Spike sorting for large, dense electrode arrays

Cyrille Rossant[1,2,3], Shabnam N. Kadir[1,2,3], Dan F. M. Goodman[4], John Schulman[5], Maximilian L.D. Hunter[2,3], Aman B. Saleem[6], Andres Grosmark[7], Mariano Belluscio[7], George H. Denfield[8], Alexander S. Ecker[8], Andreas S. Tolias[8], Samuel Solomon[9], Gyorgy Buzsaki[7], Matteo Carandini[6], Kenneth D. Harris[2,3,10]

1 – Equal contribution.
2 – UCL Department of Neuroscience, Physiology and Pharmacology, London, UK
3 – UCL Institute of Neurology, London, UK
4 – Department of Electrical and Electronic Engineering, Imperial College, London, UK
5 – Department of Electrical Engineering and Computer Sciences, UC Berkeley, Berkeley, CA
6 – UCL Institute of Ophthalmology, London, UK
7 – NYU Neuroscience Institute, Langone Medical Center, New York, NY
8 - Department of Neuroscience, Baylor College of Medicine, Houston, TX
9 – UCL Institute of Behavioural Neuroscience, Department of Experimental Psychology, London, UK
10 – Correspondence: kenneth.harris@ucl.ac.uk

Abstract

Developments in microfabrication technology have enabled the production of neural electrode arrays with hundreds of closely-spaced recording sites, and electrodes with thousands of sites are currently under development. These probes in principle allow the simultaneous recording of very large numbers of neurons. However, use of this technology requires the development of techniques for decoding the spike times of the recorded neurons, from the raw data captured from the probes. Here, we present a set of novel tools to solve this problem, implemented in a suite of practical, user-friendly, open-source software. We validate these methods on data from the cortex, hippocampus, and thalamus of rat, mouse, macaque, and marmoset, demonstrating error rates as low as 5%.

Introduction

One of the most powerful techniques for neuronal population recording is extracellular electrophysiology using microfabricated electrode arrays[1-3]. Advances in microfabrication have continuously increased the number of recording sites available on neural probes, and the number of recordable neurons is further increased by having closely spaced recording sites. Indeed, while a single sharp electrode can provide good isolation of one or two neurons, placing as few as four recording sites together in a “tetrode” can reveal the firing patterns of 10-20 simultaneously recorded cells[4-7]. This increase is possible because each recorded neuron produces extracellular action potential waveforms (“spikes”) with a characteristic spatiotemporal profile across the recording sites[8-10]. The process of using these waveforms to decipher the firing times of the recorded neurons is known as “spike sorting”[11, 12].

Spike sorting, as currently applied in nearly all labs using extracellular recordings, involves a manual operator. While some labs use a fully manual system, lower error rates can be achieved with a semi-automated process[8], consisting of four steps. First, spikes are detected, typically by high-pass filtering and thresholding. Second, each spike waveform is summarized by a compact “feature vector”, typically by principal component analysis. Third, these vectors are divided into groups corresponding to putative neurons using cluster analysis. Finally, the results are manually curated, to adjust any errors made by automatic algorithms[13]. This last step is necessary because although fully automatic spike sorting would be a powerful tool, the output of current algorithms cannot be accepted without human verification. A similar situation arises in many fields of data-intensive science: in electron microscopic connectomics, for example, automatic methods can only be used under the supervision of human operators[14].

For tetrode data this semi-automatic process performs well, reaching error rates of 5% or lower, as assessed by ground truth data obtained with simultaneous intracellular recording[8]. However, spike sorting methods developed for tetrodes do not work for a newer generation of larger electrode arrays[15, 16]. This failure occurs for two reasons. First, the automated component can fail in high dimensions, for example due to the “curse of dimensionality” that affects cluster analysis in high-dimensional spaces[17]. Second and perhaps more critically, the process of manual curation -- while manageable with low-count probes -- cannot scale to the high-count case without software that guides the operator to only those decisions that cannot be made reliably by a computer. While many different methods for spike sorting have been proposed (e.g. refs. [18-24]), no method has yet solved these problems robustly enough to be widely adopted by the experimental community.

Here we describe a system for the spike sorting of high-channel count electrode data, implemented in a suite of freely available software. While the spike sorting problem has attracted considerable theoretical research, our goal was to produce a practical system that can be immediately used by working neurophysiologists. The ability to process large datasets (millions of spikes in hundreds of dimensions) in reasonable human and computer time was deemed essential; error rates comparable to those of commonly-used tetrode methods were deemed acceptable. We tested the software on data recorded from rat neocortex with 32-site shank electrodes, as well as data from other species and brain regions. While traditional methods performed extremely poorly on this data, the new algorithms gave close to theoretically optimal performance. The techniques and software have been developed in a community-led manner, through extensive feedback from a user base of over 320 scientists in 50 neurophysiology labs. The software is downloadable and documented at http://cortexlab.net/tools, and is supported by a highly active user-group mailing list, klustaviewas@groups.google.com.

Results

Our spike sorting pipeline involves three steps: (1) spike detection and feature extraction, (2) cluster analysis, and (3) manual curation. We describe these steps in order.

Spike Detection

The first step of the pipeline is spike detection and feature extraction, implemented by the program *SpikeDetekt*.

The primary difference between spike detection for high count silicon probes and for tetrodes is that temporally overlapping spikes are extremely common in the former. This phenomenon can be seen by examining of a segment of raw data recorded with high count probes (**Fig. 1**). The spikes seen in these data are diverse, with some detected on only one or two channels, and others spanning large numbers of channels, as expected of pyramidal cells whose apical dendrites are aligned parallel to the shank[25]. In these data, simultaneous firing of multiple neurons is common. However, simultaneously firing neurons are usually detected on distinct sets of channels.

To deal with the problem of temporally overlapping spikes, we therefore sought to detect spikes as local spatiotemporal events (**Fig. 2**). This step requires knowledge of the probe geometry, which is specified by the user in the form of an "adjacency graph" (**Fig. 2a**). We illustrate the spike detection process with reference to a small segment of data containing two temporally overlapping but spatially separated spikes (**Fig. 2b**).

The first stage of the algorithm is high-pass filtering the raw data to remove the slow local field potential signal (Butterworth in forward-backward mode; **Fig. 2c**). Next, spikes are detected using a double-threshold flood fill algorithm (**Fig. 2d,e**). Specifically, spikes are detected as spatiotemporally connected components, in which the filtered signal exceeds a "weak threshold" $\theta_w$for every point, and in which at least one point exceeds a "strong threshold" θ_s (optimal values for these parameters were found to be 4 and 2 times the standard deviation of the filtered signal, as described below). Two points are considered neighboring if they are on a single channel and separated by one time sample, or at a single timepoint on channels joined by the adjacency graph; this allows the algorithm to work with probes of any geometry, not just linear ones. The dual-threshold approach avoids spurious detection of small noise events, since isolated islands in which only the weak threshold is exceeded are not retained. Conversely, spikes will not be erroneously split due to noise, as areas joined by weak threshold crossings are merged.

After detection, spikes are temporally realigned to subsample resolution, to the center of mass of the spike's suprathreshold components, weighted by a power parameter p (see Methods). Visual inspection showed that spike times detected with this method correspond closely to those that would be assigned by a human operator (**Fig. 2e**).

The waveforms of each spike are summarized by two vectors. First, a "feature vector" is found by principal component analysis of the realigned waveforms on each channel (3 principal components were kept for the current analysis). All channels are used in computing the feature vector; thus our two example spikes have similar feature vectors, as their central times are similar (**Fig. 2f**). Second, a "mask vector" is computed from the peak spike amplitude on each detected channel, rescaled and clipped so channels outside the connected component have mask 0, and channels with amplitude above θ_s have mask 1. The mask vector allows temporally overlapping spikes to be clustered as separate cells. Indeed, although the feature vectors of our two example spikes were very similar, their mask vectors are completely different (**Fig. 2g**).

Performance Validation and parameter optimization

To quantify the performance and optimize the parameters of this algorithm requires "ground truth": knowledge of when the recorded neurons actually fired. We created a simulated ground truth dataset by repeatedly adding the spikes of a "donor cell" identified in one recording, to a second "acceptor" recording made with same probe; since the extracellular medium is a linear conductor[26], addition of spike waveforms serves as a sufficient model for overlapping spikes. To evaluate the performance of the system, we chose 10 donor cells with a variety of amplitudes and waveform distributions (**Fig. 3a**), using recordings from rat cortex with a 32-channel probe shank. To model the variability of waveforms produced by a single neuron due to phenomena such as bursting[27-29], we scaled each spike to a random amplitude in a range that varied by a factor of 2 (see Methods). We refer to the spikes added to the acceptor dataset as "hybrid spikes", and the result as a "hybrid dataset".

To evaluate spike detection performance, we used a heuristic criterion to identify which spikes detected by the algorithm corresponded to which hybrid spikes (see Methods). We measured performance as a function of three algorithm parameters (θ_w, θ_s, and p), using four performance statistics.

The first statistic was the fraction of hybrid spikes detected (**Fig. 3b**). This showed a strong dependence on the thresholds: values of θ_s above 4 times standard deviation (4 SD) resulted in poor detection, particularly for low-amplitude cells. The dependence of performance on θ_w was more complex: poor performance resulted not just from overly high values (>2.5 SD), but also overly low values (<2 SD). Examination of example errors (not shown) indicated that overly low values of θ_w led to inappropriate merging of temporally overlapping but spatially separated spikes, while overly high values led to artificial splitting of single spikes.

The second statistic was the total number of detection events (**Fig. 3c**). Because this includes noise events as well as true spikes of the hybrid and background cells, this number should be as small as possible provided the fraction correctly detected remains high. We found that this statistic most critically depended on the strong threshold, increasing markedly for values below 4SD.

The third statistic was timing jitter: the standard deviation of the difference between the detected and actual times of each hybrid spike (**Fig. 3d**). Jitter was in all cases less than one sample, and improved for larger values of θ_s and θ_w, indicating that spike times are best estimated from a minority of larger amplitude spikes. For all hybrid cells, jitter was worse for $p < 1$; for low amplitude cells it showed a further worsening for $p > 2$, reflecting noise introduced by overweighting of peak amplitude times.

The final statistic was mask accuracy (**Fig. 3e**), which measures how closely the detected mask vectors match those expected from the ground truth (see Methods). This showed strongest dependence on θ_w with a peak around 2 SD, and less pronounced dependence on θ_s peaking around 5 SD.

We conclude that close to optimal performance can be obtained using a strong threshold of 4 SD, a weak threshold of 2 SD and a power weight of 2. Furthermore, using these parameters yields around 95% correctly detected spikes, and spike timing jitter of 0.5 samples.

Cluster Analysis

The second step of our spike sorting pipeline is automatic cluster analysis, implemented in the program *KlustaKwik.*

For tetrode data, we previously found that cluster analysis using a mixture of Gaussians fit gave close to optimal performance[8]. This approach cannot be directly ported to high-channel-count data for two reasons. The first is the "curse of dimensionality": in high dimensions, noise measured on the large number of uninformative channels will swamp signals measured on the smaller number of informative channels. Second, because temporally overlapping spikes have similar feature vectors (Fig. 2F), further information such as the mask vectors must be used to distinguish these spikes.

To solve this problem, we designed a novel method, the "masked EM algorithm"[30]. This algorithm fits the data as a mixture of Gaussians, but with each feature vector replaced by a virtual ensemble in which features with masks near zero are replaced by a noise distribution (see Methods). Channels with low mask values are thus "disenfranchised", and do not contribute to cluster assignment; the probabilistic nature of this disenfranchisement means false clusters are not created when amplitudes cross an arbitrary threshold. The computational complexity of this algorithm is better than that of the traditional EM algorithm, scaling with the mean number of unmasked channels per spike (which does not increase for larger arrays), rather than the total number of channels.

To evaluate the performance of this algorithm, we used the hybrid datasets described above. For each dataset, we identified the cluster containing most hybrid spikes and computed the false discovery rate (fraction of spikes in the cluster that were not hybrids), and the true positive rate (fraction of all hybrid spikes assigned to the cluster). To estimate the theoretical optimum performance that could be expected, we used the Best Ellipsoid Error Rate (BEER) measure[8], which fits a quadratic decision boundary using ground truth data, and evaluates its performance with cross-validation, varying the parameters of the classifier to obtain an ROC curve showing optimal performance.

The masked EM algorithm's performance on an example hybrid dataset was close to the optimum estimated by the BEER measure but the classical EM algorithm's performance was poor, with error rates typically exceeding 50% (**Fig. 4a**). Across all hybrid datasets, we found no significant difference between the total error of the masked EM algorithm and theoretical optimal performance (p = 0.8, t-test), but a significant difference between the performance of the Classical and Masked EM algorithms (p = 0.005, t-test; **Fig. 4b**). To ensure the poor performance of the classical EM algorithm did not simply reflect incorrect parameter choice, we reran it for multiple values of the penalty parameter (which determines the number of clusters found), but this could not improve Classical EM performance. This analysis also demonstrated that the error rates of the masked EM algorithm were largely independent of the penalty parameter; using a value corresponding to the Bayesian Information Criterion seems a good option for penalty choice, as it led to a reasonably small number of clusters without compromising error rates (**Fig. 4c,d**).

We conclude that the performance of the Masked EM algorithm is close to optimal for this clustering problem, yielding false positive and false discovery rates both of the order 5%.

Manual Curation

The final step of the spike sorting pipeline is manual verification and adjustment of cluster assignments, which are implemented in the program *KlustaViewa.*

Although semi-automatic clustering provides more consistency and lower error rates than fully manual spike sorting[8], further manual corrections are typically required, such as merging of clusters split due to electrode drift, bursting, or other reasons[27-29]. These waveform shifts are hard to model and correct mathematically, but can usually be identified by inspection of waveforms, auto- and cross-correlograms, and cluster shapes. It is essential that this step be done with a minimum of human operator time, a particularly acute problem with the very large numbers of neurons recorded by large dense electrode arrays. Specifically, if N clusters are produced automatically, it is impractical for a human operator to inspect all order N^2 potential merges.

We addressed this problem using a semi-automatic "Wizard," that reduces the number of potential merges to order N. The Wizard works by presenting the operator with pairs of potentially mergeable clusters, ordered by a measure of pairwise cluster similarity. Because the Wizard is used iteratively, this measure must be computable in a fraction of a second, even for datasets containing millions of spikes. Thus, only metrics based on summary statistics of each cluster, rather than individual points, are suitable. We evaluated several candidate similarity measures. The Kullback-Leibler divergence between two Gaussian distributions was unsuitable as it overweighted differences in covariance matrix relative to differences in the mean. However, good performance was obtained using a single step of the masked EM algorithm to compute the similarity of the mean of one cluster to each of the others (**Fig. 5a**). To verify the accuracy of this measure, we simulated automatic clustering errors by splitting the ground truth clusters in the hybrid datasets into two subclusters containing high and low amplitude spikes. In all cases, the similarity measure correctly identified the other half of the artificially split cluster (**Fig. 5b**).

The manual stage can take several hours of operator time, and human error is lowest during the start of this period. The Wizard therefore iteratively presents the operator with decisions that can be made quickly, with the most important decisions presented first. The Wizard iterates through all clusters starting with the best currently unsorted spikes. The remaining clusters are ordered by similarity to the best unsorted cluster, and the decision of whether to merge, split, or delete each merge candidate is in turn made by the operator (**Fig. 5c,d**). Once satisfied that no more potential merges exist for the currently best unsorted cluster, the operator either accepts it as a well-isolated neuron, or rejects it as multiunit activity or noise, and the top-level iteration begins again.

Although the Wizard guides the operator through the decision process, the operator at all times has free access to all data required to make rapid decisions, provided by KlustaViewa's user-friendly and easily-navigable graphical user interface (**Figure 6**). Using this software, the time taken for manual curation scales linearly with the number of clusters, with a scaling factor that varies between operators and is generally about 1 minute per cluster, regardless of probe size. This software therefore allows for thorough manual curation of a dense-array recording in a few hours.

We assessed the performance of 8 human operators (5 experienced spike-sorters, 3 novices) using this system (**Fig. 7a**). First, we asked whether the operators would correctly fix a misclustering that was produced by the masked EM algorithm in simulation of electrode drift (described further below). All experienced operators, and all but one of the novices did this correctly. Second we asked how consistent the results of these operators would be on the same dataset (**Fig. 7b-d)**. We separately assessed consistency on spikes that all operators had identified be in "good" clusters, on spikes that at least one operator had identified to be in a good cluster, and on all remaining spikes. Similarity was assessed with the Fowlkes-Mallows index[31], which gives a score between 1 for complete agreement, and 0 for complete disagreement. For all operators apart from one of the novices, consistency was extremely high for those spikes identified as good by at least one operator (**Fig. 7e,f**); nevertheless the judgement of whether a cluster should be considered well-isolated varied between operators (**Fig. 7g**). We conclude that experienced operators are likely to make accurate and consistent judgements on cluster merging identification, but that the judgement on which clusters to term "good" is inconsistent; we therefore recommend that quantitative metrics[32, 33] be used to determine isolation quality.

Additional tests

We used the system described above to answer several additional questions regarding the process of spike sorting, and the design of electrodes.

First, we used our simulated ground truth dataset to ask how spike sorting performance would change for different electrode designs. We considered two cases. In the first (“site thinning”; **Supplementary Figs. 1 and 2**), the electrode was made less dense by omitting alternating channels on both sides. We evaluated the performance of spike detection and clustering using the same hybrid spikes described earlier, but only on this subset of channels (the adjacency graph was modified to join any two channels that both connected to a missing channel). Spike detection was strongly impacted, with correct detection rates dropping to an average of below 80% (Supplementary Fig. 1). Clustering performance was also impacted, as assessed both by the theoretical optimum, and by the Masked EM algorithm. While some cells saw little decrease in clustering performance (typically those found on multiple channels), others were strongly impacted by both metrics (Supplementary Figure 2). We conclude that performance in rat cortex decreases substantially for site spacing larger the 40μm same-side site spacing of these test probes.

Next, we simulated removing one side of the probe (**Supplementary Figs. 3 and 4**). Of the 10 hybrid cells analyzed, 6 were only detectable on one of the probe’s two sides, while the other 4 could be detected on both sides to a greater or lesser extent (**Supplementary Table 1)**. The effect of side removal was different to that of site thinning. The performance of each unit’s “preferred side” was comparable to that of the full probe. However, for the 4 units that were visible on both sides of the probe, performance on the “unpreferred side” was substantially worse than performance on the full probe, as assessed both by theoretical optimum performance and the actual results of the masked EM algorithm. We conclude that in staggered probes, the probe’s two sides function largely independently: the primary benefit of two-sided shanks is not to increase the isolation quality of a cell already well isolated on one side of the probe, but to record from a larger number of units.

Next, we asked whether similar performance to that seen in neocortex could also be obtained in other brain structures and species. We first generated an additional 5 hybrid cells using 10-site recordings from rat CA1 (**Supplementary Figs. 5 and 6**). Good performance was again obtained; furthermore, the spike detection parameters found to be optimal in cortical data were also optimal in CA1 data. We then ran the same code on high-count data collected from a wider range of preparations: V1 of awake mouse and awake macaque monkey (**Supplementary Figs. 7-9**), and LGN thalamus of anesthetized marmoset (**Supplementary Fig. 10**). Additional confidence in the method was provided both by further analyses of hybrid data (**Supplementary Fig. 11**) and by the observation of sharp orientation-tuned responses (Supplementary Fig. 7c-l), including amongst cells of apparently similar waveforms that were nevertheless separated by the spike sorting procedure (Supplementary figure 7m).

Next, we asked how well the system would deal with non-stationarity in spike amplitudes. Such non-stationarity can occur both because of electrode drift, and also because of activity-related changes in spike amplitude such as after bursts or prolonged periods of firing[27]. Examination of data from acute recordings (where electrode drift is often stronger than with chronic probes), showed that the algorithm often tracked drift successfully, but in other cases split the spikes of a single drifty cell into multiple clusters requiring manual merging (**Supplementary Fig. 12**).

To simulate nonstationarity, we constructed 6 hybrid datasets in which spike amplitude drifted throughout the recording as a geometric random walk (**Supplementary Fig. 13**). Spike detection was hardly impacted by this nonstationarity (**Supplementary Fig. 14**). For clustering, only one of the 6 drifty hybrid datasets required manual curation, and once this was performed, accuracy of the masked EM algorithm was comparable to the theoretical optimum (**Supplementary Fig. 15**). A different type of nonstationarity, in which the hybrid cell simply stopped firing halfway through the recording, also had no effects on performance (p=0.75; two-sample t-test on total errors; **Supplementary Fig. 16**). As an important task is often to track cells between recordings made over multiple days – i.e. where drift occurs in non-recorded periods – we also asked whether the Wizard's similarity metric might be used for this purpose. Although ground truth data was not available, a conservative criterion gave encouraging results, as indicated by the similarities of the autocorrelograms of the units associated to each other (**Supplementary Fig. 17**).

A strategy sometimes used to deal with nonstationarity is to include time as an additional feature in the cluster analysis algorithm, in principle allowing the algorithm to track slow changes in amplitude. To our surprise, we found that this actually worsened clustering performance, which could not always be overcome by manual curation (Supplementary Fig. 15). We conclude that nonstationarity (at least of the type modelled here) does not present a serious problem to automatic sorting performance if time is not added as an additional feature, and if manual curation is performed when required.

Discussion

We have produced a software suite for spike sorting of data from large, dense electrode arrays. Analysis of simulated ground-truth data indicated that error rates of this approach are frequently of the order 5%.

A critical step in this system, and all others currently in wide use for *in vivo* data, is manual curation. Extracellular array recordings are subject to numerous sources of error including electrode drift, overlapping spikes, and the fact that neuronal spike waveforms are not constant, but change according to firing patterns including but not limited to bursting[27-29]. While most working neurophysiologists have a good understanding of these potential artifacts, formalizing this knowledge into a reliable mathematical model has proved challenging. Because spike sorting errors could lead to erroneous scientific conclusions[29], it remains essential that a scientist is able to inspect the results produced by an automatic algorithm, then correct or discard its results. We found that experienced operators tended to make similar judgements during the manual curation process, but that their judgements of which units were well-isolated were subjective. Fortunately, quantitative criteria exist for assessing the quality of unit isolation[32, 33], and we therefore recommend that these be used, rather than human judgements, when deciding which cells to include in further scientific analysis.

The current performance of the system is sufficient for practical analysis of data produced by current, commercially-available silicon probes. Nevertheless, there remain areas for further improvement. The first of these concerns execution time. KlustaKwik is several orders of magnitude faster than standard mixture of Gaussians fitting; nevertheless, when running on large datasets, it can take hours or even days to complete on a standard single-core machine. Hardware acceleration such as GPUs[34] or cloud computing[35] may speed up this analysis stage, as may alternative cluster analysis algorithms that exclude the most computationally expensive step of covariance matrix estimation (e.g. Refs. [36, 37]). Faster versions of the code presented here, currently under development, are available at https://github.com/kwikteam/klustakwik2 and https://github.com/kwikteam/phy. A second opportunity for improvement regards the detection of spatiotemporally overlapping spikes. While the current algorithm can detect the majority of temporally overlapping spikes, which occur on distinct sets of channels, it cannot resolve spikes that overlap in both space and time. Template-matching algorithms have solved this problem in the case of *in vitro* retinal array data[38, 39], but these data are much less noisy than *in vivo* brain recordings. While recent research suggests that certain forms of template matching may succeed at least for tetrode data *in vivo*[18, 21], such methods are not at present widely applied to *in vivo* recordings, and numerous challenges need to be overcome, most critically regarding the manual curation step. The platform we have described here constitutes both a practical solution to today's spike sorting challenges, and also a framework from which to develop solutions for future generations of electrodes containing thousands of channels.

Contributions

C.R., D.F.M.G., S.N.K. and J.S. wrote SpikeDetekt. K.D.H, S.N.K., and D.F.M.G. designed the Masked EM algorithm and wrote KlustaKwik. C.R. and M.L.D.H. wrote KlustaViewa. C.R wrote Galry. S.N.K. analyzed algorithm performance. Rat data were recorded by A.G., M.B. and G.B.. Mouse data were recorded by A.S and M.C.. Marmoset data were recorded by S.S. The procedure for non-chronic laminar recordings with Neuronexus Vector probes in awake, behaving macaques was developed by G.H.D., A.S.E., A.S.T., who also collected the data. K.D.H., S.N.K., and C.R. wrote the manuscript with inputs from all authors.

Acknowledgements

We thank the 200+ members of the klustaviewas@groups.google.com mailing list for their feedback, bug reports, and suggestions. This work was supported by EPSRC (K015141, I005102, KDH) and the Wellcome Trust (95668, 100154, KDH, MC).

References

1. Buzsaki, G. Large-scale recording of neuronal ensembles. *Nat Neurosci* **7**, 446-451 (2004).
2. Wise, K.D. & Najafi, K. Microfabrication techniques for integrated sensors and microsystems. *Science* **254**, 1335-1342 (1991).
3. Csicsvari, J. *et al.* Massively parallel recording of unit and local field potentials with silicon-based electrodes. *Journal of Neurophysiology* **90**, 1314-1323 (2003).
4. McNaughton, B.L., O'Keefe, J. & Barnes, C.A. The stereotrode: a new technique for simultaneous isolation of several single units in the central nervous system from multiple unit records. *J Neurosci Methods* **8**, 391-397 (1983).
5. Gray, C.M., Maldonado, P.E., Wilson, M. & McNaughton, B. Tetrodes markedly improve the reliability and yield of multiple single-unit isolation from multi-unit recordings in cat striate cortex. *J Neurosci Methods* **63**, 43-54 (1995).
6. Wilson, M.A. & McNaughton, B.L. Dynamics of the hippocampal ensemble code for space. *Science* **261**, 1055-1058 (1993).
7. Recce, M. & O'Keefe, J. The tetrode: a new technique for multi-unit extracellular recording. *Soc Neurosci Abstr* **15**, 1250 (1989).

8. Harris, K.D., Henze, D.A., Csicsvari, J., Hirase, H. & Buzsaki, G. Accuracy of tetrode spike separation as determined by simultaneous intracellular and extracellular measurements. *J.Neurophysiol.* **84**, 401-414 (2000).

9. Henze, D.A. *et al.* Intracellular features predicted by extracellular recordings in the hippocampus In vivo. *J.Neurophysiol.* **84**, 390-400 (2000).

10. Gold, C., Henze, D.A., Koch, C. & Buzsaki, G. On the origin of the extracellular action potential waveform: A modeling study. *J Neurophysiol* **95**, 3113-3128 (2006).

11. Einevoll, G.T., Franke, F., Hagen, E., Pouzat, C. & Harris, K.D. Towards reliable spike-train recordings from thousands of neurons with multielectrodes. *Curr Opin Neurobiol* **22**, 11-17 (2012).

12. Lewicki, M.S. A review of methods for spike sorting: the detection and classification of neural action potentials. *Network* **9**, R53--R78 (1998).

13. Hazan, L., Zugaro, M. & Buzsaki, G. Klusters, NeuroScope, NDManager: a free software suite for neurophysiological data processing and visualization. *J Neurosci Methods* **155**, 207-216 (2006).

14. Briggman, K.L., Helmstaedter, M. & Denk, W. Wiring specificity in the direction-selectivity circuit of the retina. *Nature* **471**, 183-188 (2011).

15. Berenyi, A. *et al.* Large-scale, high-density (up to 512 channels) recording of local circuits in behaving animals. *J Neurophysiol* **111**, 1132-1149 (2014).

16. Du, J., Blanche, T.J., Harrison, R.R., Lester, H.A. & Masmanidis, S.C. Multiplexed, high density electrophysiology with nanofabricated neural probes. *PLoS One* **6**, e26204 (2011).

17. Bouveyron, C. & Brunet-Saumard, C. Model-based clustering of high-dimensional data: A review. *Comput Stat Data An* **71**, 52-78 (2014).

18. Ekanadham, C., Tranchina, D. & Simoncelli, E.P. A unified framework and method for automatic neural spike identification. *J Neurosci Methods* **222**, 47-55 (2014).

19. Carlson, D.E. *et al.* Multichannel electrophysiological spike sorting via joint dictionary learning and mixture modeling. *IEEE Trans Biomed Eng* **61**, 41-54 (2014).

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

21. Franke, F., Natora, M., Boucsein, C., Munk, M.H. & Obermayer, K. An online spike detection and spike classification algorithm capable of instantaneous resolution of overlapping spikes. *J Comput Neurosci* **29**, 127-148 (2010).

22. Quiroga, R.Q., Nadasdy, Z. & Ben-Shaul, Y. Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering. *Neural Comput* **16**, 1661-1687 (2004).

23. Swindale, N.V. & Spacek, M.A. Spike sorting for polytrodes: a divide and conquer approach. *Front Syst Neurosci* **8**, 6 (2014).

24. Swindale, N.V. & Spacek, M.A. Spike detection methods for polytrodes and high density microelectrode arrays. *J Comput Neurosci* (2014).

25. Buzsaki, G. & Kandel, A. Somadendritic backpropagation of action potentials in cortical pyramidal cells of the awake rat. *J Neurophysiol* **79**, 1587-1591 (1998).

26. Logothetis, N.K., Kayser, C. & Oeltermann, A. In vivo measurement of cortical impedance spectrum in monkeys: implications for signal propagation. *Neuron* **55**, 809-823 (2007).

27. Harris, K.D., Hirase, H., Leinekugel, X., Henze, D.A. & Buzsaki, G. Temporal interaction between single spikes and complex spike bursts in hippocampal pyramidal cells. *Neuron* **32**, 141-149 (2001).

28. Quirk, M.C., Blum, K.I. & Wilson, M.A. Experience-Dependent Changes in Extracellular Spike Amplitude May Reflect Regulation of Dendritic Action Potential Back-Propagation in Rat Hippocampal Pyramidal Cells. *J.Neurosci.* **21**, 240-248 (2001).

29. Quirk, M.C. & Wilson, M.A. Interaction between spike waveform classification and temporal sequence detection. *J.Neurosci.Methods* **94**, 41-52 (1999).

30. Kadir, S.N., Goodman, D.F. & Harris, K.D. High-Dimensional Cluster Analysis with the Masked EM Algorithm. *Neural Comput*, 1-16 (2014).

31. Fowlkes, E.B. & Mallows, C.L. A Method for Comparing 2 Hierarchical Clusterings. *J Am Stat Assoc* **78**, 553-569 (1983).

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

33. Hill, D.N., Mehta, S.B. & Kleinfeld, D. Quality metrics to accompany spike sorting of extracellular signals. *J Neurosci* **31**, 8699-8705 (2011).

34. Owens, J.D. *et al.* GPU computing. *Proceedings of the Ieee* **96**, 879-899 (2008).

35. Freeman, J. *et al.* Mapping brain activity at scale with cluster computing. *Nature methods* **11**, 941-950 (2014).

36. Comaniciu, D. & Meer, P. Mean shift: A robust approach toward feature space analysis. *Ieee T Pattern Anal* **24**, 603-619 (2002).

37. Rodriguez, A. & Laio, A. Machine learning. Clustering by fast search and find of density peaks. *Science* **344**, 1492-1496 (2014).

38. Marre, O. *et al.* Mapping a complete neural population in the retina. *J Neurosci* **32**, 14859-14873 (2012).

39. Pillow, J.W., Shlens, J., Chichilnisky, E.J. & Simoncelli, E.P. A model-based spike sorting algorithm for removing correlation artifacts in multi-neuron recordings. *PLoS One* **8**, e62123 (2013).

Figure Legends

Figure 1: High-count silicon probe recording.

(**a**)**,** Layout of the 32-site electrode array used to collect test data. (**b**)**,** Short segment of data recorded in rat neocortex with this array. Color of traces indicates recording from the corresponding colored site in (a). Black rectangles highlight action potential waveforms; note the frequent occurrence of temporally overlapping spikes on separate recording channels.

Figure 2: Local spike detection algorithm.

(**a**)**,** Adjacency graph for the 32-channel probe. (**b**)**,** Segment of raw data showing two simultaneous action potentials on spatially separated channels (scale bars indicate 0.5mV / 10 samples). (**c**)**,** High-pass filtered data shown in pseudocolor format (units of standard deviation). Vertical lines on the colorbar indicate strong and weak thresholds, θ_s and θ_w (respectively 4 and 2 times standard deviation). (**d**)**,** Gray-scale representation showing samples which cross the weak threshold (gray), and the strong threshold (white). (**e**)**,** Results of two-threshold flood fill algorithm, showing connected components corresponding to the two spikes in orange and brown. Note that isolated weak threshold crossings resulting from noise are removed. White lines indicate alignment times of the two spikes. (**f**), Pseudocolor representation of feature vectors for the two detected spikes (top and bottom). Each set of three dots represents three principal components computed for the corresponding channel (arbitrary units). Note the similarity of the feature vectors for these two simultaneous spikes (top and bottom). (**g**)**,** Mask vectors obtained for the two detected spikes (top and bottom; 0 represents completely masked, 1 completely unmasked). Unlike the feature vectors, the mask vectors for the two spikes differ. Each set of three dots represents the three identical components of the mask vector for the corresponding channel.

Figure 3: Evaluation of spike detection performance.

(**a**)**,** Waveforms of the 10 donor cells used to test spike detection performance, in order of increasing peak amplitude (left to right). (**b**)**,** Fraction of correctly detected spikes as a function of strong threshold θ_s (left), weak threshold θ_w (center), and power parameter p (right). Colored lines indicate performance for the correspondingly colored donor cell waveform shown in A; black line indicates mean over all donor cells. (**c-e**)**,** Dependence of the total number of detected events, timing jitter, and mask accuracy on the same three parameters.

Figure 4: Evaluation of automatic clustering performance.

(**a**)**,** Receiver-Operating Characteristic (ROC) Curve showing the performance of the Masked EM algorithm (blue) and Classical EM algorithm (red) on one of the 10 hybrid datasets; each dot represents performance for a different value of the penalty parameter. The cyan curve shows a theoretical upper bound for performance, the best ellipsoid error rate (BEER) measure obtained by cross-validated supervised learning. (**b**)**,** Mean and standard error of the total error (false discovery plus false positive) over all 10 hybrid datasets for theoretical optimum (BEER measure), Masked EM and Classical EM algorithms. For each dataset and measure, the parameter setting leading to best performance was used. (**c**)**,** Effect of varying the penalty parameter (as a multiple of the AIC penalty) on the total error for both algorithms. The dotted line indicates the parameter value corresponding to BIC. Note that the Masked EM algorithm performed well for all penalty values. (**d**)**,** The number of clusters returned by the Masked EM algorithm as a function of the penalty parameter.

Figure 5: The "Wizard" for computer-guided manual correction.

(**a**)**,** Illustration of the measure used to quantify cluster similarity. p_{ij} represents the posterior probability with which the EM algorithm would assign of the mean of cluster i to cluster j. (**b**)**,** To test this measure, the clusters corresponding to hybrid spikes were artificially cut into halves of high and low amplitude. In each case, the similarity measure identified the second half as the closest merge candidate. (**c**)**,** The Wizard identifies the best unsorted cluster as the one with highest quality (top), and finds the closest match to it using the similarity matrix. (**d**)**,** The Wizard algorithm. The best unsorted cluster and closest match are identified. The operator can choose merge the closest match into the best unsorted, ignore the closest match, or delete it by marking it as multiunit activity or noise; the wizard then presents the next closest match to the operator (blue arrows). After a sufficient number of matches have been presented, the operator can decide that no further potential matches could have come from the same neuron, and either accept the best unsorted cluster as a well-isolated neuron, or delete it as multiunit activity or noise. The wizard then finds the next best unsorted cluster to present to the operator (orange arrows).

Figure 6: Screenshot of the KlustaViewa graphical user interface.

In order to make the decisions presented by the Wizard, the operator has access to information including waveforms (center panel; gray waveforms correspond to masked channels), principal component features (top right), auto- and cross-correlograms (bottom right), and an automatically computed similarity metric for each pair of clusters (inset). To enable rapid navigation, all views are integrated; for example, clicking on a particular channel in the Waveform View will update other views to show the selected channels or clusters.

Figure 7: Consistency of manual curation across operators.

(**a**)**,** Performance of 8 human operators (5 experts, 3 novices) on a "drifty" hybrid cell requiring manual curation (see supplementary figure 13b). A tick indicates correct merging of the split hybrid cell, a cross indicates this merge was not performed. (**b-d**), consistency of assignments of multiple operators over all cells in this dataset. Each submatrix shows the conditional probability of the first operator's cluster assignments given the assignments of the second operator (color scale at bottom of (d)). (**b**)**,** consistency of cluster assignments for spikes marked as well-isolated by all operators; (**c**)**,** consistency of cluster assignments for spikes marked as well-isolated by at least one operator; (**d**)**,** consistency of whether spikes were marked as well-isolated by different operators. (**e-g**)**:** Operator consistency for the analyses of (b-d) was quantified using the Fowlkes-Mallows index, for which 1 represents complete agreement and 0 complete disagreement. Note that while cluster assignments were highly consistent between all expert operators, the operators were often inconsistent in their judgements of which units were well-isolated.

Methods

A supplementary Methods checklist is available.

Test data

To test the algorithm, we created simulated ground truth data using a method termed “hybrid datasets”. The primary raw data used to construct this ground truth (shown in the main text figures) consisted of two separate recordings from somatosensory cortex (−3.8 mm from bregma, 3 mm lateral to midline, 1mm depth) of sleeping adult rats, using silicon probes with 32 non-activated platinum-plated recording sites of size 10x16 µm arranged in a staggered shank configuration (vertical spacing 20 µm between adjacent sites on opposite sides of the shank, 40 µm between adjacent sites on the same side), mounted on a home-made microdrive. Ground and reference electrodes were stainless steel screws over the cerebellum. Data was continuously recorded wideband (1Hz-Nyquist), at a sampling rate of 20 kHz. During the recording session, the signals were amplified (1000x), bandpass filtered (1 to 5000 Hz), and acquired continuously at 20 kHz on a 128-channel DataMax system (16-bit resolution; RC Electronics). All protocols were approved by the Institutional Animal Care and Use Committee of Rutgers University.

To perform additional tests (supplementary figures 5-12), we analyzed data collected in additional brain structures and species. Data was collected from the septal third of hippocampal CA1 region in male rats using 10-site silicon probes using the same methods as above. All protocols were approved by the Institutional Animal Care and Use Committee of Rutgers University. To obtain recordings in mouse V1, mice were implanted with a custom-built head post and recording chamber (4 mm inner diameter) under isoflurane anesthesia. After several days acclimatization to head-fixation, animals were anesthetized under isoflurane and a ~1 mm craniotomy was performed over area V1 one day prior to the first recording (see Refs. [40, 41] for further details). Data were recorded with an acutely-inserted 32-site Neuronexus Edge probe (20 micron spacing). Experiments were conducted according to the UK Animals (Scientific Procedures) Act, 1986 under personal and project licenses issued by the Home Office following ethical review. Non-chronic recordings were obtained from cortical area V1 of two awake, behaving, adult male rhesus monkeys (*macaca mulatta*) using Neuronexus Poly2 and custom-designed Edge (60 micron spacing) Vector probes. Animals were first implanted with scleral search coils and fit with a custom-built titanium head post and recording chamber (see Refs. [42, 43] for further details). Subsequently, a 2-3mm diameter trephination was performed through which daily penetrations would be made. Data were acquired as broad-band signals (0.5–16 kHz, sampled at 32 kHz), digitized at 24-bits using PXI-4498 cards (National Instruments, Austin, TX). All procedures were conducted in accordance with the ethical guidelines of the National Institutes of Health and were approved by the Baylor College of Medicine IACUC. To obtain recordings from dorsal lateral geniculate nucleus (LGN) of sufentanil-anaesthetised adult male marmoset monkey (Callithrix jacchus), a craniotomy was made over the right LGN and a Neuronexus A16x2 probe (500µm probe separation, 50µm spacing between contact points on each shank) was lowered into LGN and allowed to settle for at least 30 minutes before recording. Data were band-pass filtered (0.3–5kHz, sampled at 24kHz), and digitized by a Tucker-Davis

Technologies RZ2 real time processor (see Ref. [44] for further details). All procedures were approved by the University of Sydney Animal Ethics Committee and conform to Australian National Health and Medical Research Council (NHMRC) policies on the use of animals in neuroscience research.

Hybrid datasets

To create the hybrid datasets, we first completed a full spike sorting of each dataset, including manual verification. Five clusters were chosen from each dataset, corresponding to neurons spanning the range of amplitudes and channel distributions observed in the data (Figure 3A). The mean unfiltered waveform of each neuron was computed, its mean was subtracted, and its value at each end was set to exactly zero by tapering with a Hamming function. These “donor waveforms” were added at prescribed times to the raw unfiltered data of the other “acceptor” recording. To simulate amplitude variability, we linearly scaled each added waveform by a random factor chosen from the range $[\sqrt{2}/2, \sqrt{2}\,]$ causing amplitudes to vary by a factor of two, which suffices to capture the variability typical of bursting neurons [27]. The interspike intervals typical of bursting neurons were not simulated as this does not affect the spike detection or clustering process; instead, hybrid spikes were added regularly at rates in the range 2-4 spikes per second. To ensure that the simulated data tested the ability of our software to realign spikes to subsample resolution, each added spike was shifted by a random subsample offset using cubic spline interpolation. For simulations of drifty cells, amplitude was as geometric random walk (i.e. the exponential of a Brownian random walk), which was then normalized so that the mean amplitude remained the same as its non-drifty counterpart.

File format

To implement the software, we designed an HDF5-based file format to store raw data, intermediate analysis results (such as extracted spike waveforms and feature vectors), as well as final data such as spike times and cluster assignments [45]. The format makes use of HDF5 links to allow a single, small file (the “.kwik file”) containing all data required for scientific analysis (e.g. spike times, cluster assignments, unit isolation quality measures). Bulky raw data and intermediate processing steps such as feature vectors are stored in separate files (the “.kwd” and “.kwx” files). This “detachable” format is designed for data sharing applications, allowing users to download as much data as required for their needs. A full specification of the format can be found at https://phycortexlab.net/format.

SpikeDetekt

Spike detection was implemented by SpikeDetekt, a custom program written in Python 2.7 using the packages NumPy, SciPy, and PyTables.

The first step of the program is to filter the raw voltage trace data to remove the low-frequency local field potential (LFP). This is achieved with a 3rd order Butterworth filter used in the forward-backward mode to ensure zero phase distortion. Filter parameters can be specified by the user; for the analyses described here we used a band-pass filter of 500 Hz to 0.95*Nyquist.

The second step is threshold determination. Spike detection thresholds are specified as multiples of the standard deviation of the filtered signal; at the option of the user, a single threshold is used for all channels in order to avoid emphasizing noise from low-amplitude channels. To boost execution speed while minimizing the chance of biased estimates, the standard deviation is estimated from five data chunks of length 1 second each, picked randomly from throughout the recording. The standard deviation is computed with a robust estimator, $\mathrm{median}(|V|)/.6745$, to avoid contamination by spike waveforms.

The next step is spike detection. The spike detection code operates on consecutive chunks of data (1s length) for memory efficiency. Spatiotemporally connected regions of weak threshold crossing are detected using a non-recursive flood fill algorithm, with spatial continuity defined using a user-specified adjacency graph. Only connected components for which at least one point exceeds the strong threshold are kept for further analysis.

Spike alignment is computed based on a scaled and clipped transformation of the filtered voltage $V(t,c)$:

$$\psi(t,c) = \min\left(\frac{-V(t,c)-\theta_w}{\theta_s-\theta_w}, 1\right)$$

Note that $\psi(t,c)$ can never be negative within a spike, as the floodfill algorithm only finds points for which $-V(t,c) > \theta_w$. The center time for each spike S is computed as

$$\overline{t_S} \; = \frac{\sum_{(t,c)\in S} t\, \psi(t,c)^p}{\sum_{(t,c)\in S} \psi(t,c)^p}$$

where $(t,c) \in S$ denotes the set of times and channels, for all points assigned to this spike by the floodfill algorithm. If $p = 1$, this formula measures the spike's center of mass; if $\mathrm{p} = \infty$, it measures the time of the spike peak.

Spikes were realigned on $\overline{t_S}$ to subsample resolution using cubic spline interpolation (note that the center time will, in general, not be an integer number of samples). Feature vectors are computed for each channel separately by principal component analysis; the number of features per channel is a user settable parameter, with default value 3. Finally, mask vectors are computed for each spike S as zero for channels not appearing in the connected component, and as the maximum scaled waveform for all channels inside the component:

$$m_{c,S} = \max_{t:(t,c)\in S} \psi(t,c)$$

To evaluate the performance of SpikeDetekt, required identifying which detected spikes correspond to ground truth spikes. This was done with a dual criterion: the difference between the detected time and ground truth needed to be less than 2 samples, and the detected mask vector $\mathbf{m_s}$ needed to have a similarity to the ground truth mask vector $\mathbf{m_G}$ of at least 0.8, defined by the mask similarity measure

$$\frac{\mathbf{m_S}\cdot\mathbf{m_G}}{|\mathbf{m_S}||\mathbf{m_G}|}$$

Note that mask similarity cannot exceed 1, by the Cauchy-Schwartz inequality. The validity of this criterion was verified by showing that detected spike timing jitter rapidly increased for similarity threshold for values less than 0.8, but was insensitive to threshold value above 0.8. Once the detected spikes corresponding to ground truth had been identified, the four measures in figure 3 were computed. This analysis used the Python library Joblib to prevent unnecessary recomputation.

KlustaKwik

Automatic clustering was performed by KlustaKwik, a custom program written in C++. The first version of this program was designed for tetrode data, implemented a hard EM algorithm for maximum-likelihood fitting of a mixture of arbitrary-covariance Gaussians, and was released in 2000 but not specifically described in a published manuscript. Here, we have implemented several modifications of this software to enable automatic sorting of high-count probe data. The program now implements a novel "masked EM algorithm" [30] designed for high-dimensional classification, as well as other features such as cache optimization resulting in a speed increase of over 10,000%.

The masked EM algorithm takes as input both feature vectors and mask vectors. It works by fitting a mixture of Gaussians to a virtual dataset in which each feature vector is replaced by a probability distribution:

$$\tilde{x}_{n,S} \sim \begin{cases} x_{n,S} & \text{prob } \; m_{n,S} \\ N(\nu_n, \sigma_S^2) & \text{prob } \; 1 - m_{n,S} \end{cases}$$

Here, $x_{n,S}$ represents the nth component of the feature vector for spike S; $m_{n,S}$ represents the nth component of the mask vector for spike S; and $N(\nu_n, \sigma_S^2)$ denotes a univariate Gaussian distribution with mean and variance equal to those of the subthreshold noise distribution of the nth feature.

The masked EM algorithm consists of alternation of an "E step" in which each spike is assigned to the cluster for which it has highest posterior probability, and an "M step" in which the means and covariances of each cluster are estimated. We have derived analytic formulas for the expectation of the cluster assignment probability used in the E-step, and the cluster mean and variance used in the M step, over the virtual probability distribution $\tilde{x}_{n,i}$ [30]. Thus, explicit sampling from the virtual distribution does not need to be performed; furthermore, these expectations can be computed much faster than those of the full EM algorithm as they scale with the square of the number of unmasked features, rather than the square of the total number of features.

KlustaKwik automatically determines the number of clusters that best fit the data, determined using a penalty function that encodes a preference for fits with smaller numbers of clusters. We have found a modification of the Bayesian Information Criterion to deal with masked data works well in practice [30]. Because the algorithm allows for dynamic splitting and merging of clusters during the fitting process, a search for the optimal number of clusters can be achieved in a single run of the algorithm. We have found that starting the algorithm from an initial clustering determined heuristically from the mask vectors avoids the problem of local maxima, and allows good results to be obtained from a single run.

KlustaViewa

Manual correction of automatic clustering is performed with KlustaViewa, a custom program written in Python 2.7. The manual stage requires interactive visualization of very large numbers of data points, for which existing libraries such as matplotlib were not suitable. We therefore designed a new Python library for rapid interactive data visualization named Galry [46]. Galry leverages the computational power of modern graphics processing units [34] through the OpenGL graphics library [47]. High performance is achieved by porting most visualization computations to the GPU using custom shaders, and by minimizing the number of OpenGL API calls through batch rendering techniques.

To ensure rapid adoption by the experimental community, we designed KlustaViewa's user interface by the integrating novel features necessary for high-count probes into a user interface as similar as possible to existing manual spike sorting environments such as Klusters [13]. In addition to data views familiar from previous spike sorting systems (such as waveform, auto- and cross-correlograms, and similarity matrix), we implemented several new features. The most important of these is the Wizard (described in the main text), that automatically leads the user through the manual verification and merging process, while always allowing the user free access to all of the views familiar from standard spike sorting systems. In addition, a number of enhancements were designed specifically to make the sorting of high-count probe data tractable. These include features to allow display of masking information; rapid and automatic display of the channels relevant to selected clusters; transient color brushing [48]; and automatic downsampling to ensure low latency display when dealing with very large datasets.

The Wizard is based on a metric of similarity for each pair of clusters. This was computed by running a single step from the EM algorithm to compute the posterior probability for assigning the mean of cluster i to cluster j:

$$p_{ij} = \frac{w_j N(\mu_i|\mu_j; C_j)}{\sum_k w_k \ \ (\mu_i|\mu_k; C_k)}$$

Here w_j represents the weight of cluster j (i.e. the fraction of points already assigned to this cluster); μ_j and C_j represent its mean and covariance as computed by the M-step of the masked EM algorithm. The quality of each cluster j was defined as the diagonal element p_{jj}, i.e. the posterior probability for classifying cluster j's mean as coming from cluster j itself. A high value for p_{jj} therefore indicates that cluster j has no close neighbors.

The difference between two clusterings C, C', consisting of K and K' clusters, respectively, and confusion matrix entries, $n_{kk'}$ where measured using the Fowlkes-Mallows[31] index, $\sqrt{W_1 W_2}$, where:

$$W_1(C, C') = \frac{\sum_{k,k'} n_{kk'}(n_{kk'} - 1)/2}{\sum_k n_k(n_k - 1)/2}, \qquad W_2(C, C') = \frac{\sum_{k,k'} n_{kk'}(n_{kk'} - 1)/2}{\sum_{k'} n'_{k'}(n'_{k'} - 1)/2}$$

$n_{k'} = \sum_k n_{kk'}$, $n'_{k'} = \sum_{k'} n_{kk'}$, , $k = 1, \dots, K, k' = 1, \dots, K'$. W_1 is the probability that a pair of points which are in the same cluster under the clustering C is also in the same cluster in C'. W_2 is the same with the two clusterings interchanged. The Fowlkes-Mallows index symmetrizes these two asymmetric quantities by taking their geometric mean.

40. Saleem, A.B., Ayaz, A., Jeffery, K.J., Harris, K.D. & Carandini, M. Integration of visual motion and locomotion in mouse visual cortex. *Nat Neurosci* **16**, 1864-1869 (2013).

41. Ayaz, A., Saleem, A.B., Scholvinck, M.L. & Carandini, M. Locomotion controls spatial integration in mouse visual cortex. *Curr Biol* **23**, 890-894 (2013).

42. Ecker, A.S. *et al.* State dependence of noise correlations in macaque primary visual cortex. *Neuron* **82**, 235-248 (2014).

43. Ecker, A.S. *et al.* Decorrelated neuronal firing in cortical microcircuits. *Science* **327**, 584-587 (2010).

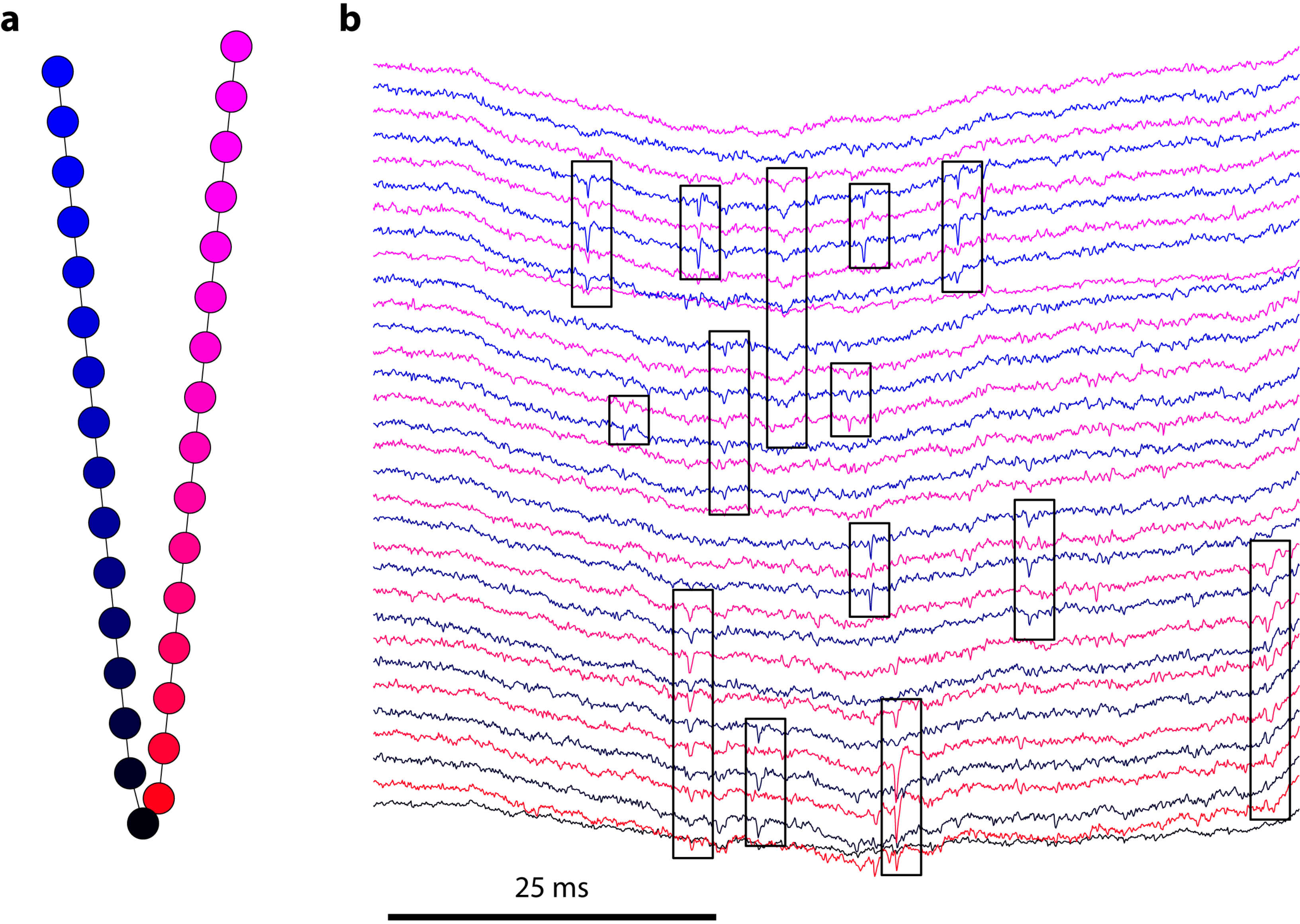
44. Zeater, N., Cheong, S.K., Solomon, S.G., Dreher, B., Martin, P.R. Binocular responses in the primate lateral geniculate nucleus. *Curr Biol* **25**, 3190-3195 (2015).

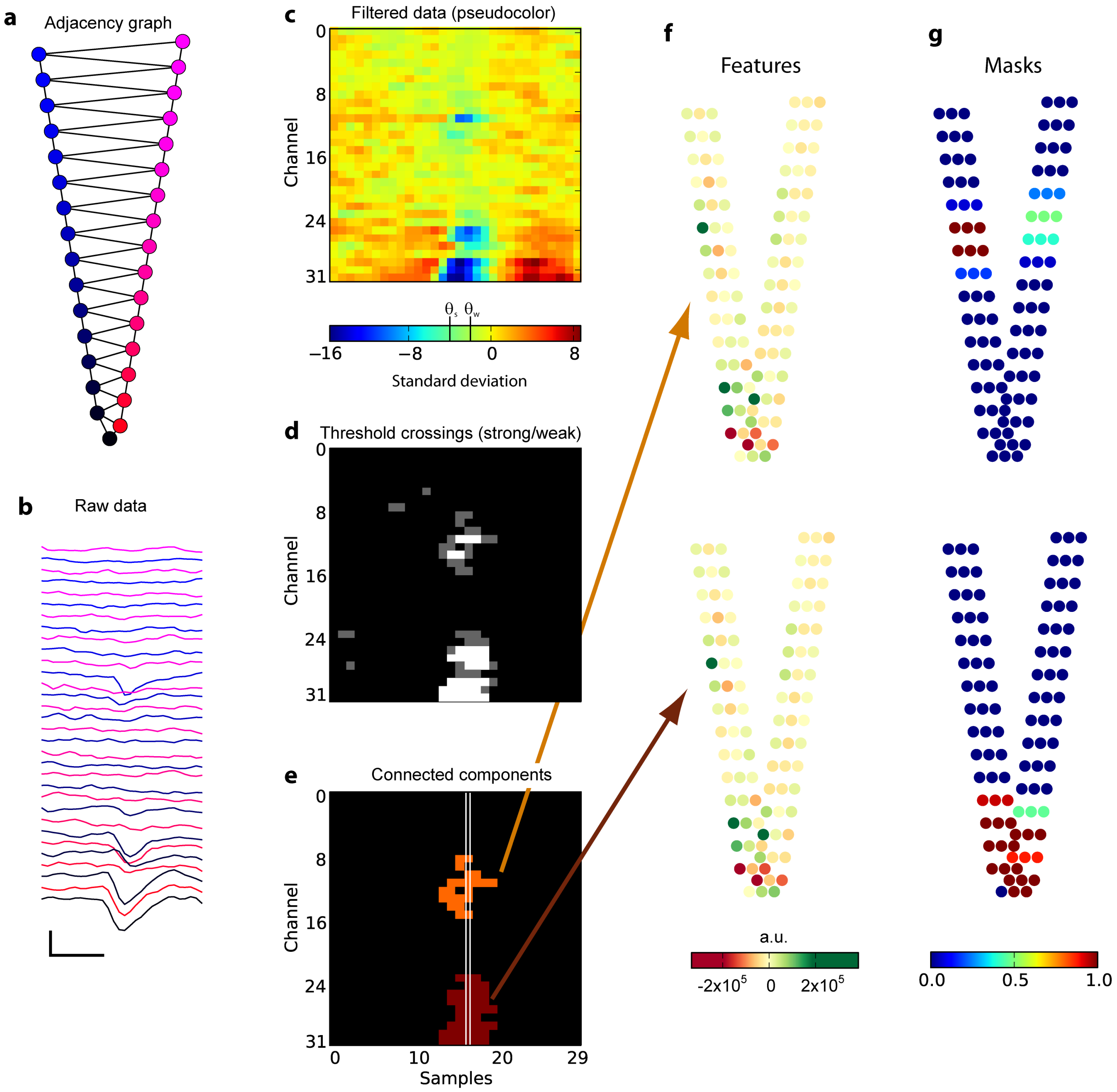
45. The HDF Group. Hierarchical Data Format, version 5. http://www.hdf5group.org/HDF5. (1997-2014).

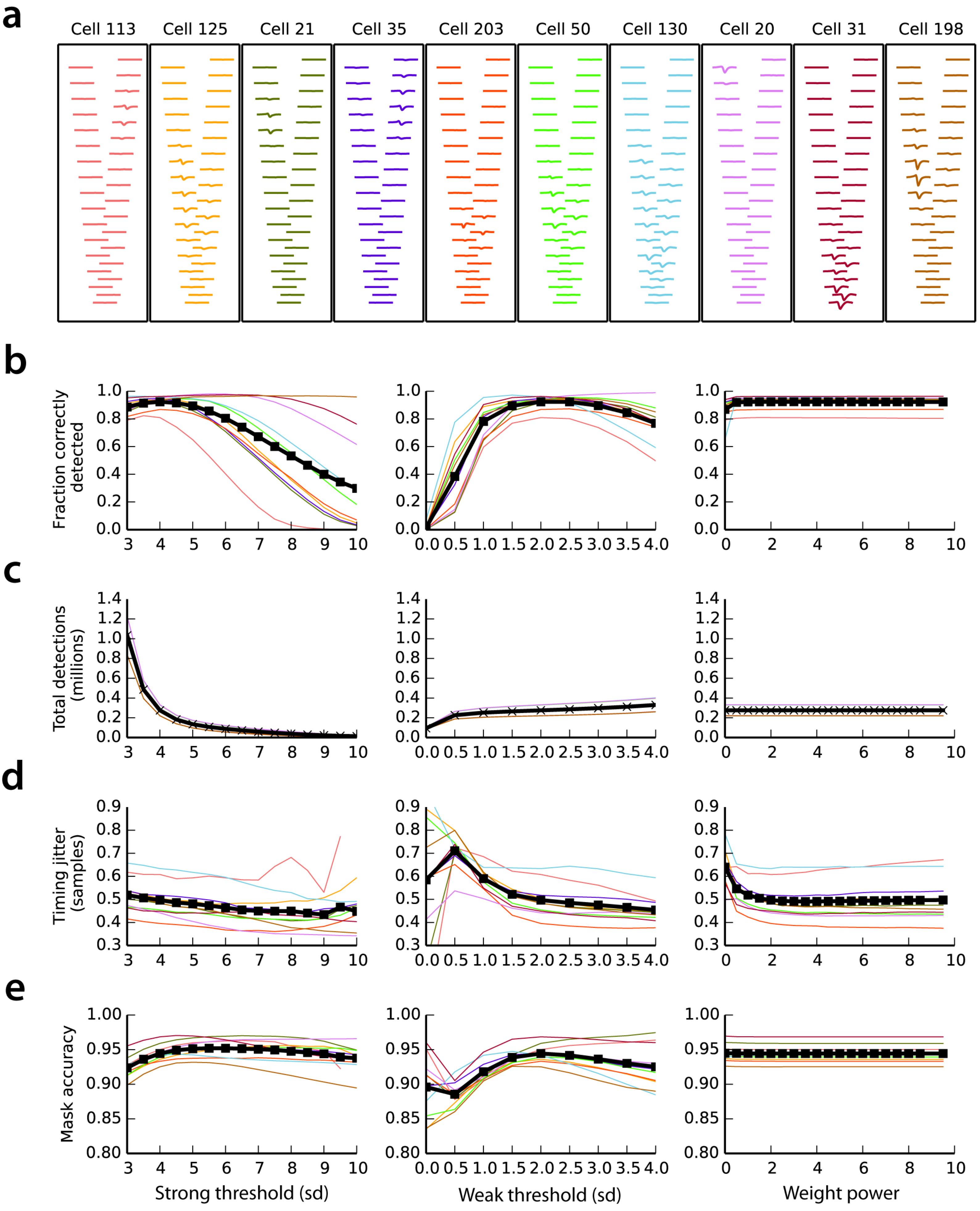
46. Rossant, C. & Harris, K.D. Hardware-accelerated interactive data visualization for neuroscience in Python. *Frontiers in neuroinformatics* **7**, 36 (2013).

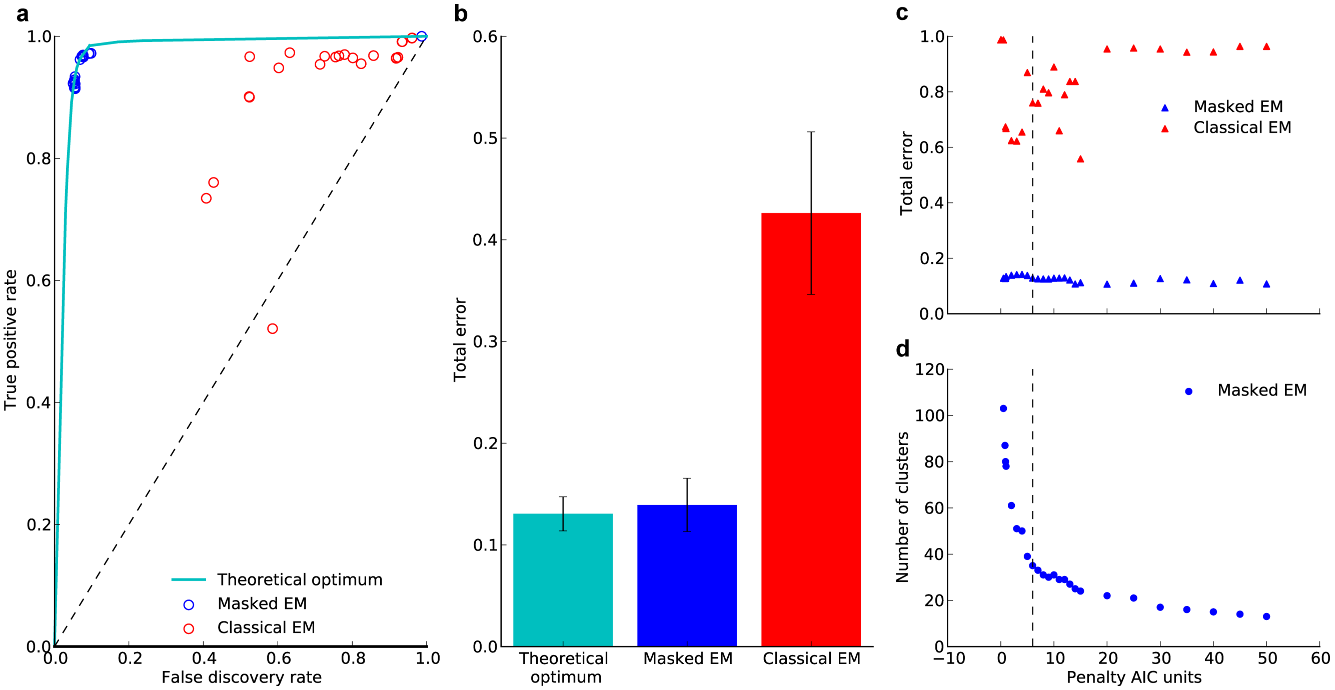
47. Shreiner, D., Sellers, G., Kessenich, J.M., Licea-Kane, B. & Khronos OpenGL ARB Working Group. *OpenGL programming guide : the official guide to learning OpenGL, version 4.3*, Edn. Eighth edition.

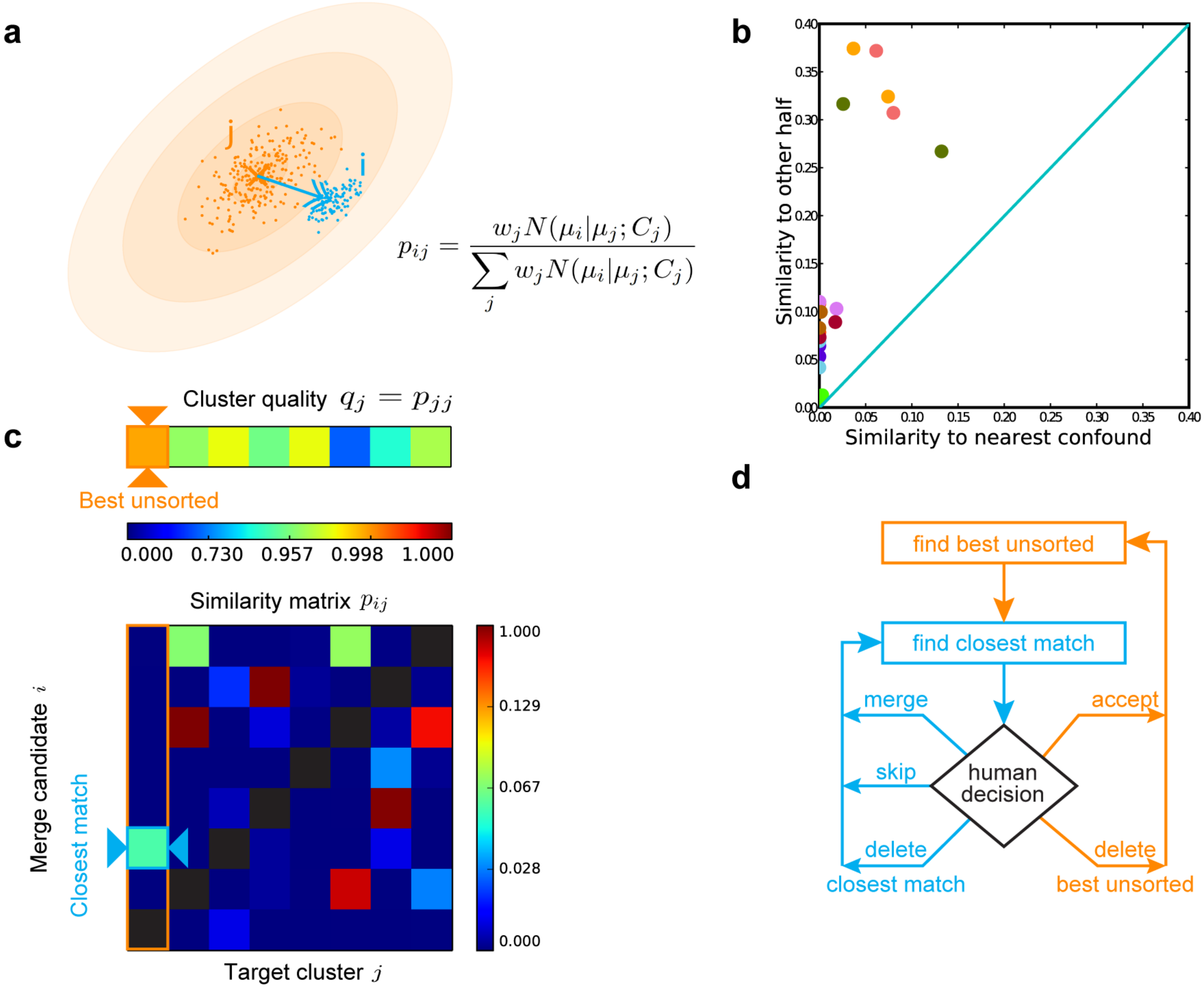
48. Swayne, D.F., Cook, D. & Buja, A. XGobi: Interactive dynamic data visualization in the X Window System. *J Comput Graph Stat* **7**, 113-130 (1998).

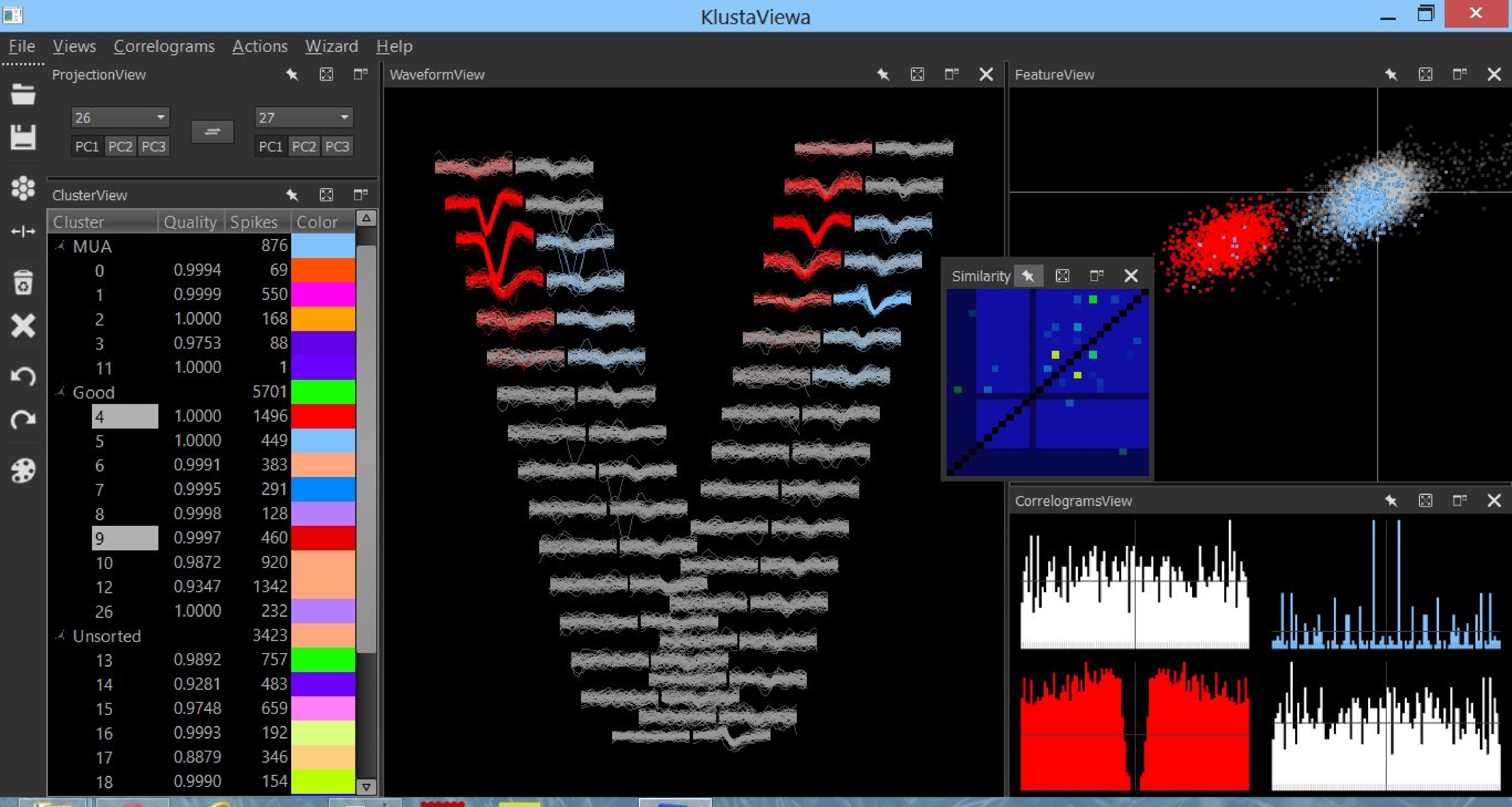

a
b
25 ms


a
Adjacency graph
b
Raw data
c
Filtered data (pseudocolor)
0
8
16
24
31
Channel
θs θw
–16
–8
0
8
Standard deviation
d
Threshold crossings (strong/weak)
0
8
16
24
31
Channel
e
Connected components
0
8
16
24
31
Channel
0
10
20
29
Samples
f
Features
a.u.
-2x10^5
0
2x10^5
g
Masks
0.0
0.5
1.0

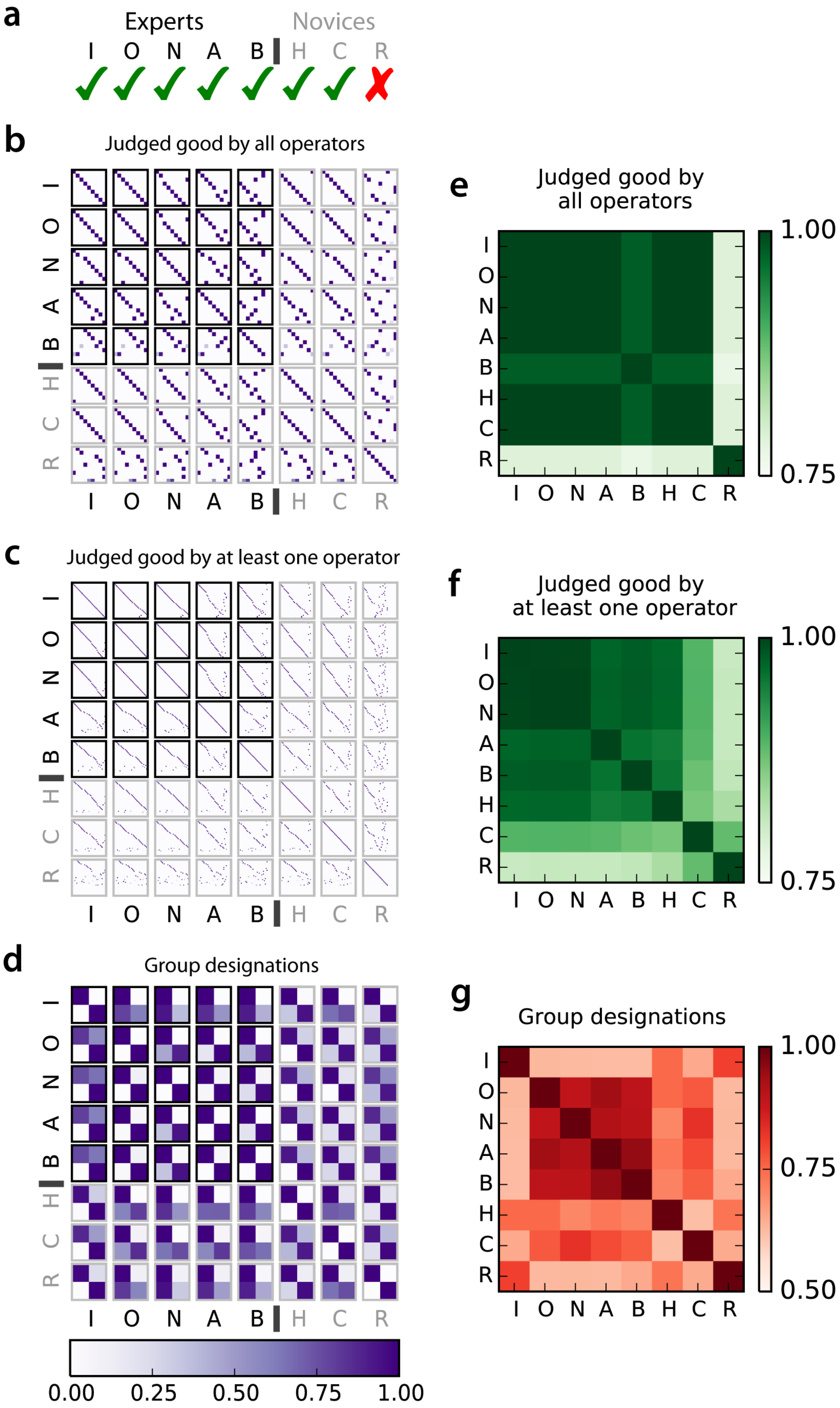

a
Cell 113
Cell 125
Cell 21
Cell 35
Cell 203
Cell 50
Cell 130
Cell 20
Cell 31
Cell 198
b
Fraction correctly detected
1.0
0.8
0.6
0.4
0.2
0.0
3 4 5 6 7 8 9 10
0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 4.0
0 2 4 6 8 10
c
Total detections (millions)
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
d
Timing jitter (samples)
0.9
0.8
0.7
0.6
0.5
0.4
0.3
e
Mask accuracy
1.00
0.95
0.90
0.85
0.80
Strong threshold (sd)
Weak threshold (sd)
Weight power


a
True positive rate
False discovery rate
0.0
0.2
0.4
0.6
0.8
1.0
Theoretical optimum
Masked EM
Classical EM
b
Total error
0.0
0.1
0.2
0.3
0.4
0.5
0.6
Theoretical optimum
Masked EM
Classical EM
c
Total error
Masked EM
Classical EM
d
Number of clusters
0
20
40
60
80
100
120
Masked EM
Penalty AIC units
−10
0
10
20
30
40
50
60


a
j
i
$p_{ij} = \frac{w_j N(\mu_i|\mu_j;C_j)}{\sum_j w_j N(\mu_i|\mu_j;C_j)}$
b
Similarity to other half
Similarity to nearest confound
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
c
Cluster quality $q_j = p_{jj}$
Best unsorted
0.000
0.730
0.957
0.998
1.000
Similarity matrix p_{ij}
1.000
0.129
0.067
0.028
0.000
Merge candidate i
Closest match
Target cluster j
d
find best unsorted
find closest match
merge
skip
delete
closest match
human decision
accept
delete
best unsorted



Cluster	Quality	Spikes	Color
MUA		876	🟦
0	0.9994	69	🟧
1	0.9999	550	🟪
2	1.0000	168	🟧
3	0.9753	88	🟪
11	1.0000	1	🟪
Good		5701	🟩
4	1.0000	1496	🟥
5	1.0000	449	🟦
6	0.9991	383	🟧
7	0.9995	291	🟦
8	0.9998	128	🟪
9	0.9997	460	🟥
10	0.9872	920	🟧
12	0.9347	1342	🟧
26	1.0000	232	🟪
Unsorted		3423	🟧
13	0.9892	757	🟩
14	0.9281	483	🟪
15	0.9748	659	🟪
16	0.9993	192	🟨
17	0.8879	346	🟧
18	0.9990	154	🟩


a
Experts
Novices
I O N A B H C R
✓ ✓ ✓ ✓ ✓ ✓ ✓ ✗
b
Judged good by all operators
I O N A B H C R
I O N A B H C R
c
Judged good by at least one operator
I O N A B H C R
I O N A B H C R
d
Group designations
I O N A B H C R
I O N A B H C R
0.00
0.25
0.50
0.75
1.00
e
Judged good by all operators
I O N A B H C R
I O N A B H C R
1.00
0.75
f
Judged good by at least one operator
I O N A B H C R
I O N A B H C R
1.00
0.75
g
Group designations
I O N A B H C R
I O N A B H C R
1.00
0.75
0.50