Proceedings of CONF-MLA 2026 Symposium: Intelligent Systems and Automation: AI, IoT, Robotic Engineering & Algorithm
DOI: 10.54254/2755-2721/2026.LD35136

Transparent and Reproducible Spike Sorting: Baseline Construction and Experimental Analysis for Simulated Neural Signals

Hongda Chen

School of Computer Science and Cybersecurity (Demonstrative Software Institute), Chengdu University of Technology, Chengdu, China
chenhd1213@163.com

Abstract. Spike sorting is a fundamental step in extracellular neural signal analysis, but educational implementations are often difficult to inspect and reproduce because processing assumptions, parameter choices, and evaluation procedures are distributed across multiple stages. This paper constructs a transparent and reproducible baseline workflow for simulated neural signals rather than proposing a new sorting algorithm. The workflow integrates band-pass filtering, threshold-based spike detection, fixed-window waveform extraction, principal component analysis, KMeans clustering, one-to-one temporal matching, and automated metric export in a configurable Python implementation. Experiments are conducted on repository-generated synthetic datasets with easy, medium, and hard noise settings and on the Wave_Clus Easy1 noise series. Detection F1-score decreases from 0.812 to 0.457 as the synthetic setting becomes harder, while Wave_Clus Easy1 results range from 0.340 to 0.907 across noise conditions. The low clustering scores observed in several settings further show that accurate event detection does not necessarily imply reliable unit separation. The study contributes an auditable baseline, a reproducible evaluation process, and a teaching-oriented example that makes the distinction between detection and clustering performance explicit.

Keywords: Spike sorting, Reproducibility, Threshold detection, PCA, Baseline workflow

1. Introduction

Extracellular recording provides a practical means of observing the electrical activity of neuronal populations, and spike sorting is the basic process that converts recorded voltage traces into candidate spike events and putative neuronal units [1]. Reliable spike-train analysis depends on both accurate event detection and credible unit assignment [2]. However, waveform overlap, recording noise, parameter sensitivity, and the absence of transparent evaluation procedures make the task difficult to reproduce, particularly for students and researchers who need to understand how each processing decision affects the final result [3]. Classical pipelines remain useful in this setting because their assumptions can be inspected directly, but many implementations do not provide a single auditable path from signal loading to metric export.

DOI: 10.54254/2755-2721/2026.LD35136

This study addresses that reproducibility gap by constructing a transparent Python baseline. In this paper, the term repository refers to the complete collection of source code, configurations, datasets, validation logs, metrics, tables, and figures used to reproduce the reported workflow. The implementation separates filtering, detection, waveform extraction, feature reduction, clustering, matching, and evaluation into configurable stages. It also exports intermediate and final artifacts automatically, allowing users to trace numerical claims to generated files. This design supports reproducible benchmarking practices [4] and provides an accessible entry point to unified spike sorting workflows [5].

The experiments combine repository-generated synthetic recordings with the Wave_Clus Easy1 series. Synthetic data provides controlled easy, medium, and hard conditions for debugging and sensitivity analysis, whereas Wave_Clus provides externally developed simulated recordings with ground-truth spike information [6]. The latter Wave_Clus implementation also provides relevant software and automation context [7]. The main contributions are therefore threefold: an end-to-end transparent baseline, an evidence-linked evaluation workflow, and an empirical demonstration that detection quality and clustering quality can diverge under increasing noise.

2. Background and related work

2.1. Classical spike sorting

Classical spike sorting follows a sequence of signal preprocessing, threshold-based event detection, waveform extraction, feature reduction, clustering, and quantitative evaluation [1]. Threshold detection provides an interpretable mechanism for locating candidate events, while a refractory interval suppresses duplicate detections [6]. PCA reduces waveform dimensionality so that dominant shape variation can be represented compactly, and KMeans supplies a simple clustering baseline with explicit cluster assignments [8]. This combination is not intended to model every form of waveform variability; its value is that each stage and parameter can be inspected, reproduced, and evaluated independently [3].

2.2. Reproducibility and benchmark data

Benchmark data connect reproducible implementation with measurable performance because known spike times and labels permit direct comparison between detections, clusters, and ground truth. Wave_Clus simulated recordings have therefore been used in prior spike sorting evaluations [6]. Related evaluations have examined how recording conditions affect the number of separable neurons [9]. MEArec provides a customizable framework for generating ground-truth extracellular activity [10]. Reproducible benchmark platforms such as SpikeForest make sorter configurations and validation results easier to compare [4]. SpikeInterface provides a unified framework for executing and evaluating spike sorting workflows [5]. In this paper, detection performance is summarized by precision, recall, and F1-score, whereas clustering agreement is summarized by accuracy, adjusted Rand index (ARI), and normalized mutual information (NMI). ARI corrects the partition agreement for chance [11]. NMI measures shared information between predicted clusters and reference labels [12]. Modern systems address dense electrode arrays [13]. Other toolboxes support ground-truth-validated large-scale sorting [14]. Recent template-based methods include Kilosort4 [15]. Fully automated processing is another important comparison direction [16].

DOI: 10.54254/2755-2721/2026.LD35136

3. Methodology

3.1. Data loading and preprocessing

The pipeline supports repository-generated synthetic .npz recordings and Wave_Clus .mat recordings. Each input is standardized into a signal, ground-truth spike times, optional spike labels, sampling rate, and metadata before processing. The voltage trace is then band-pass filtered from 300 Hz to 3000 Hz to suppress low-frequency drift and high-frequency noise while retaining the principal spike-frequency band. Dataset validation is performed before experimentation so that missing arrays, unsupported formats, or unclear keys are reported rather than silently inferred.

3.2. Detection, feature extraction, clustering, and evaluation

Candidate spikes are detected from the filtered signal using a negative-polarity threshold with multiplier 3 and a 1 ms refractory interval. For every valid detection, the pipeline extracts a waveform containing 20 samples before and 20 samples after the detected index; detections too close to a signal boundary are rejected because a complete waveform cannot be formed. PCA then reduces each waveform to two components, and KMeans with k=2 assigns the resulting feature vectors to clusters. Detected spikes are paired with ground-truth events through one-to-one temporal matching with a 100-sample tolerance. This constraint is necessary because it prevents multiple detections from receiving credit for the same reference spike. Precision measures the proportion of detections that are matched, recall measures the proportion of ground-truth events recovered, and F1-score summarizes their balance. When matched unit labels are available, accuracy, ARI, and NMI quantify clustering agreement [11]. NMI provides a complementary information-theoretic measure of cluster-label consistency [12].

4. Experimental results and analysis

4.1. Experimental setup

The evaluation uses three synthetic datasets at 10000 Hz, representing easy, medium, and hard noise conditions, and five Wave_Clus Easy1 datasets at 24000 Hz, representing noise01 through noise05. All reported runs use a threshold multiplier of 3, negative-polarity detection, two PCA components, K-Means k=2, and a 100-sample matching tolerance. The verification grid processes at most 120000 samples and at most six parameter combinations per dataset, producing 18 synthetic runs and 30 Easy1 runs. This bounded design supports efficient pipeline verification and parameter comparison, but it is not an exhaustive hyperparameter search.

4.2. Results and discussion

Tables 1 and 2 summarize the core quantitative results. On the synthetic datasets, F1-score decreases from 0.812 on easy data to 0.675 on medium data and 0.457 on hard data, showing that event recovery deteriorates as the generated noise condition becomes more difficult. ARI and NMI remain low across all three conditions, indicating that the PCA-KMeans stage does not recover strong unit-label separation even when detection recall is high. On Wave_Clus Easy1, F1-score reaches 0.907 on noise01, falls to 0.340 on noise04, and returns to 0.831 on noise05. The noise05 configuration combines high recall with lower precision and zero ARI/NMI, which confirms that successful event

DOI: 10.54254/2755-2721/2026.LD35136

detection and successful cluster recovery are distinct objectives. Mean runtime for each six-run Easy1 summary remains below 0.13 s in the limited, truncated-sample grid.

Table 1. Performance on synthetic datasets

Dataset	Precision	Recall	F1-score	Accuracy	ARI	NMI
easy	0.700	0.966	0.812	0.413	0.004	0.010
medium	0.671	0.679	0.675	0.459	0.023	0.076
hard	0.601	0.368	0.457	0.463	0.035	0.042

Table 2. Performance on Wave_Clus Easy1 datasets

Dataset	Precision	Recall	F1-score	Accuracy	ARI	NMI
noise01	0.854	0.968	0.907	0.528	0.189	0.263
noise02	0.829	0.794	0.811	0.601	0.213	0.308
noise03	0.814	0.469	0.595	0.632	0.219	0.243
noise04	0.650	0.230	0.340	0.538	0.044	0.041
noise05	0.731	0.965	0.831	0.389	0.000	0.000

The figures provide complementary visual evidence: Figures 1-3 show the synthetic recordings under increasing difficulty, and Figure 4 summarizes the Easy1 F1-score across noise levels. Together with the tables, these results support three conclusions. First, higher noise generally reduces detection performance, although individual Easy1 conditions do not form a strictly monotonic sequence. Second, PCA-K-Means has a limited capacity to separate overlapping or variable waveforms in the tested configuration. Third, detection metrics and clustering metrics must be interpreted separately because strong recall can coexist with weak cluster-label agreement. The principal value of the study is therefore not competitive sorter performance, but a transparent, reproducible, and teaching-oriented baseline in which every processing stage and reported metric can be audited.

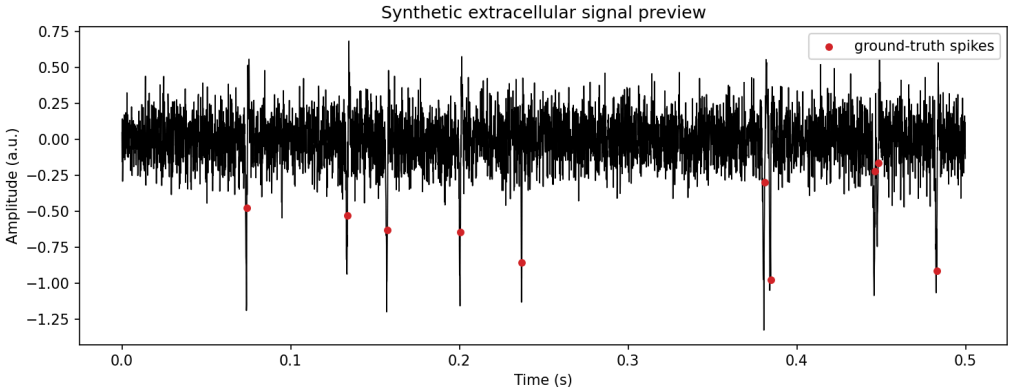

Figure 1. Synthetic easy extracellular signal preview

DOI: 10.54254/2755-2721/2026.LD35136

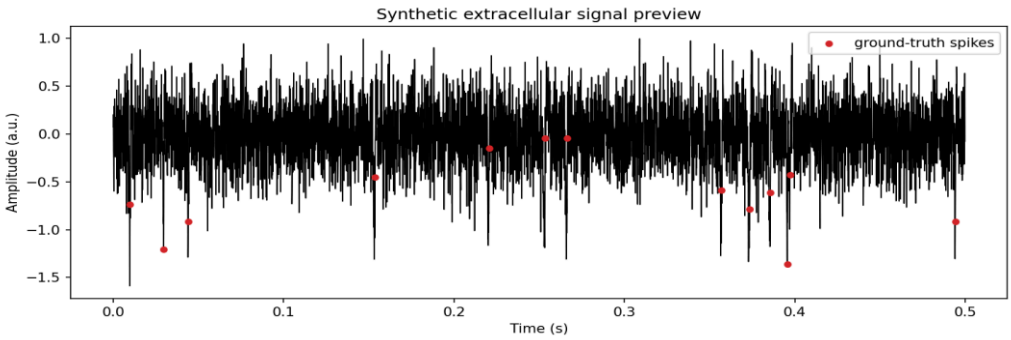

Figure 2. Synthetic medium extracellular signal preview

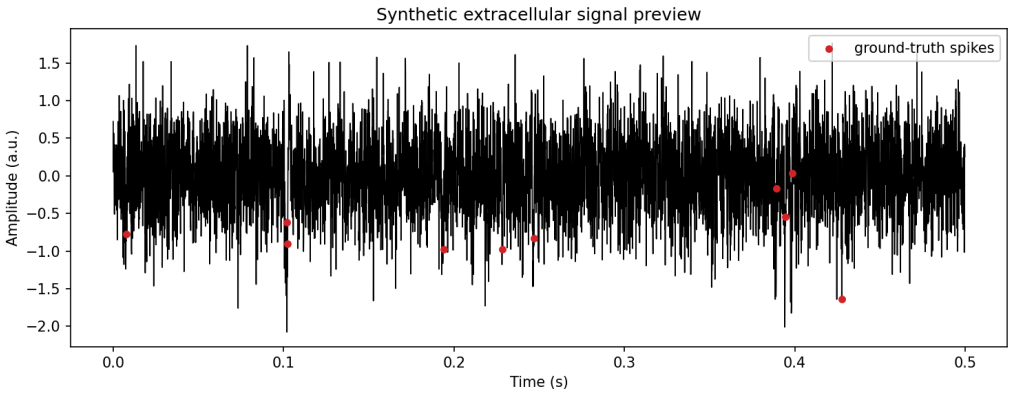

Figure 3. Synthetic hard extracellular signal preview

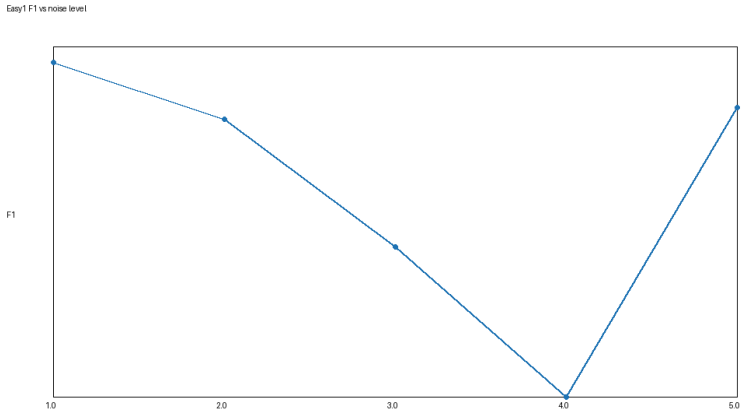

Figure 4. F1-score variation across the Wave_Clus Easy1 noise series

5. Conclusion

This paper investigated how a transparent classical pipeline can support reproducible spike sorting experiments on simulated extracellular neural signals. The resulting Python workflow integrates validated data loading, band-pass filtering, threshold-based detection, fixed-window waveform extraction, PCA feature reduction, KMeans clustering, one-to-one temporal matching, and automated metric export. Experiments on synthetic easy, medium, and hard conditions and on the

DOI: 10.54254/2755-2721/2026.LD35136

Wave_Clus Easy1 series show that detection performance is sensitive to noise and parameter settings. Synthetic F1-score decreases from 0.812 to 0.457 as the task becomes harder, while the Easy1 experiments produce F1-scores between 0.340 and 0.907. More importantly, low ARI and NMI values in several conditions demonstrate that recovering spike events does not guarantee reliable separation of neuronal units. This directly supports the paper's central argument that transparent evaluation must distinguish detection quality from clustering quality.

The study is limited by its use of simulated data, a finite verification grid, and a classical PCA-KMeans model. The conclusions should therefore not be generalized to all extracellular recording conditions or interpreted as a benchmark against state-of-the-art sorters. The exact interpretation of several Wave_Clus fields is based on repository validation logic rather than independently verified dataset documentation. In addition, the available Frontiers material does not include supported raw neural recordings, so no real-data experiment is reported. Future work should expand the parameter search, validate the pipeline on runnable real recordings with ground truth where possible, and compare the baseline with modern large-array, template-based, and automated spike sorting systems. Within its intended scope, however, the workflow provides a reproducible educational reference and an auditable foundation for more advanced experiments.

References

[1] H. G. Rey, C. Pedreira, and R. Quian Quiroga, "Past, present and future of spike sorting techniques, " Brain Research Bulletin, vol. 119, pp. 106-117, 2015, doi: 10.1016/j.brainresbull.2015.04.007.

[2] G. T. Einevoll, F. Franke, E. Hagen, C. Pouzat, and K. D. Harris, "Towards reliable spike-train recordings from thousands of neurons with multielectrodes, " Current Opinion in Neurobiology, vol. 22, no. 1, pp. 11-17, 2012, doi: 10.1016/j.conb.2011.10.001.

[3] M. S. Lewicki, "A review of methods for spike sorting: the detection and classification of neural action potentials, " Network: Computation in Neural Systems, vol. 9, no. 4, pp. R53-R78, 1998, doi: 10.1088/0954-898X/9/4/001.

[4] J. Magland et al., "SpikeForest, reproducible web-facing ground-truth validation of automated neural spike sorters, " eLife, vol. 9, Art. no. e55167, 2020, doi: 10.7554/eLife.55167.

[5] A. P. Buccino et al., "SpikeInterface, a unified framework for spike sorting, " eLife, vol. 9, Art. no. e61834, 2020, doi: 10.7554/eLife.61834.

[6] R. Quian Quiroga, Z. Nadasdy, and Y. Ben-Shaul, "Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering, " Neural Computation, vol. 16, no. 8, pp. 1661-1687, 2004, doi: 10.1162/089976604774201631.

[7] F. J. Chaure, H. G. Rey, and R. Quian Quiroga, "A novel and fully automatic spike-sorting implementation with variable number of features, " Journal of Neurophysiology, vol. 120, no. 4, pp. 1859-1871, 2018, doi: 10.1152/jn.00339.2018.

[8] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations, " in Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, 1967, pp. 281-297. [Online]. Available: https: //projecteuclid.org/euclid.bsmsp/1200512992

[9] C. Pedreira, J. I. Martinez, M. J. Ison, and R. Quian Quiroga, "How many neurons can we see with current spike sorting algorithms?, " Journal of Neuroscience Methods, vol. 211, no. 1, pp. 58-65, 2012, doi: 10.1016/j.jneumeth.2012.07.010.

[10] A. P. Buccino and G. T. Einevoll, "MEArec: A fast and customizable testbench simulator for ground-truth extracellular spiking activity, " Neuroinformatics, vol. 19, no. 1, pp. 185-204, 2021, doi: 10.1007/s12021-020-09467-7.

[11] L. Hubert and P. Arabie, "Comparing partitions, " Journal of Classification, vol. 2, no. 1, pp. 193-218, 1985, doi: 10.1007/BF01908075.

[12] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: variants, properties, normalization and correction for chance, " Journal of Machine Learning Research, vol. 11, pp. 2837-2854, 2010. [Online]. Available: https: //www.jmlr.org/papers/v11/vinh10a.html

[13] C. Rossant et al., "Spike sorting for large, dense electrode arrays, " Nature Neuroscience, vol. 19, no. 4, pp. 634-641, 2016, doi: 10.1038/nn.4268.

DOI: 10.54254/2755-2721/2026.LD35136

[14] P. Yger et al., "A spike sorting toolbox for up to thousands of electrodes validated with ground truth recordings in vitro and in vivo, " eLife, vol. 7, Art. no. e34518, 2018, doi: 10.7554/eLife.34518.
[15] M. Pachitariu, S. Sridhar, J. Pennington, and M. Carandini, "Spike sorting with Kilosort4, " Nature Methods, vol. 21, pp. 914-921, 2024, doi: 10.1038/s41592-024-02232-7.
[16] J. E. Chung et al., "A fully automated approach to spike sorting, " Neuron, vol. 95, no. 6, pp. 1381-1394.e6, 2017, doi: 10.1016/j.neuron.2017.08.030.