SpikeForest: reproducible web-facing ground-truth validation of automated neural spike sorters

Jeremy F. Magland[1], **James J. Jun**[1], **Elizabeth Lovero**[2], **Alexander J. Morley**[3], **Cole L. Hurwitz**[4], **Alessio P. Buccino**[5], **Samuel Garcia**[6], **Alex H. Barnett**[1]

[1]Center for Computational Mathematics, Flatiron Institute, New York, NY, USA; [2]Scientific Computing Core, Flatiron Institute, New York, NY, USA; [3]Medical Research Council Brain Network Dynamics Unit, University of Oxford, Oxford, United Kingdom; [4]School of Informatics, University of Edinburgh, United Kingdom; [5]Centre for Integrative Neuroplasticity (CINPLA), University of Oslo, Oslo, Norway; [6]Centre de Recherche en Neuroscience de Lyon, CNRS, Lyon, France

Abstract

Spike sorting is a crucial but time-intensive step in electrophysiological studies of neuronal activity. While there are many popular software packages for spike sorting, there is little consensus about which are the most accurate under different experimental conditions. SpikeForest is an open-source and reproducible software suite that benchmarks the performance of automated spike sorting algorithms across an extensive, curated database of electrophysiological recordings with ground truth, displaying results interactively on a continuously-updating website. With contributions from over a dozen participating laboratories, our database currently comprises 650 recordings (1.3 TB total size) with around 35,000 ground-truth units. These data include extracellular recordings paired with intracellular voltages, state-of-the-art simulated recordings, and hybrid synthetic datasets. Ten of the most frequently used modern spike sorting codes are wrapped under a common Python framework and evaluated on a compute cluster using an automated pipeline. SpikeForest validates and documents community progress in automated spike sorting, and guides neuroscientists to an optimal choice of sorter and parameters for a wide range of probes and brain regions.

Introduction

Direct electrical recording of extracellular potentials (***Buzsáki, 2004***; ***Seymour et al., 2017***) is one of the most popular modalities for studying neural activity since it is possible to determine, with sub-millisecond time resolution, individual firing events from hundreds (potentially thousands) of cells, and to track the activity of individual neurons over hours or days. Recordings are acquired either from within the living animal (*in vivo*) or from extracted tissue (*ex vivo*), at electrodes separated by typically 5-25 μm, with sensitivities of order 10 μV, and 10–30 kHz sampling rate. Probes for *in vivo* use—which are usually needle-like to minimize tissue damage during insertion—include microwire monotrodes (***Hubel, 1957***; ***Nicolelis et al., 1997***), tetrodes (***Gray et al., 1995***; ***Harris et al., 2000***; ***Dhawale et al., 2017***), and multi-shank probes (with typically 1–4 columns of electrodes per shank) on silicon (***Csicsvari et al., 2003***; ***Buzsáki, 2004***; ***Jun et al., 2017a***) or polymer (***Kuo et al., 2013***; ***Chung et al., 2019***) substrates. Multiple such probes are often combined into arrays to cover a larger volume in tandem. For *ex vivo* use (e.g., retinal sections), planar, two-dimensional multi-electrode arrays (MEAs) are common, allowing channel counts of up to tens of thousands (***Eversmann et al., 2003***; ***Litke et al., 2004***; ***Berdondini et al., 2005***; ***Yuan et al., 2016***; ***Tsai et al., 2017***).

Spike sorting is an essential computational step needed to isolate the activity of individual neurons, or units, within extracellular recordings which combine noisy signals from many neurons. Historically, this procedure has relied on manual steps (***Hazan et al., 2006***; ***Prentice et al., 2011***; ***Rossant et al., 2016***): putative waveforms crossing an amplitude threshold are visualized in a low-dimensional space (either using peak amplitudes or dimensionality reduction techniques), then clusters are separated by eye. While manual spike sorting is manageable with small numbers of recording channels, the rapid growth in channel counts and data volume in recent years as well as the requirement for reproducibility and objectivity demand automated approaches.

Most automated algorithms apply a sequence of steps that include filtering, detection, dimension reduction, and clustering, although these may be combined with (or replaced by) many other approaches such as template matching (***Prentice et al., 2011***; ***Pillow et al., 2013***; ***Pachitariu et al., 2016***), dictionary learning or basis pursuit (***Carlson et al., 2013***; ***Ekanadham et al., 2013***), and independent component analysis (***Takahashi et al., 2002***; ***Buccino et al., 2018***). The past 20 years have seen major efforts to improve these algorithms, with recent work focusing on the challenges arising from probe drift (changing spike waveform shapes), spatiotemporally overlapping spikes, and massive data volumes. We will not attempt a full review here, but instead refer the reader to, e.g., ***Fee et al.*** (***1996***); ***Lewicki*** (***1998***); ***Quiroga*** (***2012***); ***Einevoll et al.*** (***2012***); ***Rey et al.*** (***2015***); ***Lefebvre et al.*** (***2016***); ***Hennig et al.*** (***2019***); ***Carlson and Carin*** (***2019***).

In the last few years, many automated spike sorters have been released and are in wide use. Yet, there is little consensus about which is the best choice for a given probe, brain region and experiment type. Often, decisions are based not on evidence of accuracy or performance but rather on the ease of installation or usage, or historical precedent. Thus, the goals of extracting the highest quality results from experiments and of improving reproducibility across laboratories (***Denker et al., 2018***; ***Harris et al., 2016***) make objective comparison of the available automated spike sorters a pressing concern.

One approach to assessing spike sorter accuracy is to devise intrinsic quality metrics that are applied to each sorted unit, quantifying, for instance, the feature-space isolation of a cluster of firing events (***Pouzat et al., 2002***; ***Schmitzer-Torbert et al., 2005***; ***Hill et al., 2011***; ***Neymotin et al., 2011***; ***Barnett et al., 2016***; ***Chung et al., 2017***). Another approach is to use biophysical validation methods such as examining cross-correlograms or discovered place fields (***Li et al., 2015***; ***Chung et al., 2017***). However, the gold standard, when possible, is to evaluate the sorter by comparing with ground-truth data, i.e., using recordings where the spike train for one or more units is known *a priori*. Laboratory acquisition of such recordings is difficult and time-consuming, demanding simultaneous *paired* extracellular and intra-/juxta-cellular probes (***Harris et al., 2000***; ***Franke et al., 2015***; ***Neto et al., 2016***; ***Yger et al., 2018***; ***Allen et al., 2018***; ***Marques-Smith et al., 2018a***). Since the number of ground-truth units collected in this way is currently small (one per recording), *hybrid* recordings (where known synthetic firing events are added to experimental data) (***Marre et al., 2012***; ***Steinmetz, 2015–2018***; ***Rossant et al., 2016***; ***Pachitariu et al., 2016***; ***Wouters et al., 2019***), and biophysically detailed *simulated* recordings (***Camuñas-Mesa and Quiroga, 2013***; ***Hagen et al., 2015***; ***Gratiy et al., 2018***; ***Buccino and Einevoll, 2019***), which can contain 1–2 orders of magnitude more ground-truth units, have also been made available for the purpose of method validation.

Recently, such ground-truth data have been used to compare new spike sorting algorithms against preexisting ones (***Einevoll et al., 2012***; ***Pachitariu et al., 2016***; ***Chung et al., 2017***; ***Jun et al., 2017b***; ***Lee et al., 2017***; ***Yger et al., 2018***). However, the choice of accuracy metrics, sorters, data sets, parameters, and code versions varies among studies, making few of the results reproducible, transparent, or comprehensive enough to be of long-term use for the community. To alleviate these issues, a small number of groups initiated web-facing projects to benchmark spike sorter accuracy, notably `G-Node` (***Franke et al., 2012***), a `phy` hybrid study (***Steinmetz, 2015–2018***) and `spikesortingtest` (***Mitelut, 2016–2019***). To our knowledge, these unmaintained projects are either small-scale snapshots or are only partially realized. Yet, in the related area of calcium imaging, leaderboard-style comparison efforts have been more useful for establishing community benchmarks (***Freeman, 2015–2018***; ***Berens et al., 2018***).

We have addressed the above needs by creating and deploying the SpikeForest software suite. SpikeForest comprises a large database of electrophysiological recordings with ground truth (collected from the community), a parallel processing pipeline that benchmarks the performance of many automated spike sorters, and an interactive website that allows for in-depth exploration of the results. At present, the database includes hundreds of recordings, of all three types discussed above (paired, hybrid, and state-of-the-art biophysical simulation), contributed by more than a dozen laboratories and containing more than 30,000 ground-truth units. Our pipeline (see Fig. 1) runs the various sorters on the recordings, then finds, for each ground-truth unit, the sorted unit whose firing train is the best match, and finally computes metrics involving the numbers of correct, missing, and false positive spikes. A set of accuracy evaluation metrics are then derived per ground-truth unit for each sorter. By averaging results from many units of a similar recording type, we provide high-level accuracy summaries for each sorter in various experimental settings (Fig. 2). In order to understand the failure modes of each sorter, SpikeForest further provides various interactive plots, e.g., accuracy vs. signal-to-noise ratio (SNR) (Fig. 3, left panel), and multi-channel waveforms of random samples of individual firing events (Fig. 3, right panel).

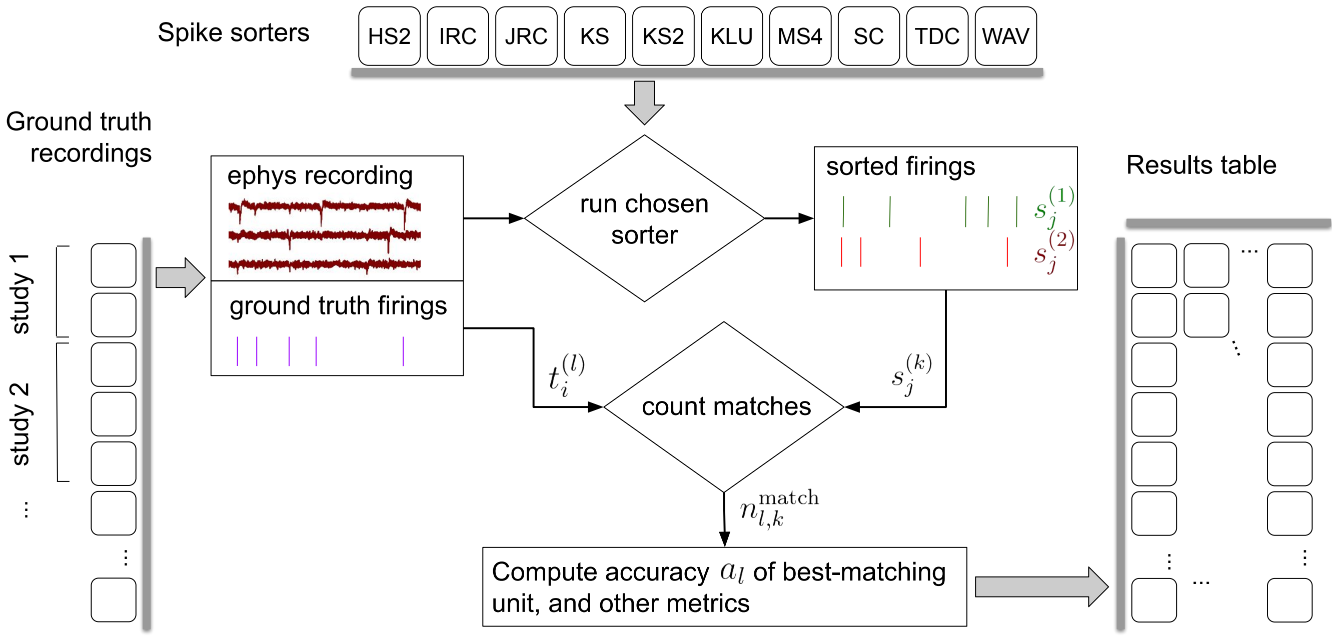


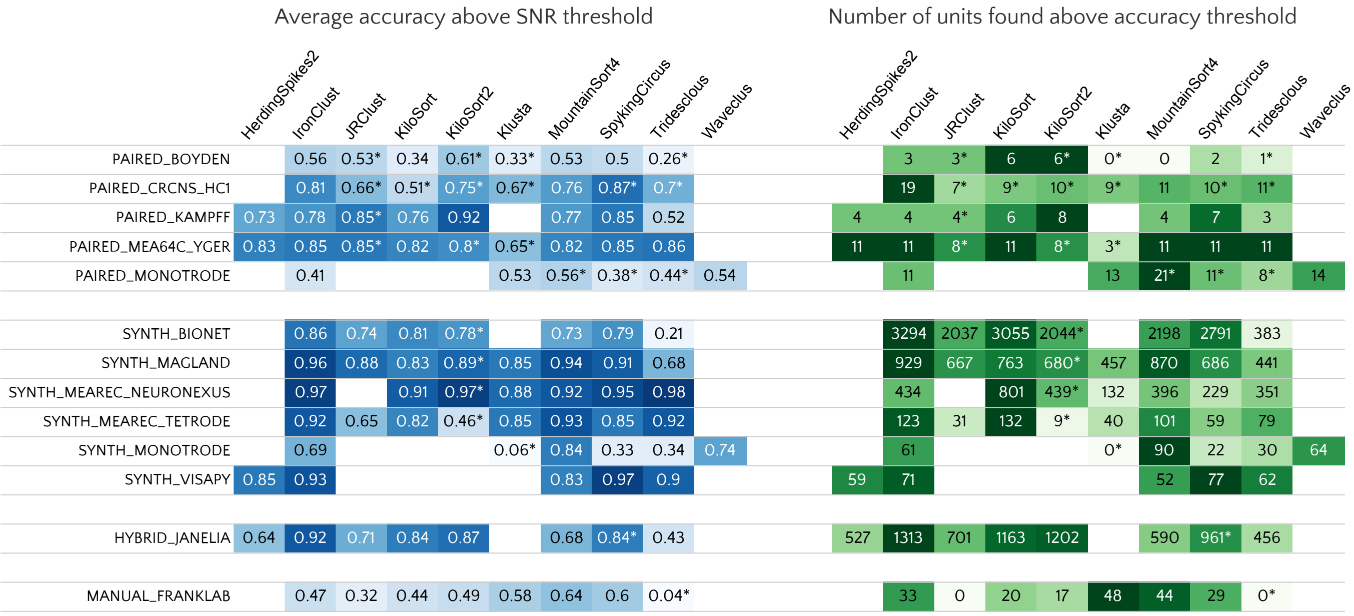
Figure 1. Simplified flow diagram of the SpikeForest analysis pipeline. Each in a collection of spike sorting codes (top) are run on each recording with ground truth (left side) to yield a large matrix of sorting results and accuracy metrics (right). See the section on comparison with ground truth for mathematical notations. Recordings are grouped into "studies", and those into "study sets"; these share features such as probe type and laboratory of origin. The web interface summarizes the results table by grouping them into study sets (as in Fig. 2), but also allows drilling down to the single study and recording level. Aspects such as extraction of mean waveforms, representative firing events, and computation of per-unit SNR are not shown, for simplicity.

Implemention on this scale (close to 400 core-hours are required just to run the sorters) necessitates standardization and fault-tolerance. At the core of our software is a Python package which, in conjunction with the SpikeInterface project (***Buccino et al., 2019***), provides standardized wrappers for most popular spike sorters. These include: HerdingSpikes2 (***Hilgen et al., 2017***), IronClust (***Jun et al., 2020***), JRCLUST (***Jun et al., 2017b***), KiloSort (***Pachitariu et al., 2016***), KiloSort2 (***Pachitariu et al., 2018–2019***), Klusta (***Rossant et al., 2016***), MountainSort4 (***Chung et al., 2017***), SpyKING CIRCUS (***Yger et al., 2018***), Tridesclous (***Garcia and Pouzat, 2018–2019***), and WaveClus (***Chaure et al., 2018***; ***Quiroga et al., 2004***). Since each of these spike sorters operates within a unique computing environment, we utilize Docker (***Merkel, 2014***) and Singularity (***Kurtzer et al., 2017***) containers to rigorously encapsulate the versions and prerequisites for each algorithm, ensuring independent verifiability of results, and circumventing software library conflicts.

Our framework includes a system for job management, batching, and auto-caching that automatically reruns portions of the analysis whenever it detects a modification to the GitHub repositories associated with algorithms, the sorting parameters, or the set of registered ground-truth recordings. It also handles the parallel execution of hundreds of simultaneous spike sorter runs on a compute cluster, and is able to report failures if a job crashes or does not complete within a reasonable time.

A central aim of this project is to maximize the transparency and reproducibility of the analyses. To this end, all data—the set of recordings, their ground-truth firings, and firing outputs from all sorters—are available for public download via our Python API. SpikeForest itself is open-source, as are the wrappers to all sorters, the Docker containers, and all of the parameter settings used in the current study results. In fact, code to rerun any sorting task may be requested via the web interface, and is auto-generated on the fly. Both the code and the formulae (for accuracy, SNR, and other metrics) are documented on the site, with links to the source code repositories.

Although human curation is still part of the spike sorting analysis pipeline in most labs, the increase in the potential yield from recently developed high-density recording devices will soon make this step infeasible. We therefore adopt the philosophy that spike sorting algorithms should be evaluated in an automated, reproducible fashion and that, when used for benchmarking purposes, the algorithms should be wrapped and run server-side without the possibility of human curation of their output. This follows evaluation efforts for automated clustering algorithms such as ClustEval (***Wiwie et al., 2015***).



	Average accuracy above SNR threshold										Number of units found above accuracy threshold									
	HerdingSpikes2	IronClust	JRClust	KiloSort	KiloSort2	Klusta	MountainSort4	SpykingCircus	Tridesclous	Waveclus	HerdingSpikes2	IronClust	JRClust	KiloSort	KiloSort2	Klusta	MountainSort4	SpykingCircus	Tridesclous	Waveclus
PAIRED_BOYDEN		0.56	0.53*	0.34	0.61*	0.33*	0.53	0.5	0.26*			3	3*	6	6*	0*	0	2	1*	
PAIRED_CRCNS_HC1		0.81	0.66*	0.51*	0.75*	0.67*	0.76	0.87*	0.7*			19	7*	9*	10*	9*	11	10*	11*	
PAIRED_KAMPFF	0.73	0.78	0.85*	0.76	0.92		0.77	0.85	0.52		4	4	4*	6	8		4	7	3	
PAIRED_MEA64C_YGER	0.83	0.85	0.85*	0.82	0.8*	0.65*	0.82	0.85	0.86		11	11	8*	11	8*	3*	11	11	11	
PAIRED_MONOTRODE		0.41				0.53	0.56*	0.38*	0.44*	0.54		11				13	21*	11*	8*	14
SYNTH_BIONET		0.86	0.74	0.81	0.78*		0.73	0.79	0.21			3294	2037	3055	2044*		2198	2791	383	
SYNTH_MAGLAND		0.96	0.88	0.83	0.89*	0.85	0.94	0.91	0.68			929	667	763	680*	457	870	686	441	
SYNTH_MEAREC_NEURONEXUS		0.97		0.91	0.97*	0.88	0.92	0.95	0.98			434		801	439*	132	396	229	351	
SYNTH_MEAREC_TETRODE		0.92	0.65	0.82	0.46*	0.85	0.93	0.85	0.92			123	31	132	9*	40	101	59	79	
SYNTH_MONOTRODE		0.69				0.06*	0.84	0.33	0.34	0.74		61				0*	90	22	30	64
SYNTH_VISAPY	0.85	0.93					0.83	0.97	0.9		59	71					52	77	62	
HYBRID_JANELIA	0.64	0.92	0.71	0.84	0.87		0.68	0.84*	0.43		527	1313	701	1163	1202		590	961*	456	
MANUAL_FRANKLAB		0.47	0.32	0.44	0.49	0.58	0.64	0.6	0.04*			33	0	20	17	48	44	29	0*	

Figure 2. Main results table from the SpikeForest website showing aggregated results for 10 algorithms applied to 13 registered study sets. The left columns of the table show the average accuracy (see (5)) obtained from averaging over all ground-truth units with SNR above an adjustable threshold, here set to 8. The right columns show the number of ground-truth units with accuracy above an adjustable threshold, here set to 0.8. The first five study sets contain paired recordings with simultaneous extracellular and juxta- or intra-cellular ground truth acquisitions. The next six contain simulations from various software packages. The HYBRID_JANELIA, obtained from ***Pachitariu et al.*** (***2018–2019***), is real data with synthetic spike waveforms superimposed at known times. The last study set is a collection of human-curated tetrode data. An asterisk indicates an incomplete (timed out) or failed sorting on a subset of results; in these cases, missing accuracies are imputed using linear regression as described in the Methods. These results reflect the analysis run of January 9th, 2020.

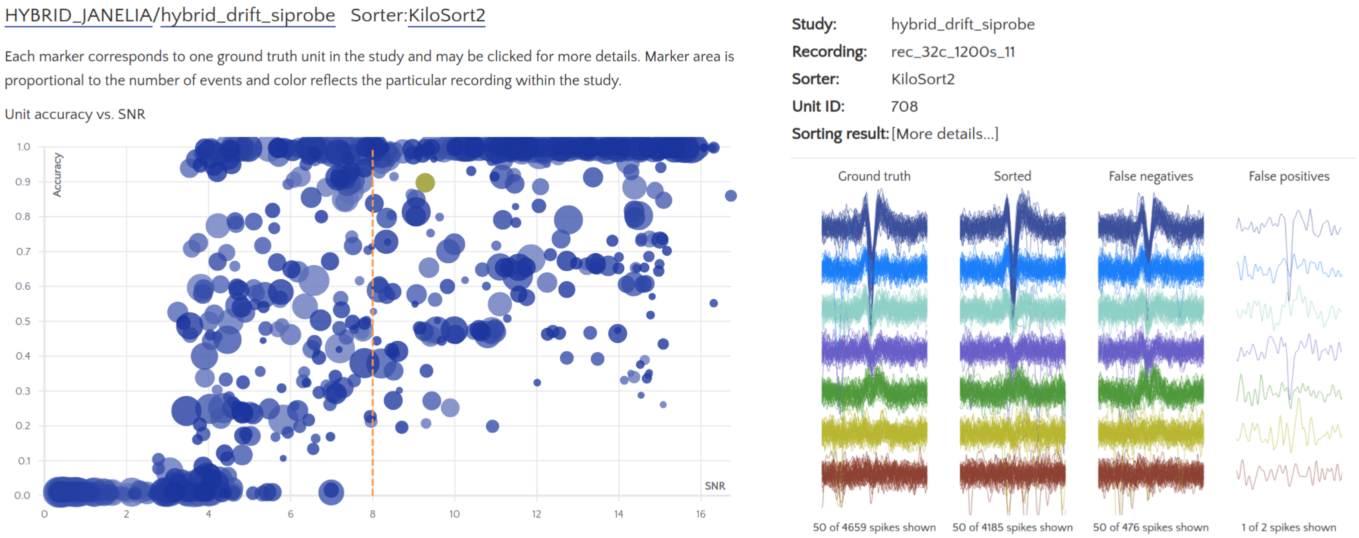
We constrast this to competition-style efforts (***Franke et al., 2012***; ***Freeman, 2015–2018***; ***Berens et al., 2018***) which allow contributions of (potentially non-reproducible) sorting results, and which report accuracy on held-out data whose ground truth are necessarily private, and thus cannot be interrogated by the community.

Our work has three main objectives. The primary goal is to aid neuroscientists in selecting the optimal spike sorting software (and algorithm parameters) for their particular probe, brain region, or application. A second goal is to spur improvements in current and future spike sorting software by providing standardized evaluation criteria. This has already begun to happen as developers of some spike sorting algorithms have already made improvements in direct response to this project. As a byproduct, and in collaboration with the SpikeInterface project (***Buccino et al., 2019***), we achieve a third objective of providing a software package which enables laboratories to run a suite of many popular, open-source, automatic spike sorters, on their own recordings via a unified Python interface.

Results

Electrophysiology recordings (together with ground-truth information) registered in SpikeForest are organized into *studies*, and studies are then grouped into *study sets*. Table 1 details all study sets presently in the system. Recordings within a study set share a common origin (e.g., laboratory) and type (e.g., paired), whereas recordings within the same study are associated with very similar simulation parameters or experimental conditions.

The results of the latest SpikeForest analysis may be found at https://spikeforest.flatironinstitute.org and are updated on a regular basis as the ground-truth recordings, sorting algorithms, and sorting parameters are adjusted based on community input. The central element of this web page is the main results matrix (Fig. 2) which summarizes results for each sorter listed in Table 2 (using formulae defined later by (5)). The average accuracies are mapped to a color scale (heat map), with darker blue indicating higher accuracy, using a nonlinear mapping designed to highlight differences at the upper end. For the average accuracy table on the left, only ground-truth units with SNR above a user-adjustable threshold are included in the average accuracy calculations; the user may then explore interactively the effect of unit amplitude on the sorting



HYBRID_JANELIA/hybrid_drift_siprobe Sorter:KiloSort2

Each marker corresponds to one ground truth unit in the study and may be clicked for more details. Marker area is proportional to the number of events and color reflects the particular recording within the study.

Unit accuracy vs. SNR

Study: hybrid_drift_siprobe
Recording: rec_32c_1200s_11
Sorter: KiloSort2
Unit ID: 708
Sorting result: [More details...]

Figure 3. Screenshots from the SpikeForest website. (left) Scatter plot of accuracy vs. SNR for each ground-truth unit, for a particular sorter (KiloSort2) and study (a hybrid drift dataset from the HYBRID_JANELIA study set). The SNR threshold for the main table calculation is shown as a dashed line, and the user-selected unit is highlighted. Marker area is proportional to the number of events, and the color indicates the particular recording within the study. (right) A subset of spike waveforms (overlaid) corresponding to the selected ground truth unit, in four categories: ground truth, sorted, false negative, and false positive.

accuracies of all sorters. If a sorter either crashes or times out (>1 hour run time) on any recording in a study set, an asterisk is appended to that accuracy result, and the missing values are imputed using linear regression as described in the Methods section (there is also an option to simply exclude the missing data from the calculation).

The right table displays the number of ground truth units with accuracy above a user-adjustable threshold (0.8 by default), regardless of SNR. This latter table may be useful for determining which sorters should be used for applications that benefit from a high yield of accurately sorted units and where the acceptable accuracy threshold is known. The website also allows easy switching between three quality metrics (accuracy, precision, and recall) as described in the section on comparison with ground truth.

Clicking on any result expands the row into its breakdown across studies. Further breakdowns are possible by clicking on the study names to reveal individual recordings. Clicking on any result brings up a scatter plot of accuracy vs. SNR for each ground-truth unit for that study/sorter pair (e.g., Fig. 3, left side). Additional information can then be obtained by clicking on the markers for individual units, revealing individual spike waveforms (e.g., Fig. 3, right side).

Since neuroscientist users also need to compare the efficiencies (speeds) of algorithms, we measure total computation time for each algorithm on each study, and provide this as an option for display on the website via a heat map. Run times are measured using our cluster pipeline, which allocates a single core to each sorting job on shared-memory multi-core machines (with GPU resources as needed). Since many jobs share I/O and RAM bandwidth, these cannot be taken as accurate indicators of speeds in ideal settings. We will not discuss speed results in detail, but only note that older sorters such as Klusta can be over 30 times slower than more recent GPU sorters such as KiloSort and IronClust.

At present, the total compute time for the 650 recordings and 10 sorters is 380 core hours, yet it takes only 3–4 hours (excluding failing jobs) to complete this analysis when run in parallel on our compute cluster with up to 100–200 jobs running simultaneously (typically 14 jobs per node). Since the system automatically detects which results require updating, the pipeline may be run on a daily basis utilizing minimal compute resources for the usual situation where few (if any) updates are needed.

We now draw some initial conclusions about the relative performances of the spike sorters based on the threshold choices in Fig. 2. No single spike sorter emerged as the top performer in all study sets, with IronClust, KiloSort2, MountainSort4, and SpyKING CIRCUS each appearing among the most accurate in at least six of the study sets.

The higher average accuracy of KiloSort2 over its predecessor KiloSort is evident, especially for paired recordings. However, in synthetic studies, particularly tetrodes, KiloSort finds more units above accuracy 0.8 than KiloSort2. Scatter plots (e.g., Fig. 3, left side) show that KiloSort2 can retain high accuracy down to lower SNR than other sorters, but for not for all such low-SNR units. While KiloSort2 was among the best performers for six of the study sets, KiloSort and KiloSort2 had higher numbers of crashes than any of the other sorters, including crashing on every one of the SYNTH_VISAPY recordings. It is likely that modifications to sorting parameters could reduce the number of crashes, but attempts so far, including contacting the author, have not yet fixed this problem. In the synthetic datasets, KiloSort2 had the largest number of *false positive units* (distinct from the false positive rate of a single unit), but this is not currently reported by SpikeForest (see Discussion).

IronClust appears among the top average accuracies for eight of the study sets, and is especially strong for the simulated and drifting recordings. For most study sets, IronClust has improved accuracy over its predecessor JRCLUST, and is also improved in terms of speed and reliability (no crashes observed). Although a substantial portion of the development of the IronClust software took place while it had access to the SpikeForest ground truth datasets, the same sorting parameters are used across all studies, limiting the potential for overfitting (see Discussion).

MountainSort4 is among the top performers for six of the study sets (based on the average accuracy table) and does particularly well for the low-channel-count datasets (monotrodes and tetrodes). It is not surprising that MountainSort4 is the top performer for MANUAL_FRANKLAB because that data source was used in development of the algorithm (***Chung et al., 2017***).

When considering the left table (average accuracy), SpyKING CIRCUS is among the best sorters for ten study sets. However, it ranks a lot lower in the unit count table on the right of Fig. 2. This was an example of a sorter that improved over a period of months as a result of using SpikeForest for benchmarking.

HerdingSpikes2 was developed for high-density MEA probes and uses a 2D estimate of the spike location, hence was applied only for recordings with a sufficiently planar electrode array structure (this excluded tetrodes and linear probes). For PAIRED_MEA64C_YGER its performance was similar to other top sorters, but in the other study sets, it was somewhat less accurate. One advantage of HerdingSpikes2 not highlighted in the results table is that it is computationally efficient for large arrays, even without using a GPU.

Tridesclous is among the top performers for both MEAREC study sets and for PAIRED_MEA64C_YGER, but had a substantially lower accuracy for most of the other datasets. This algorithm appears to struggle with lower-SNR units.

Klusta is substantially less accurate than other sorters in most of the study sets, apart from MANUAL_FRANKLAB where, surprisingly, it found the most units above accuracy 0.8 of any sorter. It also has one of the highest crash/timeout rates.

The version of WaveClus used in SpikeForest is only suited for (and only run on) monotrodes; a new version of WaveClus now supports polytrodes, but we have not yet integrated it. We included both paired and synthetic monotrode study sets with studies taken from selected single electrodes of other recordings. Four sorters (HerdingSpikes2, JRCLUST, KiloSort, and KiloSort2) were unable to sort this type of data. Of those that could, MountainSort4 was the most accurate, with accuracies slightly higher than WaveClus.

An eleventh algorithm, Yet Another Spike Sorter (YASS) (***Lee et al., 2017***), was not included in the comparison because, even after considerable effort and reaching out to the authors, its performance was too poor, leading us to suspect an installation or configuration problem. We plan to include YASS in a future version of the analysis.

Methods

Ground-truth recordings

The thirteen study sets included in the SpikeForest analysis are detailed in Table 1. All but the last belong to one of the three categories described in the introduction: paired, simulated, or hybrid.

We selected 145 paired recordings from raw extracellular recordings that were publicly released by four laboratories (***Henze et al., 2000***; ***Harris et al., 2000***; ***Henze et al., 2009***; ***Neto et al., 2016***; ***Allen et al., 2018***; ***Marques-Smith et al., 2018a***,b; ***Yger et al., 2018***; ***Spampinato et al., 2018***). The intracellular spike times were taken from the author-reported values unless they were not provided (***Henze et al., 2009***). We prepared 93 paired recordings from the Buzsaki laboratory (***Henze et al., 2000***; ***Harris et al., 2000***; ***Henze et al., 2009***) with ground truth based on the intracellular traces after excluding time segments containing artifacts due to movement and current injection. The paired recordings were obtained from tetrodes or conventional silicon probes in rat hippocampus (PAIRED_CRCNS_HC1) (***Henze et al., 2000***; ***Harris et al., 2000***; ***Henze et al., 2009***), high-density silicon probes in mice cortex (PAIRED_KAMPFF and PAIRED_BOYDEN) (***Neto et al., 2016***; ***Allen et al., 2018***; ***Marques-Smith et al., 2018a***,b), and high-density MEAs in mice retina (PAIRED_MEA64C_YGER) (***Yger et al., 2018***; ***Spampinato et al., 2018***). We generated a monotrode version of the paired recordings (PAIRED_MONOTRODE) by randomly sampling

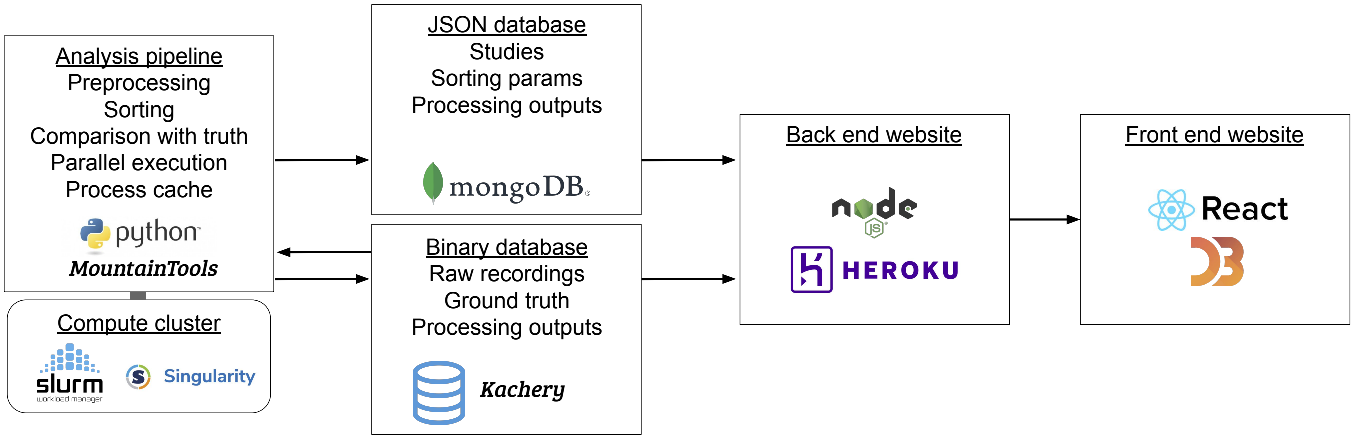


Figure 4. Interaction of software and hardware components of the SpikeForest system, showing the flow of data from the server-side analysis (left) to the user's web browser (right). The processing pipeline automatically detects which sorting jobs need to be updated and runs these in parallel as needed on a compute cluster. Processing results are uploaded to two databases, one for relatively small JSON files and the other for large binary content. A NodeJS application pulls data from these databases in order to show the most up-to-date results on the front-end website.

one channel from each recording session to uniformly span the SNR range (25 units per study).

To overcome the limited number of units offered by the paired ground truth, we added simulated ground-truth study sets that were independently generated by five laboratories (*Camuñas-Mesa and Quiroga, 2013*; *Hagen et al., 2015*; *Chung et al., 2017*; *Gratiy et al., 2018*; *Buccino and Einevoll, 2019*). The simulators vary in their biophysical details, computational speeds, and configurable parameters. Simulations based on phenomenological models tend to be fast and easily configurable (e.g., SYNTH_MAGLAND, identical to the simulations in *Chung et al.* (*2017*) except with iid Gaussian noise), while biophysical simulators such as SYNTH_VISAPY (*Hagen et al., 2015*) and SYNTH_BIONET (*Gratiy et al., 2018*) use synaptically connected, morphologically detailed neurons to achieve high fidelity at the expense of computational speed. SYNTH_MONOTRODE (*Camuñas-Mesa and Quiroga, 2013*) and SYNTH_MEAREC (*Buccino and Einevoll, 2019*) take an intermediate approach by generating spike waveform templates based on single-neuron simulations and randomly placing the spike waveforms conforming to pre-specified ISI distributions.

SYNTH_BIONET was generated from the Allen Institute's BioNet simulator (*Gratiy et al., 2018*; *Jun et al., 2017b*) running on the computing resources provided by the Neuroscience Gateway (*Sivagnanam et al., 2013*). We simulated a column of synaptically connected neurons ($n = 708$, $200 \times 200 \times 600 \mu$m^3) based on the rat cortical NEURON models (*Hines and Carnevale, 1997*; *Ascoli et al., 2007*; *Mitelut, 2017*; *Jun et al., 2017b*) by capturing the spike waveforms at four vertical columns of densely-spaced electrodes (2 μm vertical, 16 μm horizontal pitch, 600 channels). Linear probe drift was simulated by subsampling the electrodes to match the Neuropixels site layout (20-25 μm pitch) and vertically shifting the electrode positions as a function of time. To achieve smooth linear motion, a 2D-interpolation based on a Gaussian kernel was applied (0.5 μm vertical spacing, 16 μm total displacement for 16 minutes). Based on the linear probe drift simulation, we generated a 'shuffled' version to mimic the fast probe displacements during animal movement by subdividing the recordings into 32 time segments and randomly shuffling (30 s per segment).

Hybrid ground truth combines the advantages of both paired and simulated recordings by simulating a spike train based on the spike waveforms obtained from experimental recordings. The simulation entitled HYBRID_JANELIA was developed by J. Colonell (*Pachitariu et al., 2018–2019*) and uses average unit waveforms from a densely spaced electrode array (5 μm pitch, 15×17 layout) collected by the Kampff laboratory (*Kampff, 2018*). The waveform templates were inserted at randomly selected channels and time points after being multiplied by an amplitude-scaling factor drawn from a Gamma distribution. The baseline noise was randomly generated to match the power spectrum observed from a Neuropixels recording, and a spatiotemporal smoothing was applied to induce correlation between nearby channels and time samples. The original simulation generated either static or sinusoidal drift ($\pm 10 \mu$m displacement, 600 s period) with 1200 s duration (30 KS/s) on the Neuropixels layout (64 channels, 25 μm pitch, staggered four columns). We trimmed the original simulations to generate shorter recording durations (600, 1200 s) and channel counts (4, 16, 32, 64 channels) to study the effects of these parameters on sorting accuracy.

Based on the requests of the electrophysiology community, we included a single study set of *manually* curated tetrode sortings (MANUAL_FRANKLAB) (***Chung et al., 2017***); we emphasize that, for this study set only, *there is no ground-truth data*. Accuracies are reported relative to three independent manual sortings of the same tetrode recording session in a chronically implanted rat hippocampus. We subdivided the entire recording duration to generate three different durations (600, 1200, 2400 s) to study the effect of duration on sorting accuracy, but did not find a consistent relationship across all sorters.

A content-addressable storage database is used to store the file content of each recording, and all data is available for public download via our Python API. Since these files are indexed according to their SHA-1 hashes, their content is guaranteed not to have changed even when the mechanism for retrieving the data evolves over time, ensuring long-term repeatability of the analysis.

Sorting algorithms and wrappers

Table 2 provides details on the ten algorithms included in the SpikeForest analysis. For each spike sorter, SpikeForest contains a Python wrapper and a Docker (and Singularity) container defining the exact execution environment including all necessary dependencies. For the sorters that are implemented in MATLAB, the MATLAB Compiler tool was used to generate standalone binary files inserted into the containers so that a MATLAB license is not required to run spike sorting. The SpikeForest framework flexibly allows running of each sorter either within the native operating system or within the container. The former method is useful during development or in an environment where the spike sorting software is already installed. The latter is crucial for ensuring reproducibility and for avoiding conflicts between sorters due to incompatible dependencies. The Python wrappers make use of the `spikesorters` package of the SpikeInterface project (***Buccino et al., 2019***) for passing the parameters and executing the sorter. All sorters operate on the raw (unfiltered) recordings.

An eleventh algorithm, YASS, was incorporated into the Python package, but was not included in the comparison, as discussed in the Results section.

Comparison with ground truth

Depending on the experimental context, false negatives (missed events) and false positives may have different relative importance for the researcher. Thus the SpikeForest website allows switching between three evaluation metrics for the comparison with ground truth: precision (which penalizes only false positives), recall (which penalizes only false negatives), and an overall accuracy metric (which balances the two). For each sorter-recording pair these are computed by comparing the output of spike sorting (spike times and labels) with the ground-truth timings associated with the recording, as follows.

We first consider a sorted unit k and a ground-truth unit l, and describe how the events in the spike train for k are matched to ground-truth events of l. Let $s_1^{(k)}, \ldots, s_{M_k}^{(k)}$ be the spike train (timestamps) associated with sorted unit k, and let $t_1^{(l)}, \ldots, t_{N_l}^{(l)}$ be the spike train for ground-truth unit l (see Fig. 1). Let Δ be an acceptable firing time error, which we assume is shorter than half the refractory period of any true neuron. We set Δ at 1 millisecond; the results are rather insensitive to its exact value. The number of matches of sorted unit k to the ground-truth unit l is

$$n_{l,k}^{\text{match}} := \#\{i : |t_i^{(l)} - s_j^{(k)}| \le \Delta \text{ for some } j\}. \tag{1}$$

Note that even if more than one sorted event falls within $\pm\Delta$ of a true event, at most one is counted in this matching. The reverse situation—more than one ground-truth event from the same neuron matching to a given sorted event—cannot happen by our assumption about the refractory period. The number of missed events and false positives are then, respectively,

$$n_{l,k}^{\text{miss}} := N_l - n_{l,k}^{\text{match}}, \qquad n_{l,k}^{\text{fp}} := M_k - n_{l,k}^{\text{match}}, \tag{2}$$

where N_l is the total number of firings of ground-truth unit l, and M_k is the total number of found events of sorted unit k. Following ***Jun et al.*** (***2017a***) we define the *accuracy* for this pair as

$$a_{l,k} := \frac{n_{l,k}^{\text{match}}}{n_{l,k}^{\text{match}} + n_{l,k}^{\text{miss}} + n_{l,k}^{\text{fp}}}. \tag{3}$$

Note that this definition of accuracy is a balance between precision and recall, and is similar, but not identical, to the F_1 metric (***Zaki and Meira Jr, 2014***, Eq. (17.1)) used to evaluate clustering methods.

Study set	**# Rec. / # Elec. / Dur.**	**Source lab.**	**Description**
Paired intra/extracellular			
PAIRED_BOYDEN	19 / 32ch / 6-10min	E. Boyden	Subselected from 64, 128, or 256-ch. probes, mouse cortex
PAIRED_CRCNS_HC1	93 / 4-6ch / 6-12min	G. Buzsaki	Tetrodes or silicon probe (1 shank) in rat hippocampus
PAIRED_KAMPFF	15 / 32ch / 9-20min	A. Kampff	Subselected from 374, 127, or 32-ch. probes, mouse cortex
PAIRED_MEA64C_YGER	18 / 64ch / 5min	O. Marre	Subselected from 252-ch. MEA, mouse retina
PAIRED_MONOTRODE	100 / 1ch / 5-20min	Boyden, Kampff, Marre, Buzsaki	Subselected from paired recordings from four labs
Simulation			
SYNTH_BIONET	36 / 60ch / 15min	AIBS	BioNet simulation containing no drift, monotonic drift, and random jumps; used by JRCLUST, IronClust
SYNTH_MAGLAND	80 / 8ch / 10min	Flatiron Inst.	Synthetic waveforms, Gaussian noise, varying SNR, channel count and unit count
SYNTH_MEAREC_NEURONEX	60 / 32ch / 10min	A. Buccino	Simulated using MEAREC, varying SNR and unit count
SYNTH_MEAREC_TETRODE	40 / 4ch / 10min	A. Buccino	Simulated using MEAREC, varying SNR and unit count
SYNTH_MONOTRODE	111 / 1ch / 10min	Q. Quiroga	Simulated by Quiroga lab by mixing averaged real spike waveforms
SYNTH_VISAPY	6 / 30ch / 5min	G. Einevoll	Generated using VISAPy simulator
Hybrid			
HYBRID_JANELIA	60 / 4-64ch / 5-20min	M. Pachitariu	Distributed with KiloSort2, with and without simulated drift
Human curated			
MANUAL_FRANKLAB	21 / 4ch / 10-40min	L. Frank	Three manual curations of the same recordings

Table 1. Table of study sets included in the SpikeForest analysis. Study sets fall into four categories: paired, synthetic, hybrid, and curated. Each study set comprises one or more studies, which in turn comprise multiple recordings acquired or generated under the same conditions.

Sorting algorithm	Language	Notes
HerdingSpikes2	Python	Designed for large-scale, high-density multi-electrode arrays. See *Hilgen et al.* (*2017*).
IronClust	MATLAB and CUDA	Derived from JRCLUST. See *Jun et al.* (*2020*).
JRCLUST	MATLAB and CUDA	Designed for high-density silicon probes. See *Jun et al.* (*2017b*).
KiloSort	MATLAB and CUDA	Template matching. See *Pachitariu et al.* (*2016*).
KiloSort2	MATLAB and CUDA	Derived from KiloSort. See *Pachitariu et al.* (*2018–2019*).
Klusta	Python	Expectation-Maximization masked clustering. See *Rossant et al.* (*2016*).
MountainSort4	Python and C++	Density-based clustering via ISO-SPLIT. See *Chung et al.* (*2017*).
SpyKING CIRCUS	Python and MPI	Density-based clustering and template matching. See *Yger et al.* (*2018*).
Tridesclous	Python and OpenCL	See *Garcia and Pouzat* (*2018–2019*).
WaveClus	MATLAB	Superparamagnetic clustering. See *Chaure et al.* (*2018*); *Quiroga et al.* (*2004*).

Table 2. Table of spike sorting algorithms included in the SpikeForest analysis. Each algorithm is registered into the system via a Python wrapper. A Docker recipe defines the operating system and environment where the sorter is run.

Fixing the ground-truth unit l, we define its *best matching sorted unit* $\hat{k}_l$ as the sorted unit k with highest accuracy,

$$\hat{k}_l := \arg\max_k \ a_{l,k} \ . \tag{4}$$

Now restricting to this best match, we define the accuracy for ground-truth unit l by

$$a_l := a_{l,\hat{k}_l} \ , \tag{5}$$

and the corresponding precision and recall for this unit by

$$p_l := \frac{n^{\text{match}}_{l,\hat{k}_l}}{n^{\text{match}}_{l,\hat{k}_l} + n^{\text{fp}}_{l,\hat{k}_l}} \qquad\qquad r_l := \frac{n^{\text{match}}_{l,\hat{k}_l}}{n^{\text{match}}_{l,\hat{k}_l} + n^{\text{miss}}_{l,\hat{k}_l}} \ . \tag{6}$$

Averages of these metrics are then computed for all units l within each study, without weighting by their numbers of events (i.e., treating all units equally). Note that, in the case of a recording with more than one ground-truth unit, it is possible that more than one such unit could share a common best-matching sorted unit, but this could only happen if these ground-truth units had extremely correlated events or if the sorted was highly inaccurate.

The spike sorted units that are considered in the computation of these metrics are only those that are matched to ground-truth units. Therefore, the results shown in the main table do not account for false positive units, i.e., units found by the spike sorters that are not present in the recording.

Compensating for missing data

As described above, when a sorting run fails (either crashes or times out), an asterisk is appended to the corresponding table cell, and the average accuracy is calculated based on imputed data using linear regression. There is also an option to simply exclude the missing values, but the problem with this method (which we have encountered) is that sometimes an algorithm will happen to crash on recordings with relatively difficult units, resulting in artificially elevated scores. Imputing by zero has the problem of yielding deceptively low values. We have thus opted to use linear regression to fill in the missing data using values predicted based on the accuracies of other sorters. Specifically, for a given sorter with missing data and a given study set (or study, depending on the level of aggregation), a linear model is estimated for fitting the non-missing values based on the values of all sorters with no missing data. That model is then applied to estimate the missing data. To give some intuition, if a sorter typically has somewhat lower scores than the other sorters and crashes on one recording, then for the purpose of computing average accuracy, the accuracy for that recording will be filled in with a value that is also somewhat lower than the other sorters.

In constrast, when reporting total numbers of units found above an accuracy cut-off (e.g., Fig. 2, right side), we do not impute, but simply sum the number of non-missing units.

Signal-to-noise ratio per unit

We define SNR as a property of a single neural unit, either a ground-truth unit or a unit as output by a sorter. It is reported on the website and is used as a cut-off for selection of a subset of ground-truth units for computing average accuracy. SNR is computed on the bandpass-filtered timeseries data.

We first describe our filter used to compute SNR (noting that this is distinct from various filters used internally by the spike sorters). This filter is a bandpass from $f_{\text{min}} = 300$ to $f_{\text{max}} = 6000$, in Hz units. It is applied to each channel by taking the FFT, multiplying by the real-valued frequency filter function

$$A(f) = \frac{1}{2}\sqrt{1 + \text{erf}((f - f_{\text{min}})/100)}\sqrt{1 - \text{erf}((f - f_{\text{max}})/1000)}\ , \tag{7}$$

where erf is the error function, then taking the inverse FFT. Here, the parameters 100 and 1000 control the roll-off widths in Hz at the low and high ends respectively. From this filtered timeseries and the set of firing times of a unit, the average spike waveform is extracted. SNR is then defined as the ratio between the peak absolute amplitude of this average spike waveform and the estimated noise on the channel where this peak amplitude occurs. The noise is estimated as the median absolute deviation of the filtered timeseries data divided by 0.6745, which gives a robust estimate of the standard deviation of the noise (***Quiroga et al., 2004***, Sec. 3.1).

Analysis pipeline

The analysis pipeline of SpikeForest depicted in Fig. 4 is built using Python utilities developed by the authors for creating shareable and reproducible scientific workflows. This system provides a formal method for creating well-defined Python procedures that operate on input parameters and files and produce output files. These are known as *processors*. SpikeForest defines processors for running the spike sorters, computing properties of ground-truth units (e.g., SNR), comparing the spike sorting outputs with ground truth, and computing summary data for the plots shown on the website. Once these processors are set up, the framework provides several advantages including: (a) automatic execution of processing inside Singularity or Docker containers; (b) automatic caching of processing results; and (c) job management and queueing mechanisms for running batches of processing on a compute cluster. This allows the analysis pipeline to be defined as a standard Python script, with a simple nested loop iterating through the sorters and ground-truth datasets (as in Fig. 1). The script may then be configured to run in a variety of settings: on a standard workstation for development, testing, or reproducing of a subset of results; a shared computer with large memory and many cores; or on a compute cluster.

A crucial and novel feature of our framework is that all files (both input and output) are represented by SHA-1 URIs, for example of the form:

```
sha1://88b62db6fc467b83ba0693453c59c5f538e20d5c/firings_true.mda
```

The hexadecimal code embedded in the URI is the SHA-1 hash of the content of the file, and therefore this URI uniquely identifies the desired file (in this case the ground-truth firing data for one of the SpikeForest recordings) without specifying the actual location of the data. In contrast, explicit location references (e.g., a path on the local computer, an IP address, or a web URL) can be problematic because over time data archives may stop being maintained, may change locations, or files may be renamed or updated with new content. The SHA-1 URI system alleviates these difficulties by separating the mechanism for storing archives of files from the representation of these files, via universal hash strings within scripts.

All SpikeForest input recordings, ground-truth data, sorting outputs, and other processing data are stored in a public content-addressable storage (CAS) system called *kachery*. The API of a kachery database simply allows downloading of files referenced only by their SHA-1 hashes (or URIs). Therefore, as long as the environment is configured to point to a kachery database with the relevant files available, the SpikeForest pipeline may be executed (in whole or in part) on any computer connected to the internet.

The automatic caching capability we developed is also crucial for the SpikeForest system. If updates are made to either the database of ground-truth recordings or to the sorting algorithms and parameters, the system can automatically detect which processing needs to be rerun. In this way, the pipeline can simply be executed in full at regular intervals, and the website will continuously update with the latest changes. This facilitates a conceptually simple method for adding new datasets or sorting algorithms because changes are represented by modifications to pipeline configuration files.

A limitation of the reliance on caching is the implicit assumption that the sorting algorithms are deterministic, i.e., given the same inputs, parameters, and code, the outputs should be exactly the same. However, this assumption can be violated by unfixed random seeds or even differences between implementations of floating-point arithmetic on different hardware. Controlling these factors is currently out of scope for SpikeForest, but we aim to measure the degree of stability (***Barnett et al., 2016***) with respect to repeated runs of each sorting task in future versions.

To ensure that all results may be reproduced at a later date and/or independently verified by a third party, the Python wrapper for each spike sorter includes a reference to a Docker image containing the entire operating system and environment where the sorter is installed. Our system handles the automatic download of these images (if not already on the local computer) and the execution of the sorting inside Docker or singularity containers. For development and testing purposes, it is also possible to configure the pipeline to run outside the container, provided that the sorter software is installed on the operating system.

The output of processing is a single JSON file (only around 15 MB at this time) containing SHA-1 URI references to all recording files, ground-truth data, containers, spike sorting parameters, sorting outputs, comparisons with ground truth, and other information used by the website. It also contains meta-information about the sorting runs such as the execution times and console outputs. This JSON file is itself uploaded to the kachery database and is represented by a SHA-1 URI accessible on the website. The archive section of the website contains references to these files for all past analyses, allowing tracking of sorter performance over time.

Finally, the data from this JSON output file is loaded into a MongoDB database for efficient access by the website's front end.

Website front end

The primary user interface for the SpikeForest platform is an isomorphic JavaScript web application with overall structure as shown on the right side of Fig. 4. Built from reusable React components, the front end utilizes the D3 library to render the interactive tables and plots. A Node.JS backend, organized using the Redux state container, queries a MongoDB database to retrieve JSON files for each datatype and generate the comparative visualizations. For more detailed plots, like the spike sprays, larger data objects are retrieved via the content-addressable storage database.

We optimized the interface for efficient hierarchical navigation through the results and rapid loading and interactive response (for instance, when a user adjusts the SNR or accuracy cut-off slider bar). When the user clicks on results at the individual recording or unit level, they are taken to a page (with an auto-generated, shareable URL) specific to the study in question. This page allows one-click comparison of the sorters on this study. Clicking on individual units in the scatter plot for a given sorter-study pair brings up spike waveforms for that unit (Fig. 3) and a link to sub-page with details specific to the particular sorter-recording run. This latter page includes sorter parameters used, the console output of the run and a link to an auto-generated python script with human-readable documentation to reproduce that sorter run within SpikeForest. A permanent top-level menu bar allows access to all meta-data about sorters (as in Table 2), study sets (as in Table 1, also allowing hierarchical drilling down to the individual recording level), historical snapshots, metrics used, and an explanation of the project. All of the website data and results sub-pages are automatically generated from the SpikeForest databases.

Discussion

We have introduced a Python framework and public website for evaluating and comparing many popular spike sorting algorithms by running them on a large and diverse set of curated electrophysiology datasets with ground truth. We have described the principal features of the website and the methodology used to prepare the ground truth data of varying types (paired, synthetic, hybrid, and manually-curated) and apply the spike sorting algorithms in a uniform, transparent and unbiased manner. We have summarized initial findings comparing sorter accuracies, and invite the reader to explore the continually updating results on the live site.

One conclusion is that (as of this time) there is no single sorter that is the most accurate across a diversity of probe types, experimental conditions, metric types and SNR cut-offs. Rather, several different sorters are optimal in different settings. For example, MountainSort4 performs especially well for low channel counts (tetrodes and monotrodes), IronClust excels in the simulated and drifting datasets, while KiloSort2 retains higher accuracy (for units that are found) at lower SNR than other sorters.

Our contribution also helps to overcome some practical issues with neurophysiology research in the laboratory. Traditionally, spike sorting codes bring special requirements for installation and custom input/output file formats, and also require expertise in sorting parameter selection and using visualization and curation utilities intimately tied to that code. This problem has been exacerbated by the use of GPU computing to improve performance in recently developed codes. It is therefore challenging, and unusual, for a laboratory to have more than one or two installed sorters for in-house evaluation. Part of this obstacle is overcome by the SpikeInterface project (***Buccino et al., 2019***), which SpikeForest utilizes, but another part is solved by the Docker and Singularity containers that we have developed to fully capture the operating system environments needed by each sorter.

In the past, comparisons between spike sorting algorithms have been biased or limited. Almost all have been presented in the context of a new or improved method, and so the choice of simulations used for evaluation can often be (unintentionally) biased toward showing that the new method is superior. Given only such reports, it is also impractical for readers to verify that the various sorters were used properly and with sensible parameters. The openness and reproducibility of SpikeForest aims to remedy this situation.

There are many ongoing challenges to the validation of spike sorters. With SpikeForest we made the decision to package sorters and run them on our own compute cluster, on recordings that we make publicly available; this contrasts "competition"-style comparisons on held-out test data (discussed in the Introduction). Although some effort has been made to choose good parameters for each sorter, optimal choices are not guaranteed. We anticipate that a use case for SpikeForest will be finding the best combination of sorter and parameters for a particular recording type. However, automated parameter optimization (as in ClustEval by ***Wiwie et al.*** (***2015***)) has not yet been implemented since it would multiply the total CPU cost by a large factor. For now, we encourage the community to contact us with improved settings or algorithms. Indeed, we reached out to the developers of all ten tested sorters in mid-2019 with preliminary results, and several developers (including at least those of SpyKING CIRCUS, HerdingSpikes2, IronClust, and Tridesclous) have already used SpikeForest to improve their software. We emphasize that, for reproducibility, sorter versions and parameter choices used for each date-stamped analysis are available in the downloadable analysis archives linked from the website.

This raises the issue of potential *overfitting*. Because all SpikeForest data are public, the community may continue to optimize sorters using SpikeForest as a benchmark, so one might wonder if this will lead to SpikeForest metrics which overestimate true real-world performance. We believe that such bias will be small, and certainly smaller than the bias of studies conducted in order to propose a single sorting algorithm, for the following reasons. 1) Our database is large and diverse, comprising 13 different types of study sets. It is difficult to imagine overfitting to 650 different recordings by optimizing even dozens of parameters, and, while we allow the potential (as yet unused) for different parameter choices for different studies, multiple recordings would still have to be overfit by one parameter set. 2) All code and parameter settings are available for public inspection, making the above style of parameter and algorithm hacking obvious, were it to occur. We feel that the benefit to the community of improved, validated sorters far outweighs the risk of overfitting. Naturally, SpikeForest results *are* biased towards the types of neurons and conditions for which ground-truth data exist; by keeping results for different study sets separate we expose such biases as much as possible. In the long run, we appreciate that *both* held-out and open data benchmarks will play a useful role in comparison and standardization.

Another issue is the paucity of paired ground-truth units in the database, a consequence of the time-consuming nature of their collection. For instance the study set PAIRED_KAMPFF has 15 units, of which only 11 have sufficient SNR to be sortable by any of the sorters. Therefore a *single* false split or merge by a sorter can lead to variations in reported study-averaged accuracy as large as ± 0.05, and dictate the most accurate sorter for that study set. In (small-scale) laboratory pipelines it is possible that such splits or merges would be detected by expert human curation; the point of SpikeForest is to measure the performance of purely automated algorithms. Such sensitivity will be reduced as the size of ground-truth studies increases.

In the future we plan to report false positive units. Currently only the one sorted unit which best matches each ground-truth unit is assessed for accuracy. Thus there is no penalty for a sorter which generates many spurious units that are not present in the data (this is necessarily true for KiloSort in most cases, since the number of returned units is set in advance by the user). Results from such a sorter would of course then be less credible, and, even if examined by an expert, much harder to curate. This failure mode is *not* revealed by paired recordings; yet, in simulations (where every single firing is ground truth) a new metric capturing the fraction of such spurious units could be included (see ***Buccino et al.*** (***2019***)).

These issues help inform several specific future goals:

1. SpikeForest does not directly address the common laboratory task of deciding, given a new sorting of electrophysiology recordings, which neural units are to be trusted and which discarded. Assessing the credibility of putative neural units output by spike sorters remains a crucial question, and many quality metrics are used in practice, including SNR, firing rate, metrics of cluster isolation or separation from the "noise cluster", stability, and cross-correlograms (***Pouzat et al., 2002***; ***Schmitzer-Torbert et al., 2005***; ***Hill et al., 2011***; ***Neymotin et al., 2011***; ***Barnett et al., 2016***; ***Chung et al., 2017***). Yet the SpikeForest database provides a unique opportunity to tabulate such quality metrics for multiple sorter outputs, then correlate each to the actual ground-truth accuracy with the goal of assessing the *predictive power* of each metric, or combinations thereof. Note that we have already made one step in this direction with scatter plots of accuracy vs. SNR (Fig. 3, left side), but plan to add other quality metrics.
2. It is currently not as easy as it could be to identify common failure modes such as false merges and false splits. We plan to display *confusion matrices* (see, e.g., ***Barnett et al.*** (***2016***)) between ground-truth and a given sorting; this would also allow comparison between two different sorter outputs.
3. We plan to add accuracy metrics that address specific tasks, such as the ability to handle probe drift and long recording durations (currently only two hybrid and one synthetic study explicitly test drift). The website could visualize successful tracking as a function of recording duration. We also plan to include accuracy metrics that penalize false positive units, as discussed above, or that evaluate the stability with respect to repeated runs (see (***Barnett et al., 2016***)).
4. The comparison of CPU time for each sorter is currently sub-optimal, because multiple sorters may be running on one node. We plan to include special timing comparisons on dedicated nodes in order to reflect actual laboratory use cases.
5. The number of sorter job crashes and time-outs needs be reduced, which requires detailed diagnosis on many sorter-recording pairs. For instance, currently KiloSort2 has a larger proportion of crashes or time-outs than many other sorters, yet this is not fully understood and not quantified in accuracy summary results.
6. Finally, we envision that the machinery of SpikeForest could be used for a *web-based* spike sorting platform, to which users would upload their data (which becomes public), run possibly more than one sorter, visualize, curate, and download the results. This could render all spike sorting and human curation affecting the downstream science analysis accessible and reproducible.

We anticipate that, as the use and scale of spike sorting as a tool continues to grow, SpikeForest will become a resource for comparison and validation of sorter codes and encourage more rigorous reproducibility in neuroscience. To this end, we seek contributions and input from the electrophysiology community, both to optimize parameter settings for existing algorithms, and to further expand the set of algorithms and ground-truth recordings included in the analysis.

Acknowledgments

We have benefited from discussions with Leslie Greengard, Eftychios Pnevmatikakis, Andrea Giovannucci, Karel Svoboda, and Loren Frank. The BioNet simulations were made possible by Catalin Mitelut and the Neuroscience Gateway Portal. We are grateful to the organizers of the Janelia spike sorting workshop, held in March 2018, which helped to inspire this project. The Flatiron Institute is a division of the Simons Foundation.

References

Allen BD, Moore-Kochlacs C, Bernstein JG, Kinney J, Scholvin J, Seoane L, Chronopoulos C, Lamantia C, Kodandaramaiah SB, Tegmark M, Boyden ES. Automated in vivo patch clamp evaluation of extracellular multielectrode array spike recording capability. J Neurophysiol. 2018; 120(5):2182–2200.

Ascoli GA, Donohue DE, Halavi M. NeuroMorpho.Org: a central resource for neuronal morphologies. J Neurosci. 2007; 27(35):9247–9251.

Barnett AH, Magland JM, Greengard LF. Validation of neural spike sorting algorithms without ground-truth information. J Neurosci Meth. 2016; 264:65–77.

Berdondini L, van der Wal PD, Guenat O, de Rooij NF, Koudelka-Hep M, Seitz P, Kaufmann R, Metzler P, Blanc N, Rohr S. High-density electrode array for imaging in vitro electrophysiological activity. Biosens Bioelectron. 2005; 21(1):167–74. http://www.ncbi.nlm.nih.gov/pubmed/15967365, doi: 10.1016/j.bios.2004.08.011.

Berens P, Freeman J, Deneux T, Chenkov N, McColgan T, Speiser A, Macke JH, Turaga SC, Mineault P, Rupprecht P, Gerhard S, W FR, Friedrich J, Paninski L, Pachitariu M, Harris KD, Bolte B, Machado TA, Ringach D, Stone J, et al. Community-based benchmarking improves spike rate inference from two-photon calcium imaging data. PLoS Comput Biol. 2018; 14(5):e1006157.

Buccino AP, Einevoll GT. MEArec: a fast and customizable testbench simulator for ground-truth extracellular spiking activity. bioRxiv. 2019; doi: 10.1101/691642.

Buccino AP, Hagen E, Einevoll GT, Häfliger PD, Cauwenbergh G. Independent Component Analysis for Fully Automated Multi-Electrode Array Spike Sorting. In: *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)* IEEE; 2018. p. 2627–2630.

Buccino AP, Hurwitz CL, Magland J, Garcia S, Siegle JH, Hurwitz R, Hennig MH. SpikeInterface, a unified framework for spike sorting. bioRxiv. 2019; doi: 10.1101/796599.

Buzsáki G. Large-scale recording of neuronal ensembles. Nat Neurosci. 2004; 7(5):446–451.

Camuñas-Mesa LA, Quiroga RQ. A detailed and fast model of extracellular recordings. Neural Comput. 2013; 25(5):1191–1212.

Carlson DE, Vogelstein JT, Wu Q, Lian W, Zhou M, Stoetzner CR, Kipke D, Weber D, Dunson DB, Carin L. Multichannel electrophysiology spike sorting via joint dictionary learning & mixture modeling. IEEE Trans Biomed Eng. 2013; .

Carlson D, Carin L. Continuing progress of spike sorting in the era of big data. Current opinion in neurobiology. 2019; 55:90–96.

Chaure FJ, Rey HG, Quian Quiroga R. A novel and fully automatic spike-sorting implementation with variable number of features. J Neurophysiol. 2018; 120(4):1859–1871.

Chung JE, Joo HR, Fan JL, Liu DF, Barnett AH, Chen P, Geaghan-Breiner C, Karlsson MP, Lee KY, Liang H, Magland JF, Pebbles JA, Tooker AC, Greengard LF, Tolosa VM, Frank JM. High-density, long-lasting, and multi-region electrophysiological recordings using polymer electrode arrays. Neuron. 2019; 101:21–31.

Chung JE, Magland JF, Barnett AH, Tolosa VM, Tooker AC, Lee KY, Shah KG, Felix SH, Frank LM, Greengard LF. A fully automated approach to spike sorting. Neuron. 2017; 95(6):1381–1394.

Csicsvari J, Henze DA, Jameison B, Harris KD, Sirota A, Barthoó P, Wise KD, Buzsáki G. Massively Parallel Recording of Unit and Local Field Potentials With Silicon-Based Electrodes. J Neurophys. 2003; 90:1314–1323.

Denker M, Einevoll GT, Franke F, Grün S, Hagen E, Kerr JND, Nawrot MP, Ness TV, Ritz R, Smith LS, Wachtler T, Wójcik DK, 1st INCF Workshop on Validation of Analysis Methods; 2018. https://doi.org/10.7490/f1000research.1115887.1.

Dhawale AK, Poddar R, Wolff SB, Normand VA, Kopelowitz E, Ölveczky BP. Automated long-term recording and analysis of neural activity in behaving animals. eLife. 2017; 6:e27702. doi: 10.7554/eLife.27702.

Einevoll GT, Franke F, Hagen E, Pouzat C, Harris KD. Towards reliable spike-train recordings from thousands of neurons with multielectrodes. Curr Opin Neurobiol. 2012; 22(1):11–17. http://dx.doi.org/10.1016/j.conb.2011.10.001, doi: 10.1016/j.conb.2011.10.001.

Ekanadham C, Tranchina D, Simoncelli EP. A unified framework and method for automatic neural spike identification. J Neurosci Meth. 2013; 222:47–55. doi: 10.1016/j.jneumeth.2013.10.001.

Eversmann B, Jenkner M, Hofmann F, Paulus C, Brederlow R, Holzapfl B, Fromherz P, Merz M, Brenner M, Schreiter M, Gabl R, Plehnert K, Steinhauser M, Eckstein G, Schmitt-Landsiedel D, Thewes R. A 128 × 128 CMOS Biosensor Array for Extracellular Recording of Neural Activity. IEEE J Solid-State Circuits. 2003; 38(12):2306–2317.

Fee MS, Mitra PP, Kleinfeld D. Automatic sorting of multiple unit neuronal signals in the presence of anisotropic and non-Gaussian variability. J Neurosci Meth. 1996; 69:175–188.

Franke F, Pröpper R, Alle H, Meier P, Geiger JR, Obermayer K, Munk MH. Spike sorting of synchronous spikes from local neuron ensembles. J Neurophysiol. 2015; 114:2535–49.

Franke F, Meier P, Obermayer KH, Watchler T, Sobolev A, Tiziano Z, Einevoll G, Hagen E, Ness TV, Khosrowshahi A, Teleńczuk B, Bielievtsov D, Kadir S, G-Node Spikesorting Evaluation Project (website); 2012. http://spike.g-node.org/.

Freeman J, NeuroFinder (website); 2015–2018. http://neurofinder.codeneuro.org.

Garcia S, Pouzat C, Tridesclous (software); 2018–2019. https://github.com/tridesclous/tridesclous.

Gratiy SL, Billeh YN, Dai K, Mitelut C, Feng D, Gouwens NW, Cain N, Koch C, Anastassiou CA, Arkhipov A. BioNet: A Python interface to NEURON for modeling large-scale networks. PLoS One. 2018; 13(8):e0201630.

Gray C, Maldonado P, Wilson M. Tetrodes markedly improve the reliability and yield of multiple single-unit isolation from multi-unit recordings in cat striate cortex. J Neurosci Meth. 1995; 63:43–54.

Hagen E, Ness TV, Khosrowshahi A, Sørensen C, Fyhn M, Hafting T, Franke F, Einevoll GT. ViSAPy: a Python tool for biophysics-based generation of virtual spiking activity for evaluation of spike-sorting algorithms. Journal of neuroscience methods. 2015; 245:182–204.

Harris KD, A HD, Csicsvari J, Hirase H, Buzsaki G. Accuracy of Tetrode Spike Separation as Determined by Simultaneous Intracellular and Extracellular Measurements. J Neurophysiol. 2000; 84:401–414.

Harris KD, Quiroga RQ, Freeman J, Smith SL. Improving data quality in neuronal population recordings. Nat Neurosci. 2016; 19(9):1165.

Hazan L, Zugaro M, Buzsáki G. Klusters, NeuroScope, NDManager: a free software suite for neurophysiological data processing and visualization. Journal of neuroscience methods. 2006; 155(2):207–216.

Hennig MH, Hurwitz C, Sorbaro M. Scaling Spike Detection and Sorting for Next-Generation Electrophysiology. In: *In Vitro Neuronal Networks* Springer; 2019.p. 171–184.

Henze D, Harris K, Borhegyi Z, Csicsvari J, Mamiya A, Hirase H, Sirota A, Buzsáki G, Simultaneous intracellular and extracellular recordings from hippocampus region CA1 of anesthetized rats. CRCNS.org; 2009. doi: http://dx.doi.org/10.6080/K02Z13FP, data available from https://crcns.org/data-sets/hc/hc-1.

Henze DA, Borhegyi Z, Csicsvari J, Mamiya A, Harris KD, Buzsaki G. Intracellular features predicted by extracellular recordings in the hippocampus in vivo. J Neurophysiol. 2000; 84(1):390–400.

Hilgen G, Sorbaro M, Pirmoradian S, Muthmann JO, Kepiro IE, Ullo S, Ramirez CJ, Encinas AP, Maccione A, Berdondini L, Murino V, Sona D, Cella ZF, Sernagor E, Hennig MH. Unsupervised spike sorting for large-scale, high-density multielectrode arrays. Cell Rep. 2017; 18(10):2521–2532.

Hill DN, Mehta SB, Kleinfeld D. Quality metrics to accompany spike sorting of extracellular signals. J Neurosci. 2011; 31(24):8699–8705. doi: 10.1523/JNEUROSCI.0971-11.2011.

Hines ML, Carnevale NT. The NEURON simulation environment. Neural Comput. 1997; 9(6):1179–1209.

Hubel DH. Tungsten Microelectrode for Recording from Single Units. Science. 1957; 125(3247):549–550.

Jun JJ, Magland JF, Mitelut C, Barnett AH, IronClust: Scalable and drift-resistant spike sorting for long-duration, high-channel count recordings; 2020. In preparation.

Jun JJ, Steinmetz NA, Siegle JH, Denman DJ, Bauza M, Barbarits B, Lee AK, Anastassiou CA, Andrei A, Aydın Ç, Barbic M, Blanche T, Bonin V, Couto J, Dutta B, Gratiy S, Gutnisky D, Häusser M, Karsh B, Harris TD. Fully integrated silicon probes for high-density recording of neural activity. Nature. 2017; 551(7679):232.

Jun JJ, Mitelut C, Lai C, Gratiy S, Anastassiou C, Harris TD. Real-time spike sorting platform for high-density extracellular probes with ground-truth validation and drift correction. bioRxiv. 2017; doi: 10.1101/101030.

Kampff A, Ultra Dense Extracellular Survey (website); 2018. http://www.kampff-lab.org/ultra-dense-survey.

Kuo JT, Kim BJ, Hara SA, Lee CD, Gutierrez CA, Hoang TQ, Meng E. Novel flexible Parylene neural probe with 3D sheath structure for enhancing tissue integration. Lab on a Chip. 2013; 13(4):554–561.

Kurtzer GM, Sochat V, Bauer MW. Singularity: Scientific containers for mobility of compute. PLoS One. 2017; 12(5):e0177459.

Lee JH, Carlson DE, Razaghi HS, Yao W, Goetz GA, Hagen E, Batty E, Chichilnisky EJ, Einevoll GT, Paninski L. YASS: yet another spike sorter. In: *Adv. Neural. Inf. Process. Syst. (NeurIPS)*; 2017. p. 4002–4012.

Lefebvre B, Yger P, Marre O. Recent progress in multi-electrode spike sorting methods. J Physiol Paris. 2016; 110(4):327–335.

Lewicki MS. A review of methods for spike sorting: the detection and classification of neural action potentials. Netw Comput Neural Syst. 1998; 9:R53–R78.

Li PH, Gauthier JL, Schiff M, Sher A, Ahn D, Field GD, Greschner M, Callaway EM, Litke AM, Chichilnisky EJ. Anatomical Identification of Extracellularly Recorded Cells in Large-Scale Multielectrode Recordings. J Neurosci. 2015; 35(11):4663–4675. http://www.jneurosci.org/cgi/doi/10.1523/JNEUROSCI.3675-14.2015, doi: 10.1523/JNEUROSCI.3675-14.2015.

Litke AM, Bezayiff N, Chichilnisky EJ, Cunningham W, Dabrowski W, Grillo AA, Grivich M, Grybos P, Hottowy P, Kachiguine S, Kalmar RS, Mathieson K, Petrusca D, Rahman M, Sher A. What does the eye tell the brain? Development of a system for the large scale recording of retinal output activity. IEEE T Nucl Sci. 2004; 51(4):1434–1440.

Marques-Smith A, Neto JP, Lopes G, Nogueira J, Calcaterra L, Frazão J, Kim D, Phillips MG, Dimitriadis G, Kampff A. Recording from the same neuron with high-density CMOS probes and patch-clamp: a ground-truth dataset and an experiment in collaboration. bioRxiv. 2018; doi: 10.1101/370080.

Marques-Smith A, Neto JP, Lopes G, Nogueira J, Calcaterra L, Frazão J, Kim D, Phillips MG, Dimitriadis G, Kampff A, Simultaneous patch-clamp and dense CMOS probe extracellular recordings from the same cortical neuron in anaesthetized rats. CRCNS.org; 2018. doi: http://dx.doi.org/10.6080/K0J67F4T, data available from http://dx.doi.org/10.6080/K0J67F4T.

Marre O, Amodei D, Deshmukh N, Sadeghi K, Soo F, Holy TE, Berry II MJ. Mapping a Complete Neural Population in the Retina. J Neurosci. 2012; 32(43):14859–14873.

Merkel D. Docker: lightweight Linux containers for consistent development and deployment. Linux J. 2014; 2014(239):2. doi: 10.5555/2600239.

Mitelut CC, Spikesortingtest (archived website); 2016–2019. https://web.archive.org/web/20190511041642/http://spikesortingtest.com.

Mitelut CC. Characterizing single neuron activity patterns and dynamics using multi-scale spontaneous neuronal activity recordings of cat and mouse cortex. PhD thesis, University of British Columbia; 2017.

Neto JP, Lopes G, Frazão J, Nogueira J, Lacerda P, Baião P, Aarts A, Andrei A, Musa S, Fortunato E, Barquinha P, Kampff A. Validating silicon polytrodes with paired juxtacellular recordings: method and dataset. J Neurophysiol. 2016; 116(2):892–903. doi: 10.1152/jn.00103.2016.

Neymotin SA, Lytton WW, Olypher AV, Fenton AA. Measuring the quality of neuronal identification in ensemble recordings. J Neurosci. 2011; 31(45):16398–16409. doi: 10.1523/JNEUROSCI.4053-11.2011.

Nicolelis MAL, Ghazanfar AA, Faggin B, Votaw S, Oliveira LMO. Reconstructing the Engram: Simultaneous, Multisite, Many Single Neuron Recordings. Neuron. 1997; 18:529–537.

Pachitariu M, Steinmetz NA, Colonell J, Kilosort2 (software); 2018–2019. https://github.com/MouseLand/Kilosort2.

Pachitariu M, Steinmetz NA, Kadir SN, Carandini M, Harris KD. Fast and accurate spike sorting of high-channel count probes with KiloSort. In: *Adv. Neural. Inf. Process. Syst. (NeurIPS)*; 2016. p. 4448–4456.

Pillow JW, Shlens J, Chichilnisky EJ, Simoncelli EP. A Model-Based Spike Sorting Algorithm for Removing Correlation Artifacts in Multi-Neuron Recordings. PLoS One. 2013; 8(5):e62123.

Pouzat C, Mazor O, Laurent G. Using noise signature to optimize spike-sorting and to assess neuronal classification quality. J Neurosci Meth. 2002; 122:43–57.

Prentice JS, Homann J, Simmons KD, Tkačik G, Balasubramanian V, Nelson PC. Fast, scalable, Bayesian spike identification and multi-electrode arrays. PLoS One. 2011; 6(7):e19884. doi: 10.1371/journal.pone.0019884.

Quiroga RQ. Spike sorting. Curr Biol. 2012; 22(2):R45–R46.

Quiroga RQ, Nadasdy Z, Ben-Shaul Y. Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering. Neural Comput. 2004; 16(8):1661–1687.

Rey HG, Pedreira C, Quiroga RQ. Past, present and future of spike sorting techniques. Brain Res Bull. 2015; 119:106–117.

Rossant C, Kadir SN, Goodman DF, Schulman J, Hunter ML, Saleem AB, Grosmark A, Belluscio M, Denfield GH, Ecker AS, Tolias A, Solomon S, Buzsáki G, Carandini M, Harris KD. Spike sorting for large, dense electrode arrays. Nat Neurosci. 2016; 19(4):634. doi: 10.1038/nn.4268.

Schmitzer-Torbert N, Jackson J, Henze D, Harris K, Redish AD. Quantitative measures of cluster quality for use in extracellular recordings. Neuroscience. 2005; 131(1):1–11. doi: 10.1016/j.neuroscience.2004.09.066.

Seymour JP, Wu F, Wise KD, Yoon E. State-of-the-art MEMS and microsystem tools for brain research. Microsyst Nanoeng. 2017; 3:16066.

Sivagnanam S, Majumdar A, Yoshimoto K, Astakhov V, Bandrowski A, Martone ME, Carnevale NT. Introducing the Neuroscience Gateway. In: *IWSG, volume 993 of CEUR Workshop Proceedings*; 2013. p. 10. http://ceur-ws.org.

Spampinato G, Esposito E, Yger P, Duebel J, Picaud S, Marre O, Ground truth recordings for validation of spike sorting algorithms. Zenodo; 2018. doi: http://dx.doi.org/10.5281/zenodo.1205233, data available from https://zenodo.org/record/1205233#.Xc8fdTJKh3k.

Steinmetz N, Sorting Comparison (webpage); 2015–2018. http://phy.cortexlab.net/data/sortingComparison.

Takahashi S, Sakurai Y, Tsukada M, Anzai Y. Classification of neuronal activities from tetrode recordings using independent component analysis. Neurocomputing. 2002; 49:289–298.

Tsai D, Sawyer D, Bradd A, Yuste A, Shepard KL. A very large-scale microelectrode array for cellular-resolution electrophysiology. Nat Commun. 2017; 8:1802.

Wiwie C, Baumbach J, Röttger R. Comparing the performance of biomedical clustering methods. Nat Methods. 2015; 12(11):1033–1038.

Wouters J, Kloosterman F, Bertrand A. SHYBRID: A graphical tool for generating hybrid ground-truth spiking data for evaluating spike sorting performance. bioRxiv. 2019; doi: 10.1101/734061.

Yger P, Spampinato GL, Esposito E, Lefebvre B, Deny S, Gardella C, Stimberg M, Jetter F, Zeck G, Picaud S, Duebel J, Marre O. A spike sorting toolbox for up to thousands of electrodes validated with ground truth recordings in vitro and in vivo. eLife. 2018; 7:e34518.

Yuan X, Kim S, Juyon J, D'Urbino M, Bullmann T, Chen Y, Stettler A, Hierlemann A, Frey U. A microelectrode array with 8,640 electrodes enabling simultaneous full-frame readout at 6.5 kfps and 112-channel switch-matrix readout at 20 kS/s. In: *2016 IEEE Symposium on VLSI Circuits (VLSI-Circuits)*; 2016. p. 1–2. doi: 10.1109/VLSIC.2016.7573558.

Zaki MJ, Meira Jr W. Data mining and analysis: fundamental concepts and algorithms. Cambridge University Press, New York, NY; 2014.