A Modular Implementation to Handle and Benchmark Drift Correction for High-Density Extracellular Recordings

Samuel Garcia, Charlie Windolf, Julien Boussard, Benjamin Dichter, Alessio P. Buccino, Pierre Yger

▶ **To cite this version:**

Samuel Garcia, Charlie Windolf, Julien Boussard, Benjamin Dichter, Alessio P. Buccino, et al.. A Modular Implementation to Handle and Benchmark Drift Correction for High-Density Extracellular Recordings. eNeuro, 2024, 11 (2), pp.ENEURO.0229 - 23.2023. 10.1523/eneuro.0229-23.2023 . hal-04546246

HAL Id: hal-04546246
https://hal.science/hal-04546246

Submitted on 15 Apr 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

eNeuro

Research Article: New Research
Novel Tools and Methods

A Modular Implementation to Handle and Benchmark Drift Correction for High-Density Extracellular Recordings

Samuel Garcia,[1] Charlie Windolf,[2] Julien Boussard,[2] Benjamin Dichter,[3] Alessio P. Buccino,[3,4]* and Pierre Yger[5,6]*

[1]Centre de Recherche en Neuroscience de Lyon, CNRS, Lyon 69675, France, [2]Columbia University, New York, New York 10027, [3]CatalystNeuro, Benicia, California 94510, [4]Allen Institute for Neural Dynamics, Seattle, Washington 98109, [5]Institut de la Vision, Sorbonne Université, INSERM, Paris 75012, France, and [6]Lille Neurosciences & Cognition (lilNCog)—U1172 (INSERM, Lille), Univ Lille, Centre Hospitalier Universitaire de Lille, Lille 59800, France

Abstract

High-density neural devices are now offering the possibility to record from neuronal populations in vivo at unprecedented scale. However, the mechanical drifts often observed in these recordings are currently a major issue for "spike sorting," an essential analysis step to identify the activity of single neurons from extracellular signals. Although several strategies have been proposed to compensate for such drifts, the lack of proper benchmarks makes it hard to assess the quality and effectiveness of motion correction. In this paper, we present a benchmark study to precisely and quantitatively evaluate the performance of several state-of-the-art motion correction algorithms introduced in the literature. Using simulated recordings with induced drifts, we dissect the origins of the errors performed while applying a motion correction algorithm as a preprocessing step in the spike sorting pipeline. We show how important it is to properly estimate the positions of the neurons from extracellular traces in order to correctly estimate the probe motion, compare several interpolation procedures, and highlight what are the current limits for motion correction approaches.

Key words: benchmark; drift; electrophysiology; ground-truth; neuropixel; spike sorting

Significance Statement

High-density extracellular recordings allow experimentalists to get access to the spiking activity of large neuronal populations, via the procedure of spike sorting. However, it is widely known that spike sorters are affected by *drifts*, i.e., the fact that neurons move with respect to the recording electrodes. While several algorithms have been proposed to handle drifts, a systematic comparison on the performance of these algorithms is still lacking. In this contribution, we performed a large comparison study to benchmark and understand the limitations of state-of-the-art drift correction methods. Our results suggest that they all have some intrinsic limitations that should be taken into account by analysis pipelines.

Introduction

Recording from increasingly larger neuronal populations is a crucial challenge to unravel how information is processed by the brain. The recent development of complementary metal-oxide semiconductor-based high-density multi-electrode arrays (HD-MEAs), both for in vitro (Berdondini et al., 2009; Frey et al., 2009) and in vivo applications, such as Neuropixels probes (Jun et al., 2017; Angotzi et al., 2019; Steinmetz et al., 2021), has dramatically boosted the yield of extracellular electrophysiology experiments. To maximize the return from these novel probes, the "spike sorting" step, i.e., the process which

Received June 29, 2023; revised Dec. 20, 2023; accepted Dec. 21, 2023.

The authors declare no competing financial interests. The research reported here has no relation with the economic activities of the company Hyland Switzerland Sarl, for which it is currently working.

S.G., A.P.B., and P.Y. designed research; S.G., A.P.B., and P.Y. performed research; S.G., A.P.B., and P.Y. analyzed data; S.G., C.W., J.B., B.D., A.P.B., and P.Y. wrote the paper.

We thank Nick Steinmetz and Maxime Juventin for acquiring and sharing the experimental data used in Fig. 1.

*A.P.B. and P.Y. share senior authorship.

Correspondence should be addressed to Pierre Y ger at pierre.yger@inserm.fr.

Copyright © 2024 Garcia et al. This is an open-access article distributed under the terms of the Creative Commons Attribution 4.0 International license, which permits unrestricted use, distribution and reproduction in any medium provided that the original work is properly attributed.

extracts single neuron activities from the recorded traces, has also undergone major advances both in algorithmic development (Lefebvre et al., 2016; Pachitariu et al., 2016; Chung et al., 2017; Yger et al., 2018; Lee et al., 2020; Magland et al., 2020; Steinmetz et al., 2021; Buccino et al., 2022; Pachitariu et al., 2023) and quantitative benchmarks (Buccino et al., 2020; Magland et al., 2020; Garcia et al., 2022).

The layout of HD probes for in vivo applications usually consists of a long shank (e.g., ~1 mm for Neuropixels 1.0; Jun et al., 2017) to allow the electrodes to span multiple regions of the brain and reach deep structures. Due to the different mechanical properties of the probe and the brain tissue, it is very common to observe a relative movement of the tissue with respect to the probe. This phenomenon is known as *drift*. The origins and types of such drifts can be diverse. For example, cells are likely to slowly drift from initial positions because of the pressure release in the tissue after an acute probe insertion; abrupt and discontinuous drift events could be caused by sudden rig instabilities and movement artifacts. When a neuron moves with respect to the recording electrodes, its waveforms are distorted (Fig. 1*A*), challenging the operation of spike sorting algorithms which mainly rely on waveform similarities to cluster different neurons in the recordings.

Due to the spatial regularities of the recordings, which makes drifts relatively coherent in space (Fig. 1*B* shows a *raster map* of a recording with an induced triangular probe movement; Steinmetz, 2021), a common approach is to mimic drift correction procedures that have been applied in the field of calcium imaging (Greenberg and Kerr, 2009). First, motion is estimated from the spiking activity extracted from the recording (Fig. 1*C*, left). Then, the estimate is used to interpolate or register the recording as a preprocessing step (Fig. 1*C*, right). Different methods have been recently proposed to correct drifts before spike sorting (Boussard et al., 2021; Steinmetz et al., 2021; Varol et al., 2021; Pachitariu et al., 2023), but their evaluation is mainly qualitative. A quantitative evaluation of the performance and effectiveness of different strategies is still lacking, probably due to the complexity of performing such benchmarks. To benchmark drift correction methods, one would need to know the ground-truth (GT) motion that is generating drifts. In addition, these methods are usually integrated into complicated spike sorting pipelines, and this makes it hard to track down what part of the error was really tied to the motion correction step.

In this work, we extensively benchmarked state-of-the-art methods for drift correction to solve the above-mentioned challenges. To obtain recordings with GT motion, we used biophysically detailed simulations with drifting neurons. This allowed us to systematically explore a wide range of drifting recordings of varying complexity. We further identified and broke down existing drift correction strategies into a modular architecture that allowed us to properly evaluate the performance of different methods for estimating motion, interpolating the traces, and their overall impact on spike sorting

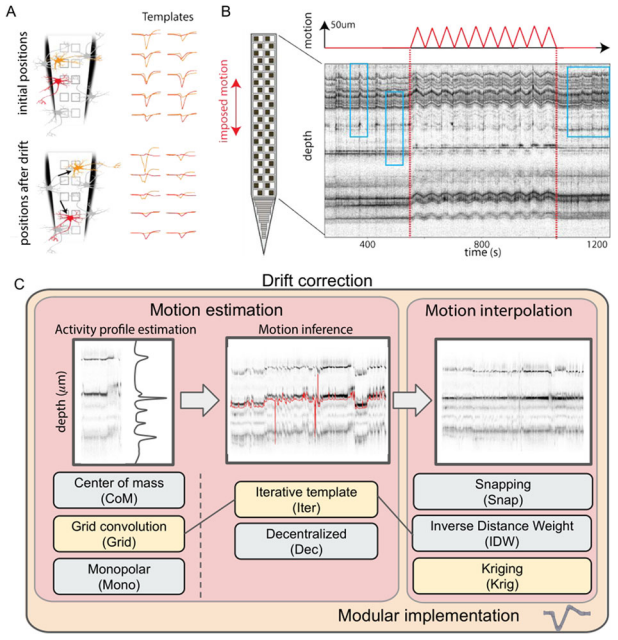

Figure 1. Common pipeline for drift correction algorithms. ***A***, Illustration of the problems raised by neuronal drifts. Two neurons (red and orange) have two distinct waveforms, but after some non-rigid drifts (movements), the templates are changed. ***B***, Illustration of the drift in vivo. A Neuropixels 1.0 probe in the cortex of the mice is moved up and down with an imposed triangular movement of amplitude 50 m (top trace; Steinmetz, 2021; Steinmetz et al., 2021)—`p1` dataset. The row shows the depth of all the peaks detected *via* signal threshold (*positional* raster plot, see text) as a function of time. When the probe starts moving (red dotted line), so does the depth of the peaks, since neurons are drifting across channels. While the drift here is imposed, spontaneous drifts can happen locally and non-homogeneously over the whole probe (see inside light blue boxes for before and after imposed motion). Panels A and B are adapted from Buccino et al. (2022), with permission. ***C***, Key algorithmic steps for the motion correction pipeline. For each step, the various implementations available in `SpikeInterface` are listed. The yellow boxes connected with arrows correspond to a `Kilosort` 2.5/3-like approach (Pachitariu et al., 2023).

outcomes. All the source code used to generate the synthetic recordings and to reproduce results and figures (including Extended Data) is available at https://github.com/samuelgarcia/spike_sorting_motion_benchmark_paper_2023.

Materials and Methods

A modular implementation of drift correction

In order to properly evaluate and benchmark the latest in vivo drift correction methods available nowadays in the literature (Varol et al., 2021; Pachitariu et al., 2023), we first identified the individual sub-components of the different approaches and implemented a modular system to be able to substitute different steps. In fact, most of the available drift correction methods acting at the preprocessing level (Boussard et al., 2021; Pachitariu et al., 2023) share similar ideas, so that the global algorithmic pipeline can be decomposed and summarized as follows (Fig. 1*C*).

First, a sweep of spike detection (with a peak detection algorithm) is performed and the putative peak depths (defined as the position along probe's insertion direction) for each action potential are estimated (this first step is usually done on a subset of peaks to speed up the computation time). The estimated depths are binned in short time windows (e.g., 2 s) to obtain an **activity profile** (Fig. 1*C*, left) as a function of depth. It is important to stress that a time window that is too small might not contain enough spikes to properly estimate the activity profile, while too large time windows might result in *blurred* activity profiles because of the drift within each time window. From the activity profiles varying in time, the **motion inference** step aims at recovering the motion signals (red line in Fig. 1*C*, middle). Finally, one can interpolate the traces with the negative motion signals to compensate for the estimated motion (**motion interpolation**, Fig. 1*C*, right).

The general pattern of these three processing stages is seen across different spike sorting pipelines, though the details and implementations of the different steps vary from algorithm to algorithm. For example, in Steinmetz et al. (2021) and Pachitariu et al. (2023), an average activity template is computed as the mean of activity histograms computed over all temporal windows and used as a reference for motion registration. Nevertheless, this *average template* strategy assumes a stationary spiking activity over the course of the recording. Recent works have proposed a decentralized method (Varol et al., 2021; Windolf et al., 2023) to relax this assumption, which uses all the pairwise displacements between activity profiles, instead of using a single reference template, to directly estimate a motion signal to register the traces.

We used the modular architecture of the `sortingcomponents` module of the `SpikeInterface` package (Buccino et al., 2020) to factorize and benchmark all the steps in a modular manner. As shown in the bottom of Figure 1*C*, we have implemented three major algorithms to estimate the positions of detected spike (see Methods), two methods to estimate the motion given these positions (the one from `Kilosort` , termed "íterative template," and the one from Boussard et al., 2021 termed "decentralized," see Methods), and three methods to interpolate the traces.

This modular implementation enables us first to benchmark the different drift correction steps in a sorter-agnostic manner. It is worth highlighting that these correction methods can be applied as a preprocessing step before any spike sorting algorithm. In addition, the modular implementation can also replicate existing drift correction strategies, including the one similar to `Kilosort` (since v2.5 (The implementation was ported from pyKilosort—https://github.com/int-brain-lab/pykilosort/); shown in yellow in Fig. 1*C*), and the motion estimation introduced in Boussard et al. (2021) and Varol et al. (2021) (note that in this case, the motion interpolation step was not used).

Motion estimation

As illustrated in Figure 1*C*, motion estimation includes two distinct steps: activity profile estimation and motion inference.

Notations

Throughout the article, vector variables are represented by the $\vec{}$ notation. We use $\vec{w}_i(t) \in \mathbb{R}^{N\times M}$ to represent the spatio-temporal waveforms emitted by the neuron i at time t, where N is the total number of channels and M is the number of time samples (by default, we cut out 2 ms signal at a sampling rate of 32 kHz, which makes $M = 64$). We further use the term GT to refer to the fully controlled variables in our synthetic recordings (either the motion signal or the spike times of the units).

Activity profile estimation (peak localization)

To estimate a histogram of the activity of the cells in a given time window τ_{win} (default is 2 s) as a function of depth, we used, in all cases, the estimated peak depths obtained via peak localization. We first detected peaks as negative threshold crossings with a spatio-temporal exclusion zone (see the "locally exclusive" method of `SpikeInterface`, with parameters *detection_threshold* $= 10$, *local_radius_um* $= 50$ m, and *exclude_sweep_ms* $= 0.2$ ms). From the detected peaks, we estimated their depths in three ways:

Center of mass (CoM). Assuming that the spike i has its waveform $\vec{w}_i(t)$ defined on several channels $c \in \{1, \ldots, N\}$, we computed the peak-to-peak values $\text{ptp}_i(c)$ on every channel (also referred to as *peak-to-trough* amplitude). Since every

channel has a physical position $\vec{p}_c = (x_c \ y_c)$ in the 2D space, we can obtain, for every spike i, its CoM(i) as

$$\text{CoM}(i) = \frac{\sum_c \text{ptp}_i(c)\vec{p}(c)}{\sum_c \text{ptp}_i(c)}. \quad (1)$$

Grid convolution (Grid). Introduced in Pachitariu et al. (2023), the idea behind this localization method is to create an exhaustive catalog $\mathcal{F}$ of artificial templates $\vec{f}_{n\in 1\dots k}(t)$ at known positions $\vec{p}_n = (x_n \ y_n)$ and to estimate the position of a given waveform $\vec{w}_i(t)$ projected onto this catalog. If the spatial resolution of this grid is finer than the one of the recording channels, one could expect to enhance the resolution of the localization estimates. First, the scalar products $\omega_{in} = \vec{w}_i^F(t)\vec{f}_n^F(t)$ of the waveform with all the templates in the catalog are computed; then, the spike position Grid(i) is estimated as a weighted sum of the positions, with weights equal to the scalar product between the waveform and these templates (negative scalar products are discarded):

$$\text{Grid}(i) = \frac{\sum_n \omega_{in}\vec{p}_n}{\sum_n \omega_{in}}. \quad (2)$$

To create the artificial templates, the typical waveform $\vec{H}(t)$ of all detected peaks on a single channel is estimated as a median over 1,000 normalized waveforms, and then duplicated on all nearby channels, with a spatial decay in amplitude σ such that on every channel c at a position $\vec{p}_c = (x_c \ y_c)$ we have:

$$\vec{f}_n(c \ t) = e^{-(\vec{p}_c - \vec{p}_n)^2/(2\sigma^2)}\vec{H}(t). \quad (3)$$

To extend the catalog, one can use multiple σ values (in the range of 10–50 m). Moreover, in order to reduce the spreading of the scalar products if too many templates are in the catalog, we only perform the estimation in Equation 2 on the top 10% of the scalar product values.

Monopolar triangulation (Mono). The idea of this method is to consider the cell as a monopolar current source (Buccino et al., 2018; Boussard et al., 2021) and infer its position by triangulation given the amplitudes of the waveforms recorded on all channels. Assuming the cell behaves as a monopolar current source, the extracellular peak-to-peak values $\text{ptp}_i(c)$ on a channel c generated by a neuron i at position $\vec{p}_i = (x_i \ y_i \ z_i)$ can be expressed as

$$\text{ptp}_i(c) = \frac{k_i}{\sqrt{(x_c - x_i)^2 + (y_c - y_i)^2 + (z_c - z_i)^2}} \quad (4)$$

with the k_i term including the magnitude of the current and propagation properties of the tissue (see Buccino et al., 2018). Therefore, to estimate the location of the spike, one can simply solve an optimization problem with the cost function $\Phi(x, y, z, k)$ and find the (x_i, y_i, z_i) estimated position (and (k_i)) from the monopolar approximation by minimization:

$$\Phi(x \ y \ z \ k) = \sum_c \left(\text{ptp}_i(c) - \frac{k}{\sqrt{(x_c - x)^2 + (y_c - y)^2 + (z_c - z)^2}} \right)^2. \quad (5)$$

Such that the spike position Mono(i) is

$$\text{Mono}(i) = \text{argmin}_{xyzk}\Phi(x \ y \ z \ k). \quad (6)$$

To minimize the cost function, we used the `scipy.optimize` (Virtanen et al., 2020) function with the Broyden–Fletcher–Goldfarb–Shanno algorithm.

Motion inference

From the activity profiles, the next step is to infer motion. Although the peak localization methods return 2D (CoM and Grid) and even 3D (Mono) locations, the motion inference step currently only uses the y coordinate, i.e., the depth of the probe. Therefore, the y estimates of the peak localization are also referred to as spike depths.

For this step, we considered two different methods to estimate the drift vectors, i.e., the putative drift at a given place in time and depth:

Iterative template (Iter). The first method is the one implemented in `Kilosort` 2.5 (Steinmetz et al., 2021; Pachitariu et al., 2023). From the activity profiles, a 3D histogram is constructed using the spike counts, spike depths, and log-transformed spike amplitudes (i.e., the values of the trace when a putative spike is detected on the channel it was detected on) as dimensions. The spike counts are binned over 2 s temporal windows, depths using 5 μm bins, and amplitudes in 20 equally distributed bins. Next, a rigid registration is computed across the time dimension by iteratively finding the shift that maximizes the correlation between each shifted temporal bin of the histogram and a target *average* template (maximum 15 shifts, each corresponding to one spatial bin of 5 μm), which is also iteratively updated (for the first iteration, an activity profile sample at the center of the recording is used). The correlation is computed as the mean of the element-wise product between the shifted temporal bin and the average template. Next, the depth is divided into sub-blocks of 50 μm to refine the estimation by taking into consideration non-rigid effects. For each block, a similar procedure, but without iteration, is performed to find the best alignment for each spatial block. All parameters were the same as the default parameters of `Kilosort2.5`. The only minor difference is that `Kilosort2.5` defines the temporal bins in number of samples (default 65'600 ~ 2.18 s), while we used 2 s.

Decentralized (Dec). The second method is from Varol et al. (2021). First, it constructs a 2D motion histogram from the activity profile, by binning peak locations in time and depth, using 2 s temporal bins and 5 μm spatial bins. The main difference between this approach and the iterative template one is that the correlation is not computed with respect to a target average template, but, instead, a correlation value is computed between each pair of temporal bins (optionally, a time *horizon* can be used to limit the correlation computations to bins within a certain interval to speed up the computation, e.g., 120 s). This results in a pairwise correlation matrix, which is then used to estimate the motion that maximizes all pairwise correlations, using the LSMR solver (for details, refer to Varol et al., 2021). To achieve non-rigid motion estimation, the same approach is applied on spatial blocks of 50 μm and a spatial prior is used to enforce smoothness between consecutive spatial blocks (Windolf et al., 2023). Note that differently from the *Iter* method, where first a global inference is performed, followed by a non-rigid refinement, here the motion of each spatial block is inferred separately, and a final smoothing operation between blocks is applied.

Motion interpolation

Once the drift vectors have been estimated, the next step of the pipeline is to interpolate the extracellular traces. Given all the values of the signals $s_c(t)$ at channels c at positions $\vec{p}_c = (x_c \ y_c)$, the goal is to determine the value $s(t)$ of the extracellular traces at a drift-corrected position $\vec{p} = (x \ y)$ which accounts for the estimated motion. Here again, several methods to interpolate the extracellular traces from the motion vector have been implemented. To be more efficient, interpolation kernels are computed at the temporal resolution of the time histograms used to compute the activity profiles in a chunk-wise manner, not at every time point.

Snapping (Snap). In this case, given an estimated position $\vec{p}$, the value of the extracellular traces at the closest electrode from the drift vector is used:

$$s(t) = s_{c^*}(t) \tag{7}$$

with c^* such that we have $\min_c \|\vec{p} - \vec{p}_c\|_2$.

Inverse distance weighting. The interpolated value is a weighted sum of the extracellular values on the channels in the vicinity of the estimated position $\vec{p}$, with weights determined by the distances:

$$s(t) = \frac{\sum_c w_c s_c(t)}{\sum_c w_c} \tag{8}$$

with $w_c = \|\vec{p} - \vec{p}_c\|_2$. In practice, this weighed sum is restricted to the three nearest channels.

Kriging (Krig). Finally, we also implemented the kriging method, as used in `Kilosort` (Pachitariu et al., 2023). To be more precise, we computed K_{xx} as the negatively exponentiated distance matrix between all the positions of the known channels $\vec{p}_c$. To account for the different x and y scales of the Neuropixels layout, `Kilosort` uses different scaling factors σ_x and σ_y, and we reuse the exact same formula for comparison purposes. However, a more generic framework has been implemented in `SpikeInterface` to account for the general distances. The distance considered for K_{xx} in this paper is thus a city-block distance between all channels i, j such that:

$$K_{xx} = e^{-|x_i - x_j|/\sigma_x - |y_i - y_j|/\sigma_y}. \tag{9}$$

In practice, we set $\sigma_x = 20\,\mu m$ and $\sigma_y = 30\,\mu m$. Then we computed K_{yx} also as the negatively exponentiated distance matrix between $\vec{p}$ and all the channels $\vec{p}_c$. The kriging kernel W is obtained as

$$W = K_{yx}(K_{xx} + 0.01 \cdot I)^{-1}. \quad (10)$$

The kernel is sparsified so that all values below $\epsilon = 0.001$ are set to 0 and normalized so that each column of W sums up to 1.

Simulated datasets

We used the `MEArec` simulator (Buccino and Einevoll, 2020; version 1.9.0) to generate 10-min-long synthetic GT recordings. `MEArec` uses biophysically detailed multicompartment models to simulate extracellular action potentials, or so-called "templates." For this study, we used 13 *default* cell models from layer 5 of a juvenile rat somatosensory cortex (Markram et al., 2015; Ramaswamy et al., 2015) to simulate a dictionary of biologically plausible templates on a Neuropixels probe layout (cropped to 128 electrodes in 4 columns and hexagonal arrangement, a *x*- and *y*-pitch of 18 and 22 μm, respectively, and an electrode size of 12 μm per side). To simulate drift, the templates (100 for each cell model) were obtained by virtually moving the cell models along a 100 μm straight vertical line with 1 μm steps starting from random initial positions. For each recording, we then selected 256 neurons and generated corresponding spike trains. Templates and spike trains were then convolved following the drift signals and adding a slight modulation in amplitude to add physiological variability. Finally, uncorrelated Gaussian noise with 5 μV standard deviation (STD) was added to the traces. The sampling rate of the simulated recordings was set to 32 kHz. Please refer to Figure 6 of Buccino and Einevoll (2020) for additional details on simulating drift.

To challenge the drift correction algorithms, we generated various drift recordings using different drift signals, depth distributions, and firing rate profiles. In doing so, we tried to capture the key features that could be observed in experimental recordings obtained in vivo, especially for Neuropixels probes.

Drift signals. The first obvious source of variability is the nature of the drift itself. Drifts can indeed be slow, rigid, or non-rigid as a function of the positions of the cells, but can also be abrupt and discontinuous, with jumps of few tens of microns (Steinmetz et al., 2021). We therefore generated three drift signals:

- *Zigzag (rigid):* For this drift mode, we used a rigid drift (all cells move homogeneously across depth) with a triangular slow oscillation, an amplitude of 30 μm and a 30 μm/min speed. Figure 2, *A* and *B*, shows sample raster maps using this drift mode.
- *Zigzag (non-rigid):* This mode is similar to Zigzag (rigid), but in this case, the drift is non-rigid, leading to a non-homogeneous movement of all the cells as a function of their depths. In this non-rigid case, the drift vector is modulated by a linear gradient, ranging from 0.4 (12 μm maximum displacement) at the upper part of the probe to 1 (30 μm maximum displacement) at the bottom. Figure 2, *C* and *D*, shows sample raster maps using this drift mode.
- *Bumps (non-rigid):* To reproduce the abrupt transient drifts that can often be observed in vivo (Durand et al., 2023), we also modeled fast, abrupt drifts that can happen irregularly during the recording. These abrupt events happened at random times (between 30 and 90 s) and moved all cells in a non-rigid manner, with minimal amplitudes between −20 and 20 μm reached at the top of the electrode and maximal ones between −40 and 40 μm at the bottom of the probe. The drift signals also experience an additional small 3 μm sinusoidal modulation with 40 Hz frequency. Figure 2*E* shows sample raster maps using this drift mode.

Depth distributions. A second feature that could affect motion estimation is the presence of probe regions with very low spiking activity. In fact, due to the paucity of localized spiking information, this might lead to a wrong estimation of motion. To account for this level of complexity, we simulated different depth distributions:

- *Uniform:* The depths of the selected neurons (256) were drawn from a uniform distribution, as in Figure 2*A* and *C–E*.
- *Bimodal:* The neuron depths follow a bimodal distribution with two peaks at the top and bottom of the probe and a central region with a lower density of cells, as in Figure 2*B*.

Firing rates. A third level of complexity is represented by non-stationary firing rates in the spiking activity. This scenario can be particularly challenging for motion inference methods that rely on an *average* template, such as the iterative template method. We therefore simulated two cases:

- *Homogeneous:* The activity of all neurons was modeled as homogeneous Poisson sources with 5 Hz mean firing rate, as in Figure 2*A*, *B*, *D*, and *E*.

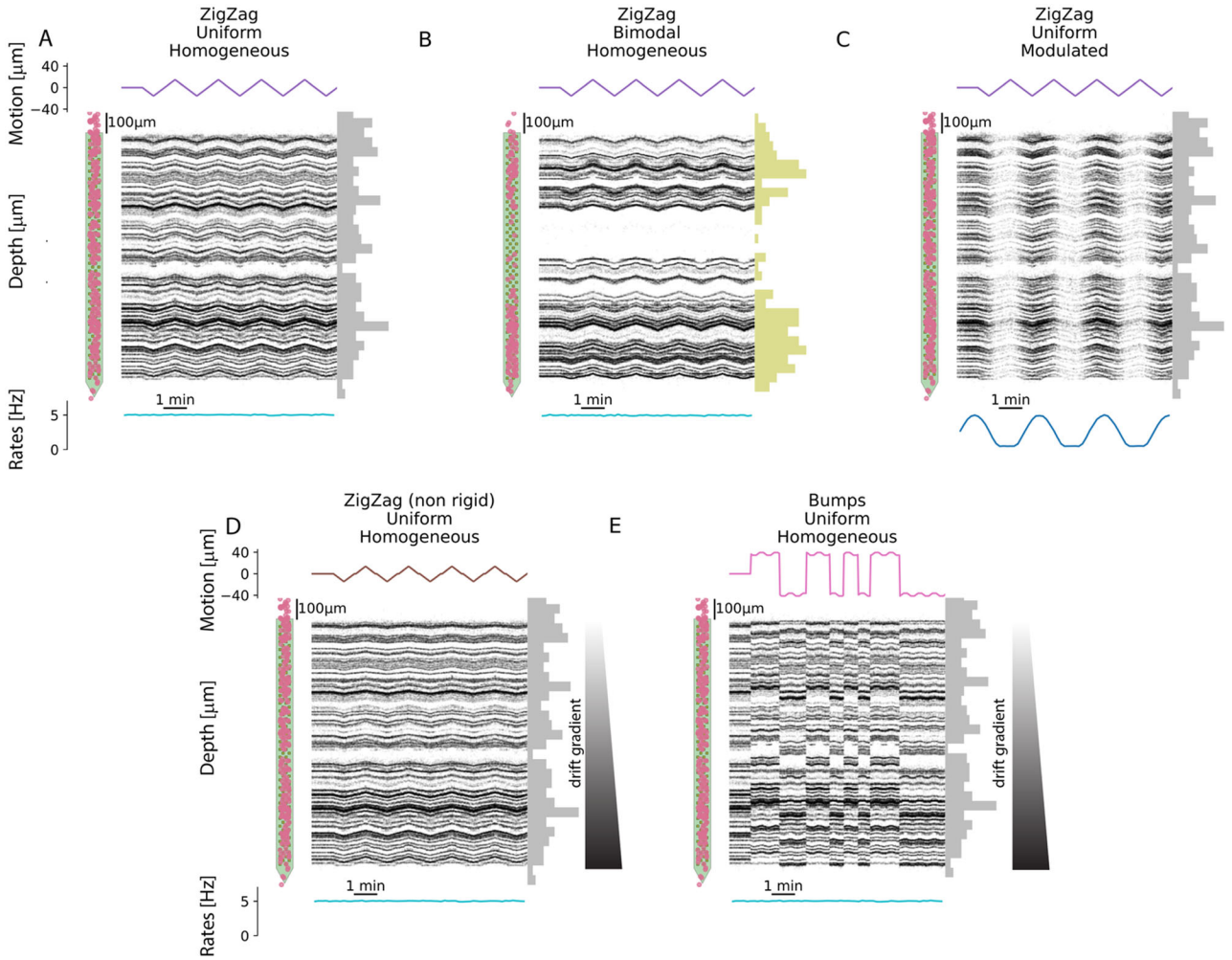

Figure 2. Examples of simulated drift recordings. For each panel, the top shows the layout of the probe with superimposed starting position of each cell (left), the positional raster plot with overlaid GT motion signals (center), and the depth distribution of the 256 neurons. The bottom part displays the firing rate modulation. Here we display 5 out of 12 recordings (the remaining 7 can be found as Extended data Figure on the github repo of the project). ***A***, Rigid ZigZag drift with uniform depth distribution and homogeneous firing rates. ***B***, Rigid ZigZag drift with bimodal depth distribution and homogeneous firing rates. ***C***, Non-rigid ZigZag with uniform depth distribution and modulated firing rates. ***D***, Non-rigid ZigZag with uniform depth distribution and homogeneous firing rates. ***E***, Bumps with uniform depth distribution and homogeneous firing rates.

- *Sine-modulated:* The spike trains were obtained following a non-homogeneous Poisson distribution modulated with a slow sine wave with a 3-min period, as in Figure 2*C*. In practice, to avoid too few spikes, the sinusoidal profile is clipped at 0.5 Hz.

We generated a total of 12 scenarios by combining the above-mentioned option (3 drift signals × 2 depth distributions × 2 firing rates). Five of these cases are shown in Figure 2, the rest can be found as a Extended data Figure available on the github repo of the project. In all cases, the drift started after 60 s, so that the first portion of the recording is drift-free. Owing to the reproducibility option of the `MEArec` simulator, which allows one to set random seeds to control all steps of the simulation, for each scenario we generated a *static* counterpart, with the exact same neurons, spiking activity, and noise profile, but only without drifts. We used these *static* recordings to benchmark the motion interpolation step.

Evaluation

To evaluate the impact of the motion estimation step, we took advantage of the known GT motion signals at different depths. We computed the estimation error as the Euclidean distance between the GT motion and the one estimated by the different methods. Note that to compensate the offsets between estimated and GT motions, we aligned the medians of the GT and the estimated motions. From these error signals, we could then visualize both the evolution of the error over time and the error profile along the depth dimension (by averaging accordingly).

To benchmark motion interpolation, we used the GT motion signals to interpolate the traces and then utilized the GT spike timing to extract single spike waveforms from the motion-corrected and associated *static* recording. Assuming that the neuron i has fired N spikes with waveforms $\vec{w}_i^{1\cdots N}(t)$ (defined on a subset of their five best channels, i.e., where amplitudes are minimal), we calculated the dispersion around the mean template $\vec{T}_i(t)$ as the STD σ_i to measure of how "variable" these spikes are. Normalization is achieved by dividing σ_i by the root mean square (RMS) of the template $\vec{T}_i$. To compare the effect of different motion interpolation procedures, we further computed the ratio $\sigma_i^{\text{interpolated}}/\sigma_i^{\text{static}}$ for every neuron i. The rationale for such a metric is that spike sorting algorithms assume that waveforms are "sterotypical," but drift is increasing the variability due to movements. Interpolation should therefore reduce this extra variability introduced by drifts.

Finally, and most importantly, we evaluated how drift correction affects spike sorting outcomes. To do so, we focused on the `Kilosort` 2.5 algorithm, which first introduced drift correction as a preprocessing step to spike sorting (Steinmetz et al., 2021). We ran all spike sorting jobs in the `SpikeInterface` framework (Buccino et al., 2020) on a 40-core Intel(R) Xeon(R) Silver 4210 CPU @ 2.20GHz, with 64GB of RAM and an 8GB Quadro RTX 4000 GPU. We used default parameters, but ran the preprocessing and motion correction steps in SpikeInterface instead of `Kilosort` 2.5. To match the `Kilosort` 2.5 processing, before motion correction, we applied a highpass filter (cut-off frequency at 150 Hz), a common median reference, and finally a local whitening step (Yger et al., 2018; Pachitariu et al., 2023; using, for each channel, the neighbor electrodes within 150 μm radius).

Knowing the GT spike timing, we could compute the *accuracy* of each GT unit i as

$$acc_i = \frac{TP_{ij}}{TP_{ij} - N_i - N_j} \tag{11}$$

where j is the sorted unit matched to the GT unit i, N_i and N_j are the number of spikes in the GT and matched sorted unit, respectively, and TP_{ij} is the number of *true positive* spikes, i.e., the spikes found both in the GT and sorted spike trains. From this accuracy metric, we further classified spike sorted units as:

- *well detected*: units with an accuracy greater than or equal to 80%.
- *overmerged*: units with an agreement above 20% with more than one GT unit.
- *redundant*: units with an agreement above 20% with a GT unit that are not the best matching unit. These units can be either oversplit or duplicated sorted units.
- *false positive*: sorted units with an agreement below 20%.
- *bad*: the sum of *overmerged*, *redundant*, and *false positive* units.

Results

Benchmarking motion estimation

To benchmark the performance of different motion estimation strategies, we generated several artificial recordings with 256 neurons and various drift cases (see Materials and Methods, Simulated datasets). Before diving into the fine details on the performance of different methods, we first looked at the global averaged errors over all algorithmic pipelines tested in this paper (see Methods); as a function of the drift features, we decided to explore while generating the artificial datasets. Since error samples are not independent (e.g., due to time and depth dependencies), we decided not to perform statistical test, but only qualitatively observe global trends. As in Figure 3*A*, we show the error distributions as a function of the drift signal type. Clearly, when drifts are discontinuous (bumps), all motion estimation algorithms show larger errors. Instead, there is no evident difference between the rigid and non-rigid ZigZag, which suggests that the methods that are used are properly able to handle non-rigid motion. Regarding the other drift features, a bimodal depth distribution results in slightly higher errors than a uniform one (Fig. 3*B*), and a modulate firing rate profile is more challenging than a homogeneous one (Fig. 3*C*). In all cases, another important general trend to highlight is that errors are larger at the borders of the probe (bottom panels). This is because channels at the borders have less spatial information to properly localize the peaks.

In Figure 4, we first show the results for a representative subset of five drift recordings (one per row). All the remaining cases can be found as a Extended data Figure available on the github repo of the project. The left column displays the raster maps with the overlaid drift signals at multiple depths, which give a qualitative view of the drift over the entire recording. The right part of the figure focuses on the errors of different *peak localization + motion inference* combinations: the left panels show the errors over time, the central panels the error distributions, and the right panels the errors over probe depth (left-bottom and right-top). The first thing that we should highlight is that motion estimation, irregardless of the method, is generally good. Average errors for relatively smooth drifts (*A*–*D*) are less than 5 μm, which is well below the inter-electrode distance. Even in the case of *Bumps* (*E* and *F*), where errors are more pronounced, some but not all methods can achieve average errors below 5 μm. Second, the errors tend to be larger at the borders of the probe. This is expected because we have less spatial information there, thus the motion inference is less accurate. Another general observation is related to the peak localization methods, used to estimate the activity profile: the CoM (*CoM*—blue

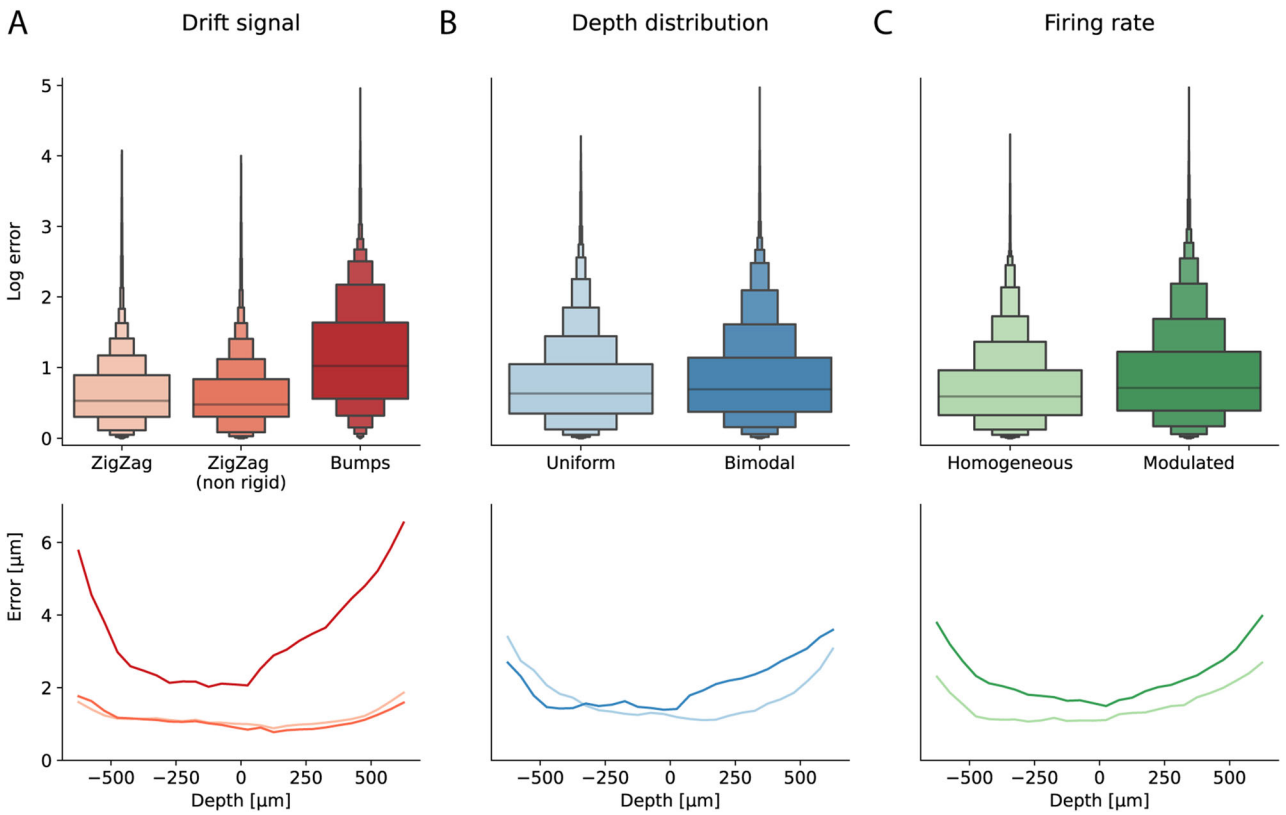

Figure 3. Motion estimation performance for different simulation modes. ***A***, Top: distribution of the log errors for all the errors pooled over all motion correction pipelines tested in this paper, as a function of the types of drift (ZigZag, ZigZag (non-rigid), or bumps). Bottom: averaged error over all methods, as a function of the probe depth and the nature of the drift. ***B***, Same as in *A*, but as a function of the type of the depth distribution (uniform and bimodal) of the neurons. ***C***, Same as in *A*, but as a function of the firing rate profiles.

color) method yields higher errors, and hence performs worse than all other estimation methods (but it is the fastest—see Fig. 5*C*). The monopolar triangulation (*Mono*—oranges) and the grid convolution (*Grid*—green) perform similarly, with the former achieving slightly lower error distributions. As for the motion inference step, the decentralized method (*Dec*—dark shades) generally produces lower errors than the iterative template (*Iter*—light shades) approach, except for the ZigZag/bimodal/modulated case (panel B). We should also note that the errors from the decentralized method, despite lower on average, show increased variance, due to some time bins being highly misestimated (see *spikes* in the second column of Fig. 4*B*, *E*, and *F*). During these particular time bins, the increased errors can dramatically affect the motion interpolation step and the overall spike sorting results.

To further investigate what is exactly happening during a complicated case, we decided to focus on the bumps/uniform/homogeneous case in Figure 5. As one can see in Figure 5*B*, the errors made by the decentralized algorithm (either via *CoM*—blue, *Mono*—orange, or *Grid*—green—localization) are lower, on average, compared to the ones made with the iterative template algorithm (see global errors). The method seems to be less sensitive to the depth (see Fig. 5*B*, depth error), with lower errors on average. The computational cost of the motion estimation part is slightly higher (see Fig. 5*C*), but most of the computational cost is due to the peak localization methods (the monopolar approximation is more accurate, but also slower as the results of an optimization problem). This is a point not explored here, since we are computing the position for all peaks, but only in a 10-min-long recording. For longer recordings, to limit the computational burden, sub-sampling the peaks prior to localization might be necessary (in the current implementation). Figure 5*D* shows the errors as a function of both depth and time, for all motion estimation options. Large transient errors (yellow stripes) can be observed for the decentralized method (and also in one case for the iterative template method) and do not seem to be correlated with depths. Such large errors, although transient, can have a very negative impact at the spike sorting level, because during these time bins, spikes are unlikely to be correctly recovered.

Benchmarking motion interpolation

After quantifying the performance of motion estimation, we evaluated and compared different motion interpolation methods: snapping (Snap), inverse distance weighting (IDW), and kriging (Krig) (see Materials and Methods, Motion interpolation for details). Note that kriging is the method implemented in `Kilosort` (Pachitariu et al., 2023).

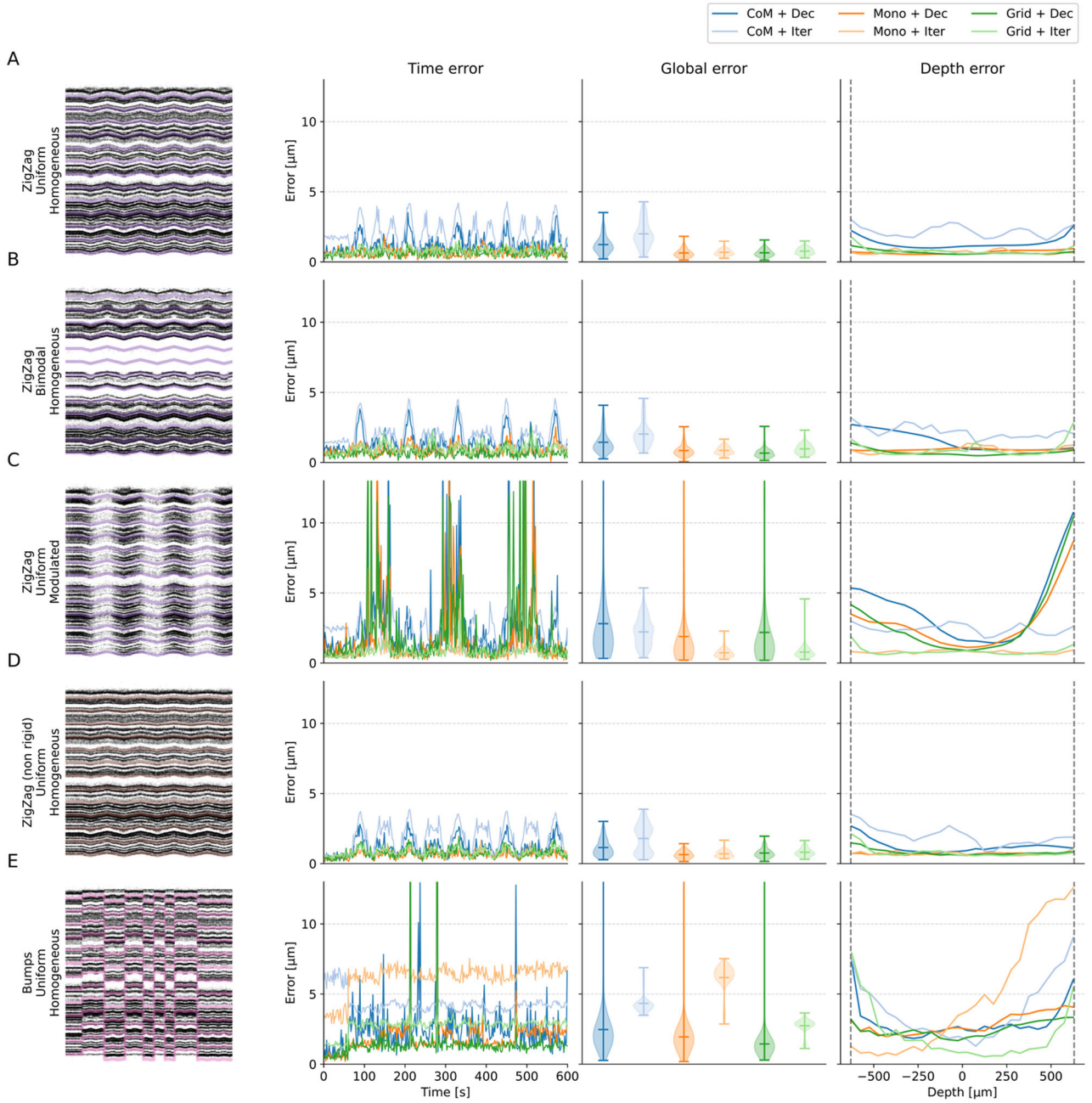

Figure 4. Performance in motion estimation for various drift scenarios. For various situations of the drifts with overlaid GT motion (left column), errors (from left to right) as a function of time (time error), averaged (global error), or as a function of the depths (depth error) and for various motion estimation pipelines. Here we display 5 out of 12 recordings (the remaining 7 can be found in the github repo of the project). ***A***, Zigzag (rigid), bimodal positions, homogeneous rates. ***B***, Zigzag (rigid), bimodal positions, homogeneous rates. ***C***, Zigzag (non-rigid), uniform positions, modulated rates. ***D***, Zigzag (non-rigid), uniform positions, homogeneous rates. ***E***, Bump (non-rigid), uniform positions, homogeneous rates.

As a representative example, we focused on one recording (bumps/uniform/homogeneous) for this section. Figure 6*A* shows a portion of the raster maps (from 0 to 400 μm depth) for the *static* (gray) and drifting (red) recordings, and the interpolated recordings with kriging (green), IDW (orange), and snapping (blue). The raster maps are estimated from the interpolated traces after applying filtering, common median referencing, and z-scoring and using monopolar triangulation to estimate peak depths. To isolate the effects of interpolation, we used here the GT motion vector as input for the various interpolation methods. Already from the raster maps, one can see that the kriging and IDW procedures seem to compensate well for the added motion, but in both cases some *wiggles* can be observed, indicating imperfections in the interpolation. In the ideal case, after interpolation, one would expect straight bands similar to the *static* case. The snapping method, instead, is not capable to compensate for such large drifts, also due to the relatively low spatial density of the

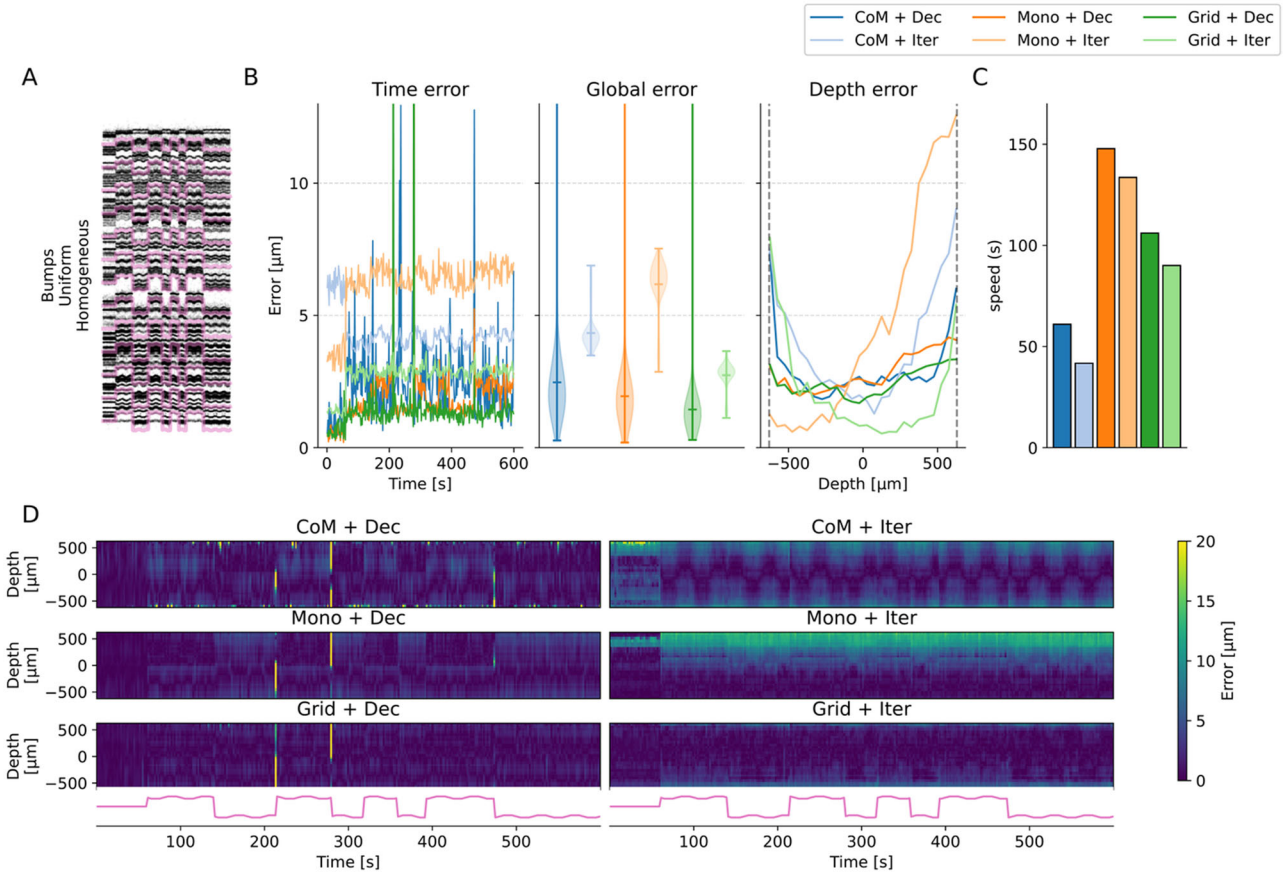

Figure 5. Motion estimation performance for the bumps (non-rigid) drift case. ***A***, Positional raster plot to show the movement of the cells, during the simulated drift. ***B***, The errors with respect to the GT motion vector as a function of the time (time error), averaged over time (global error), or as a function of the depth (depth error) for several motion estimation pipelines. ***C***, Run time of the different pipeline to estimate the motion (see legend below). ***D***, Errors, as a function of depth and time, for all the motion estimation methods considered. The time course of the averaged drift vector is shown at the bottom.

Neuropixels 1.0 configuration (with electrode distances of ~ 20 μm). Higher electrode densities might improve the results of this method.

To have a better insight on how interpolation affects spike waveforms, in Figure 6*B* we show the average template of one unit (top) and the temporal variance around the template (bottom) on the five channels with the largest amplitude. It is important to look at waveform variability since the main assumption of all spike sorting algorithms is that waveforms from a given neuron are reliable and stationary. A deviation from this assumption will therefore likely translate to worse spike sorting performances. For all interpolation methods, the interpolated template is smaller than the static counterpart (gray) and larger than the drifting recording (dashed red). This is probably due to the inevitable spatial smoothing that results from averaging the signal from different channels (for kriging and IDW). The temporal evolution of the STD also shows an increase, specifically in correspondence of the template peaks. Nevertheless, the STD is largely reduced with respect to the drifting case, indicating that motion interpolation *should* improve spike sorting results.

In Figure 6*C*, we show, for each neuron, the ratio of the waveform STD with respect to the static waveforms, depending on the neuron depth (left) and the distribution of these ratios (right) for different cases (red—drifting, green—kriging, orange—IDW, blue—snapping). We remind that this view is an idealized situation in the drift correction pipeline, because here the exact GT motion vector is provided to interpolate the traces. As expected, the waveform variability is systematically increased for all cells in the drifting case compared to static. For IDW and kriging, the overall ratio not only is reduced, but it is even less than one on average, probably due to the spatial smoothing resulting from the interpolation procedures, which reduces the additive uncorrelated noise. Since the test recording has non-rigid drifts, we can also observe a slight correlation with depth, with smaller variabilities as the depth gets positive (and the drift amount reduces). A strong border effect at the borders of the probe is also visible, with STD ratios increasing regardless of the methods. This is also expected, since all motion interpolation procedures, close to the probe borders, can only rely on partial information to interpolate the traces.

Global impact at the spike sorting level

After evaluating the performance at the motion estimation and motion interpolation stage, we now investigate how a complete drift correction pipeline affects spike sorting results. From the previous section, we showed that IDW and kriging

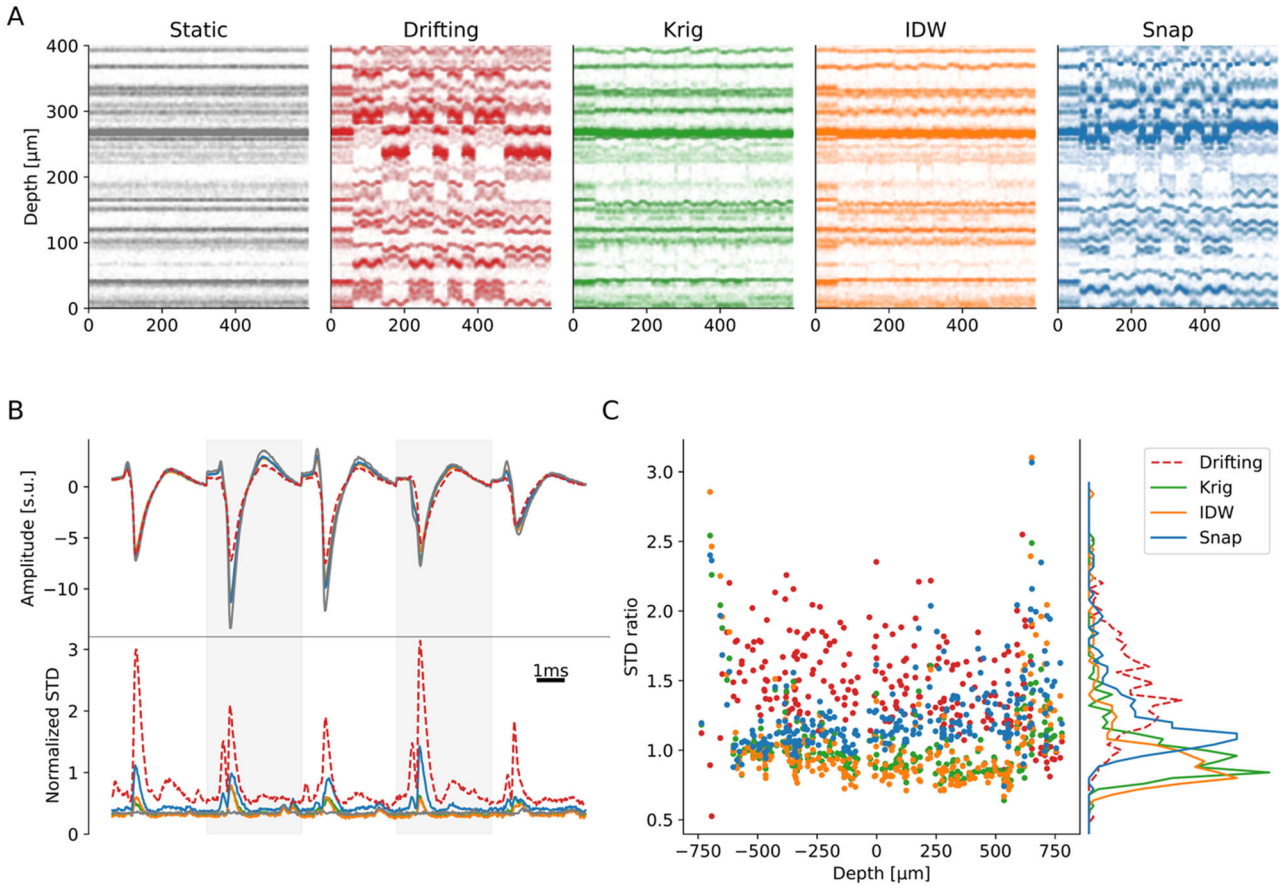

Figure 6. Quantification of the interpolation methods. ***A***, Positional raster plots with a bumps (non-rigid) drift (uniform/homogeneous) for the static (gray), drifting (red), and interpolated traces *via* Kriging (green), IDW (orange), or Snap (blue), using the GT motion vector. Note that the probe is cropped here for visibility purposes. ***B***, Top: flattened template (on a subset of channels) for a single neuron in all the conditions (static, drifting, and the three interpolation methods), in units of the median absolute deviation (mad) of the noise on the considered channels. Bottom: STD over time, normalized by the template RMS, of all the individual spikes for this particular neuron, in the same conditions. ***C***, Ratios of waveform STD compared to the static case for all the cells in the recording, in all conditions, and as a function of the depth of the neurons. Right panel displays the overall STD ratio distributions. In this case, an STD ratio above 1 means an increase in waveforms dispersion with respect to the static case (and a decrease for ratios below 1).

are the best interpolation options. For this analysis, we used kriging since it is also used by `Kilosort` 2.5 and it therefore allowed us to perform a direct comparison.

We ran spike sorting using `Kilosort` 2.5 on three different datasets—ZigZag, ZigZag (non-rigid), and bumps—all with uniform depth distributions and homogeneous firing rates (Fig. 2*A*, *C*, and *E*). For each recording, we ran five spike sorting options, and note that the cell distributions and spiking activity for all three recordings are the same, resulting in the exact same *static* recordings:

- *Static—no interpolation:* spike sorting on the *static* recording, turning off motion correction in `Kilosort` 2.5
- *Static—using KS2.5:* spike sorting on the *static* recording, turning on motion correction in `Kilosort` 2.5
- *Using GT:* spike sorting on the drifting recording, using the GT motion signals and kriging for motion interpolation in `SpikeInterface`
- *Using Mono + Dec:* spike sorting on the drifting recording, using the estimated motion with monopolar triangulation + decentralized inference for motion estimation and kriging for motion interpolation in `SpikeInterface`
- *Using KS2.5:* spike sorting on the drifting recording, letting `Kilosort` 2.5 preprocess the recording and correct for drifts (equivalent to grid localization + iterative template inference for motion estimation and kriging for motion interpolation).

Figure 7, *A*, *D*, and *G*, shows the accuracy of the spike sorting for all GT units (sorted by accuracy). For all test cases, we observe a lower performance on the drift-corrected recordings with respect to the static ones (red lines). This drop is particularly dramatic in the complex case of Bumps (Fig. 7*G*). Nevertheless, running motion correction using the Mono + Dec estimation in `SpikeInterface` (green lines) produces better results than using the `Kilosort` 2.5 correction procedure

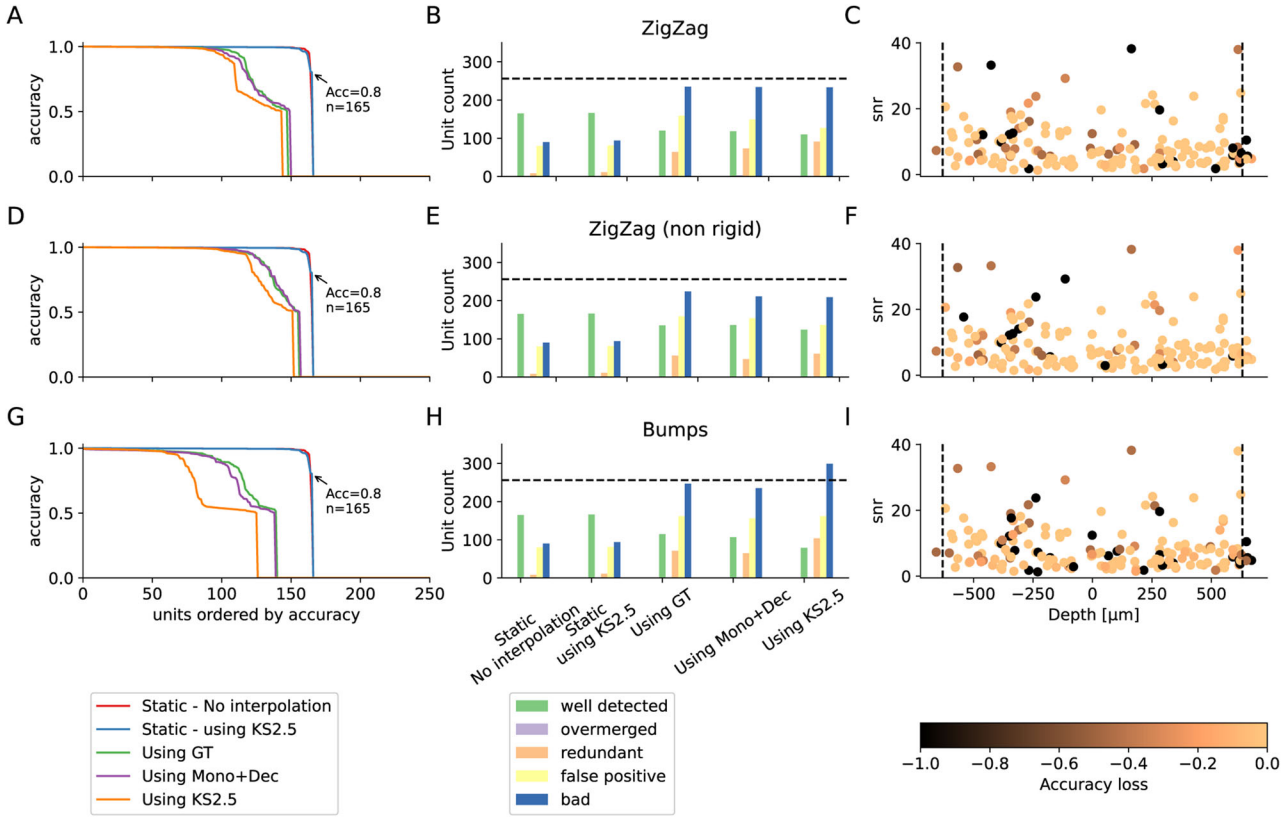

Figure 7. Quantification of the interpolation on sorting accuracy. ***A***, ***D***, ***G***, Accuracy of all units (sorted by accuracy) in the simulated recordings when launching `Kilosort` 2.5 in various conditions for the Zigzag rigid case (***A***), the Zigzag (non-rigid case) (***D***), or the bumps (non-rigid) case (***G***). All recordings have uniform distributions and homogeneous firing rates. ***B***, ***E***, ***H***, Number of well detected/overmerged/redundant/false positive and bad units found by `Kilosort` , with various drift correction methods, for the recordings explained in *A, D, G*. ***C***, ***F***, ***I***, The loss in accuracy for the well-detected neurons (accuracy higher than 0.8—$N = 164$) between the static case and the one using Mono + Dec estimation, as a function of the depth of the neurons (*x* axis) and the SNR (*y* axis).

(purple lines) and it yields similar performance to using the GT motion signals for interpolation (blue lines). In some cases, the accuracy using the estimated motion is even slightly better than the one using GT signals. This could be explained by the fact that the estimated motion is fully data-driven, while the effect of the GT motion on the traces can also depend on the neuron location, morphology, and relative orientation with respect to the electrodes. Note that even in all cases, the accuracy loss is rather large, indicating that the interpolation step is degrading spike sorting outputs. It is however interesting to note that the motion correction feature of `Kilosort` 2.5 is not harming the results when no drift is present. This is because when `Kilosort` does not detect any drift, no interpolation of the traces is performed and thus the extracellular traces are not modified, leading to very similar results with respect to static recordings (the slight differences are due to run-to-run variability of `Kilosort` itself).

When we look at the classification of sorted units (Fig. 7*B*, *E*, and *H*), the number of well-detected units is 165 for the static recording, but it drops to 118 for Mono + Dec and 110 for `Kilosort` on the ZigZag rigid recording; 136 for Mono + Dec and 124 for `Kilosort` on the ZigZag non-rigid recording; and 107 for Mono + Dec and 79 for `Kilosort` on the Bumps recording. In addition, a much larger number of bad units is observed after drift correction, resulting from both false positive and redundant units. However, correcting with `SpikeInterface` also results in a reduced number of bad units with respect to `Kilosort` : for example, for the Bumps example, Mono + Dec finds 238 bad units and `Kilosort` finds 302.

In Figure 7*C*, *F*, and *I*, we display the drop in accuracy for the well-detected units in the static recording ($N = 165$) between the static and the *estimated Mono + Dec* cases depending on the unit depths (*x* axis) and signal-to-noise ratio (SNR; *y* axis). While a clear trend is not apparent, some units with large SNR seem to exhibit a large drop in accuracy. These drops in accuracy could partially be due to oversplits due to residual drifts, which would also explain the large number of redundant units. In order to assess this, we performed a merging procedure of the sorted results using the GT information. In brief, for each GT unit, all the sorted units with an accuracy greater than or equal to 20% were merged into a single unit. Even after this merging step, there is still an observable drop in accuracy after interpolation compared to static (see Extended data Fig. 3*A*, *D*, and *G*, available on the github repo of the project). The number of well-detected units is increased, redundant units are almost gone, and the accuracy drops of high-SNR units are mostly recovered. However,

we want to highlight that this precise merging procedure is only possible for GT data, and other strategies will need to be employed for experimental data, such as manual merging or alternative data-driven automated methods (Llobet et al., 2022).

Drift correction with `SpikeInterface`

In the previous sections, we benchmarked the different steps involved in drift correction and their global impact on spike sorting outcomes. As mentioned in Section "A modular implementation of drift correction," we implemented all the methods in `SpikeInterface` with a modular architecture. For example, here is a code snippet that shows how to estimate motion signals from a Neuropixels recording:

```
import spikeinterface as si
import spikeinterface.extractors as se
# modular implementation
from spikeinterface.sortingcomponents.peak_detection import detect_peaks
from spikeinterface.sortingcomponents.peak_localization import detect_peaks
from spikeinterface.sortingcomponents.motion_estimation import estimate_motion
from spikeinterface.sortingcomponents.motion_interpolation import interpolate_motion
# Load Neuropixels recording
recording = se.read_spikeglx(spikeglx_folder)
# Step 1: peak detection
peaks = detect_peaks(recording,
method='locally_exclusive'
)
# Step 2: peak localization
peak_locations = localize_peaks(recording,
peaks,
method='monopolar_triangulation'
)
# Step 3: motion estimation
estimated_motion, temporal_bins, spatial_bins = estimate_motion(recording,
peaks,
peak_locations,
method='decentralized'
)
# Step 4: motion interpolation
recording_corrected = interpolate_motion(recording,
estimated_motion,
temporal_bins,
spatial_bins,
method='kriging',
border_mode='remove_channels'
)
```

Note that the `border_mode='remove_channels'` removes the channels at the border of the probe based on the maximum estimated motion and should reduce the inaccuracies observed at the borders of the probe.

Discussion

In this work, we have presented a comparison of state-of-the-art methods to correct for motion in extracellular electrophysiology recordings. We generated a wide variety of simulated benchmark datasets using a Neuropixels 1.0 probe design with varying levels of complexity, using Zigzag and/or *bumping* drift signals, rigid or non-rigid motion, and uniform or non-uniform depth distributions and firing rates. Knowing the GT motion and spiking activity, we were able to finely evaluate the two different steps involved in drift correction, i.e., motion estimation and motion interpolation.

For motion estimation, we showed that the combination of monopolar triangulation (for peak localization) and decentralized registration generally outperforms all other options on the test datasets. Nevertheless, in its current implementation, localizing peaks with monopolar triangulation is more computationally expensive than other strategies (center of mass and grid interpolation), and the decentralized motion estimation method has a tendency to provide short but high spurious errors during some time bins. It is also important to notice that the accuracy in estimating motion signals is lower close to the borders of the probe, owing to the partial view of drifting activity with neurons that might move in and out of the *field of view* and thus be harder to track.

To evaluate motion interpolation, we compared the motion-corrected traces to a matching *static* simulation without added drift. We found that the *kriging* method (same interpolation method as 2.5; Pachitariu et al., 2023) achieves an equally good performance in reconstructing the static traces compared to IDW scheme. Similarly to the drift estimation, the activity at the borders of the probe cannot be fully recovered/interpolated, because of missing information. Therefore, since both motion estimation and interpolation perform poorly at the borders of the probe, it would be advisable, for recordings with apparent drift, to discard portions of the probe close to the borders from downstream analysis. The design of metrics and/or quantitative criteria to perform such a slicing of the probes should be the subject of further studies and have already been implemented in `SpikeInterface`.

Finally, we evaluate the overall impact of the drift correction on spike sorting results. We found that applying the best motion estimation strategy (monopolar + decentralized registration) outperforms the `Kilosort` 2.5 implementation (grid interpolation + iterative template registration) and yields the same accuracy as using GT motion signals for the motion interpolation step. Importantly, the reader should note that even applying the *best* drift correction dramatically reduces spike sorting accuracy with respect to a static recording. In other words, even using the best correction strategy will not recover the same spike sorting accuracy as with a static recording. Electrophysiologists should therefore pay additional care in trying to minimize any major source of drifts, for example, by stabilizing the rig, lowering insertion speeds (Fiáth et al., 2019), and partially retracting the probe after insertion (Durand et al., 2023). On the algorithmic side, re-interpolating the input traces to correct for drifts prior to spike sorting might not be the favorable solution moving forward, since not only the waveforms are distorted but also the noise levels. Given the high accuracy of the motion estimation step, an alternative approach could therefore be to use the motion signals *within* the spike sorting pipeline, for example, by correcting waveform features before clustering or by using evolving templates over time in template matching (Boussard et al., 2023).

In this work, we used only simulated datasets on Neuropixels 1.0-like probe designs as a benchmark. While clearly artificial data cannot fully substitute experimental ones, they provide a very controlled and customizable framework to test specific aspects of recordings, including drifts. A quantitative evaluation of drift correction methods on real data would in fact be very challenging due to the lack of GT. Even in the case of mechanical manipulation of the probe to inject drift (Steinmetz, 2021), the relative movement of different layers of tissue with respect to the rigid probe movement would be hard to control. Furthermore, while here we focus on Neuropixels 1.0 probes, commercialized in 2019 and already widely used in neuroscience research, further benchmarks following a similar approach could target different probe layouts, such as Neuropixels 2.0 (Steinmetz et al., 2021), Neuropixels Ultra (a prototype version with closely packed electrodes at 5 μm pitch Andrew M Shelton et al., 2023), or other HD-MEA probe designs, such as the SiNAPS probe (Angotzi et al., 2019). Different probe geometries and electrode densities could in fact affect motion correction in non-trivial ways. The Neuropixels 2.0 layout was designed with drift correction in mind and is believed to improve the performance of motion correction with respect to the staggered layout of Neuropixels 1.0 (Steinmetz, 2021). For such configurable probes, users should keep in mind that choosing lower density configurations might badly affect the motion correction step. In this light, the presented simulation-based benchmark could be used in the in silico test and design of novel, more drift-resistant, probe layouts and geometries. It is also worthwhile noting that behavioral brain states might also have a direct impact on motion correction pipelines. However, as for real experimental data, such states can be hard to model and might depend a lot on the animals and experimental conditions. Finally, in this work we did not explore the relatively large parameter space associated to the different methods that we used, but we rather chose default parameters suggested by the original implementations. Future improvements could target fine-tuning the parameters to further improve the performance of motion correction methods.

All motion estimation methods evaluated in this analysis use the detected peaks to construct activity histograms utilized to reconstruct drift signals. While these approaches are appropriate for recordings in rodents, they may fall short when dealing with recordings experiencing drifts at faster scales, such as the heart-beat and breathing modulations observed in recent recordings from humans (Paulk et al., 2022; similar drifts could be expected from non-human primates; Trautmann et al., 2023). For drift at such fast timescales, local field potential signals can be used instead of peak location estimates to build activity histograms (Paulk et al., 2022; Windolf et al., 2023).

Finally, all the methods and drift correction options evaluated in this work are readily available to the electrophysiology community within the `SpikeInterface` package and can be immediately deployed with a few lines of code prior to any spike sorting process, as shown in Section “Drift correction with SpikeInterface.”

References

Angotzi GN, Boi F, Lecomte A, Miele E, Malerba M, Zucca S, Casile A, Berdondini L (2019) SiNAPS: an implantable active pixel sensor CMOS-probe for simultaneous large-scale neural recordings. Biosens Bioelectron 126:355–364.

Berdondini L, Imfeld K, Maccione A, Tedesco M, Neukom S, Koudelka-Hep M, Martinoia S (2009) Active pixel sensor array for high spatio-temporal resolution electrophysiological recordings from single cell to large scale neuronal networks. Lab Chip 9:2644–2651.

Boussard J, Varol E, Dong Lee H, Dethe N, Paninski L (2021) Three-dimensional spike localization and improved motion correction for neuropixels recordings. Adv Neural Inf Process Syst 34:22095–22105.

Boussard J, Windolf C, Hurwitz C, Lee HD, Yu H, Winter O, Paninski L (2023) "DARTsort: A Modular Drift Tracking Spike Sorter for High-Density Multi-electrode Probes." bioRxiv, pp 2023–08.

Buccino AP, Einevoll GT (2020) Mearec: a fast and customizable testbench simulator for ground-truth extracellular spiking activity. Neuroinformatics 19:185–204.

Buccino AP, Kordovan M, Ness TV, Merkt B, Häfliger PD, Fyhn M, Cauwenberghs G, Rotter S, Einevoll GT (2018) Combining biophysical modeling and deep learning for multielectrode array neuron localization and classification. J Neurophysiol 120:1212–1232.

Buccino AP, Hurwitz CL, Garcia S, Magland J, Siegle JH, Hurwitz R, Hennig MH (2020) Spikeinterface, a unified framework for spike sorting. Elife 9:e61834.

Buccino AP, Garcia S, Yger P (2022) Spike sorting: new trends and challenges of the era of high-density probes. Prog Biomed Eng 4:022005.

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

Durand S, et al. (2023) Acute head-fixed recordings in awake mice with multiple neuropixels probes. Nat Protoc 18:424–457.

Fiáth R, Márton AL, Mátyás F, Pinke D, Márton G, Tóth K, Ulbert I (2019) Slow insertion of silicon probes improves the quality of acute neuronal recordings. Sci Rep 9:1–17.

Frey U, Egert U, Heer F, Hafizovic S, Hierlemann A (2009) Microelectronic system for high-resolution mapping of extracellular electric fields applied to brain slices. Biosens Bioelectron 24:2191–2198.

Garcia S, Buccino AP, Yger P (2022) How do spike collisions affect spike sorting performance? Eneuro 9: ENEURO.0105-22.2022.

Greenberg DS, Kerr JND (2009) Automated correction of fast motion artifacts for two-photon imaging of awake animals. J Neurosci Methods 176:1–15.

Jun JJ, et al. (2017) Fully integrated silicon probes for high-density recording of neural activity. Nature 551:232.

Lee JH, et al. (2020) "YASS: Yet Another Spike Sorter Applied to Large-Scale Multi-electrode Array Recordings in Primate Retina." bioRxiv.

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

Llobet V, Wyngaard A, Barbour B (2022) "Automatic Post-processing and Merging of Multiple Spike-Sorting Analyses with Lussac." bioRxiv, pp 2022–02.

Magland J, Jun JJ, Lovero E, Morley AJ, Lincoln Hurwitz C, Paolo Buccino A, Garcia S, Barnett AH (2020) SpikeForest, reproducible web-facing ground-truth validation of automated neural spike sorters. Elife 9:e55167.

Markram H, et al. (2015) Reconstruction and simulation of neocortical microcircuitry. Cell 163:456–492.

Pachitariu M, Steinmetz NA, Kadir SN, Carandini M, Harris KD (2016) Fast and accurate spike sorting of high-channel count probes with KiloSort. In: *Advances in neural information processing systems* (Lee D, Sugiyama M, Luxburg U, Guyon I, Garnett R, eds), Vol. 29, pp 4448–4456. Curran Associates, Inc.

Pachitariu M, Sridhar S, Stringer C (2023) "Solving the Spike Sorting Problem with Kilosort." bioRxiv, pp 2023–01.

Paulk AC, et al. (2022) Large-scale neural recordings with single neuron resolution using neuropixels probes in human cortex. Technical report, Nature Publishing Group.

Ramaswamy S (2015) The neocortical microcircuit collaboration portal: a resource for rat somatosensory cortex. Front Neural Circ 9:44.

Steinmetz NA (2021) "imposed motion datasets" from Steinmetz et al. Science 2021. Available at: https://figshare.com/articles/dataset/_Imposed_motion_datasets_from_Steinmetz_et_al_Science_2021/14024495/1.

Steinmetz NA (2021) Neuropixels 2.0: a miniaturized high-density probe for stable, long-term brain recordings. Science 372:eabf4588.

Trautmann EM, et al. (2023) "Large-Scale High-Density Brain-Wide Neural Recording in Nonhuman Primates." bioRxiv, pp 2023–02.

Varol E, Boussard J, Dethe N, Winter O, Urai A, Laboratory TIB, Churchland A, Steinmetz N, Paninski L (2021) Decentralized motion inference and registration of neuropixel data. In: ICASSP 2021–2021 IEEE international conference on acoustics, speech and signal processing (ICASSP), pp 1085–1089. Toronto, ON, Canada: IEEE.

Virtanen P, et al. (2020) SciPy 1.0: fundamental algorithms for scientific computing in python. Nat Methods 17:261–272.

Windolf C (2023) Robust online multiband drift estimation in electrophysiology data. In: ICASSP 2023–2023 IEEE international conference on acoustics, speech and signal processing (ICASSP), pp 1–5. Rhodes Island, Greece: IEEE.

Ye Z, et al. (2023) "Ultra-High Density Electrodes Improve Detection, Yield, and Cell Type Specificity of Brain Recordings." bioRxiv, pp 2023–08.

Yger P, et al. (2018) A spike sorting toolbox for up to thousands of electrodes validated with ground truth recordings in vitro and in vivo. Elife 7:e34518.