Wesley Pub. Co., Reading.

Calabrese, A., Paninski, L., 2011. Kalman filter mixture model for spike sorting of non-stationary data. J. Neurosci. Methods 196, 159–169.

Carlson, D.E., Rao, V., Vogelstein, J., Carin, L., 2013. Real-time inference for a gamma process model of neural spiking. Adv. Neural Inf. Process. Syst., 2805–2813.

Dempster, A.P., Laird, N.M., Rubin, D.B., 1977. Maximum likelihood from incomplete data via the EM algorithm. J. R. Stat. Soc. 39, 1–38.

Dhawale, A.K., Poddar, R., Kopelowitz, E., Normand, V., Wolff, S.B., Ölveczky, B.P., 2015. Automated long-term recording and analysis of neural activity in behaving animals. bioRxiv, 033266.
Fee, M.S., Mitra, P.P., Kleinfeld, D., 1996. Variability of extracellular spike waveforms of cortical neurons. J. Neurophysiol. 76, 3823–3833.
Feldman, D., Faulkner, M., Krause, A., 2011. Scalable training of mixture models via coresets. Adv. Neural Inf. Process. Syst., 2142–2150.
Fournier, J., Mueller, C.M., Shein-Idelson, M., Hemberger, M., Laurent, G., 2016. Consensus-based sorting of neuronal spike waveforms. PLOS ONE 11, e0160494.
Franke, F., Natora, M., Boucsein, C., Munk, M.H.J., Obermayer, K., 2010. An online spike detection and spike classification algorithm capable of instantaneous resolution of overlapping spikes. J. Comput. Neurosci. 29, 127–148.
Franke, F., Pröpper, R., Alle, H., Meier, P., Geiger, J.R.P., Obermayer, K., Munk, M.H.J., 2015. Spike sorting of synchronous spikes from local neuron ensembles. J. Neurophysiol. 114, 2535–2549.
Harris, K.D., Quiroga, R.Q., Freeman, J., Smith, S.L., 2016. Improving data quality in neuronal population recordings. Nat. Neurosci. 19, 1165–1174.
Hill, D.N., Mehta, S.B., Kleinfeld, D., 2011. Quality metrics to accompany spike sorting of extracellular signals. J. Neurosci. 31, 8699–8705.
Kadir, S.N., Goodman, D.F.M., Harris, K.D., 2014. High-dimensional cluster analysis with the masked EM algorithm. Neural Comput. 26, 2379–2394.
Lee, S., McLachlan, G.J., 2014. Finite mixtures of multivariate skew t-distributions: some recent and new results. Stat. Comput. 24, 181–202.
Lewicki, M.S., 1998. A review of methods for spike sorting: the detection and classification of neural action potentials. Netw. Comput. Neural Syst. 9, R53–R78.
Magdon-Ismail, M., Purnell, J.T., 2010. Approximating the covariance matrix of GMMs with low-rank perturbations. In: Intelligent Data Engineering and Automated Learning – IDEAL 2010, Springer, pp. 300–307.
McLachlan, G., Peel, D., 2000. Finite Mixture Models. John Wiley & Sons.
Meng, X., Bradley, J., Yavuz, B., Sparks, E., Venkataraman, S., Liu, D., Freeman, J., Tsai, D.B., Amde, M., Owen, S., Xin, D., Xin, R., Franklin, M.J., Zadeh, R., Zaharia, M., Talwalkar, A., 2016. MLlib: machine learning in apache spark. J. Mach. Learn. Res., 17.
Mukhopadhyay, S., Ray, G.C., 1998. A new interpretation of nonlinear energy operator and its efficacy in spike detection. IEEE Trans. Biomed. Eng. 45, 180–187.
Neal, R.M., Hinton, G.E., 1998. A view of the EM algorithm that justifies incremental, sparse, and other variants. In: Learning in Graphical Models. Springer, Dordrecht, Netherlands, pp. 355–368.
Pachitariu, M., Steinmetz, N., Kadir, S.N., Carandini, M., Harris, K.D., 2016. Kilosort: realtime spike-sorting for extracellular electrophysiology with hundreds of channels. bioRxiv, 061481.
Paninski, L., Ahmadian, Y., Ferreira, D.G., Koyama, S., Rad, K.R., Vidne, M., Vogelstein, J., Wu, W., 2010. A new look at state-space models for neural data. J. Comput. Neurosci. 29, 107–126.
Peel, D., McLachlan, G.J., 2000. Robust mixture modelling using the *t* distribution. Stat. Comput. 10, 339–348.
Pillow, J.W., Shlens, J., Chichilnisky, E.J., Simoncelli, E.P., 2013. A model-based spike sorting algorithm for removing correlation artifacts in multi-neuron recordings. PLoS ONE 8, e62123.
Pouzat, C., Delescluse, M., Viot, P., Diebolt, J., 2004. Improved spike-sorting by modeling firing statistics and burst-dependent spike amplitude attenuation: a Markov chain Monte Carlo approach. J. Neurophysiol. 91, 2910–2928.
Prentice, J.S., Homann, J., Simmons, K.D., Tkačik, G., Balasubramanian, V., Nelson, P.C., 2011. Fast, scalable, Bayesian spike identification for multi-electrode arrays. PLoS ONE 6, e19884.
Rossant, C., Kadir, S.N., Goodman, D.F.M., Schulman, J., Hunter, M.L.D., Saleem, A.B., Grosmark, A., Belluscio, M., Denfield, G.H., Ecker, A.S., Tolias, A.S., Solomon, S., Buzsáki, G., Carandini, M., Harris, K.D., 2016. Spike sorting for large, dense electrode arrays. Nat. Neurosci. 19, 634–641.
Schmitzer-Torbert, N., Jackson, J., Henze, D.A., Harris, K.D., Redish, A.D., 2005. Quantitative measures of cluster quality for use in extracellular recordings. Neuroscience 131, 1–11.
Segev, R., Goodhouse, J., Puchalla, J., Berry, M.J., 2004. Recording spikes from a large fraction of the ganglion cells in a retinal patch. Nat. Neurosci. 7, 1155–1162.
Shalchyan, V., Farina, D., 2014. A non-parametric Bayesian approach for clustering and tracking non-stationarities of neural spikes. J. Neurosci. Methods 223, 85–91.
Shoham, S., Fellows, M.R., Normann, R.A., 2003. Robust, automatic spike sorting using mixtures of multivariate *t*-distributions. J. Neurosci. Methods 127, 111–122.
Snider, R.K., Bonds, A.B., 1998. Classification of non-stationary neural signals. J. Neurosci. Methods 84, 155–166.
Tolias, A.S., Ecker, A.S., Siapas, A.G., Hoenselaar, A., Keliris, G.A., Logothetis, N.K., 2007. Recording chronically from the same neurons in awake, behaving primates. J. Neurophysiol. 98, 3780–3790.
Udell, M., Horn, C., Zadeh, R., Boyd, S., 2016. Generalized low rank models. Found. Trends Mach. Learn. 9, 1–118.
Ueda, N., Nakano, R., Ghahramani, Z., Hinton, G.E., 2000. SMEM algorithm for mixture models. Neural Comput. 12, 2109–2128.
Wolf, M.T., Burdick, J.W., 2009. A Bayesian clustering method for tracking neural signals over successive intervals. IEEE Trans. Biomed. Eng. 56, 2649–2659.
Yatsenko, D., Reimer, J., Ecker, A.S., Walker, E.Y., Sinz, F.H., Berens, P., Hoenselaar, A., Cotton, R.J., Siapas, A.G., Tolias, A.S., 2015. DataJoint: managing big scientific data using MATLAB or Python. bioRxiv, 031658.
Yger, P., Spampinato, G.L.B., Esposito, E., Lefebvre, B., Deny, S., Gardella, C., Stimberg, M., Jetter, F., Zeck, G., Picaud, S., Duebel, J., Marre, O., 2016. Fast and accurate spike sorting in vitro and in vivo for up to thousands of electrodes. bioRxiv, 067843.