Semi-supervised spike sorting using pattern matching and a scaled Mahalanobis distance metric

Douglas M. Schwarz[a], Muhammad S. A. Zilany[a,b], Melissa Skevington[a], Nicholas J. Huang[a,b], Brian C. Flynn[a,b], Laurel H. Carney[a,b]

[a] *Neurobiology & Anatomy, University of Rochester, Box 603, 601 Elmwood Ave., Rochester, NY 14642, USA*
[b] *Biomedical Engineering, University of Rochester, Box 270168, Rochester, NY 14627-0168, USA*

Abstract

Sorting action potentials (spikes) from tetrode recordings can be time consuming, labor intensive, and inconsistent, depending on the methods used and the experience of the operator. The techniques presented here were designed to address these issues. A feature related to the slope of the spike during repolarization is computed. A small subsample of the features obtained from the tetrode (ca. 10,000–20,000 events) is clustered using a modified version of *k*-means that uses Mahalanobis distance and a scaling factor related to the cluster size. The cluster-size-based scaling improves the clustering by increasing the separability of close clusters, especially when they are of disparate size. The full data set is then classified from the statistics of the clusters. The technique yields consistent results for a chosen number of clusters. A MATLAB implementation is able to classify more than 5000 spikes per second on a modern workstation.

Keywords: classification, clustering, tetrode recording

1. Introduction

The study of neural processing often involves recording action potentials generated by neurons in response to sensory stimuli. Subsequent analysis generally requires determining how many neurons contributed to the observed set of spikes, which spikes were produced by each neuron, and which spikes were spurious, a process known as *spike sorting* (Abeles and Goldstein Jr, 1977; Lewicki, 1994; Fee et al., 1996a; Quian Quiroga et al., 2004; Delescluse and Pouzat, 2006, reviewed by Lewicki, 1998). The task of spike sorting involves capturing spike waveforms, computing *features* of each waveform (statistics such as peak amplitude), and classifying the spikes by grouping spikes with similar features.

It can be difficult to find features that separate the spikes produced by different neurons. One technique is to record from four closely spaced wires using a tetrode rather than a single electrode (Gray et al., 1995). The detection of a spike on any of the four wires triggers the apparatus to record a "snapshot" of all four wires simultaneously and is called an *event*. Because of their physical separation, the four wires receive slightly different signals, which can help differentiate spikes from multiple neurons. It is important to note that these signals are quite small (on the order of tens of microvolts) and distorted by the presence of background electrical activity.

The set of N features computed from an event is called a *feature vector* and can be interpreted as a point in N-dimensional space. If the spikes from different neurons have different shapes, and the features are suitably chosen, then the feature vectors will occupy discernible regions, or clusters, of that space. The purpose of clustering is to examine the feature vectors, determine the number of clusters, k (presumably related to the number of underlying contributing neurons), and classify each event into one of the k clusters (Wheeler and Heetderks, 1982; Schmidt, 1984).

The features may consist of easily computable statistics, such as peak amplitude, or they could simply be all the samples from the waveforms on all four wires. In the latter case, because N is large, it is common to employ a dimensionality-reducing algorithm, such as principal components analysis (PCA) (Glaser and Marks, 1968; Abeles and Goldstein Jr, 1977), independent components analysis (ICA) (Takahashi et al., 2003), or wavelet decomposition (Quian Quiroga et al., 2004). A frequent choice is to keep the first two or three principal components. PCA performs an orthonormal transformation (distance-preserving rotation) of the set of feature vectors in such a way that the first component of the result has the most variance, with subsequent components successively less. PCA is frequently helpful, but *components with large variance are not necessarily components with large separability.* In other words, the information that allows the separation of feature vector clusters might end up in a component deemed insignificant because it has low variance (Fig. 1).

So as not to risk this potential loss of separability information, it is desirable to choose features that result in feature vectors of low enough dimensionality that they do not require dimensionality reduction. One such choice begins with computing the cross-correlation of each spike waveform with a fixed pattern. This operation measures the similarity of the spike waveform to the pattern at each point along the waveform. Selection of the largest value of the cross-correlation gives the best match to the pattern and can be used as a feature. Because

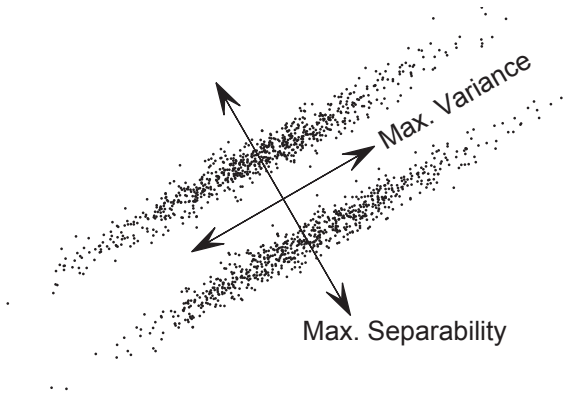


Figure 1: The distinction between variance and separability is illustrated by synthetic data that have been constructed so that the directions of maximum variance and maximum separability are orthogonal. If PCA were applied and the component with the lowest variance discarded, the two clusters would completely overlap and be impossible to distinguish. This simple example is illustrated in two dimensions, but the principle applies to data of any dimensionality.

there are four waveforms per event, such a scheme will yield a feature vector of only four coefficients. Spikes from different neurons will tend to have different shapes and match the pattern to varying degrees. The problem then becomes the choice of an appropriate pattern.

It is known from signal detection theory that when trying to detect the presence of a signal corrupted by additive Gaussian noise, the optimal pattern is the signal itself (Turin, 1960). That pattern is impractical because the chosen pattern must work with a variety of spike shapes, not known a priori, but it is often sufficient to use a pattern that matches consistent characteristics of all spike shapes. One shape characteristic shared by all spikes is the rapid negative transition during repolarization, which can be detected with the pattern

$$p = \begin{bmatrix} 1 & \dots & 1 & 0 & -1 & \dots & -1 \end{bmatrix}, \qquad (1)$$

where the numbers of ones and negative ones are equal and depend on the number of samples spanned by a spike. The best match to this pattern is indicated by the maximum value of the cross-correlation of the spike waveform and the pattern, and this value is recorded as the feature for that waveform.

After the feature vectors have been computed, it is necessary to estimate the number of neurons, k, contributing to the record, and sort the feature vectors into the k clusters. Many researchers have sought a way to do this completely automatically, so that the clustering operation can run unsupervised (Cheeseman and Stutz, 1996), while others have opted for manual sorting. The method described here is a hybrid approach, in which the clustering is done automatically in order to reduce the effects of human bias, but constrained to group the feature vectors into a number of clusters estimated manually from scatter plots of feature vectors. The clustering algorithm is a modified version of k-means (Lloyd, 1982) that uses k-means++ (Arthur and Vassilvitskii, 2007) to choose initial cluster locations and scaled Mahalanobis (1936) distance as the distance measure. The scaling reduces the tendency of a large cluster to subsume a nearby small cluster. Overall, the performance of this algorithm is similar to that of computing a Gaussian mixture model (GMM) with expectation-maximization (i.e., Bayesian clustering as mentioned in Lewicki (1998)) when the Gaussian distributions do not overlap significantly. However, for some cluster configurations, the proposed algorithm outperforms GMM.

Cluster quality is evaluated three ways: 1) by visual inspection of feature vector scatter plots, 2) by computation of a cluster separation metric, L_{ratio} (Schmitzer-Torbert et al., 2005), and 3) by examination of spike interval histograms (Hill et al., 2011). Because the clustering algorithm is not deterministic (the initialization step has a random component), it can be advantageous to recluster when the clustering is poor. The interactivity and speed of the algorithm facilitate rapid reclustering and multiple trials with different values of k. The automation provides greater clustering consistency than can be achieved by human operators, for given values of k.

2. Methods

Data were collected with a Neuralynx Cheetah system connected to four tetrodes, each independently processed. The algorithm presented here was developed using recordings made in the inferior colliculus of awake rabbit, employing general methods described previously for single-unit recordings (Nelson and Carney, 2007) except that tetrodes were used. During recording sessions, animals were seated in a plastic chair inside a sound-attenuated booth. The head was fixed to allow delivery of sound stimuli to each ear via custom earmolds. All methods were approved by the University of Rochester Committee on Animal Research and complied with NIH guidelines.

The algorithm is also illustrated using a data set recorded from a different midbrain region, the nucleus of the brachium of the inferior colliculus of awake marmoset, provided by S. Slee at Johns Hopkins University. These recordings were made with Thomas Recording tetrodes in a fixed-head preparation to allow controlled delivery of acoustic stimuli. Because all of the data of interest were in fixed-head recordings, motion artifacts associated with freely moving animals were not a concern here. Although awake animals in a head-fixed preparation occasionally shrug or fidget, the artifacts associated with these movements can typically be eliminated as outlier waveforms.

Positive- or negative-going voltage transitions of sufficient amplitude on any one of the four tetrode wires triggered events that were "snapshots" of the waveforms recorded by those wires, each consisting of 32 samples at a rate of 32,051 samples/sec. The processing described here is equally valid for spikes of opposite polarity simply by inverting the detection pattern. The waveforms were collected in a file for later analysis.

The events from a single two-hour recording session can number up to several hundred thousand. One goal was to make the evaluation of an experiment take no more than a few minutes so that adjustments to the experimental procedure, such as the position of the tetrode, could be made in "real time." To

achieve the desired speed, rather than clustering all the events, a subset is selected and used to train the clustering algorithm by collecting information that characterizes the clusters. Occasionally, it was noticed that the characteristic spike shape of a neuron changed slightly and gradually during the experiment, but not so much that the spikes become confused with those of another neuron. Consequently, in order to capture the full range of shapes, the training set must include events distributed across the whole experiment. It is also necessary to use events that are contiguous so that interval histograms can be computed. To satisfy both requirements, the full set of events is broken into blocks of contiguous events, and the blocks are uniformly distributed throughout the experiment. If the training set has M events, it is appropriate to form approximately $\sqrt{M}$ blocks of approximately $\sqrt{M}$ events each. The exact numbers are not important; in practice M might be chosen to be 10,000–20,000.

From the features computed for the training set, the number of clusters, k, is estimated visually, and the statistical properties (mean and covariance matrix) of each cluster are computed. The cluster means and covariance matrices are used to classify the full set of events.

To summarize, the processing procedure is as follows: 1) record events, 2) train clustering algorithm: 2a) choose subset of events, 2b) compute feature vectors, 2c) choose k by visual inspection of feature vector plots, 2d) cluster, 2e) repeat 2c–2d as necessary to achieve an acceptable result, 3) compute feature vectors for all events, and 4) classify all events into k clusters. Each of these steps is described in more detail below.

2.1. Feature computation with repolarization slope (RPS)

Action potentials from different neurons tend to have different shapes, but all have the general characteristics of the spike shown schematically in Fig. 2.

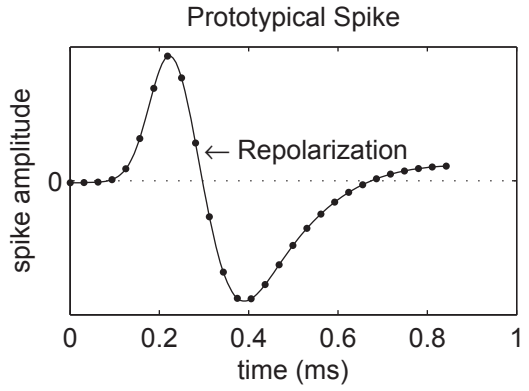


Figure 2: Prototypical spike waveform showing approximate sample positions. The slope of the waveform during the repolarization phase was used as a feature for clustering. This waveform is an average of multiple waveforms from a single cluster recorded in the rabbit inferior colliculus.

As previously stated, the proposed waveform feature is computed by taking the maximum value of the cross-correlation of the waveform with a fixed pattern, p, defined in Eq. 1. Samples of a spike waveform $s(t)$, triggered at $t = 0$, are given by

$$s[n] = \begin{cases} s(nT), & n = 0, 1, \ldots, N_s - 1 \\ 0, & \text{otherwise} \end{cases}, \tag{2}$$

where T is the sampling interval and N_s is the number of samples recorded per spike. For a pattern given by $p\,[n]$, the feature, r, is computed by

$$r = \max_n \left\{ \sum_{m=-\infty}^{\infty} s\,[n+m]\;\; p\,[m] \right\}. \tag{3}$$

Mathematically, cross-correlation is the same as convolution when the pattern has been time-reversed, so the feature can also be computed as

$$r = \max_n \left\{ \sum_{m=-\infty}^{\infty} s\,[m]\;\; q\,[n-m] \right\} = \max\{s * q\}, \tag{4}$$

where $q\,[n] = p\,[-n]$.

Representing the cross-correlation as a convolution is convenient because the operation can be interpreted as passing the spike waveform through a digital filter. Indeed, a filter whose impulse response is equal to the time-reverse of a pattern to be matched is called a *matched filter* (Turin, 1960) because the impulse response is matched to the signal being detected. It is interesting to note that the filter impulse response, q, is exactly the convolution of two other sequences,

$$q = \underbrace{\begin{bmatrix} -1 & 1 \end{bmatrix}}_{K_d} * \underbrace{\begin{bmatrix} 1 & 2 & \ldots & n & n & \ldots & 2 & 1 \end{bmatrix}}_{K_f}, \tag{5}$$

where K_d performs a derivative-like operation and K_f is a low-pass filter with the frequency response shown in Fig. 3 for $n = 4$. Therefore, convolution with q produces an approxi-

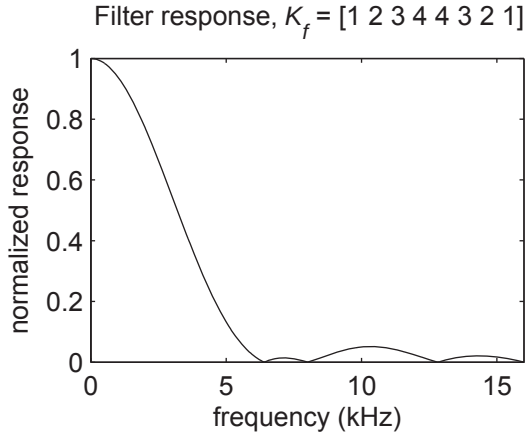


Figure 3: The action of taking the derivative of a spike waveform tends to accentuate high-frequency noise. That noise is attenuated by the low-pass filter K_f (see Eq. 5) which, for $n = 4$ and sample rate = 32.051 kHz, has the frequency response depicted here.

mation to the filtered derivative of each of the four waveforms

recorded by the tetrode, and each computed feature is approximately proportional to the magnitude of the slope of the spike in the region in which it is falling most rapidly, normally the repolarization region (Fig. 2). It is from this description that the algorithm receives its name, *repolarization slope* (RPS).

Due to noise, the exact phase at which triggering occurs varies, resulting in a random variation of the spike location along the time axis. Some feature-extraction algorithms require spike alignment to remove this temporal variation, but alignment is not necessary for this algorithm because each feature is computed from the maximum value of the whole cross-correlation function rather than from a specific sample.

The feature vector for an event is simply the concatenation of the four r values, one from each waveform recorded by the tetrode. These values are further concatenated into an $M\times4$ feature matrix, which is the set of M feature vectors for the M events.

2.2. Feature-vector visualization and selection of number of clusters

The next step in the spike sorting process is to visualize the feature vectors so that the number of clusters, k, can be determined. Unfortunately, each feature vector is a four-dimensional entity that is hard to visualize and impossible to plot. One approach to this problem is to plot the two dimensions with the greatest variance, as obtained from PCA (Glaser and Marks, 1968; Abeles and Goldstein Jr, 1977). As stated previously, use of PCA can be unsatisfactory. Instead, an array of scatter plots is displayed, similar to those shown in Fig. 4, each of which plots two different columns of the feature matrix against each other. Visual inspection of the scatter plots has usually been found to be sufficient to determine the number of clusters, k ($k = 3$ in the example shown). However, because these plots only show two dimensions at a time, it is possible for clusters to remain hidden. That has not proved to be a problem in practice, and it is easy to try different values of k in any case.

2.3. Cluster initialization

The standard k-means algorithm (Lloyd, 1982) chooses the initial cluster centers to be the coordinates of k points chosen at random from the data before proceeding with the iterative clustering. This strategy can produce cluster centers that are close together resulting in clusters that are not properly defined, so it is standard practice to run k-means several times until a good result has been obtained. The k-means++ algorithm (Arthur and Vassilvitskii, 2007) was devised to reduce or eliminate these multiple runs by making a better choice for the set of initial cluster centers. (Despite its name, k-means++ is not a clustering algorithm—it only performs the cluster initialization step.) The algorithm consists of the following steps: a) choose one point at random as the first cluster center, b) for each point to be clustered, x, compute the Euclidean distance, $D_e(x)$, to the nearest existing cluster center, c) add one point at random as a new cluster center using a weighted probability where point x is chosen with probability proportional to $D_e^2(x)$, and d) repeat b and c until k centers have been chosen. In practice, this technique usually selects cluster centers that are well separated.

2.4. Iterative cluster determination and Mahalanobis distance

The iterative clustering of any variant of k-means consists of just two steps: 1) assign each point to the nearest cluster, and 2) recompute the cluster descriptive statistics (e.g., mean). These steps are repeated until the assignments do not change from one iteration to the next.

In order to assign each point to the nearest cluster, it is necessary to define some notion of distance. Ordinary Euclidean distance does not work well when the clusters are elongated and close to each other, as they often are in this application. Mahalanobis (1936) distance is a statistical metric used to determine how well a point, x, fits in a whole distribution of points, C, taking into account the shape, size and orientation of the distribution, and is defined by

$$D(x) = \sqrt{(x-\mu_C)\Sigma_C^{-1}(x-\mu_C)^{\mathsf{T}}}\ , \tag{6}$$

where C is described by its mean, μ_C, and covariance matrix, Σ_C. The distance is unitless because it is a relative measure. An example is illustrated in Fig. 5.

To use Mahalanobis distance in the determination of the nearest cluster, Σ_C must be nonsingular. Immediately after initialization, each cluster contains only one point, and all covariance matrices are identically zero and singular. Intuitively, the inability to compute Mahalanobis distance from a cluster of one point makes sense, because it is impossible to infer anything about the size or shape of such a cluster. Consequently, whenever a cluster has only one point, as will always occur in the first iteration, some method other than Mahalanobis distance must be used to compute point-cluster distances. Euclidean distance was chosen as the alternate method, because it does not require a covariance matrix.

To illustrate the difference between the Euclidean and Mahalanobis distance measures, Fig. 6 shows equal distance contours from two clusters.

2.5. Scaling by cluster size, k-means with Scaled Mahalanobis Distance (KSMD)

With the strategy presented so far, the clustering algorithm often failed to find small clusters located close to large ones, included too many points from a small cluster in a nearby large cluster, produced cluster regions with holes (one cluster completely surrounding another cluster), or produced disjoint cluster regions. It was hypothesized that these shortcomings could be eliminated or reduced if the Mahalanobis distance measures were scaled by a factor related to the size of the cluster (Quian Quiroga et al., 2004; Harris et al., 2000; Shoham et al., 2003). The idea is that the scaled distance of a point from a large cluster should have a larger numerical value than a point with the same Mahalanobis distance from a small cluster, encouraging the point to belong to the smaller cluster.

Intuitively, the scale factor should be based on a linear measure of cluster size rather than volume, which was confirmed empirically. Consequently, the cluster size is expressed as the length of a side of the N-dimensional hypercube with the same volume as the cluster, i.e., the N^{th} root of the volume. Strictly

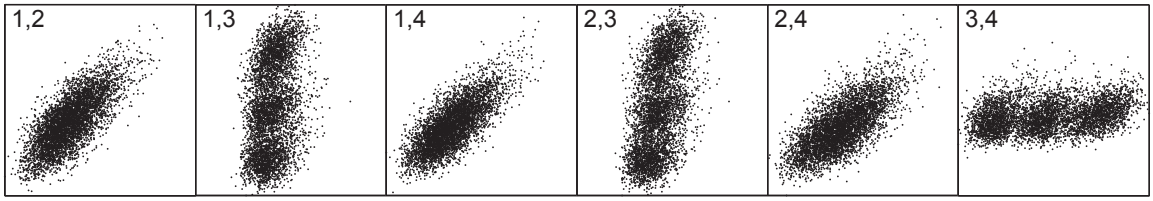


Figure 4: Scatter plots of features computed from spikes recorded in the rabbit inferior colliculus. The features computed from the four wires of a tetrode yield a single four-dimensional point for each spike. For plotting, it is necessary to project each 4-D point into two dimensions and this is done most easily by plotting the point coordinates two at a time, resulting in the six plots shown here. Labels indicate the pair of coordinates plotted, e.g., in panel "1,2", the horizontal axis represents the repolarization slope of the spike on wire 1 and the vertical axis that of the spike on wire 2. Visual inspection of these plots suggests that this recording contains spikes from three neurons.

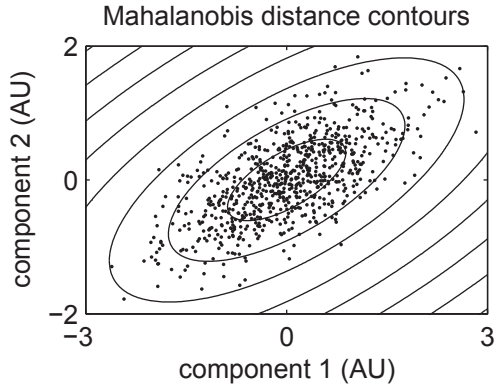


Figure 5: Illustration of contours of equal Mahalanobis distance from a distribution in two dimensions for $D = 1, 2, \ldots, 8$ using synthetic data in arbitrary units (AU) (see Eq. 6). Note how the contours follow the shape of the distribution rather than appearing as concentric circles surrounding the mean as would be obtained with contours of equal Euclidean distance from the distribution center.

speaking, a cluster is a distribution, not a polytope, and does not have a volume, but for this application the volume, V, can be defined as the product of the widths of the cluster along the N orthogonal axes of the cluster,

$$V = \prod_{i=1}^{N} \sqrt{\lambda_i}\,, \tag{7}$$

where λ_i is the i^{th} eigenvalue of the covariance matrix of the cluster members. These λ_i are exactly equal to the variances of the distribution in the axial directions if the cluster were lined up with the coordinate axes via PCA. As mentioned, the side length, ℓ, is obtained simply by taking the N^{th} root of the volume,

$$\ell = \sqrt[N]{V}. \tag{8}$$

Finally, the scale factor for cluster C_j was chosen to be

$$w_j = \ell_j^{\alpha}, \tag{9}$$

where the exponential parameter, α, has been introduced to control the scaling effect. The nominal value of $\alpha = 1$ generally succeeds in minimizing the shortcomings present without the scaling. Setting α to a value much above 1 can introduce unwanted artifacts (Fig. 7). Values of α between 0 and 1 were not tried, but might prove useful. Setting $\alpha = 0$ is equivalent to having no scaling.

2.6. Cluster evaluation

An important part of clustering is evaluation of the resulting clusters (Hill et al., 2011; Joshua et al., 2007). Whether because the wrong number of clusters was chosen or because the random component in the initialization step caused the clustering algorithm to perform poorly, sometimes it is necessary to recluster. There are several ways to perform this evaluation: visual inspection of the feature vector scatter plot, computation of a cluster separation metric, inspection of waveform histograms, and examination of interval histograms. Each approach will be described below and illustrated in the Results section.

2.6.1. Visual evaluation of feature vectors

The visual evaluation involves making a paired scatter plot in which the points from each cluster are plotted in a different color. A simple visual inspection by the user can indicate whether the clustering was successful. Poor clustering is often obvious, as in Fig. 8c. Sometimes, even when the number of clusters is easy to estimate, the clustering algorithm is unable to cluster the points as desired, a failure that is easily determined by visual observation of the scatter plots.

2.6.2. Cluster separation metric

A quantitative cluster separation metric called L_{ratio} (Schmitzer-Torbert et al., 2005) is defined by

$$L(C) = \sum_{x_i \notin C} 1 - F_{\chi^2_N}\left(D^2_C(x_i)\right) \tag{10}$$

$$L_{\text{ratio}}(C) = \frac{L(C)}{n_C}, \tag{11}$$

where $F_{\chi^2_N}$ is the cumulative distribution function of the χ^2 distribution with N degrees of freedom (N is the dimensionality of the feature space), $D_C(x_i)$ is the Mahalanobis distance of point x_i from cluster C, and n_C is the number of points in C. L represents a measure of the isolation of C so that a cluster with

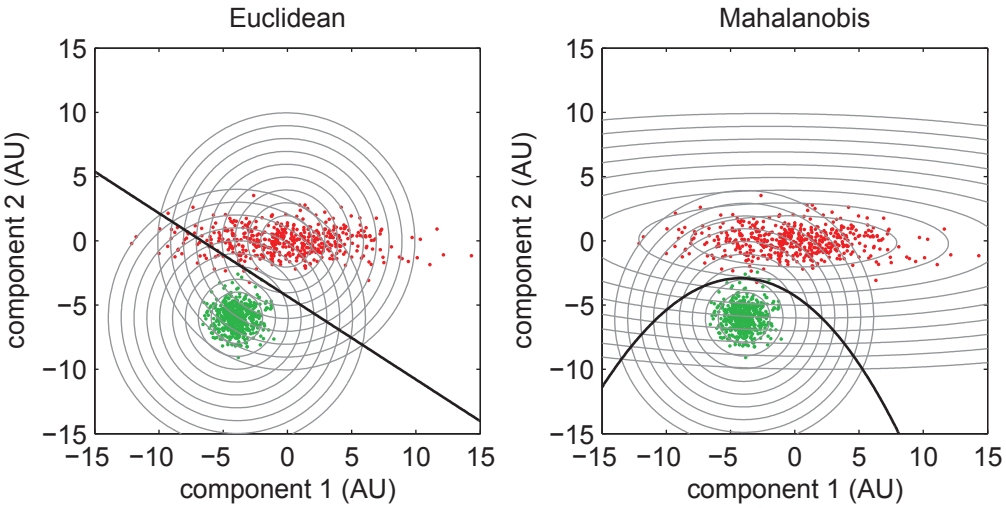


Figure 6: The gray concentric curves depict equal distance contours using the two distance measures, Euclidean (left) and Mahalanobis (right), similar to those in Fig. 5. The solid black lines depict the loci of points that are equidistant from the two clusters within each plot and are used as boundaries between the two cluster regions. Notice that the boundary based on the Mahalanobis distances results in fewer classification errors. The data are synthetic with arbitrary units (AU).

a significant gap surrounding it will have a small value. L_{ratio} normalizes $L(C)$ by the number of points in C so that if an acceptability criterion is established, a dense cluster will be able to tolerate more contamination than a sparse one. The values are mostly useful for differentiating between multiple clusterings of the same data and not intended to be used to rate different data sets. Because each $L_{\text{ratio}} \geq 0$ and small values indicate good cluster separation, the sum,

$$L_{\Sigma} = \sum_{j=1}^{k} L_{\text{ratio}}(C_j), \tag{12}$$

will only be small if each L_{ratio} is small and thus serves as an overall figure of merit for the clustering.

2.6.3. *Interval histograms*

After a neuron has fired it cannot fire again during its refractory period (Hodgkin and Huxley, 1952), which typically has a duration of about 1 ms for neurons that have relatively high discharge rates (auditory nerve: Gray, 1967; inferior colliculus: Yagodnitsyn and Shik, 1974). This characteristic can be used to test if the spikes in a given cluster could actually have been produced by a single neuron. The distribution of first-order spike interval times (the length of time between a spike and the next spike from the same neuron) is examined to see what fraction of spike intervals are less than the refractory period. If the fraction is zero, then it is possible that all the spikes in that cluster were produced by a single neuron. In practice, some cluster contamination is acceptable (spikes incorrectly assigned to that cluster), perhaps 0.5%.

Conversely, it is expected that many intervals from a spike in one cluster to the subsequent spike *in another cluster* will be less than the refractory period. Therefore, examination of cross-cluster spike intervals can indicate whether spikes from a single neuron have been assigned to more than one cluster, as happens in over-clustering.

2.6.4. *Waveform histograms*

After clustering, a natural way to evaluate whether all the spikes in a cluster have similar shapes is to plot them on a single graph. A single graph works well if there are few enough waveforms that most of them remain visible despite the clutter, but it does not accurately convey the waveform distribution when many waveforms are obscured. Instead, the waveform histogram has been defined that is simply a sequence of histograms of the waveform amplitudes at each time sample for a given cluster. To display the waveform histogram, an image is constructed by displaying each bin of each histogram as a single pixel—the number of pixels in the vertical direction is the number of bins and the number of pixels in the horizontal direction is the number of samples per waveform. The gray level of each pixel is proportional to the corresponding histogram value, leading to a fuzzy-looking spike waveform. The degree of fuzziness indicates the width of the distribution: a well-defined appearance implies a narrow distribution and a fuzzier appearance implies a broad distribution. See results for examples (Fig. 9). See also Hill et al. (2011).

Previously, it was stated that one of the advantages of RPS was that it does not require the spikes to be aligned. That is true, but the waveform histogram does require spike alignment for maximum utility. Spike alignment, using an approach that aligns the peak values of the waveforms after passing them through a low-pass filter, was included in the waveform histograms shown here.

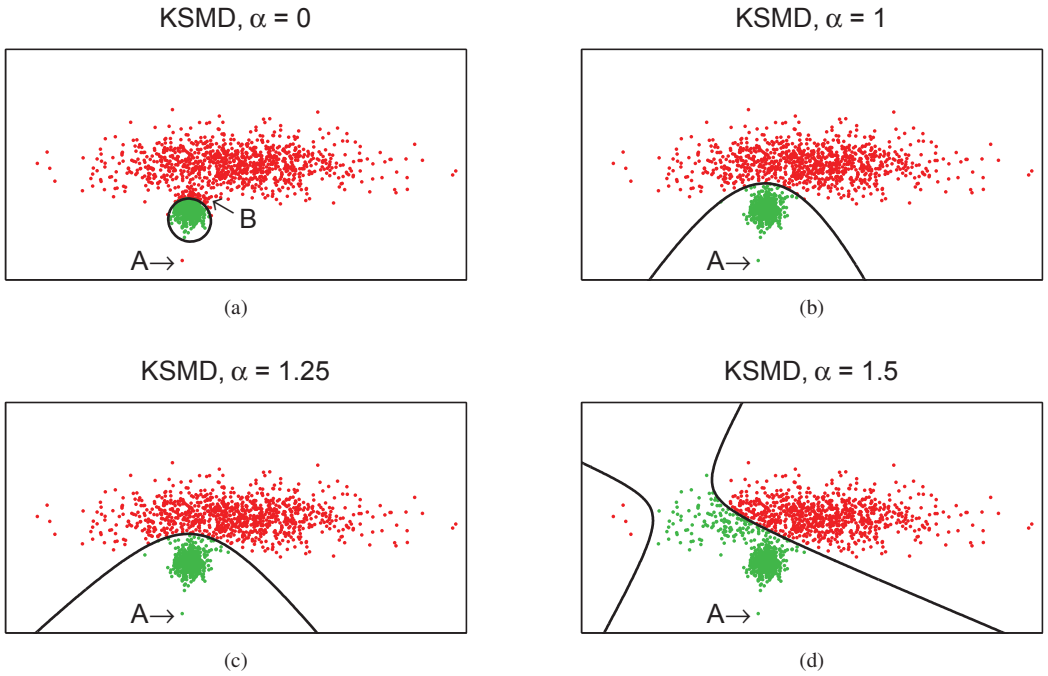


Figure 7: The effect of the KSMD scaling parameter α on clustering performance. A data set was constructed from two known, synthetic, arbitrarily scaled point distributions and clustered by KSMD with several values of α. The black lines represents the boundaries between the two clusters (colored red and green). (a) With $\alpha = 0$ (equivalent to no scaling), two shortcomings are evident: 1) the red cluster region completely surrounds the green cluster (a "hole", note point A which is on the opposite side of the green cluster from the other red cluster points), and 2) points in the region B are clustered incorrectly with the red points. (b) $\alpha = 1$, the boundary results in acceptable classification, especially in the region between the two clusters. Also, the outlier point A is included in the green cluster. (c) $\alpha = 1.25$, performance similar to (b), however, the boundary is moved towards the red distribution. In this case, more classification errors are apparent, but in other cases performance might be improved. (d) Large values of α may result in disjoint clusters and many classification errors, as in this example when $\alpha = 1.5$.

2.6.5. Cluster temporal stability

A final cluster evaluation is to examine the temporal stability of the clusters. In some applications, the signal from a neuron can change during a measurement session, perhaps due to a slight shift in position of the tetrode, resulting in apparently two (or more) clusters from a single neuron. To ensure that a single neuron has not been manifested as multiple clusters, it is sufficient to examine histograms of spike times for each cluster. If the times of the spikes in each cluster are distributed throughout the session it is likely that the clusters are stable.

2.7. Classification of all points

If the feature vectors used for training the final clustering are representative of the whole set of feature vectors, then it is reasonable to assume that the cluster mean and covariance matrices obtained from training are accurate estimates of those that would be obtained from the whole data set. The mean and covariance matrix of each cluster of the training set can be used to cluster the full set in an efficient way. All that is required is to compute the scaled Mahalanobis distance of each feature vector from each training cluster and assign each feature vector to the nearest cluster.

2.8. Implementation and clustering procedure

The algorithms described above have been implemented in MATLAB (The MathWorks, Inc., Natick, Massachusetts), in a convenient application with a graphical user interface.[1] It is important to note that RPS and KSMD are merely two of the tools that might be employed and a full-featured application supplies other tools to try if RPS or KSMD were not effective on a particular data set.

Generally, the clustering procedure is as follows: 1) load the waveforms for the training set of events, 2) compute the feature vectors using the default method (e.g., RPS) and display the paired scatter plots, 3) estimate the number of clusters, compute them with the default clustering algorithm (e.g., KSMD with $\alpha = 1$), and update the scatter plot display with different colors for each cluster, 4) check the results by examining the paired scatter plots, waveform histograms, interval histograms, and L_Σ, 5) if the results are not acceptable, try a different number of clusters, feature computation, and/or clustering algorithm until acceptable results are obtained, and 6) using the selected algorithms and cluster statistics gathered from the training set, cluster the whole data set.

[1] http://www.urmc.rochester.edu/labs/Carney-Lab/

3. Results

RPS and KSMD have been used to sort spikes from hundreds of recording sessions. RPS was not always effective and other techniques were sometimes used, but in practice, RPS/KSMD became the default methods because they worked well so often. In order to illustrate the efficacy of RPS/KSMD, the results of four sample experiments will be shown.

An exhaustive comparison of RPS/KSMD to other clustering techniques was not performed. However, in a study utilizing 49 data sets, each was clustered using the combinations of RPS/KSMD, PCA/KSMD, RPS/GMM and PCA/GMM (the PCA cases retained four principal components) (PCA: Glaser and Marks, 1968; Abeles and Goldstein Jr, 1977; GMM: Lewicki, 1998). The resulting values of L_Σ were compiled and ranged from 1.86×10^{-15} to 19.1. In 30 data sets, RPS/KSMD had the smallest L_Σ and in 14 of the remaining 19 data sets, the RPS/KSMD L_Σ was no more than 1.8 times as large as the smallest L_Σ. There were 5 data sets in which RPS/KSMD was clearly outperformed by one or more of the other techniques.

3.1. Example #1—RPS works well, PCA fails.

The first example showcases the advantages of RPS/KSMD. Application of PCA to the waveforms of all four wires, keeping the four most significant components, failed to give an indication of the number of clusters (Fig. 8a), while RPS clearly indicated two clusters (Fig. 8b). Clustering with k-means (using the standard Euclidean distance measure) produced a poor result as shown in Fig. 8c and indicated by the large value of L_Σ. The combination of RPS and KSMD was able to cluster these data properly (Fig. 8d). Note the extremely low value of L_Σ, indicating well separated clusters.

The existence of more than one cluster in this example is also indicated by the unclustered waveform histogram (Fig. 9a). Note the bimodal appearance of the waveform histograms near 0.3 ms, especially on wire 4. After sorting, the waveform histograms in Fig. 9b were obtained. Note that the waveform histograms appear sharper and unimodal, indicating a narrower distribution of waveform amplitudes at each sample point. The histograms of Fig. 9b indicate well separated spike waveforms consistent with the corresponding feature vector scatter plots of Fig 8d.

The first-order interval histograms (Fig. 10) allow tests of cluster quality based on inter-spike intervals within and across clusters. From the first-order interval histogram for cluster 1 (Fig. 10a), it can be seen that there were few intervals less than the presumed refractory period of 1 ms, indicating that cluster 1 is likely a good, single-unit cluster. Note that the refractory period of 1 ms is applicable to the inferior colliculus (Yagodnitsyn and Shik, 1974) where discharge rates can reach a few hundred spikes per second. The first-order interval histogram for cluster 2 (Fig. 10d) shows that 1.3% of the intervals were less than 1 ms, indicating that it was likely a multi-unit recording.

Overclustering can be detected by examining cross-cluster histograms. If two clusters are associated with different neurons then intervals less than the refractory period are expected. The two off-diagonal histograms (Figs. 10b and 10c) show a significant number of intervals less than the refractory period, indicating that the data have not been over-clustered. This result is consistent with other illustrations of this data set in Figs. 8 and 9.

3.2. Example #2—Cluster merging required.

Some data sets are particularly challenging for clustering with KSMD, even when different values of α are tried. In particular, if the clusters are mismatched in density as well as size, the algorithm will tend to group a small sparse cluster with a nearby larger and denser one. Also, closely spaced irregularly shaped clusters are difficult to separate.

Example #2 illustrates a case of mismatched cluster densities and a solution to the problem. From the initial feature vector visualization (Fig. 11a), it appeared that there were two clusters, but one contained many more points than the other. This presented a challenge for KSMD, as it was unable to cluster the data in a way consistent with the visual evaluation.

The solution was to increase the value of k for KSMD until the small sparse cluster visible in Fig. 11a was isolated. Then, the remaining clusters were merged into a single large cluster, resulting in two clusters. In this example, it was necessary to set $k = 4$ in order for KSMD (with $\alpha = 1$) to isolate the sparse cluster, shown in Fig. 11b. Combining the red, green, and blue clusters into a single cluster resulted in the clustering shown in Fig. 11c. Note the improvement in cluster separation as indicated by the large reduction in L_Σ from 0.712 to 0.00675.

3.3. Example #3—Large number of clusters.

From the recordings in the rabbit inferior colliculus it was rare to find a data set with more than two or three clusters. This example illustrates a recording with five clusters. RPS produced the feature vectors illustrated in Fig. 12a. The clustering was done by KSMD with $\alpha = 1$. The value of L_Σ for these data is 0.324 which is somewhat large due to the close spacing of these clusters. Remember that the L_Σ metric is only suitable for comparing multiple clusterings of the same data set and little meaning can be attached to the absolute value.

It is useful to look at the waveform histograms for this data set to confirm that the algorithms are functioning correctly. From the waveform histograms shown in Fig. 12b, it is clear that clusters 1 and 5 consist of well defined spikes. The cluster 3 pattern is similar to that of cluster 5, but the shape and amplitude of the spike on wire 3 is different enough that these are probably from different neurons. The waveform distribution on wire 1 of cluster 1 looks somewhat blurry at 0.35 ms. This blurriness is due to a slow variation of the wire 1 spike shape (and the feature value) over the course of the recording. The spike shape variation is also responsible for the somewhat irregular shape of cluster 1 (red) in Fig. 12a.

3.4. Example #4—Nucleus of the brachium of the inferior colliculus of awake marmoset.

This example describes a data set recorded in a different species by another laboratory. RPS/KSMD was used by S. Slee of Johns Hopkins University to cluster recordings made from a

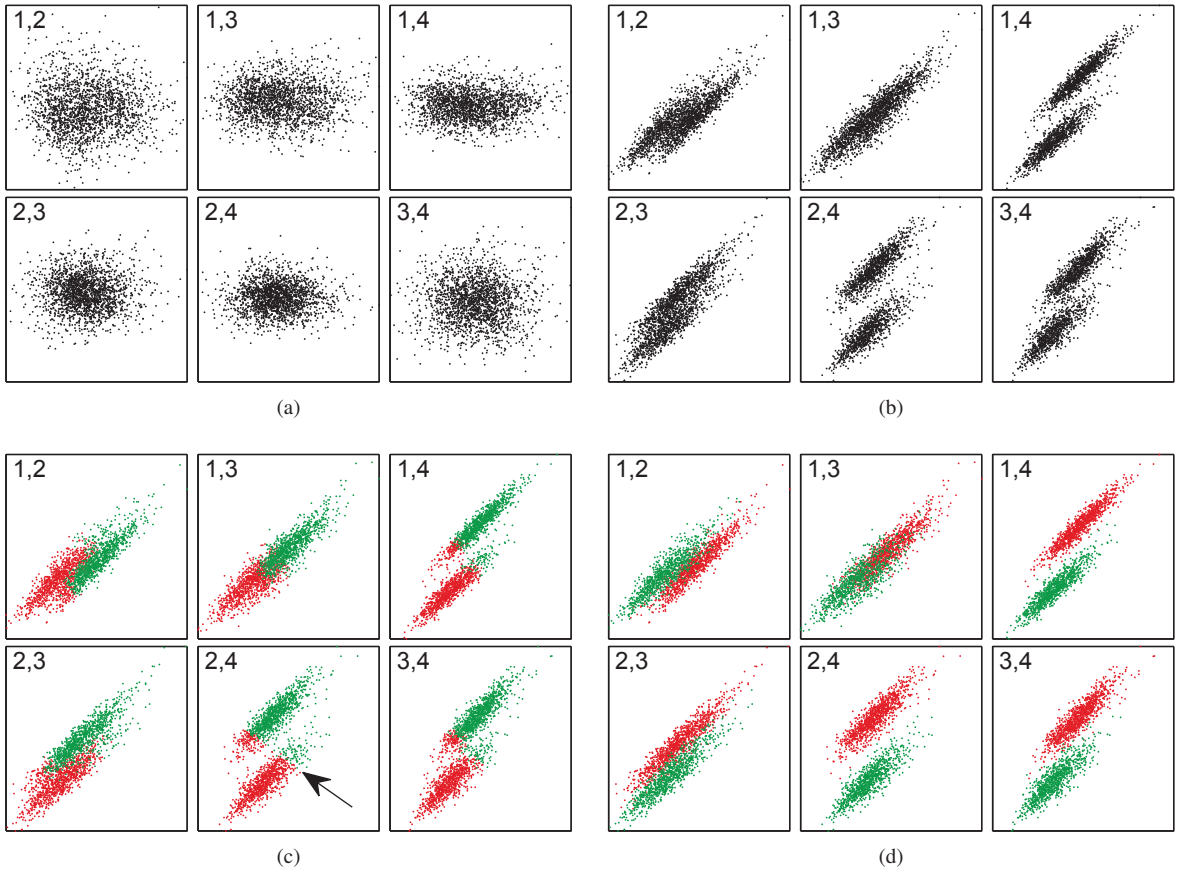


Figure 8: Example #1 in which PCA fails and RPS succeeds. (a) Unclustered feature vectors after application of PCA to the waveforms: multiple clusters are not apparent. (b) Unclustered feature vectors from RPS: two clusters are clearly visible. (c) Feature vectors from RPS were improperly clustered by k-means, readily visible in the plot of wire 4 vs. wire 2. The arrow points to the erroneous k-means cluster boundary. The value of the cluster separation metric, L_Σ, is 0.089. (d) Feature vectors from RPS clustered by KSMD with $\alpha = 1$ resulted in far fewer classification errors. The value of L_Σ was reduced to 1.15×10^{-5} (small values of L_Σ indicate better separation). These data were recorded in the rabbit inferior colliculus.

different auditory midbrain region, the nucleus of the brachium of the inferior colliculus, in the awake marmoset.

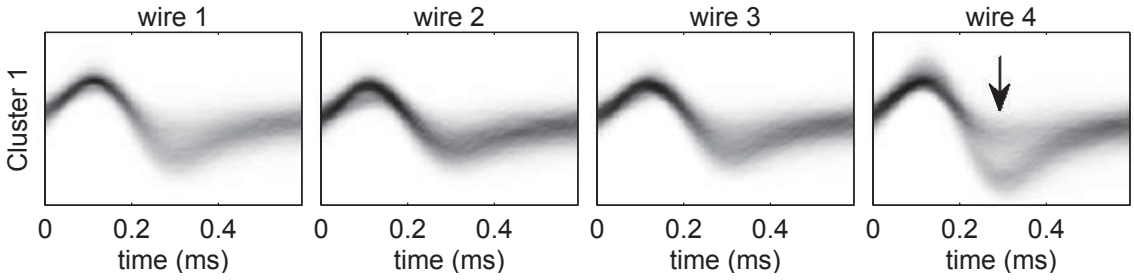
The results are shown in Fig. 13. One difference from the previous examples is that the polarity of the spikes is inverted due to the particular hardware configuration used. RPS must accommodate this polarity reversal so that the slopes in the repolarization regions (now positive) are computed, but this is done simply by inverting the signs of the pattern coefficients (Eq. 1).

4. Discussion

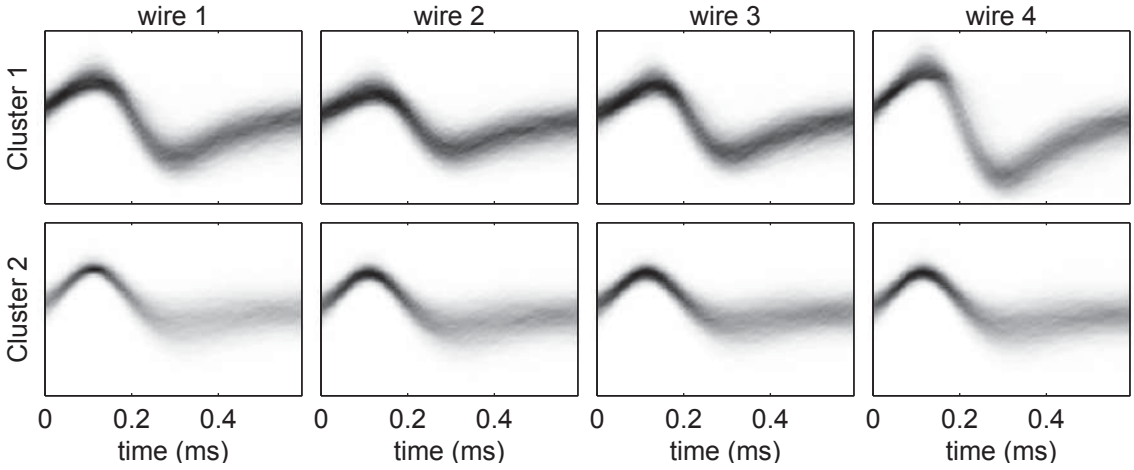
The algorithms RPS and KSMD have been shown to be effective at sorting spikes. Combined with the approach of determining cluster characteristics from a subset of spikes, the entire clustering procedure is fast, taking no more than a few minutes to cluster spikes obtained from two hours of data collection (4 tetrodes with an average of 200,000 spikes per tetrode). There are, however, some issues that would benefit from further investigation. Note that no attempt was made to detect overlapping spikes from different neurons (Lewicki, 1998; Zhang et al., 2004; Franke et al., 2010).

The scaling of the Mahalanobis distances by the size of the cluster was motivated by the observation that sometimes a small cluster located close to a large one would not be recognized as a separate cluster. The technique does seem to help the clustering algorithm work more acceptably, but it is still possible for a small cluster to be hidden inside a larger one. If that happens, it is a failure of the feature, RPS, to discriminate spikes from two neurons, presumably because the slopes of the spikes in their respective repolarization regions are nearly equal. If that occurs, a different feature computation should be used.

Visual evaluation of k can be problematic. The technique of plotting two dimensions at a time in pairs (reducing the inherently 4-D data to six 2-D views) is not guaranteed to show all the clusters, but seemed to work acceptably. A more thorough, but space-consuming approach is to plot additional views of the data, obtained by performing intermediate orthonormal transformations (akin to viewing 3-D data from a 45° angle), though there will still be no guarantee that clusters will not be



(a) Waveform histograms, unclustered data.



(b) Waveform histograms, clustered data.

Figure 9: Example #1 waveform histograms for each wire (see section 2.6.4). (a) All waveforms (unclustered). The bimodal distribution near 0.3 ms on wire 4 (see arrow) suggests the presence of more than one spike shape. (b) Waveform histograms of the same set of waveforms after being sorted into two clusters using RPS and KSMD with $\alpha = 1$. Consistent with the well separated cluster diagrams in Fig. 8d, the spike waveforms appear properly clustered, especially evident on wire 4. Vertical scales are in identical arbitrary units. These data were recorded in the rabbit inferior colliculus.

missed.

Additional strategies are possible to further automate the clustering procedure. For example, a data set can be clustered with multiple values of k and then the quality of each clustering can be evaluated using the L_Σ metric. However, this technique can fail when presented with some of the more difficult cases, such as Example #2 (see Fig. 11).

Another suggestion for future development of the algorithm would be to explore additional features automatically. The algorithm could then determine which set of features results in the best separability. This technique could result in improved performance with less operator interaction. Such a strategy might be particularly useful in brain regions, such as cortex and hippocampus, where spike shapes vary substantially across different classes of neurons (Fee et al., 1996b; Buzsáki, 2004).

RPS and KSMD are two tools used for spike sorting, but they are far from the only ones and not always the best ones. A practical approach to spike sorting employs multiple techniques with an easy way to switch between them. In fact, RPS has been implemented along with several other techniques including PCA, spectral techniques and wavelets. Likewise, both KSMD and GMM algorithms have been implemented to allow easy selection of the best clustering algorithm for a particular data set. All the algorithms are provided in an easy-to-use program with a graphical user interface. When the results of RPS are unacceptable, the user is able to try another technique simply by pressing a button. On a modern workstation, operating on 20,000 events, results are obtained in just a few seconds.

5. Acknowledgments

This work was supported by NIH-NIDCD R01-001641. We are grateful to Blair Stewart and Hannah Rasmussen for testing of the algorithms described in this document.

References

Abeles M, Goldstein Jr M. Multispike train analysis. Proceedings of the IEEE 1977;65(5):762–73.

Arthur D, Vassilvitskii S. k-means++: The advantages of careful seeding. In: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms. Society for Industrial and Applied Mathematics; 2007. p. 1027–35.

Buzsáki G. Large-scale recording of neuronal ensembles. Nature neuroscience 2004;7(5):446–51.

Cheeseman P, Stutz J. Bayesian classification (AutoClass): Theory and results. In: Fayyad U, Piatetsky-Shapiro G, Smyth P, Uthurusamy R, editors. Advances in Knowledge Discovery and Data Mining. Menlo Park, California: AAAI Press; 1996. p. 153–80.

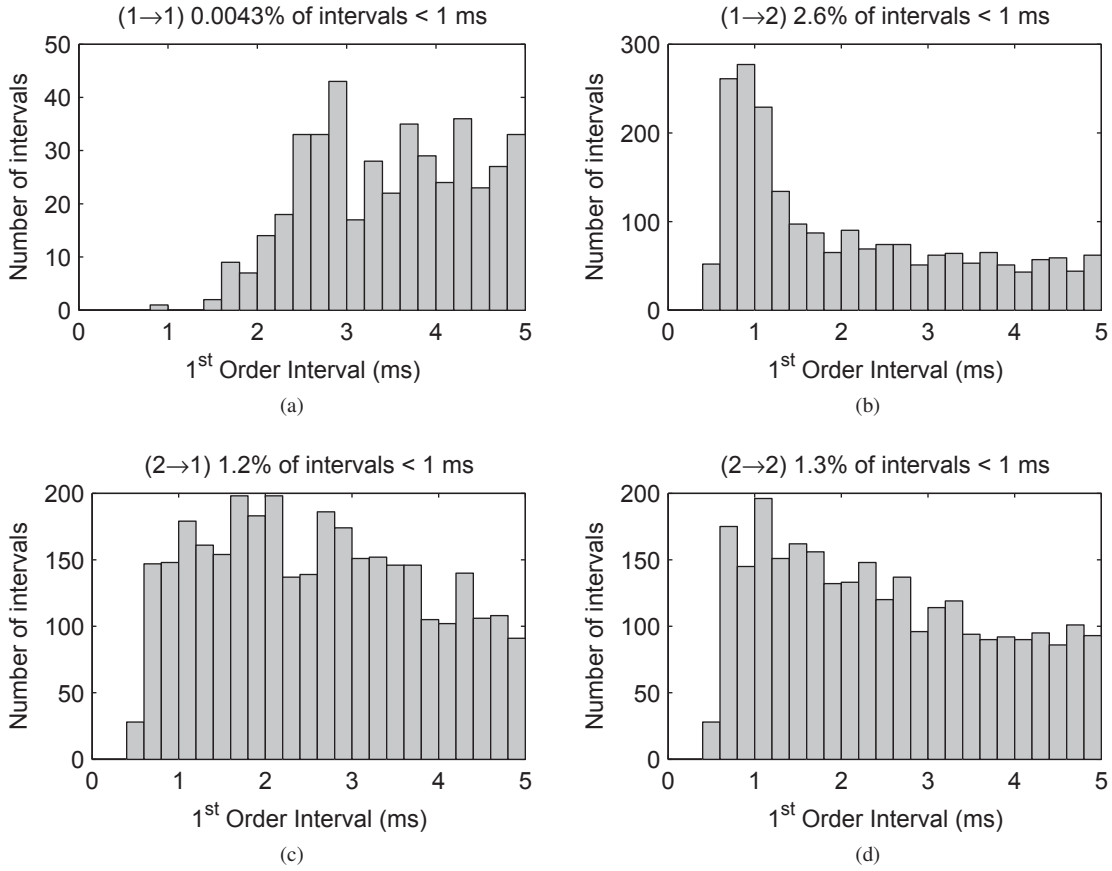


Figure 10: Example #1, First order interval histograms within and between clusters from a data set of 50,000 spikes. (a) The small number of intervals in cluster 1 shorter than the refractory period of 1 ms suggests that those spikes are generated by a single neuron. (b and c) A high percentage of intervals between clusters 1 and 2 shorter than 1 ms is consistent with those clusters being associated with different neurons. (d) A large percentage of intervals in cluster 2 shorter than 1 ms suggests that it is a multi-unit cluster. This data set was recorded in the rabbit inferior colliculus.

Delescluse M, Pouzat C. Efficient spike-sorting of multi-state neurons using inter-spike intervals information. Journal of neuroscience methods 2006;150(1):16–29.

Fee M, Mitra P, Kleinfeld D. Automatic sorting of multiple unit neuronal signals in the presence of anisotropic and non-gaussian variability. Journal of Neuroscience Methods 1996a;69(2):175–88.

Fee M, Mitra P, Kleinfeld D. Variability of extracellular spike waveforms of cortical neurons. Journal of neurophysiology 1996b;76(6):3823–33.

Franke F, Natora M, Boucsein C, Munk M, Obermayer K. An online spike detection and spike classification algorithm capable of instantaneous resolution of overlapping spikes. Journal of computational neuroscience 2010;29(1):127–48.

Glaser E, Marks W. On-line separation of interleaved neuronal pulse sequences. In: Data Acquisition and Processing in Biology and Medicine. Pergamon Press; volume 5; 1968. p. 137–56.

Gray C, Maldonado P, Wilson M, McNaughton B. Tetrodes markedly improve the reliability and yield of multiple single-unit isolation from multi-unit recordings in cat striate cortex. Journal of Neuroscience Methods 1995;63(1-2):43–54.

Gray P. Conditional probability analyses of the spike activity of single neurons. Biophysical Journal 1967;7(6):759–77.

Harris K, Henze D, Csicsvari J, Hirase H, Buzsáki G. Accuracy of tetrode spike separation as determined by simultaneous intracellular and extracellular measurements. Journal of Neurophysiology 2000;84(1):401–14.

Hill D, Mehta S, Kleinfeld D. Quality metrics to accompany spike sorting of extracellular signals. Journal of Neuroscience 2011;31(24):8699–705.

Hodgkin A, Huxley A. A quantitative description of membrane current and its application to conduction and excitation in nerve. The Journal of physiology 1952;117(4):500–44.

Joshua M, Elias S, Levine O, Bergman H. Quantifying the isolation quality of extracellularly recorded action potentials. Journal of neuroscience methods 2007;163(2):267–82.

Lewicki M. Bayesian modeling and classification of neural signals. Neural computation 1994;6(5):1005–30.

Lewicki M. A review of methods for spike sorting: the detection and classification of neural action potentials. Network: Computation in Neural Systems 1998;9(4):53–78.

Lloyd S. Least squares quantization in pcm. Information Theory, IEEE Transactions on 1982;28(2):129–37.

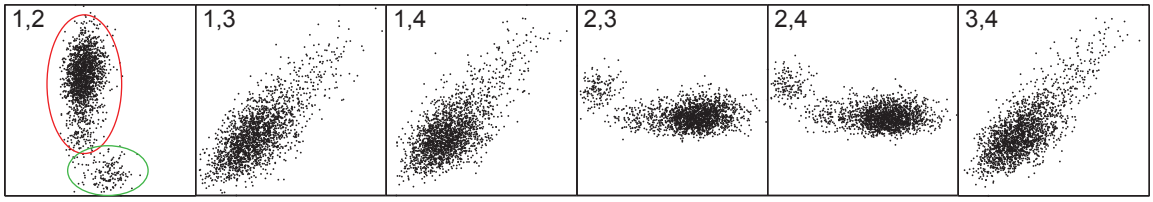
Mahalanobis P. On the generalized distance in statistics. In: Proceedings of the National Institute of Science, Calcutta. volume 12; 1936. p. 49–55.

Nelson P, Carney L. Neural rate and timing cues for detection and discrimination of amplitude-modulated tones in the awake rabbit inferior colliculus. Journal of neurophysiology 2007;97(1):522–39.

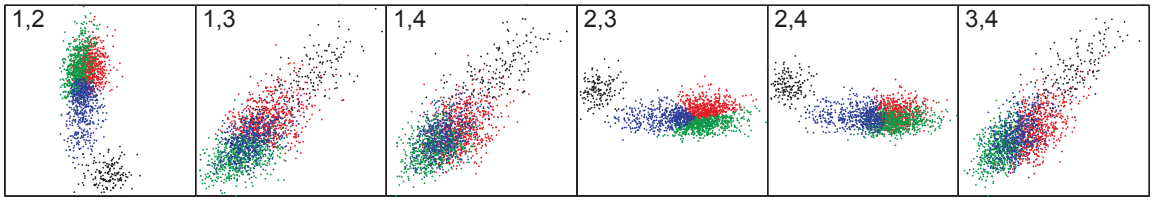
Quian Quiroga R, Nadasdy Z, Ben-Shaul Y. Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering. Neural Computation 2004;16(8):1661–87.

Schmidt E. Computer separation of multi-unit neuroelectric data. J Neurosci Meth 1984;12:95–111.

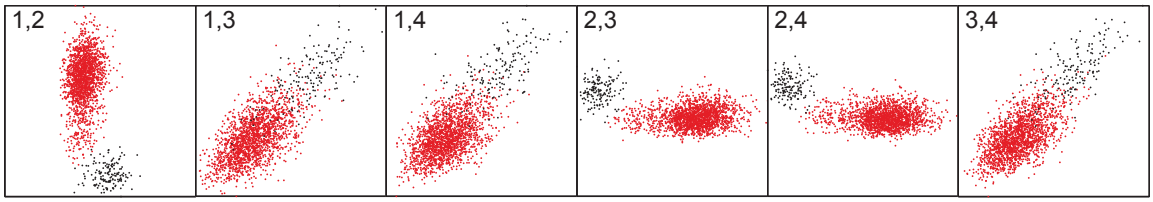
Schmitzer-Torbert N, Jackson J, Henze D, Harris K, Redish A. Quantitative measures of cluster quality for use in extracellular recordings. Neuroscience 2005;131(1):1–11.



(a)



(b)



(c)

Figure 11: Example #2 scatter plots of feature vectors from spikes recorded in the rabbit inferior colliculus. (a) From the unclustered feature vectors produced by RPS, it can be seen that there are two clusters (roughly delineated by the red and green ellipses), one being smaller and sparser than the other. KSMD with $k = 2$ did not properly separate the points in the sparse cluster (not shown). (b) It was necessary to set $k = 4$ to get KSMD to separate the sparse points (black) from the other points, but this also caused the dense cluster to be separated into three clusters (red, green and blue) for a total of four clusters, as specified. The resulting L_Σ before merging was 0.712. (c) The final clustering was achieved by merging the red, green and blue points into one cluster. The final $L_\Sigma = 0.00675$.

Shoham S, Fellows M, Normann R. Robust, automatic spike sorting using mixtures of multivariate t-distributions. Journal of Neuroscience Methods 2003;127(2):111–22.

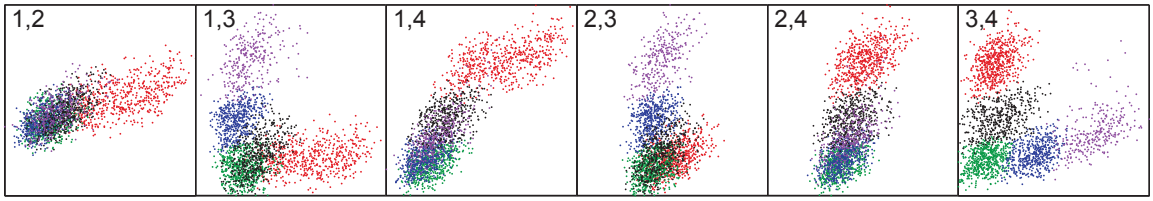
Takahashi S, Anzai Y, Sakurai Y. A new approach to spike sorting for multi-neuronal activities recorded with a tetrode–how ica can be practical. Neuroscience research 2003;46(3):265–72.

Turin G. An introduction to matched filters. Information Theory, IRE Transactions on 1960;6(3):311–29.

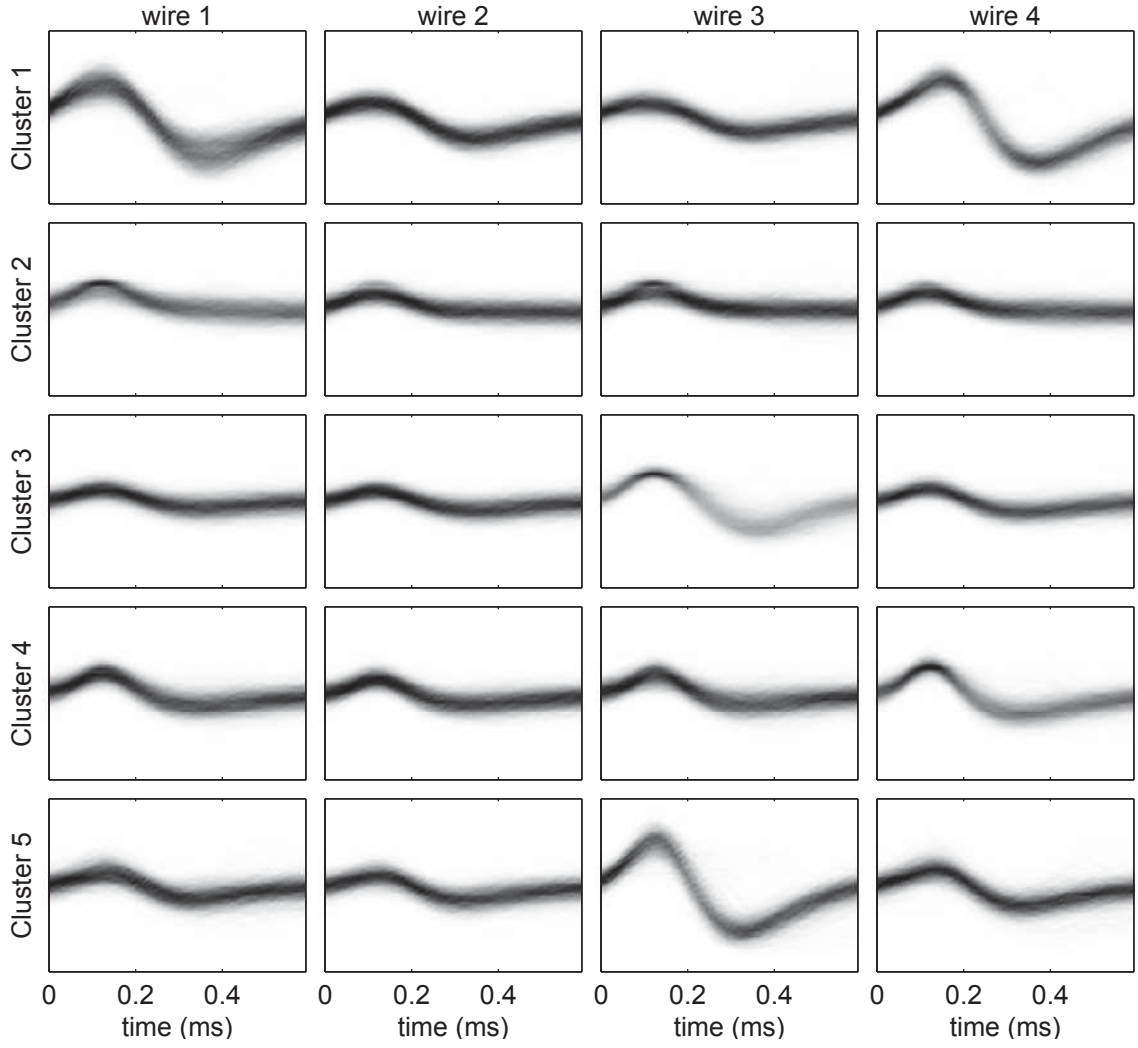
Wheeler B, Heetderks W. A comparison of techniques for classification of multiple neural signals. Biomedical Engineering, IEEE Transactions on 1982;(12):752–9.

Yagodnitsyn A, Shik M. Anisotropic connections in the cat midbrain tegmentum. Neurophysiology 1974;6(6):475–81.

Zhang P, Wu J, Zhou Y, Liang P, Yuan J. Spike sorting based on automatic template reconstruction with a partial solution to the overlapping problem. Journal of neuroscience methods 2004;135(1-2):55–65.

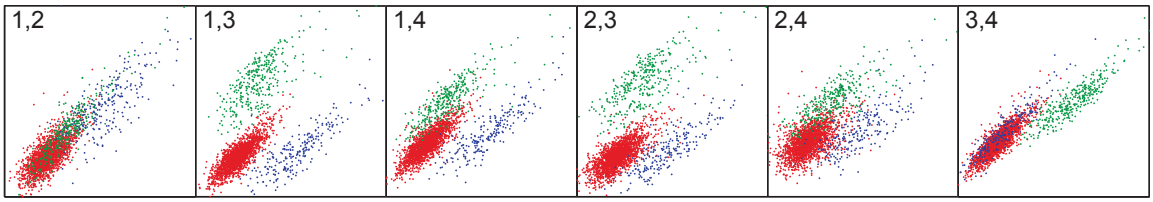


(a)

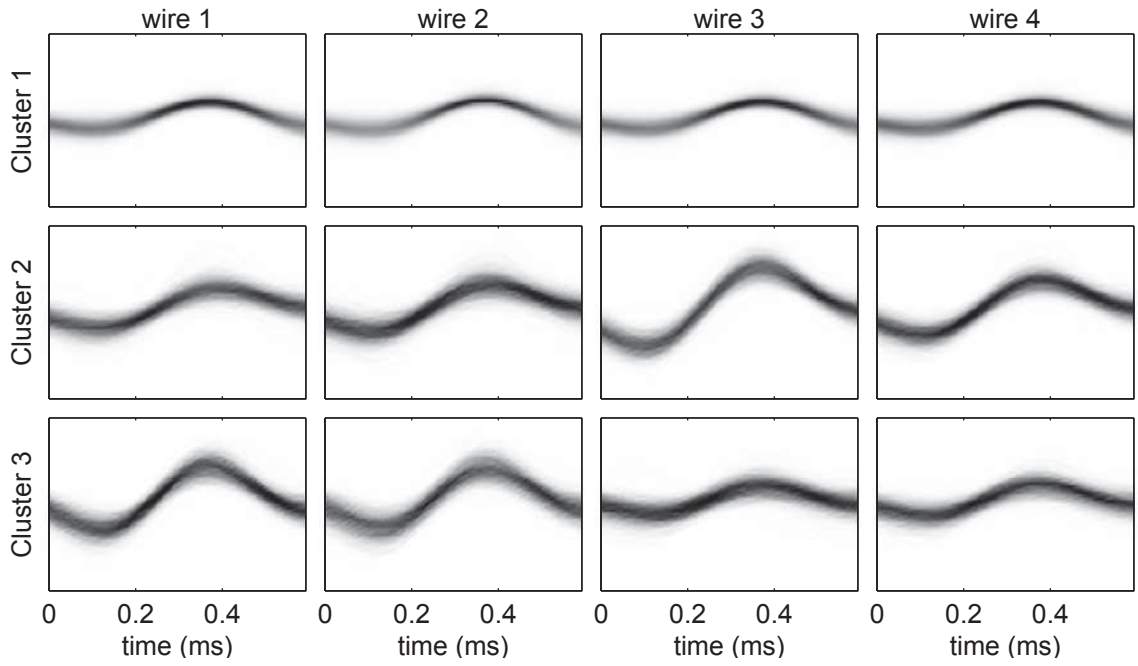


(b)

Figure 12: Example #3, data set recorded in the inferior colliculus of awake rabbit with five clusters. (a) The features illustrated here were computed by RPS and clustered by KSMD with $\alpha = 1$. All five clusters are most readily visible in the "3,4" view. Clusters 1–5 are colored red, green, blue, black and purple, respectively. (b) The waveform histograms confirm the presence of five clusters.



(a)



(b)

Figure 13: Example #4, data set recorded in the nucleus of the brachium of the inferior colliculus of awake marmoset with three clusters. (a) The features illustrated here were computed by RPS and clustered by KSMD with $\alpha = 2$. Clusters 1–3 are colored red, green and blue, respectively. (b) The waveform histograms confirm the presence of three clusters. Compared with the previous examples, the waveforms are inverted due to the hardware configuration used. This polarity reversal is accommodated by inverting the polarity of the pattern used by RPS (Eq. 1).