ELSEVIER

Journal of Neuroscience Methods 164 (2007) 1–18

JOURNAL OF NEUROSCIENCE METHODS

www.elsevier.com/locate/jneumeth

Automated spike sorting using density grid contour clustering and subtractive waveform decomposition

Carlos Vargas-Irwin *, John P. Donoghue

Department of Neuroscience, Brown University, 185 Meeting Street, Providence, RI 02912, USA

Received 9 August 2006; received in revised form 15 March 2007; accepted 28 March 2007

Abstract

In multiple cell recordings identifying the number of neurons and assigning each action potential to a particular source, commonly referred to as 'spike sorting', is a highly non-trivial problem. Density grid contour clustering provides a computationally efficient way of locating high-density regions of arbitrary shape in low-dimensional space. When applied to waveforms projected onto their first two principal components, the algorithm allows the extraction of templates that provide high-dimensional reference points that can be used to perform accurate spike sorting. Template matching using subtractive waveform decomposition can locate these templates in waveform samples despite the influence of noise, spurious threshold crossing and waveform overlap. Tests with a large synthetic dataset incorporating realistic challenges faced during spike sorting (including overlapping and phase-shifted spikes) reveal that this strategy can consistently yield results with less than 6% false positives and false negatives (and less than 2% for high signal-to-noise ratios) at processing speeds exceeding those previously reported for similar algorithms by more than an order of magnitude.

Keywords: Spike sorting; Template matching; Overlapping spikes; Multi-electrode arrays; Electrophysiology

1. Introduction

Recent technological developments have made it possible to simultaneously record the spiking activity of increasingly large populations of cortical neurons using extracellular microelectrodes. Not only does this allow the exploration of more sophisticated models of neural ensemble information processing, it also holds great promise for the development of neuroprosthetics that rely on control signals derived from the spiking patterns of multiple neurons. Recordings obtained with extracellular electrode arrays create a range of signal processing challenges because they generate many channels of recordings, with each channel potentially containing signals from several neurons. In most cases, identification of the unique spiking pattern of each single neuron is highly desirable. Isolating neural signals and assigning each recorded waveform to the neuron of origin is a time-consuming and highly non-trivial problem commonly referred to as 'spike sorting'. It can be difficult to compare data across studies without an objective and definable classification measure. The problem is made more complex for multi-channel recording systems that are fixed in place, because they cannot be moved to approximate the electrode tip to an ideal recording position that maximizes waveform separation. Fixed multi-electrode arrays often employ larger tipped electrodes, further increasing the likelihood of recording multiple neurons with similar spike shapes on a single channel. Action potentials recorded from a single neuron tend to have a stereotypical shape determined by the cell's structure and biophysical properties as well as its position relative to the electrode. A stereotypical spike shape is generally the major or even sole characteristic used to verify that a set of waveforms is attributable to a single neuron. However, other features such as the cell's firing history can introduce considerable variation in waveform shape or amplitude (Snider and Bonds, 1998; Fee et al., 1996b; Quirk and Wilson, 1999). Waveform variation is also increased by interfering signals such as the background activity of other neurons (Fee et al., 1996b), as well as other sources of noise like electrode movement (Snider and Bonds, 1998) and external electrical or mechanical interference. The similarity between spike waveforms from different cells may also be increased by the bandpass filtering methods generally applied to this kind of data.

Another challenge to spike sorting emerges from waveform misalignment. Data flow and size constraints of high channel

* Corresponding author. Tel.: +1 401 338 3639; fax: +1 401 751 2102.
E-mail address: Carlos_Vargas_Irwin@Brown.edu (C. Vargas-Irwin).

0165-0270/$ – see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.jneumeth.2007.03.025

count arrays make it necessary to limit storage to fragments of continuous records that may provide irregularly captured snapshots of desired waveforms. Misalignments may result from variable triggering of data collection due to noise or other spike events that lead to premature or delayed appearance of the waveform, which can be truncated within the sample window. Even if waveforms are aligned using local or global minima/maxima prior to sorting they may be difficult to classify if they are not fully captured within a limited event window. Recording multiple neurons on a given electrode can also produce complex sums of spike waveforms (Zouridakis and Tam, 2000). Decomposing these overlapping spikes into their single-neuron components generates the greatest computational burden on spike sorting algorithms.

A perfect solution to the spike-sorting problem is often impossible in practice. Many investigators rely on manual sorting by an expert, which can often be time consuming, arbitrary, non-verifiable and inaccurate. Wood et al. (2004) have reported an average of 23% false positives and 30% false negatives for manual spike sorting of synthetic datasets performed by experts using commercial software. Harris et al. (2000) have obtained similar estimates using simultaneous intracellular recording to determine the true firing pattern of neurons. Practical automated spike sorting applications must strike a balance between maximizing accuracy and maintaining low computational complexity. Processing speed becomes and especially important factor for the emerging variety of setups that simultaneously collect information from hundreds of electrodes.

Many algorithms have been applied to the spike-sorting problem (for a general review, see Lewicki, 1998). In this paper we focus on algorithms suited to process data obtained using multiple single electrodes (where the tissue monitored by individual electrodes does not overlap). Although the use of stereotrodes (where multiple electrodes monitor the same region) can potentially extract more information about a given volume of tissue, it is possible to perform accurate spike sorting using single-electrode data. Furthermore, it is more practical to cover a large region using multiple single electrodes, facilitating the study of neural activity at a population level. This type of electrode configuration is currently the most widely used recording method for neo-cortical electrophysiology.

The goal for the present study was to develop a spike sorting application that could be used to sort very large datasets from multi-electrode arrays (consisting of hundreds of thousands of mixed spike waveforms) in a reliable, consistent and computationally efficient manner. Our approach uses isolevel curves plotted around local density maxima in principal component space in order to extract spike templates. Instead of standard centroid-based cluster identification, we have developed a novel strategy to estimate data density for regions of variable shapes and sizes in reduced-dimensional space that requires only a single pass through the data. The second step in our algorithm performs template matching using a new approach to spike shape decomposition in order to deal with overlapping spikes without having to test all possible combinations of spike templates. Our spike-sorting algorithm was tested using a synthetic dataset that replicates the main challenges present in real data in a carefully quantified manner, as well as multi-electrode array recordings obtained from macaque primary motor cortex.

2. Methods

2.1. Hardware used to implement the algorithm

All calculations were performed using MatLab (Mathworks, Natick, MA). The sorting results for different parameter values of our algorithm were calculated using a Dell Optiplex computer running Windows XP with a 2 GHz Pentium processor and 256 MB of RAM. The comparison between our algorithm and other spike sorting methods was carried out on a different computer: a Dell Dimension running Windows XP with a 2.8 GH Pentium Processor and 4 GB of RAM.

2.2. Density grid contour clustering (DGCC) algorithm

The goal of the first phase of the algorithm is to determine the number of neurons present in the recording and identify spike templates for each one. These templates are used to classify waveforms in the second phase. The first step of the algorithm uses principal component analysis to reduce the dimensionality of the data, projecting each waveform onto two-dimensional space. After this initial data compression step, the algorithm proceeds to locate putative cluster centers by outlining arbitrarily shaped high-density regions in 2D principal component space. Template waveforms are then calculated by taking the mean of all the points whose principal component projections fall within these central regions.

2.2.1. Density grid

Instead of calculating density measures at individual data points, DGCC divides principal component space into a grid where the density can be calculated simply as the number of data points within each partition. In order to perform this calculation it is necessary to select an appropriate size for the partitions. Overlapping spikes and noise waveforms may produce outliers when projected to principal component space. If all the data is included, such outliers can stretch the density grid to a scale where individual grid partitions become too large to represent useful information and most of the grid is wasted representing areas of very low density. In order to avoid this pitfall, outlier elimination is performed before calculating the density grid. For each principal component, 0.1% of the data points (those most distant from the mean) were removed. The final bounds therefore guaranteed that the density grid would include more than 99.8% of the data. The data was then divided into 100 equal partitions spanning the range of remaining values along each principal component, resulting in a 100×100 grid. The final density matrix was smoothed with a Gaussian kernel of size equal to three times the width of the partitions and standard deviation of 0.65. The time taken to calculate the density matrix is linearly proportional to the number of samples. Using 100×100 grid (which we have empirically determined results in adequate resolution), we can represent the entire set of data from one channel with 10,000 points, regardless the duration of

the recording session. Unlike other clustering methods, DGCC does not require continuous updating of density measures. All subsequent calculations are carried out without modifying the density grid.

2.2.2. Central region identification and template extraction

Local maxima in the density grid can be easily located using an exhaustive search where each grid square is compared to its neighbors. After this preliminary step, the positions of local density maxima are specified by the coordinates of the center of their grid partition. A user-defined minimum density threshold determines which locations are examined further. Although most of the outliers have already been removed, this step prevents the extraction of large numbers of small density peaks from sparse regions of PCA space. This threshold is specified as a percentage of the maximum density value on the grid (in this implementation of the algorithm, it was set to 5%).

Isodensity curves are plotted around each of the remaining local maxima. The density surface is represented by a mesh connecting each point $[x,y]$ in the density matrix with four other points: $[x+1,y]$, $[x-1,y]$, $[x,y+1]$ and $[x,y-1]$ (except for points at the edges of the matrix). This forms a set of triangular planes ('cells') defined by each point and two of its connected neighbors. Isodensity curves are generated by finding the intersection between the plane representing the desired level and each of the grid cells. The lines drawn across each cell are then connected to form the final isodensity line. In our implementation of the algorithm, we used the MatLab 'contour' function to perform this calculation. The levels at which the curves are drawn is set according to the value of the local maximum in question, spanning from 50 to 95% of the maximum value at 5% intervals. The isodensity contours generated in this way become a polygonal representation of each high-density area in grid coordinates. After discarding all polygons smaller than a grid square, the central regions of the clusters were identified by locating polygons that contained none of the points forming any of the other polygons. The central regions can then be adjusted according to the scaling factor for each dimension to transform them to principal component space coordinates. These high-density regions do not have any shape or size restrictions and are therefore able to accurately conform to any cluster shape.

By taking the mean of the waveforms whose principal component projections lie within each central region we can obtain a representative template wave for each cluster. Noise and multiunit activity can trigger data collection at varying phases of the spike waveform. This can result in having a single unit projected to different regions in principal component space and be represented by more than one central region. Therefore, at this stage, the algorithm tends to over-estimate the number of neurons, and often detects several high-density regions associated with a given spike template.

2.2.3. Determination of the number of units

Two steps are used to select the final set of templates among the candidates obtained from high-density regions in PC space. The first step involves evaluating the deviation from the mean spike shape for the waveforms within each central region. The standard deviation for each voltage measurement is calculated. The mean of these standard deviations is multiplied by four to obtain an estimate of the range of variation not attributable to the mean spike shape. If the amplitude of the putative template (the mean of the waveforms in the central region) is less than this estimate of variation, the template is discarded. This strategy selects central regions with consistently stereotyped waveform shapes, preventing noise waveforms and overlapping spikes from being used as templates even if they form dense clusters in principal component space.

The second phase of the template selection process involves comparing the possible spike templates to each other. The number of local maxima in the density matrix generally exceeds the true number of neurons being recorded. The shapes of the templates are used to determine whether or not they represent different neurons. Each pair of templates is aligned, and the maximum difference between them is calculated. Templates are assumed to belong to the same neuron if this difference is less than 95% of the deviations in voltage observed in the absence of spikes. The first five sample points for the waveforms of each channel are used to determine this estimate of noise magnitude (these measurements take place before the threshold crossing that triggers data collection and are therefore guaranteed to contain no spikes). When a group of templates are assumed to come from the same neuron, the one associated with the central region containing the greatest number of waveforms was kept while the rest are discarded.

2.3. Template matching and overlap resolution

The degree to which a sample waveform matches a template is evaluated by subtracting the template from the waveform. The L[infinity] norm (maximum absolute value) of the resulting difference vector is taken as the estimate of fit. Setting a threshold at a given value for this function is equivalent to outlining an envelope encompassing a fixed distance around the shape of the spike template. The estimate of fit for each template is evaluated after aligning the largest deviation from zero of the template (largest peak of trough) to the largest deviation of the same kind in the sample. In this implementation of the algorithm two additional alignments flanking the global minimum or maximum are also tested. The first level of the template matching algorithm evaluates single template fits only at these three alignments. In order to identify waveforms that do not contain any spikes, an additional test is performed: if the amplitude of the difference vectors is greater than the amplitude of the original waveform for all alignments and all spike templates, the waveform is classified as noise.

Checking for template overlaps is the most time-consuming of the algorithm, so it is advantageous from the point of computational efficiency to skip this step whenever possible. Moreover, since single-spike events tend to be more common than overlapping spikes, giving priority to parsimonious single template matches also tends to improve classification accuracy. Therefore, the subroutine used to evaluate the fit for possible overlapping spikes is only activated if none of the individual templates yield estimates of fit below a preset threshold. This "overlap detec-

tion" threshold determines how close of a match is required in order to skip checking possible overlapping templates in search of better match. The value chosen for this threshold is the determining factor for the total run time. Ideally this threshold should be set to a value encompassing most of the variations in spike shape attributed to noise. We evaluated the performance of the algorithm using several settings for the overlap rejection threshold, and determined that sorting accuracy was stable over a wide range of values (see Section 3).

Fig. 1 is a flowchart description of the template matching algorithm. When a sample waveform is matched with a given template three possible difference vectors are generated (one for each alignment). If all estimates of fit are above the overlap detection threshold, overlaps of two spikes are examined. In the second level of the decision tree the estimate of fit for a combination of two templates is obtained by running the template matching subroutine on each of the first level difference vectors with all the remaining templates. If the estimate of fit for any of the difference vectors in at this level is below the overlap detection threshold the algorithm stops and returns the best estimate of fit. Otherwise, overlaps of three spikes are examined using a similar procedure with the level two difference vectors. This step produces the final estimates of fit, since the current implementation of the algorithm does not test overlaps of more than three spikes.

Once the final estimates of fit are calculated, the waveform is assigned to the template (or combination of templates) that produces the best estimate of fit only if this value is below a second preset threshold (the 'template assignment' threshold). If none of the residues satisfies this criterion the original waveform is classified as noise. Having different values for our template assignment and overlap detection thresholds improved the range of parameters over which the algorithm was effective (see Section 3).

Our waveform decomposition method drastically reduces the number of comparisons necessary to classify a waveform (and hence computational demands) in the presence of multiple, possibly overlapping templates. Instead of attempting to

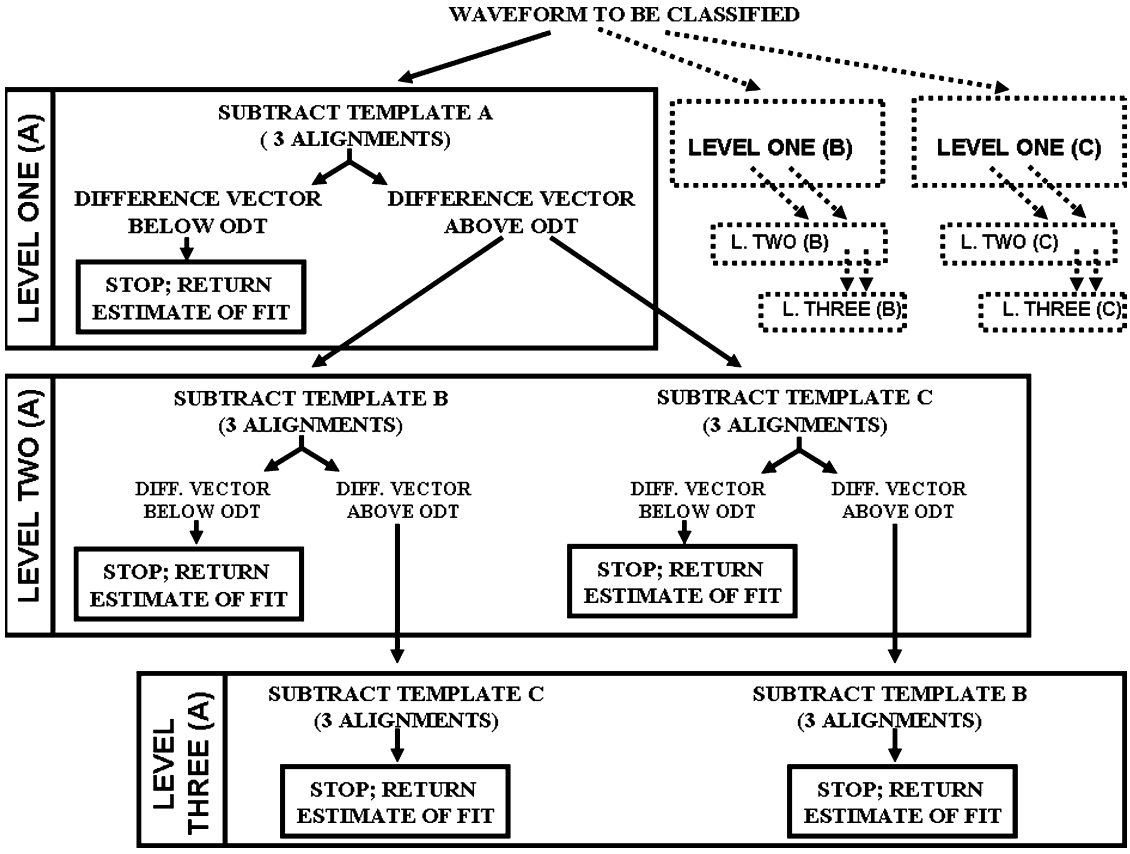


Fig. 1. Decision tree for template matching with three spike templates. Templates are sequentially subtracted from the waveform to be classified in order to generate estimates of fit. Each level of the decision tree corresponds to the number of templates being subtracted. Since only a subset of alignments is considered, the order in which the templates are subtracted makes a difference in the outcome. The diagram only shows one-third of the decision tree in detail—the section that originates when template A is subtracted first. The other parallel branches of the decision tree are summarized by dashed outlines. In the first level of the algorithm, each spike template is subtracted from the waveform to be classified at three alignments (centered on the most salient peak or trough). If there are three spike templates, a total of nine difference vectors will be generated. The maximum absolute value of each difference vector is used as an estimate of fit. If this value is below a preset overlap detection threshold (ODT) the algorithm stops and returns the estimates of fit obtained in level one. Otherwise, the algorithm proceeds to the next level, where combinations of two templates are examined by subtracting the remaining two templates from each of the level one difference vectors. Estimates of fit for the resulting 54 level two difference vectors are evaluated once more before the decision is made to move on to level three, where the last remaining template is subtracted. The waveform is assigned to the template (or combination of templates) that provides the best estimate of fit only if this value is below the template assignment threshold (otherwise it is classified as noise). The current implementation of the algorithm does not check for overlaps of more than three templates.

fit a waveform to all possible combinations of templates at all possible relative alignments, our strategy allows the putative contribution of each template to be individually removed from the waveform, ultimately leaving only a single template to fit. The most salient feature (trough or peak) of at least one spike in an overlapping combination will be close to the most salient feature (global minimum or maximum) of the summed waveform. Therefore, at least one the first set of residues generated after subtracting individual templates will have a spike subtracted out in the correct alignment. The rest of the contributing spikes are removed in the same way as the remaining residues are processed. The order in which the templates are subtracted makes a difference because it can affect the way templates are aligned before subtraction. However, even if all possible orders of subtraction are explored, the number of comparisons required remains small. For n templates only $3n + 3(n-1) + 3(n-2)\cdots+3(n-n)$ comparisons per waveform are required (taking into account three comparisons centered on the most salient feature).

By keeping track of the best-fitting template alignments for overlapping spikes, it is possible to subtract all templates save one in order to reconstruct the spike waveform originating from each putative neuron. In this way overlapping signals can be decoupled into separate waveforms that can be assigned to previously identified units. This procedure facilitates the subsequent quantification of waveform variability for each putative neuron. Template shifts can also be used to adjust the timestamps for the decoupled waveforms, enhancing the temporal resolution of recorded spike times.

2.4. Performance evaluation

2.4.1. Motivation for synthetic data

The impossibility of knowing the ground truth for spike sorting when only extracellular data is available prevents a straightforward assessment of algorithmic accuracy. Hence, synthetic data with a known spike composition was generated in order to evaluate sorting performance on four different types of events:

- Single spikes.
- Differently shaped spikes recorded simultaneously (overlapping spikes).
- Spikes shifted within the data collection time window due to variable triggering of data collection.
- Noise waveforms (without spikes).

In order to generate a realistic synthetic dataset, we incorporated spike and noise elements obtained from real cortical recordings, obtained as described below.

2.4.2. Cortical recordings

Spike waveforms were recorded from a Bionic microelectrode array (Cyberkinetics Neurotechnolgy Systems, Inc. Foxboro MA, 'CKI') chronically implanted in the arm area of the primary motor cortex of a Rhesus macaque performing a visually guided reaching task. The array contains a square grid of 100 silicon-based electrodes (96 of which are available for recording) spaced 400 μm apart (see Suner et al., 2005 for details on array performance and surgical implantation procedures). Data was recorded using a Cerebrus (CKI) multi-channel data acquisition system. Spike waveforms were sampled at 30 KHz (16 bit resolution) and processed using a digital Butterworth filter (250 Hz to 7.5 KHz). Each waveform was stored as a vector of 48 voltage measurements, spanning 1.6 ms. Data collection was triggered when the measured voltage exceeded a preset threshold individually set for each channel. When the trigger value was reached, 0.4 ms preceding the trigger as well as the following 1.2 ms was saved to disk. For channels where spikes were extracted, the trigger threshold was set to 4.5 times the root mean squared value of 2 s of continuously acquired voltage measurements. A set of 'noise' waveforms was recorded from a channel where no spike events were obvious to a trained observer. For these channels, the threshold was set to zero (such that continuous, non-overlapping 1.6 ms segments were recorded). The amplitude distribution in these pure noise recordings was approximately Gaussian in nature with a mean of zero and a standard deviation of 8.77 μV.

2.4.3. Synthetic data generation

Three spike shapes were used as templates to simulate three individual neurons. Each one was obtained by taking the mean of the waveforms attributed to a neuron identified by an expert sorter from a cortical recording performed as previously described. All of the templates were taken from the same channel. If we define the signal-to-noise ratio (SNR) as the amplitude of the template waveforms (430, 266 and 139 μV) over two times the standard deviation of the noise waveforms (25.1167 μV), the values for the synthetic units are 17.1, 10.6 and 5.5, respectively. A randomly selected sample from the 'pure noise' recordings described above was added to each artificially generated waveform. Sample waveforms are shown in Fig. 2.

The timestamps for the spikes originating from each synthetic neuron were determined by generating a Poisson process with a rate n times greater than the desired firing rate (80 Hz). The timestamps were then decimated by selecting every nth one from the original list. This resulted in a gamma distribution that simulates the refractory period naturally present in neurons (Baker and Lemon, 2000). The synthetic dataset simulated a 5 min recording session (roughly twenty thousand spikes per neuron).

A subset of the noise waveforms with troughs exceeding −40 μV were used to simulate high amplitude noise signals typically collected when trigger thresholds are set to low values during data acquistion (Fig. 2D). The mean of the maximum deviation from zero for these waveforms was 44.5 ± 5.9 μV. Twenty thousand of these high amplitude pure noise waveforms were assigned random timestamps within the data stream. Additionally, for a predetermined proportion of the single-spike events (15%), the spike template was paired with a high amplitude noise waveform and randomly shifted between 1 and 40 sample points (0.033–1.33 ms) within the data collection window to simulate premature data collection triggered by noise. When the maximum shift was applied, less than 0.3 ms of the original spike was included in the final waveform.

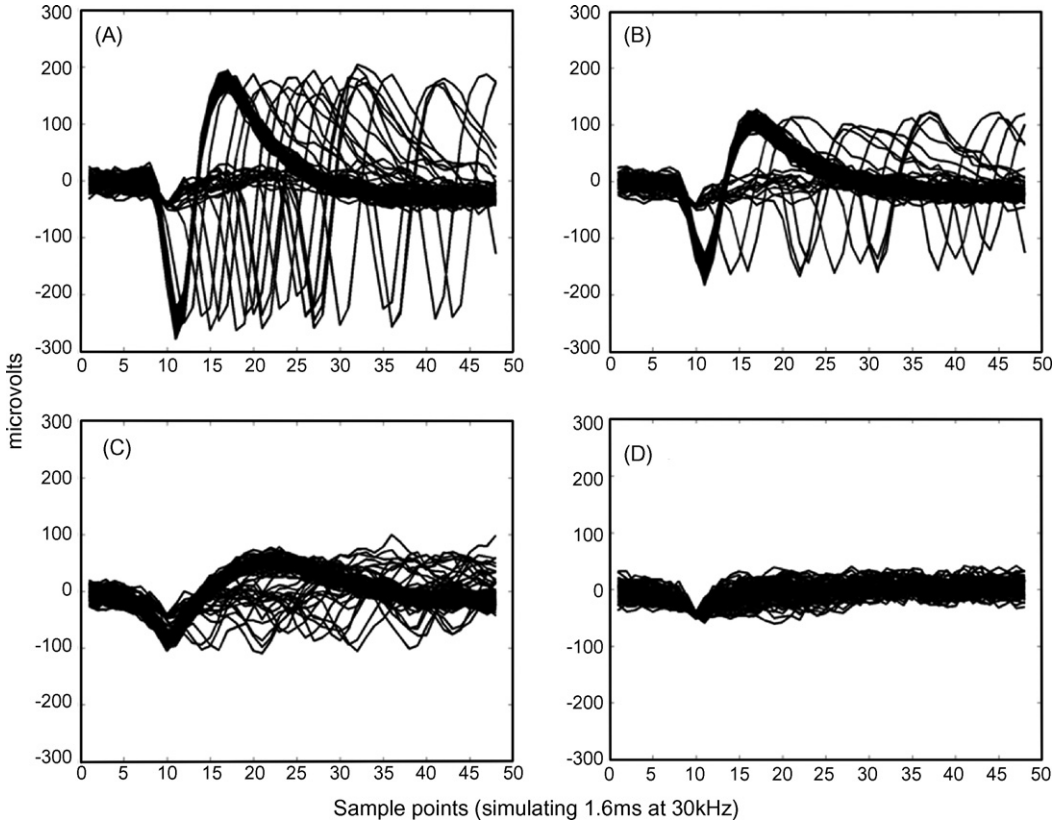


Fig. 2. Synthetic waveform samples. Panels A–C display samples of the waveforms generated for each simulated neuron. The amplitudes of the templates used to generate the waveforms were 430, 266 and 139 μV, respectively. Panel D shows a noise sample obtained from microelectrode array recordings in primate primary motor cortex. The mean of the maximum deviation from zero for these waveforms was 44.47 ± 5.9 μV. For 15% of the artificial waveforms the spike template was randomly shifted and combined with one of these noise samples in order to simulate inappropriate data collection triggering. Timestamps for each waveform were randomly generated using a decimated Poisson process. When two units had overlapping timestamps, the spike templates were additively combined to simulate overlapping spikes (not shown).

The timestamps for each waveform were analyzed to determine the number of overlaps (defined as timestamps that fell within 1.6 ms of each other). When an overlap was detected, the final waveform was constructed as the sum of multiple spike templates plus noise. The relative alignment of the templates was assigned at random.

The timestamps for each neuron were also used to estimate inter-spike intervals. The trends observed by Quirk and Wilson (1999) were used to simulate decreases in spike amplitude observed during bursting events. Bursting events were defined as any set of spikes linked by ISIs smaller than 10 ms. The amplitude of the template being inserted into each waveform was reduced according to the position of a spike within a burst. If n is the position of the spike within the burst, the amplitude was adjusted to a percentage of that of the first spike equal to $(m \times n) + b$, with $m = -0.034$ and $b = 1.05$. This adjustment, combined with the effects of added noise, replicated variations in amplitude observed in real data.

2.4.4. Comparison with other algorithms

We directly compared the performance of DGCC two other algorithms, WaveClus (Quiroga et al., 2004) and NSMpro (Zhang et al., 2004). The code for both algorithms was obtained directly from the authors and only modified to accommodate differences in the structure of the data (number of sampling points, isolated waveforms instead of continuous recordings). Default parameters were used unless otherwise noted. A general overview of the algorithms is presented below.

WaveClus takes a traditional approach to non-parametric clustering: the first phase reduces the dimensionality of the data, while the second phase subdivides the compressed representations into distinct clusters. This algorithm does not deal with overlapping spikes explicitly; all spikes are clustered in reduced-dimensional space (as is usually done in manual clustering). A wavelet transform is applied during the first phase. Each wavelet coefficient represents the input signal at different scales and times, making this method well suited to the representation of neural spikes. Since wavelet decomposition does not decrease the dimensionality of the data per se, a subset of the wavelet coefficients must be selected in order to compress the data. WaveClus uses the 10-wavelet coefficients whose distributions have the largest deviations from normality as evaluated by the Kolmogorov–Smirnov (KS) test. This strategy favors the selection of coefficients with multi-modal distributions, promoting a segregation of data points into well-isolated clusters. Unlike our approach, this method attempts to preserve an accurate representation of spike shape in the feature space generated through dimensionality reduction.

The second stage of the WaveClus algorithm uses supraparamagnetic clustering (SPC) to sort the spikes after they have been reduced to 10 dimensional vectors as described above. At the beginning of each run of SPC, each point is assigned to a state s between 1 and q. During each iteration, a point p selected at random is switched to a random state s_new. The K nearest neighbors of p are screened to determine if they will also change their state to s_new. Only points that were originally in state s are considered. The probability that they will change their state to s_new is determined based on the distance between the points and a preset 'temperature' parameter. Similar points (those closest in the reduced feature space) have a higher probability of changing together. Lower settings of the temperature parameter will also increase the probability of a state change, regardless of the relative position of the points. The process is repeated for each nearest neighbor of p that changes state, until the boundary outlining the cluster of points remains stationary. Once this happens a new random starting point is selected and a new iteration begins. Sorting is performed at various temperature settings. Whereas at low temperatures points tend to change state together more often (generating fewer clusters) at high temperatures points are more likely to change state independently (generating more clusters). When processing the synthetic dataset we adjusted the temperature parameter so that the correct number of clusters was obtained.

The strategy used in DGCC only uses dimensionality reduction to identify possible cluster centers and extract templates. Once the templates are identified, the algorithm reverts to the original full-dimensional representation of the data (in this case vectors of 48 voltage measurements) to sort each spike. The implementation of WaveClus we used also provides an option that improves sorting speed by applying a similar strategy. The algorithm will run as described above for a preset number of spikes (we used the default parameter of 30,000), but will then switch to template matching after the clusters have been established. We evaluated sorting accuracy and speed under both conditions.

The NSMpro algorithm also extracts spike templates from high-density regions in principal component space and then performs template matching. Each high-density region is centered on a high-density point identified using subtractive clustering. Density is calculated at each data point based on the distances to all other points. Once the densest point is found it is assigned as the center of the first cluster. The contribution of this putative cluster is subtracted from the density estimate of all remaining points according to their distance from the cluster being removed. The process is repeated until the remaining points are below a preset density threshold. NSMpro only outlines high-density regions using fixed diameter circles around each cluster center, while DGCC uses isodensity curves calculated around local density maxima in a density matrix to represent regions of arbitrary size and shape. When processing the synthetic dataset, we adjusted the diameter of the circles used by NSMpro so that the extracted templates closely matched the real templates used to generate the synthetic dataset.

NSMpro and DGCC implement template matching in different ways. Both algorithms examine the difference vectors resulting from template subtraction in order to generate estimates of fit. However, while our algorithm uses the maximum deviation from zero (L[infinity] norm) of the difference vector, NSMpro evaluates the variance of the entire segment. In both cases the value is compared to expected noise levels to determine if a correct match has been made. Another important difference is the number of alignments at which the templates are subtracted. NSMpro iterates through all possible alignments until an acceptable match is found, while DGCC only tests template alignments close to the largest peaks or troughs of the difference vectors.

2.4.5. Template extraction comparison

Although our synthetic dataset contained a large number of examples spanning three signal-to-noise ratios, it only represented a single channel and therefore only provided one instance to test the template extraction phase. Rather than generating a large number of synthetic channels, which would take an extensive computational effort, we tested the template extraction using 89 actual data channels recorded from primate motor cortex as previously described. The length of the recording session was approximately 20 min. In addition to being processed using the algorithm, these signals were also sorted manually (using standard cluster-cutting techniques) by an expert using commercial software (Offline-Sorter, Plexon, Inc.). Performance of DGCC was evaluated by comparing the number and shape of these two sets of templates.

3. Results

3.1. Results with synthetic data

3.1.1. Template extraction

DGCC identified four high-density regions in principal component space for the synthetic dataset (Fig. 3). One template was automatically rejected because the variability of the waveforms within its associated central region was above preset parameters (the amplitude of the template was less than four times the mean standard deviation for all the voltage measurements). The high-density region in PCA space that represented this cluster was mostly composed of not only pure noise waveforms, but also included some shifted and overlapping spikes (examples are marked with + and * in Fig. 3) The three remaining templates were used for sorting. The extracted templates closely matched the original spike waveforms used to generate the dataset (Fig. 4).

3.1.2. Effect of sorting parameters on accuracy and processing speed

We evaluated the accuracy and speed of the algorithm by running it on the synthetic channel described above, which contained three units and one set of noise waveforms. In order to determine the range of parameters that yielded optimal results, we repeated the sorting procedure using various values for the overlap detection and template assignment thresholds. The overlap detection threshold determines when a template fit is good enough to warrant skipping the phase of the algorithm that

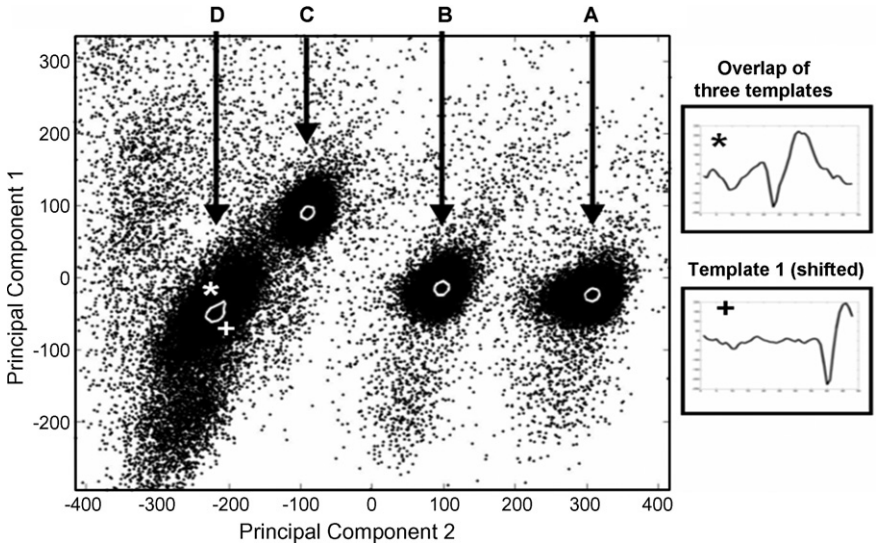


Fig. 3. Principal component projection translocation due to template shifting and spike overlaps. When projected onto the first two principal components, the waveforms of the synthetic dataset formed four main clusters (A–D). White outlines show the high-density regions identified using DGCC. Clusters A–C mainly contained waveforms from the simulated units, while cluster D contained most of the noise waveforms (as shown by the mean waveforms for each central region, shown in Fig. 4). Spike overlapping and template shifting can displace principal component projections away from the main cluster for each unit. Examples of waves affected by these factors are shown on the two panels on the right. The location of the principal component projection for these waves is marked with * and + symbols. In both cases the waveforms were displaced into the "noise" cluster. Similar translocations account for the points in the space surrounding the main clusters.

checks for overlapping spikes. The introduction of this parameter biases the algorithm towards parsimonious single template representations and reduces the running time. The template assignment threshold determines how closely a waveform must conform to a given template in order to be classified as a match to the template.

Each panel of Fig. 5 shows the percentage of false positives (solid lines) and false negatives (dashed lines) for each of the three simulated neurons as a function of the template assignment threshold. The results shown in each panel correspond to a different value of the overlap detection threshold. As expected, the neurons with the lowest SNR were associated with the highest false positive and false negative values. In all cases, as the template assignment threshold increased, false negatives tended to decrease while false positives tended to increase. However, the increase in false positives was far more gradual and tended to asymptote at levels below 5%, even for the lowest SNR examined. Two factors prevented a more dramatic increase in false

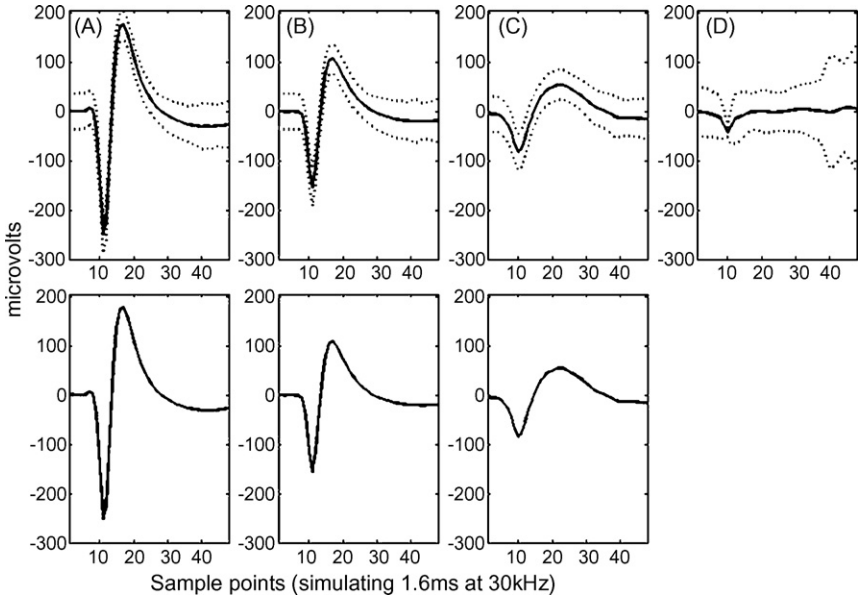


Fig. 4. Templates extracted from synthetic data. (Top row) Templates obtained by averaging the waveforms inside high-density regions shown in Fig. 2. Dashed lines show 4× the standard deviation at each point. Templates A–C were selected as valid templates for classification. They are close matches for the original spike templates used to generate the dataset (bottom row). The wave in panel D was rejected as a template for classification because the amplitude was less than 4× the mean standard deviation.

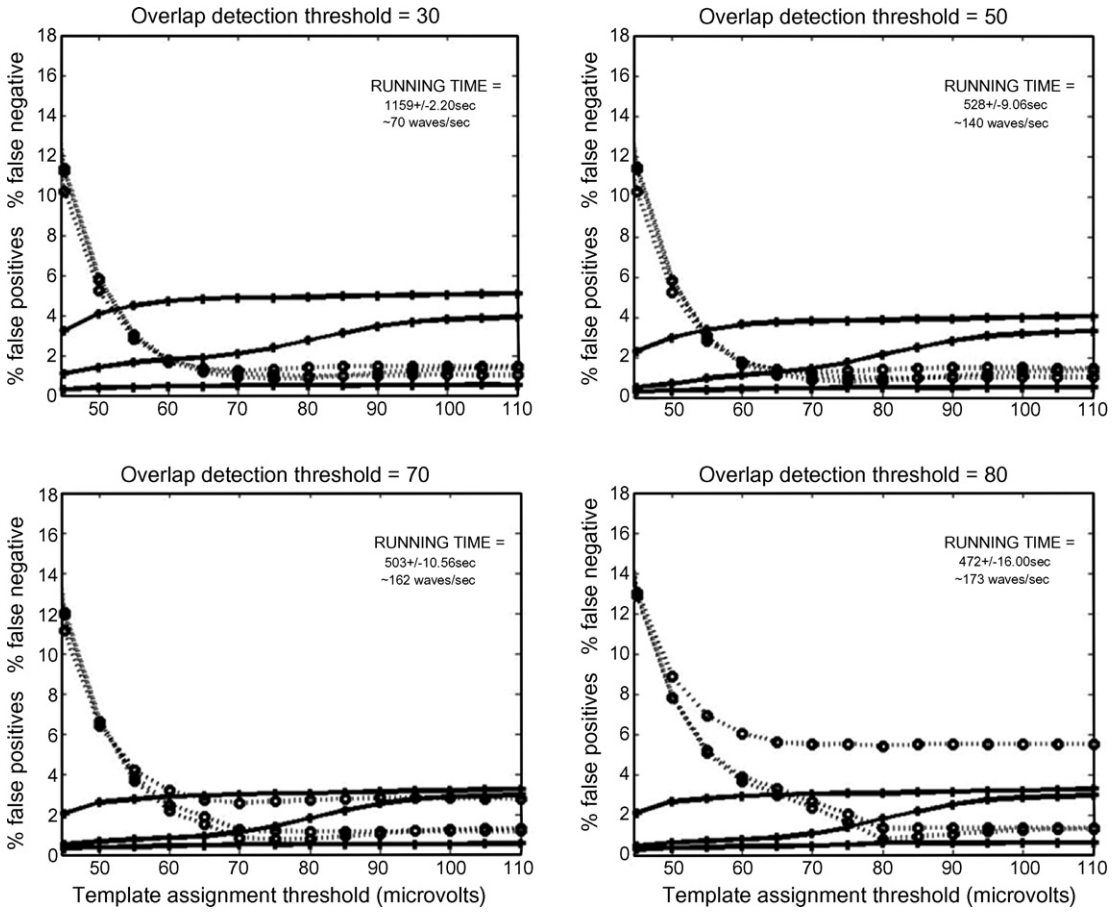


Fig. 5. Effects of threshold settings on error rates: false positives (continuous lines) and false negatives (dashed lines) are shown as a function of the template assignment threshold employed for classification. Each graph displays the results obtained with a different overlap detection threshold (see text for details). For overlap detection thresholds of up to 70 μV, both false positives and false negatives were below 5% if the template assignment threshold was greater than 55 μV.

positives. First, the noise detection step in the template matching sequence effectively pinpointed almost all of the pure noise waveforms by keeping track of the amplitude of the residual waveforms generated after subtracting candidate templates (if the amplitude of the residual was greater than that of the original waveform for all templates, the waveform was automatically classified as pure noise). Second, when many templates matched a given waveform, the waveform was only paired with the single best fitting template. Due to these two key steps in the sorting process (which do not depend on any of the other parameters) it was possible to use very liberal template assignment thresholds while maintaining near-optimal performance.

Changing the overlap rejection threshold affected the values at which false positives and false negatives reached a plateau. When this parameter was set to zero all waveforms were checked for overlaps (Fig. 5, top left panel). Low values for the overlap rejection threshold increased the number of false positives because of overfitting (i.e. giving priority to complex combinations of templates even when a single template provided an adequate fit). High values of the overlap rejection threshold can potentially generate more false negatives if the algorithm fails to recognize overlapping spikes. This situation was observed for the neuron with lowest signal-to-noise ratio (unit three, SNR = 5.5) at overlap rejection thresholds equal to or greater than 80 μV (see Fig. 5, bottom right). The false negatives for unit three originated almost entirely from overlapping spikes where the influence of this neuron (which had the smallest amplitude) was missed. Even with an overlap rejection threshold of 80 μV and a template assignment threshold set to 110 μV, more than 99% of the single-spike events for neuron three were identified. However, at this high overlap rejection threshold, the contribution of the other two templates to overlapping spikes was considered a good enough match, and the possible overlap for the smaller amplitude template was not explored. For the synthetic neuron with highest signal-to-noise ratio (17.1) performance was more stable and error rates were consistently below 2% for the full range of sorting parameters explored. The neuron with the second largest SNR (10.6) was between the other two in terms of performance and stability, maintaining error rates below 4% for the range mentioned above. A comprehensive breakdown of the waveforms giving rise to these errors is discussed in detail in Section 3.1.3. Overall, the algorithm generated less than 6% false positives and false negatives for each simulated neuron over a wide range of settings, spanning overlap detection thresholds between 0 and 70 μV at all template assignment thresholds higher than 50 μV.

Processing speed was strongly dependent on the overlap detection threshold used and ranged from 24 to 173 waveforms

Table 1
Classification results using DGCC (overlap detection threshold = 50, template assignment threshold = 75)

	Noise	A	B	C	A + B	A + C	B + C	A + B + C	No. of spikes
Actual origin									
Noise	**96.44**			3.55			0.01		15068
A	0.30	**98.47**	0.03	0.04	0.36	0.43	0.09	0.28	20264
B	0.26		**98.84**	0.08	0.11	0.02	0.45	0.23	20264
C	0.33			**99.41**			0.25	0.01	20199
A + B	1.38	0.86	0.57		**93.00**	0.17	0.17	3.84	1744
A + C	1.17	0.89		0.67	6.60	**85.35**	1.50	3.83	1802
B + C	3.84	0.06	0.85	1.81	0.45	1.19	**91.46**	0.34	1769
A + B + C	1.90					1.90	0.63	**95.57**	158

Each row represents the true category of the waveforms, while each column represents the category to which they were assigned during sorting. The rightmost column shows the total number of waveforms for each category. Entries in each row are reported as a percentage of these values. The diagonal in bold marks correctly classified waveforms. In some instances waveforms were partially misclassified: for example, the entry on the last row and sixth column shows that 1.90% of the 158 overlaps of all three templates were classified as being overlaps of only templates A and C.

per second (including the time taken to load the waveforms into MatLab). The slowest processing speed was observed when using an overlap rejection threshold of zero (Fig. 5, top left panel). At this value all waveforms were examined for possible template overlaps. Although this more thorough examination of the waveforms took longer, it did not yield the most accurate results: using higher values for the overlap rejection threshold not only resulted in faster, but also more accurate spike sorting with fewer false positives.

3.1.3. Detailed example of sorting results

Table 1 shows a detailed account of the sorting results obtained with an overlap rejection threshold of 50 μV and a template assignment threshold of 75 μV. These values are in the center of the range that resulted in stable performance for the algorithm. These values also match the bounds for the noise used to generate the synthetic dataset: more than 95% of the noise waveforms used to generate the synthetic dataset had amplitudes below 50 μV, and more than 99.9% of them had amplitudes below 75 μV.

Using these parameters more than 99% of all the single spikes and 93% of overlapping spikes for all units were correctly identified. More than 96% of all the pure noise waveforms were also correctly identified. Most of the remaining noise waveforms (3.55%) were incorrectly assigned to the template with the smallest amplitude. The total percentage of false positives (FP; templates that were detected within a waveform they had not been in fact added to, relative to the total number of detected spikes) and false negatives (FN; spikes that were added to a waveform but not detected, relative to the total number of spikes in the data) for each unit is summarized in Table 2 (rightmost column). False positives ranged from 0.48% (SNR 17.1) to 3.84% (SNR 5.5). False negatives ranged from 0.81% (SNR 17.1) to 1.33% (SNR 5.5). These error rates were enough to produce a shift in the estimated inter-spike interval distributions (Fig. 6). False positives resulted in abnormally short ISIs, shifting the distributions towards zero.

The accuracy of the shifts estimated by the algorithm to align the templates to each waveform was also evaluated. More than 98% estimated shifts for the templates that were correctly detected were within one sample point (equivalent to 1/30th of a millisecond) of the true shift. More than 0.999% were within three sample points, indicating that the algorithm reliably aligns waveforms. Misalignments generally occurred in the rare cases where the trough (global minimum) of a waveform was not within three sample points of the trough of at least one of the spikes contributing to the waveform.

Fig. 7 shows the locations in 2D principal component space of the waves attributed to units one and three by the algorithm, highlighting false positives (+) and false negatives (O). For clarity, only the first 5000 waveforms are included. Although the majority of the samples for each unit tended to be grouped in clusters, shifted as well as overlapping spikes were widely dispersed and often located in clusters that contained noise waveforms or spike waveforms of another unit. Misclassified waveforms tended to be located outside of the main clusters, but were often in close proximity to correctly identified spikes. These conditions make it impossible to accurately separate the waveforms originating from each unit by simply outlining areas in principal component space.

3.1.4. Comparison with other algorithms

The overall accuracy and processing speed for all algorithms tested is shown in Table 2. DGCC provided the most accurate results, especially for the unit with lowest signal-to-noise ratio. DGCC also had the shortest running time of all algorithms tested, as shown in Table 3. The direct comparison between the algorithms was performed on a computer with a higher processor speed than the previous analyzes (see Section 2 for

Table 2
Sorting accuracy

	WaveClus	NSMpro	DGCC
Unit A %FP	0.29	0.11	0.48
Unit A %FN	18.02	9.45	0.81
Unit B %FP	3.33	1.89	1.75
Unit B %FN	18.65	8.51	1.01
Unit C %FP	40.53	12.55	3.84
Unit C %FN	12.15	11.05	1.33

Total percentage of false positives and false negatives for each simulated neuron obtained with each of the three algorithms tested. Each component of overlapping spikes was counted independently.

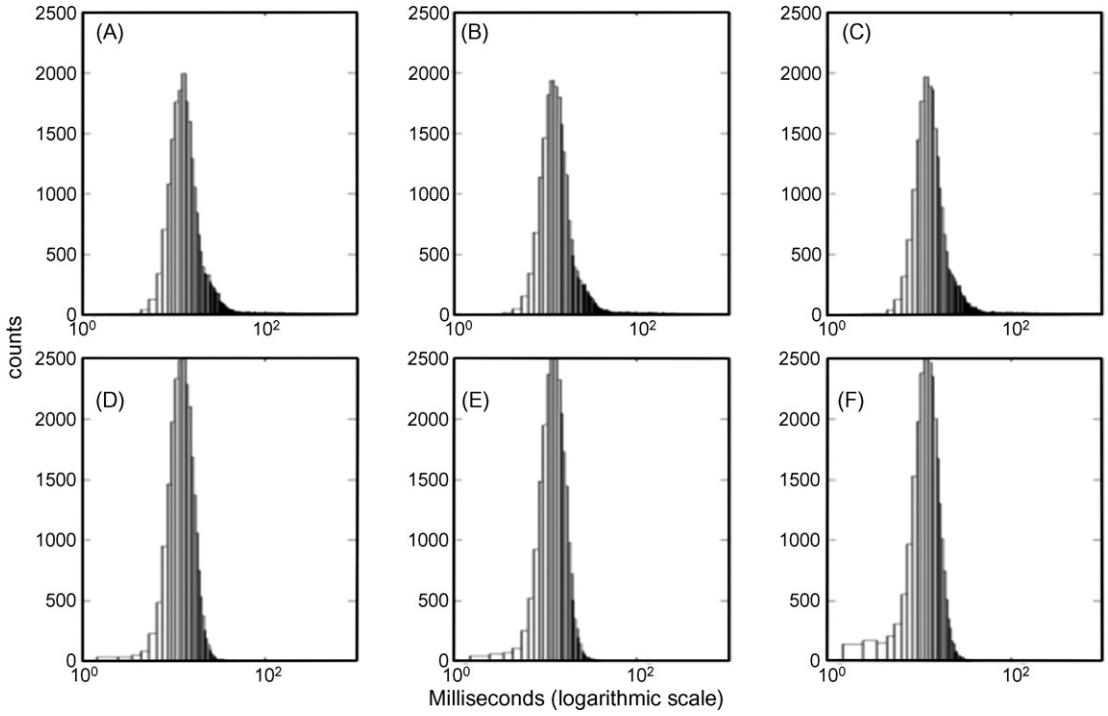


Fig. 6. Effects of sorting in inter-spike interval distributions. Panels A–C show the true inter-spike interval distribution of the timestamps for the synthetic units. Panels D–F show the inter-spike interval distributions for the timestamps attributed to each unit by the algorithm using a threshold of 60 μV. False positive rates of 0.79, 1.14 and 3.46% were enough to alter the shape of the distributions, resulting in the appearance of a few events in the refractory period.

Table 3
Processing speed

Algorithm	Waveforms/s
DGCC	211.81
WaveClus with template matching	55.84
WaveClus	18.52
NSMpro	0.18

The processing speed of each algorithm was calculated using our synthetic dataset. The WaveClus algorithm was run using only supraparamagnetic clustering (SPC) as well as using a combination of SPC (for the first 30,000 waveforms) and template matching (without overlap resolution).

details). Under these conditions DGCC performed in excess of 200 spikes/s (using an overlap rejection threshold of 50 μV and a template assignment threshold of 75 μV, as described above). Surprisingly, it performed faster than WaveClus even though this algorithm did not perform overlapping spike decomposition.

Detailed sorting results using NSMpro are summarized in Table 4. The overall distribution of sorting errors was similar to what was observed for DGCC. However, NSMpro tended to assign more spikes to the 'noise' category, even though this algorithm checks more possible spike alignments than DGCC. These missed spikes are therefore probably due to the function chosen to evaluate template matches. Taking into account the variance of the entire difference vector rather than the single largest deviation from zero may have decreased the sensitivity of spike detection.

Detailed sorting results using WaveClus are summarized in Table 5. The wavelet-space representation of the synthetic dataset displayed three clusters associated with the three spike templates (data not shown). However, overlapping spikes tend to be projected away from the main clusters. WaveClus performed well on single-spike events, but classified most of the overlapping spikes as noise.

Table 4
Classification results using NSMpro

	Noise	A	B	C	A + B	A + C	B + C	A + B + C	No. of spikes
Actual origin									
Noise	**88.80**			11.20					15068
A	7.90	**88.91**		0.06	0.01	2.40	0.67	0.05	20264
B	8.32		**90.62**	0.05		0.01	0.97	0.03	20264
C	11.22			**88.47**		0.01	0.29	0.015	20199
A + B	3.67	4.07	2.47		**62.90**	4.99	3.90	18.00	1744
A + C	6.71	2.00		1.11		**78.41**	9.88	1.890	1802
B + C	7.63	0.06	2.20	2.32		0.28	**87.28**	0.23	1769
A + B + C	3.16			0.63		6.96	10.13	**79.11**	158

Spike sorting results using NSMpro. Data is presented with the same conventions used for Table 1.

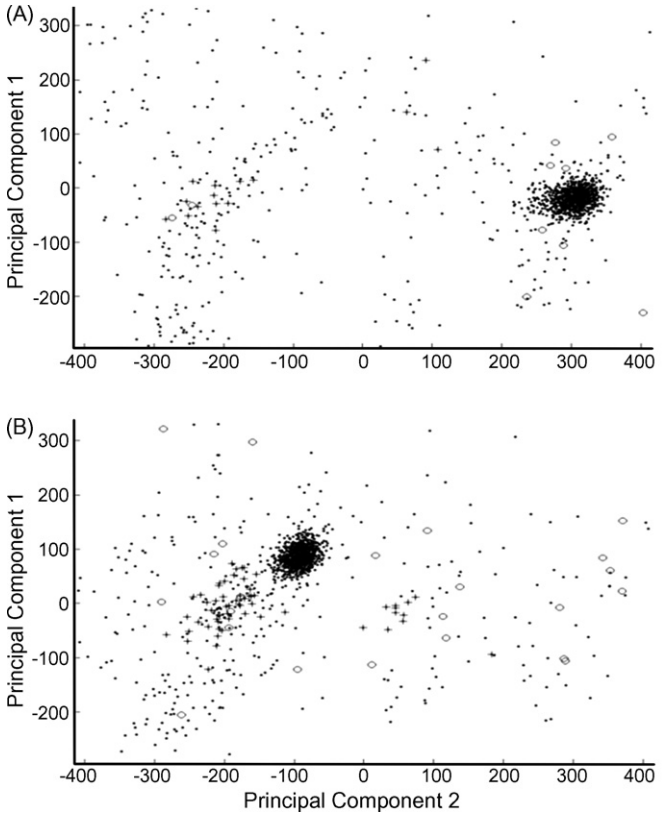


Fig. 7. Principal component projections alone are not adequate for accurate spike sorting. The dots in each plot represent the location of correctly classified waveforms for a given unit in principal component space. Panel A displays the result for unit one (highest SNR) and panel B shows the result for unit three (lowest SNR). For clarity, only the first 5000 waveforms of the dataset are shown. False positives (+) and false negatives (O) were generally surrounded by correctly classified waveforms. Although projections from each unit are more numerous in a given cluster, they are often found in the wide field of principal component space shared by all units. These locations include the cluster that contained the bulk of the noise (where most of the false positives can be observed). These results show how difficult it can be to classify waveforms based solely on low-dimensional representations.

Table 5
Classification results using WaveClus

	Noise	A	B	C	No. of spikes
Actual origin					
Noise	**12.87**			87.13	15068
A	4.49	**93.02**	0.24	2.25	20264
B	2.31	0.01	**93.97**	3.71	20264
C	0.08		0.01	**99.91**	20199
A + B	86.35	10.21	2.75	0.69	1744
A + C	58.99	33.91		7.10	1802
B + C	33.46	3.22	21.42	41.89	1769
A + B + C	86.08	6.33	1.90	5.70	158

Spike sorting results using WaveClus. Data is presented with the same conventions used for Table 1. This algorithm did not detect spike overlaps, so the corresponding columns are omitted.

3.2. Results with actual cortical recordings

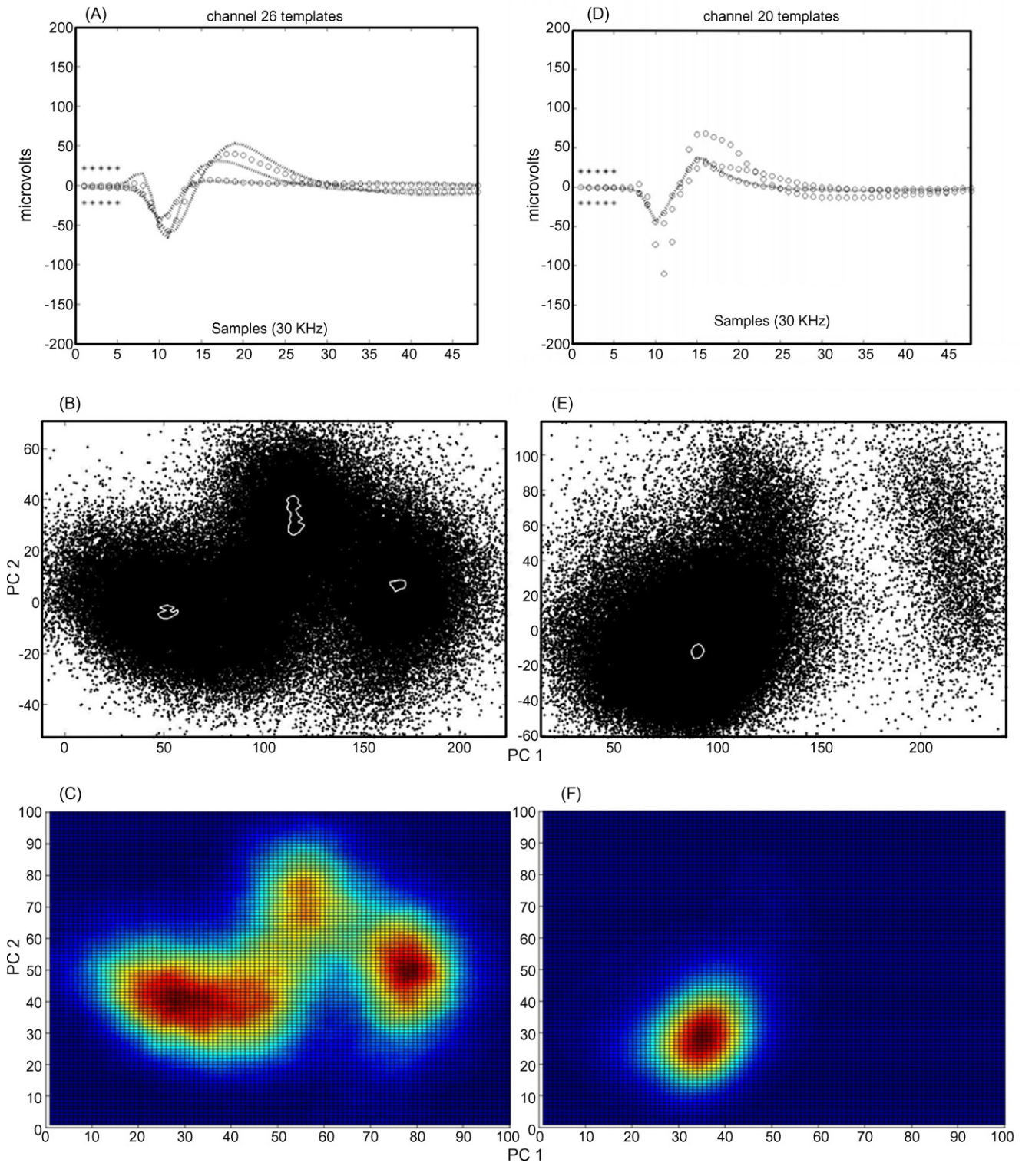
3.2.1. Template extraction

Spike templates extracted by an expert using commercial software were compared to those reported by the algorithm. For 89 channels, total of 140 different spike template shapes were identified using both techniques. The number of templates extracted by the algorithm matched the number of manually extracted templates on 80% of all channels analyzed. The shapes of the templates extracted by the algorithm closely matched those of manually extracted templates.

The algorithm extracted 17 templates that were not reported by the expert sorter (12% of the total number of templates). In 11 of these cases, the algorithm identified separate high-density regions within what appeared to be a single cluster of data points when visualized using the Offline Sorter software package. Fig. 8 shows an example of this kind. Panel A shows the templates extracted using both methods. In accordance with the template selection procedure, the differences between all templates extracted from the same channel by the algorithm exceeded the expected noise distortion. Panel B shows the principal component projections of the waveforms (similar to the display the expert sorter used). Panel C shows the density matrix generated by the algorithm from the principal component projections. By comparing panels B and C it is possible to appreciate how the separation between clusters is highlighted when the density of data points is taken into account. The remaining six templates that were only detected by the algorithm were extracted from channels where the expert sorter reported no units and represented low amplitude multi-unit activity.

Fig. 8. Template extraction results: DGCC vs. manual sorting using offline-sorter. (A–C) Template extraction results from a channel where the algorithm identified more templates than were manually extracted by an expert sorter. Panel A shows the template waveforms extracted by the expert sorter (circles) and the algorithm (dashed lines). Asterisks indicate the estimated noise levels from the pre-trigger epoch corresponding to the first five samples of each waveform (the range covering 95% of the deviations from zero). Panel B shows the principal component projections of the waveforms for this channel, highlighting the high-density regions identified by the algorithm (white outlines). Panel C shows the density matrix, where PC space is divided into a 100×100 grid where the color of each partition represents the number of data points. Although three clusters are clearly visible in the density matrix, they are less obvious in the principal component projection. When the dataset was sorted manually, the two rightmost clusters were combined, and the resulting mean waveform was representative of the low-density area between the clusters. The templates for the leftmost cluster extracted using both methods were nearly identical. (D–F) Template extraction results from a channel where the algorithm identified fewer templates than the expert sorter. These panels follow the same conventions as (A–C). Waveforms representing the leftmost cluster were extracted using both methods, producing nearly identical templates. The algorithm did not extract templates from the two other clusters. One of these templates was too similar to another template on the same channel (their maximum difference being smaller than the estimate of noise, shown by the asterisks over the first five sample points). Nearly all the templates rejected by the algorithm were of this type. The second, higher-amplitude template was missed because the representative waveforms failed to form a dense cluster in PCA space. This situation was only observed in less that 1.5% of the total number of templates analyzed (2 out of 140), and included the only two missed templates with amplitudes greater than 100 μV. In this case, the waveforms associated with other clusters outnumbered the overlooked cluster by more than 40-to-1. Over the entire recording period, the firing rate for the neuron that was missed was less than 1 Hz. This result shows that the algorithm can miss neurons with very low firing rates if they are recorded on the same channel as other more active neurons.

The algorithm did not report 12 of the templates extracted manually (9% of the total number of templates). Ten of these 12 templates were identified in the first phase of template extraction, but rejected because their shape was too similar to those of other templates on the same channel. As described in Section 2, the minimum difference allowed between templates was set by taking the voltage values encompassing 95% of the deviations from zero in the pre-trigger time interval in order to determine expected variations due to noise. All of the templates rejected because of similarities with other templates had relatively low amplitudes (less than 100 μV). The remaining two templates missed by the algorithm (less than 1.5% of the total number of templates) had relatively high amplitudes and were distinct from other templates in the same channel. However, they were



overlooked because they were represented by relatively few waveforms that did not form distinct clusters in PCA space. Fig. 8, panels D–F, show an example of this kind, where a neuron with very low firing rate was missed by the algorithm. The diffuse cluster shown on the right side of panel E is not captured by the density matrix shown in panel F, although the waveforms associated with this cluster represented a high amplitude spike template (shown in panel D).

3.2.2. *Detailed example of sorting results for cortical recordings*

We applied our sorting algorithm to a channel of cortical recordings obtained from a macaque performing a visually guided reaching task. This particular channel was the same from which the waveform templates used to construct the synthetic data were originally extracted. The recording contained 95,222 waveforms collected over 22 min. DGCC identified three high-density cluster centers corresponding to three units, the same number as an expert sorter using commercial software. In this recording sample there was no evidence of a separate noise cluster, suggesting that the data collection trigger was appropriately set to exclude almost all pure noise waveforms. Background noise levels were approximated using the first five voltage measurements in each waveform. These measurements take place between 0.4 and 0.23 ms before the threshold crossing that triggers data collection and were therefore guaranteed to contain no spikes. Ninety-five percent of these samples had deviations from zero smaller than 45 μV. This value was therefore chosen as the template rejection threshold. Using a similar approach, the template assignment threshold was set to 75 μV (which encompassed 99% of the noise sample). Sorting results using these parameters shown in Fig. 9. The top panel shows the principal component projections for the waveforms, with the high-density areas detected by the algorithm highlighted in white. The template waveform for each high-density area is displayed below. Standard deviations around the templates reflect the variability of the final sorted waveforms assigned to each template (including waveforms extracted from overlapping spikes). More than 99.99 of the waveforms were assigned to one of the three identified spike templates. About 22% of the waveforms were classified as overlapping spikes (more than twice the number in the synthetic dataset), even though the simulated neurons had average firing rates approximately three times higher than those in the real recording. Due to the increased number of overlapping spikes, processing speed was only ∼82 waveforms/s (total processing time for this 22 min recording was ∼19 min). Overlapping spikes were separated into single unit components based on the template alignment that yielded the best match. An example of how a waveform with overlapping spikes was separated is shown in the left side of Fig. 10. The original waveform is decomposed by sequentially subtracting the spike templates after aligning them with the global minimum. Although several sequences of template orders and alignments were used, only the sequence that provided the best fit for the waveform is shown. The left side of Fig. 10 shows the decomposition of one of the waveforms from the synthetic data set for comparison. When

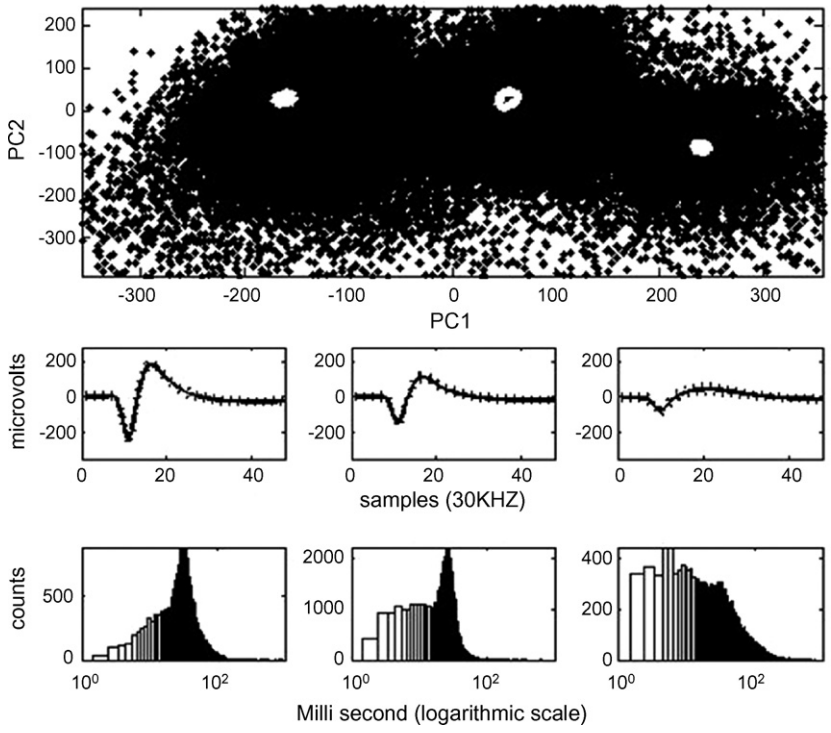


Fig. 9. Sorting results for activity recorded in primate primary motor cortex. (Top) When applied to a sample of real data; (DGCC) identified three high-density regions similar to those that were originally found by an expert sorter using commercial software. (Middle) The templates extracted from each region (solid lines) and the standard deviation of the waveforms reconstructed for each unit after resolving template overlaps (dashed lines). (Bottom) Inter-spike intervals for the sorted units, plotted on a millisecond logarithmic scale.

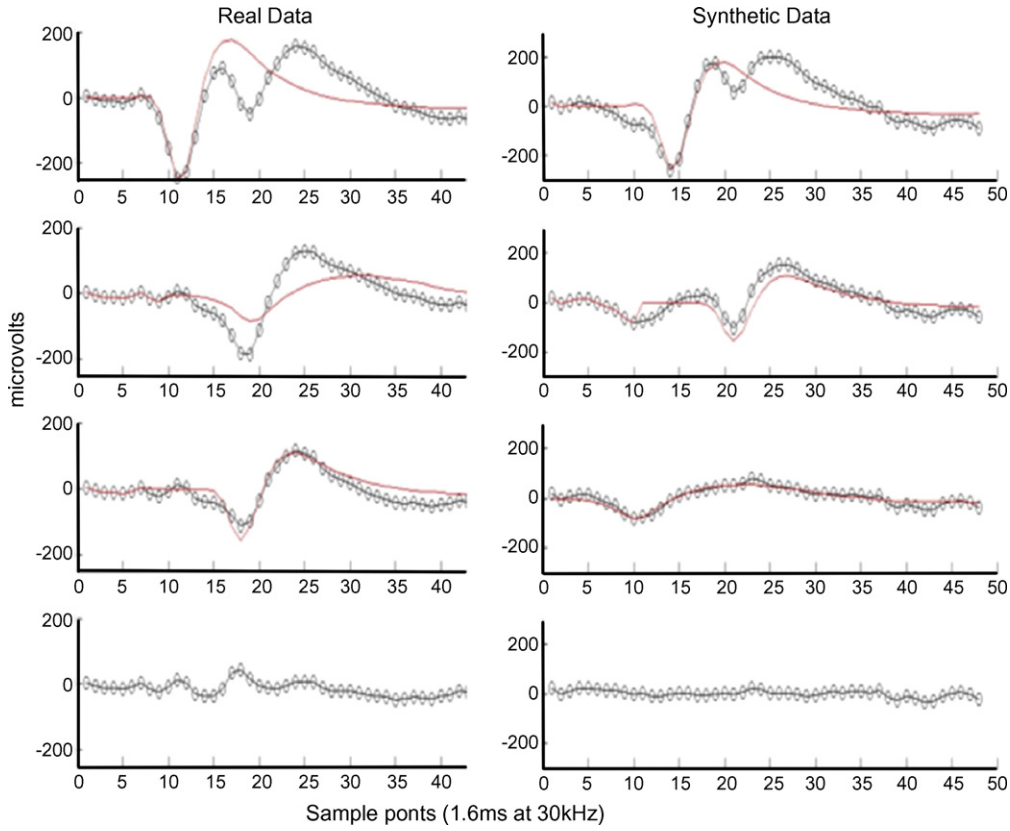


Fig. 10. Waveform decomposition example. (Left) Waveforms in the real data often showed evidence of additive combination of spikes from different neurons (overlapping spikes). Template decomposition reveals how the three identified spike templates can be used to account for the observed waveform shape. Each template (solid lines) is aligned to the global minimum and subtracted until only a low amplitude noise signature remains (the template alignment that produced the smallest residue is shown). The remainder of each subtraction is shown by dotted lines. (Right) This situation was replicated in the synthetic dataset, where the accuracy of the procedure could be accurately evaluated. In the sorted synthetic data, more than 95% of the estimated template positions were within one sample point of the true alignment used to generate the waveforms.

overlapping spikes were decomposed, the new timestamps were generated for the extracted waveforms reflecting their position in the recording window. The inter-spike interval distributions for the first two units displayed evidence of a refractory period (through a gradually decreasing number of short ISIs), suggesting that they represent individual neurons (Fig. 9, bottom). The third unit contained a significant number of events in the refractory period. This abundance of short ISIs suggests that this sorted unit actually represented the activity of more than one cell. If this is the case, the spikes for the cells involved were very similar in shape, since the standard deviations of the voltage measurements were small and comparable to those observed for the other sorted components (Fig. 9, middle). The third unit displayed the smallest amplitude, further supporting the idea that it included the activity of several relatively distant neurons.

4. Discussion

Spike sorting is an integral part of any electrophysiological analysis of neural activity at the single neuron level. This important data processing step is often taken for granted and not described in detail. The emergence of automated spike sorting methods offers the promise of much needed standardization and reproducibility in the field of electrophysiological cortical recordings.

Most spike sorting algorithms begin by applying feature extraction techniques such as principal component analysis (Zhang et al., 2004) or wavelet transformations (Quiroga et al., 2004) and then attempt to find clusters representing waveforms with similar properties in the resulting reduced-dimensional space. There are two general strategies used for clustering: parametric and non-parametric. The distribution of waveform shapes can be parameterized using mixture models where each mixture component represents the contribution of a putative single neuron (Shoham et al., 2003; Kim and Kim, 2003). Determining the number of units in the mixture can be a challenging problem. To prevent over-fitting, a balance must be struck by improving the fit of the model while penalizing large number of mixture components. Many possible mixtures must be compared as the parameters are gradually optimized. Choosing the type of functions to mix poses a similar problem: adding more parameters can improve fit, but makes optimizing the parameters more challenging. Once the model parameters are estimated, this strategy has the advantage of allowing straightforward classification using a maximum likelihood approach. However, the translation of maximum likelihood decision boundaries to the full high-

dimensional waveform space does not always yield desirable results because low-dimensional projections may fail to reflect nuances of waveform shape (Fee et al., 1996a). Although it is possible to model parameters in high-dimensional space, this greatly increases the already considerable computational effort required. Moreover, waveform shifting and spike overlapping can result in distorted waveforms that will not be located within well-defined clusters and can show up as outliers in parametrically modeled distributions (Figs. 3 and 7 in this paper; Brown et al., 2004; Takahashi et al., 2003; Fee et al., 1996a,b). The ability to process irregular waveforms including overlaps and shifts provides a more accurate and unbiased representation of neural activity. Although they represent a minority of the total number of spikes, irregular waveforms may provide valuable information. For example, inappropriate sorting of overlapping spikes can distort estimates of neuronal correlation used to evaluate local neuronal synchrony and connectivity (Bar-Gad et al., 2001). Non-parametric methods can effectively locate the centers of clusters without requiring any a priori assumptions about the distribution of waveform shapes. However, unlike their parametric counterparts, setting the boundaries to separate clusters from noise waveforms as well as each other cannot be accomplished in a straightforward statistical fashion according to previously evaluated model parameters. For this reason this kind of algorithms are generally used as a preliminary step for yet another non-parametric method: template matching. Template-based sorting provides a computationally efficient way to compare high-dimensional waveforms. If representative waveforms from each neuron recorded in a given channel are known, they can be used as templates against which unclassified waveforms can be compared in order to determine the most likely neuron of origin. Such templates are usually generated by taking the mean of waveforms thought to be near the center of a given cluster. The simplest form of template-based spike sorting requires the experimenter to visually inspect the data in the reduced feature space and manually draw contours around each cluster. This type of spike sorting can be a very tedious and arbitrary process when large datasets need to be analyzed. The speed, precision and reliability of template sorting is therefore greatly increased when it is combined with non-parametric algorithms that can automatically extract templates.

The algorithm described in this paper relies on non-parametric estimates of data density to extract spike templates from regions in principal component space. The assumption underlying this method is that non-shifted, non-overlapping waveforms from a given neuron will form a relatively dense cluster in principal component space. Even if there is an equivalent number of overlapping and shifted waveforms, the range of possible phase differences will tend to scatter them more widely. The spike templates extracted from high-density regions can be used as high-dimensional reference points that not only reflect the stereotyped shape of stable waveforms, but can also be used to decompose irregular waveforms into their single-neuron components in a fast and effective manner. Density grid contour clustering estimates the density of points within small partitions of principal component space and therefore requires only one iteration through each data point to generate a detailed density profile, considerably reducing the time required to extract templates. Our template matching procedure is also geared to maximize computational efficiency. Rather than checking for all possible template alignments, individual template fits are evaluated at a small range of alignments centered on the maximum deviation from zero of the waveform being sorted. A third factor contributing the computational efficiency of the algorithm was the use of an overlap detection threshold. This value was used to determine when a single template match was good enough to accept without checking for possible template overlaps. This allows the algorithm to expedite template matching without compromising accuracy. Indeed, giving precedence to the more parsimonious single template explanation for waveform shape actually improved sorting performance. The best results were obtained when the overlap detection threshold was set to encompass ∼95% of the noise and paired with a more liberal template assignment threshold. Having different values for the template assignment and overlap rejection thresholds expanded the range over which the two parameters produced optimal results (see Fig. 5).

4.1. Comparison with WaveClus and NSMpro in terms of accuracy and speed

DGCC processed our simulated dataset faster and more accurately than the WaveClus (Quiroga et al., 2004) and NSMpro (Zhang et al., 2004) algorithms. Optimal sorting results for DGCC were obtained using threshold values that roughly approximated the amplitude of the noise, with slightly higher values for the template assignment threshold. Sorting accuracy was not seriously compromised when the values chosen strayed from the optimal settings, a useful characteristic when dealing with real data for which estimates of noise levels may be inaccurate. Spike representations in WaveClus use 10 wavelet coefficients and undoubtedly capture more variations in waveform shape than the projections onto two principal components used in DGCC. Indeed, Quiroga et al. have shown that clustering in wavelet space outperforms clustering in principal component space as well as clustering in the space of unprocessed spike waveforms (where each voltage measurement is used as a dimension). However, our simple representation holds enough information to accurately extract spike templates. These templates can then be used to sort unprocessed waveforms accurately and quickly (avoiding the so called ‘curse of dimensionality’). NSMpro has been compared to Atiya’s method (Atiya, 1992) and found to be both faster and more accurate (Zhang et al., 2004). Although NSMpro and DGCC use the same overall template matching strategy, our algorithm was faster by more than three orders of magnitude when run on our artificial dataset. Surprisingly, overall sorting accuracy was also higher for DGCC, even though it only checked a small subset of possible template alignments examined by NSMpro. This result suggests that our template matching strategy is effectively narrowing down the search space of possible template combinations, improving processing speed without compromising sorting accuracy.

4.2. Results with electrophysiological data

It is impossible to definitively evaluate spike-sorting accuracy when processing real data where the true origin of the waveforms cannot be known. However, the results obtained suggest that the algorithm still performs well under these conditions when compared to expert human sorting.

The templates extracted by the algorithm almost exactly matched those extracted manually for 80% of the channels. In most cases where the extracted templates differed, the algorithm either joined two clusters extracted by the expert sorter or extracted two templates from a single cluster outlined by the expert. The number of neurons identified by the algorithm depended on the number of local density maxima present in the principal component projection of the data, as well as the similarity between shapes of the mean waveforms associated with each density peak. Thus, the cluster assignments determined by the algorithm are more consistent and based on more information than those performed manually using Offline-Sorter. Comparable discrepancies in the numbers of detected neurons have been reported when comparing results between different users employing the same software (Wood et al., 2004). It is also worth pointing out that almost all the disagreements on the inclusion of a given template occurred when the waveform in question had a relatively low amplitude. High-amplitude templates were only overlooked by the algorithm when a neuron with a very low firing rate (close to 1 Hz for the only two such cases observed) was recorded together with another more active neuron (with firing rates more than an order of magnitude higher) on the same channel.

The results for the template matching phase of the algorithm were examined in detail for a neural recording channel where both the algorithm and the expert sorter identified three different spike templates. Nearly all of the waveforms were assigned to at least one of the templates. Single spikes were correctly extracted from overlapping waveforms, as evidenced by relatively small variation around each spike template.

Processing speed for cortical recordings was somewhat slower than what was observed for artificially generated data as a result of a greater incidence of overlapping spikes. The increased number of spike overlaps in the real data probably corresponds to brief correlated peaks in firing not present in the synthetic dataset. Even under these conditions, processing speed still exceeded that of previously described algorithms with overlap resolution by more than three orders of magnitude.

4.3. Influence of low SNR units on sorting accuracy

Sorting accuracy for the high-amplitude templates was not compromised by the inclusion of units with smaller amplitude waves. Furthermore, it can be argued that rejecting neurons with low-amplitude representative waveforms (by setting the data collection trigger at relatively high values, for example) can introduce new sources of uncertainty to the template matching process. Our template decomposition strategy enables us to separate the influence of various neurons on a given voltage trace, so keeping track of units with low amplitude signals may allow for more efficient overlap resolution. This not only helps to account for a greater portion the variability in the waveform shape for all units on a given channel, but also results in more accurate waveform and timestamp reconstruction for overlapped waves. The presented algorithm can efficiently carry out this task even in the presence of substantial noise contamination. This is a highly desirable property when working with fixed multielectrode arrays where electrode position cannot be adjusted after implantation and recording spikes from the greatest number of possible neurons is a priority.

4.4. Future work

One possible complication for the presented algorithm arises if a pair of neurons with widely different firing rates is recorded on the same channel. It is possible that the more frequently firing neuron may mask the distribution of waveform shapes for the less active one. In this case the DGCC template extraction phase could miss the smaller peak in density. So far this situation has only been observed with neurons whose firing rates differ by more than an order of magnitude. Future versions of the algorithm could address this problem using a modified form of subtractive clustering: by isolating the waveforms classified as ‘noise’ during the first pass of the algorithm, the influence of the neurons associated with a high density of spikes could be removed, allowing these waveforms to be scanned for templates independently.

Appropriate selection of the classification parameters is a concern seldom addressed in spike sorting field. Although the algorithm performs well over a range of parameter values, future development will focus on improved automatic threshold adjustment based on inter-spike interval distributions. Modifying threshold values to eliminate events in the refractory period could contribute to the isolation of single neurons. Template matching lends itself well to this approach because it provides an estimate of fit for each template (or template combinations). A given dataset can therefore be re-sorted using a different threshold with minimal computational effort by keeping track of all the estimates of fit. Different parts of a recording session could even be sorted with different threshold values: for example, pairs of spikes with suspiciously short inter-spike intervals could be re-sorted using a more stringent template assignment threshold. Comparing estimates of fit could also be used to determine which spike in such a pair should remain in the cluster and which needs to be reassigned.

Future versions of the algorithm will also implement threshold values that can be dynamically adjusted for each neuron based on recent firing history. Although most sources of variation in spike shape affect all neurons in a given channel equally (noise in the electronics, activity of distant neurons), modulation of spike amplitude associated with rapid firing may vary considerably from neuron to neuron. Although such variations in amplitude were present in our synthetic data, our sorting process did not directly address them. Using different thresholds for each neuron depending on expected variation in amplitude may prove advantageous.

4.5. Source code availability

The current version of DGCC is written in MatLab 7 and designed to take .nev files as input. Source code for DGCC as well as the synthetic dataset used for performance evaluation are available upon request.

Acknowledgements

We thank Wilson Truccolo, John Simeral, Matthew Fellows and Juliana Dushanova for their valuable feedback during the development of the algorithm and the drafting of this paper. This work is partially supported by NIH-NINDS NS-25074, the VA, and by the Office of Naval Research, NRD-386.

References

Atiya AF. Recognition of multiunit neural signals. IEEE Trans Biomed Eng 1992;39(7):723–9.

Baker SN, Lemon RN. Precise spatiotemporal repeating patterns in monkey primary and supplementary motor areas occur at chance levels. J Neurophysiol 2000;84(4):1770–80.

Bar-Gad I, Ritov Y, Vaadia E, Bergman H. Failure in identification of overlapping spikes from multiple neuron activity causes artificial correlations. J Neurosci Methods 2001;107(1/2):1–13.

Brown EN, Kass RE, Mitra PP. Multiple neural spike train data analysis: state-of-the-art and future challenges. Nat Neurosci 2004;7(5):456–61 [review].

Fee MS, Mitra PP, Kleinfeld D. Automatic sorting of multiple unit neuronal signals in the presence of anisotropic and non-Gaussian variability. J Neurosci Methods 1996a;69(2):175–88 [erratum in: J Neurosci Methods 1997;71(February (2)):233].

Fee MS, Mitra PP, Kleinfeld D. Variability of extracellular spike waveforms of cortical neurons. J Neurophysiol 1996b;76(6):3823–33.

Harris KD, Henze DA, Csicsvari J, Hirase H, Buzsaki G. Accuracy of tetrode spike separation as determined by simultaneous intracellular and extracellular measurements. J Neurophysiol 2000;84(1):401–14.

Kim KH, Kim SJ. Method for unsupervised classification of multiunit neural signal recording under low signal-to-noise ratio. IEEE Trans Biomed Eng 2003;50(4):421–31.

Lewicki MS. A review of methods for spike sorting: the detection and classification of neural action potentials. Network 1998;9(4):R53–78 [review].

Quirk MC, Wilson MA. Interaction between spike waveform classification and temporal sequence detection. J Neurosci Methods 1999;94(1):41–52 [erratum in: J Neurosci Methods 2000;99(June (1–2)):143–145].

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

Shoham S, Fellows MR, Normann RA. Robust, automatic spike sorting using mixtures of multivariate *t*-distributions. J Neurosci Methods 2003;127(2):111–22.

Snider RK, Bonds AB. Classification of non-stationary neural signals. J Neurosci Methods 1998;84(1–2):155–66.

Suner S, Fellows MR, Vargas-Irwin C, Nakata K, Donoghue JP. Reliability of signals from chronically implanted, silicon-based electrode array in non-human primate primary motor cortex. IEEE Trans Neural Syst Rehabil Eng 2005;13(4):524–41.

Takahashi S, Anzai Y, Sakurai Y. Automatic sorting for multi-neuronal activity recorded with tetrodes in the presence of overlapping spikes. J Neurophysiol 2003;89(4):2245–58 [epub December 18, 2002].

Wood F, Black MJ, Vargas-Irwin C, Fellows M, Donoghue JP. On the variability of manual spike sorting. IEEE Trans Biomed Eng 2004;51(6):912–8.

Zouridakis G, Tam DC. Identification of reliable spike templates in multi-unit extracellular recordings using fuzzy clustering. Comput Methods Programs Biomed 2000;61(2):91–8.

Zhang PM, Wu JY, Zhou Y, Liang PJ, Yuan JQ. Spike sorting based on automatic template reconstruction with a partial solution to the overlapping problem. J Neurosci Methods 2004;135(1/2):55–65.