Executive Summary of the Thesis

Investigating Independent Component Analysis-based strategies to improve spike sorting performance for high-density micro-electrode arrays

Laurea Magistrale in Biomedical Engineering - Ingegneria Biomedica

Author: Mattia Randazzo

Advisor: Dott. Alessio Paolo Buccino

Co-advisor: Prof. Alessandra Laura Giulia Pedrocchi, Prof. Andreas Hierlemann

Academic year: 2020-2021

1. Introduction

The analysis of neuronal signals is of great importance to understand the brain and the nervous system. Neurons communicate by means of electrical signals, called Action Potentials which generate a fast variation in the electric potential of the extracellular medium. The signal is recorded by neuronal probes whose spatial resolution, i.e. the density of channels, is increasing year by year. Indeed, the higher the spatial resolution of a neuronal probe, the higher the number of neurons that can be distinguished processing the recorded signal. After properly processing the signal it is possible to get access to single neuron activities, i.e. identifying the spike times of each neurons. The method that aims at processing neuronal recordings is called **spike sorting**.
Despite the large number of existing spike sorting algorithms, none can perform well on recordings with large noise levels, meaning that they fail in detecting small neurons. A possible solution to that problem is provided by Independent Component Analysis, a blind source separation algorithm that estimates the signals of the sources generating the recordings. The source signals, called Independent Components (ICs), have a larger *Signal to Noise Ratio* (SNR) than the raw recording. ICA has already been applied to spike sorting by running event detection (spike detection and clustering) on the output of ICA [1, 5]. In this work it is proposed a different and new approach which applies ICA as a post processing tool to enhance the activity of small neurons, helping spike sorting algorithms to detect them.

2. State Of The Art

2.1. Recording Devices

Classical recording devices such as tungsten microwires, tetrodes and silicon probes could record the activity of few cells only. The development of CMOS-based probes steeply increased both the density and the number of recording electrodes. The improved spatial resolution allows to record large networks of neurons with low noise levels. That is especially true for **High-Density Micro-Electrode Arrays** (HD-MEAs) which are chips for in-vitro experiments featuring thousands of electrodes.

2.2. Spike Sorting

Spike sorting algorithms aims to estimate the spike times of neurons in the analyzed culture. They differ in the actual implementation, but it is possible to identify four main steps:
Spike detection aim is to extract all the spikes from the signal by thresholding it, e.g. evaluating when the signal exceeds n times its *Mean Absolute Deviation* (MAD) [1]. After spikes are detected, their features are extracted for example by means of PCA or wavelet features. Finally, the spikes are clustered based on their features. Clustering algorithms may fail to detect overlapping spikes, i.e. spikes that overlap both in time and space. This problem is gaining more and more importance due to the increasing number of recording channels which increase the probability of spike collisions. **Template matching** algorithms have been developed to disclose overlapping spikes. They estimate a set of templates, i.e. the average spike waveform of a neuron, by running a pre-clustering step on a subset of detected spikes. Each template is then matched with the signal, e.g. by evaluating when the correlation between the signal and the template exceeds a threshold. When a match is found, the template is subtracted from the signal so that overlapping spikes are uncovered.

2.3. Independent Component Analysis

ICA is a blind source separation technique that aims to unmix a signal into a set of statistically independent signals, called Independent Components. When ICA is run on neuronal recordings, each IC is, ideally, the activity of a single neuron. ICA assumes the signal to be an instantaneous linear mix of the sources:

$$\boldsymbol{r}(t) = A\boldsymbol{s}(t) \tag{1}$$

where $\boldsymbol{r}(t)$ is the set of recorded signals, A is called the *mixing matrix* and $\boldsymbol{s}(t)$ is the set of source signals. Since the goal is to retrieve $\boldsymbol{s}(t)$, ICA estimates the pseudo-inverse of A, which is called *unmixing matrix* and outputs the estimated source signals $\bar{\boldsymbol{s}}(t)$ as follows:

$$\bar{\boldsymbol{s}}(t) = W\boldsymbol{r}(t) \tag{2}$$

where W is the *unmixing matrix*. Three assumptions must be satisfied to apply ICA:

- The number of channels is greater than the number of neurons: this assumption is closer to be satisfied for HD-MEAs given their high spatial resolution. Therefore, the algorithm proposed by this work assumes that signals are recorded by HD-MEA probes.
- Instantaneous statistical independence of the sources: that would not be true if neurons were firing together at every time. That is not the case due to randomness of spiking activity [5].
- Recorded signals are an instantaneous mixture of the sources: That is not completely true as the spikes recorded along dendrites are phase shifted with respect to spikes recorded close to the soma because of filtering properties of the dendrites.

2.4. Aim of the Work

The hereby work has three main objectives:

- Improving ICA efficiency: ICA scales quadratically with the number of channels which leads to large estimation times. That becomes problematic if signals are recorded by high-density probes, so we attempted to decrease the dimensionality of the input dataset by wisely subsampling the input dataset.
- Increasing the SNR of raw neuronal recordings through ICA.
- Improving sorting performances by increasing the SNR of recordings: Existing spike sorting algorithms do not perform well on low-SNR recordings which leads to not to detect small units in the signal. Since ICA increases the SNR of neurons [5] we attempted to enhance the activity of small neurons through ICA so that sorting algorithms may detect them.

3. Methods

3.1. Datasets

Due to the lack of ground-truth information for large recordings, it is not trivial to evaluate the performances of spike sorting algorithms. Therefore, recordings have been simulated using the `MEArec` [3] Python package. It allows to finely control the simulated recordings with several parameters. The recordings were simu-

lated with three different probe models (a Neuronexus probe with 32 channels, a squared high-density MEA with 100 channels and a Neuropixels probe with 128 channels), for each probe 4 noise levels (standard deviation of the signal in μV) have been used: 5, 10, 20 and 30. Finally, for each noise level 5 recordings with different seeds have been simulated. The number of neurons in the recording is half the number of channels, so Neuronexus recordings carry 16 neurons, squared MEA recordings carry 50 neurons and Neuropixels recordings carry 64 neurons.
We also tested our new ICA application for spike sorting on 2 real recordings. The signals were recorded by CMOS-based HD-MEA with 26'400 recording electrodes at 15 μm pitch. The neurons plated on the chip were extracted from embryos of Wistar rats. For both datasets the signals were only recorded from the most active electrodes which ended up being 529 and 536.

3.2. Spike Sorting Framework

To read and process recordings, to run spike sorting algorithms and to evaluate their performances we used `SpikeInterface` [2], a Python framework developed to ease the processing of neuronal recordings. It allows both to run several spike sorting algorithms (*sorters*) and to compare their outputs with ground-truth data (if available). Throughout the work we used the following sorters: `Herdingspikes`, `Ironclust` and `Kilosort2`.

3.3. Peak Selection

Due to the sparsity of neuron firing rates, spikes may be spaced out by several milliseconds during which only noise is recorded. That is not a useful information for the estimation of the ICs for spike sorting purposes as the ICA model do not follow temporal patterns (see equation 2). That is why the dataset can be subsampled by focusing it on the spikes only. Spike times are first detected evaluating when the signal exceeds n times its MAD ($n = 5$ by default). For every spike a window of $n_{before} + n_{after}$ samples is selected on all the channels. The two parameters are computed as follow:

$$n_{before} = \frac{ms_{before} * fs}{1000} \qquad (3)$$

$$n_{after} = \frac{ms_{after} * fs}{1000}$$

where ms_{before} and ms_{after} are the milliseconds before and after the spike time respectively. By default $ms_{before} = 1$ and $ms_{after} = 2$ but they can be set externally. It is possible to further subsample the dataset using a (user-defined) percentage of the detected spikes. Since a random choice of the spikes may do not represent the variability of the original dataset, it is possible to balance the number of spikes across the channels so that the number of selected spikes for a channel i is:

$$n_{spikes,i} = all\_spikes_i * p \qquad (4)$$

where all_spikes_i is the number of all spikes detected on channel i and p is the percentage of spikes to be used.

3.4. Cleaning

When the number of neurons is lower than the number of channels, some ICs are not focused on any channel and they only carry noise, meaning that they need to be removed. Following the same approach used by Buccino et al. [1] the *false* neuronal sources can be removed by removing the ICs whose skewness is below a threshold (0.2 by default). The skewness computation may take some time for large recordings, that is why its computation was improved by parallelizing the process.

3.5. Units Recovery

Because of the phase shift introduced by dendrites, the ICA model may estimate duplicate sources, i.e. ICs that are focused on the same neuron. Projecting the ICs back to the original space reconstructs the original structure of the data solving the problem of duplicate sources. Moreover, after the cleaning process a portion of the noise in the signal is removed which leads to an increase of the recording SNR after back-projection. That is the starting point of the small-unit recovery algorithm. Often, after running a sorting algorithm on a recording, only the largest neurons are well detected. If their spikes are removed from the signal, only small neurons are still present in the signal. So, the small-unit recovery algorithm runs ICA on the residual recording (after peak selection) and backprojects ICs (after cleaning) to increase the SNR of the residual signals. Now, the same sorting algorithm is run again to extract the spike train of neurons that were not detected during the first round of spike sorting. If ground-truth data are available, `SpikeInterface` [2] provides the units that are *well detected* by the sorter. If ground truth is not available, we propose an *automatic curation* step to find the *well detected* units. It defines a unit i as *well detected* based on the following quality metrics [4]:

- $firing\_rate_i \geq 0.1\ Hz$
- $ISI\_violation\_ratio_i \leq 0.3$

The firing rate of a neuron is the average frequency at which it emits spikes, while the ISI violation ratio is the percentage of spikes that violate the Inter Spike Interval. The ISI is a time interval between two consecutive spikes during which a neuron is not supposed to emit spikes. The default value is equal to 1 ms but it can be set by the user. Both threshold values are the default ones but they can be set externally for a more or less strict automatic curation.

4. Results

4.1. Efficiency Improvement

To understand if the subsampling decreases the *goodness* of the estimated ICs, the accuracy of a sorting algorithm (`Ironclust`) was evaluated on the datasets subsampled with decreasing percentage of spikes. Figure 1 shows the results for recordings simulated with the Neuropixels probe. Both the elapsed time (red) and the accuracy (blue) were averaged on 5 simulated recordings with a noise level equal to 10 μV and 64 neurons. The recordings were sampled at 32 kHz and were 120 s long for a total of 3'840'000 samples per channel. The time values were normalized by the time required by ICA when run on the *full* dataset. Conversely, the accuracy was only evaluated for the detected units with a $SNR > 5$ (see equation 6). When the *full* dataset was used ICA took $5'251.59 \pm 141.27\ s$ and the accuracy was $69.62 \pm 3.59\%$. After subsampling (*all peaks*) the dimension and the elapsed time decreased to $2'458'775 \pm 119'186$ (35.97% reduction) and $2'562.60 \pm 125.79\ s$ (51.2% reduction) respectively. On the contrary, the accuracy increased to $73.12 \pm 3.74\%$. Overall, decreasing the percentage of spikes decreases the running time, but the accuracy remains almost constant.

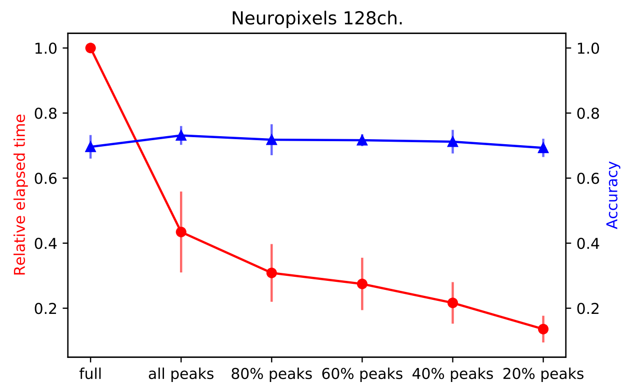


Figure 1: Average running time of ICA (red) and accuracy of Ironclust (blue) as function of the dataset dimension. The accuracy is constant with respect to the percentage of spikes used for subsampling the dataset while the running time decreases by decreasing the percentage of spikes. If *all peaks* are used, the running time is reduced by 51.2%.

4.2. SNR Improvement

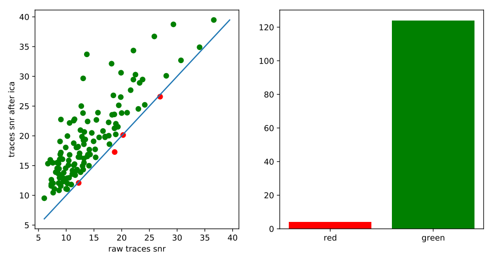
To evaluate the consequences of projecting the ICs back to the original space after cleaning, the SNR of both traces and neurons has been computed before and after **backprojection**. The traces SNR was evaluated as:

$$traces\_snr_i = \frac{max(traces_i)}{noise\_level_i} \tag{5}$$

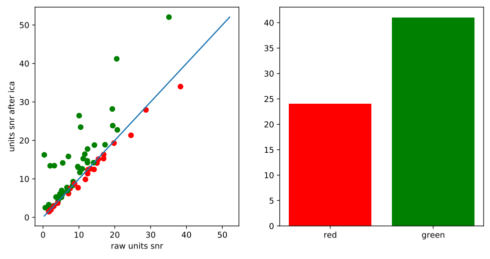
where the index i refers to the i^{th} channel and the noise level is the MAD of the channel. The neurons SNR is computed by `SpikeInterface` [2] as:

$$units\_snr_j = \frac{max(waveforms_j)}{noise\_level_j} \tag{6}$$

where $waveforms_j$ is a vector collecting all spike waveforms from unit j and $noise\_level_j$ is the noise level of the trace from which the spike given by $max(waveforms_j)$ is extracted. Figure 2 compares the channel SNRs before (horizontal axis) and after (vertical axis) **backprojection**. Each dot represents a channel and it is red if its SNR decreased, while it is green if its SNR increased. The barplot in the right panel counts the green and red dots. It can be seen that the SNR of 124 out of 128 channels improved thanks to **backprojection**.
Figure 2.B compares the units SNR before (horizontal axis) and after (vertical axis) **backprojection**. It can be seen that the SNR of 41 out



(A)



(B)

Figure 2: The scatter plots compare the SNR of traces (**A**) and units (**B**) before and after **backprojection**. Green dots show when the SNR increased, while the red ones show when the SNR decreased. On the right side, the barplots count the number of red and green points.

of 64 neurons improved thanks to **backprojection**.
The datasets processed here were simulated with Neuropixels probe (128 channels) with `noise_level=10` and 64 neurons.

4.3. Units Recovery

The algorithm for small-unit recovery was run with three different spikes sorting algorithms: `Herdingspikes`, `Ironclust` and `Kilosort2`. Figure 3 shows the accuracy improvement ($\Delta acc = acc_{after\_recovery} - acc_{before\_recovery}$) averaged over the three algorithms for three probes - Neuronexus (magenta), squared MEA (blue) and Neuropixels(orange) - with increasing noise level. The algorithm performs better on squared MEA and Neuropixels when the average improvement at `noise_level=5` is equal to 9.12±0.1% and 10.8±0.2%. For Neuronexus the improvement at the same noise level is less than half of the previously and equal to $3.87 \pm 0.2\%$. This may suggest that an higher spatial resolution of the probe used to record the signal can improve the output of ICA for spike sorting purposes. The performance decreases as the noise

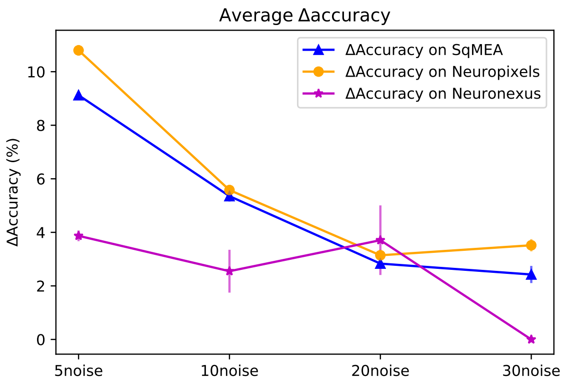


Figure 3: The plot shows the average accuracy improvement for `Kilosort2`, `Herdingspikes` and `Ironclust` algorithms on recordings simulated with Neuronexus (magenta), squared MEA (blue) and Neuropixels (orange) with increasing noise level. The improvement achieved on squared MEA and Neuropixels recordings is greater than the one achieved on Neuronexus recordings (with an exception at 20 μV of noise level).

level increases. Indeed, at `noise_level=30` the Δ accuracy on Neuronexus, squared MEA and Neuropixels respectively drops to 0%, $2.43 \pm 0.32\%$ and $3.52 \pm 0.23\%$.
If ground truth is not available, non-*well detected units* may be detected by evaluating which units have a small firing rate or a large ISI violation ratio (*automatic curation*). Figure 4 shows the average Δaccuracy between small-units recovery with *automatic curation* and small-units recovery with ground-truth (gt) curation ($\Delta acc = acc_{gt} - acc_{automatic\_curation}$). The highest Δaccuracy is related to squared MEA at `noise_level=5` and equal to 7.11% while the Δaccuracy on Neuropixels and Neuronexus is much lower and equal to 1.92% and 1.62% at the same noise level. A different behavior is shown for Neuropixels and Neuronexus as the difference of the former remains $\approx 2\%$ while the accuracy of the latter for `noise_level=20` and `noise_level=30` becomes negative and equal to -1.02% and -0.69%, meaning it increases with *automatic curation*.
The small-unit recovery algorithm was tested on real recordings. The results are shown in Figure 5. Figure **(A)** shows the results of the algorithm using `Kilosort2` as spike sorting algorithm which found 52 and 43 new units (after recovery) for a total of 239 and 244 neurons on the 529-channels recording and the 536-channels recording respectively. Figure **(B)** shows the re-

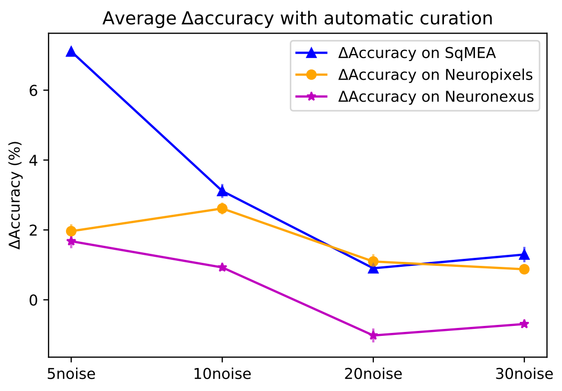
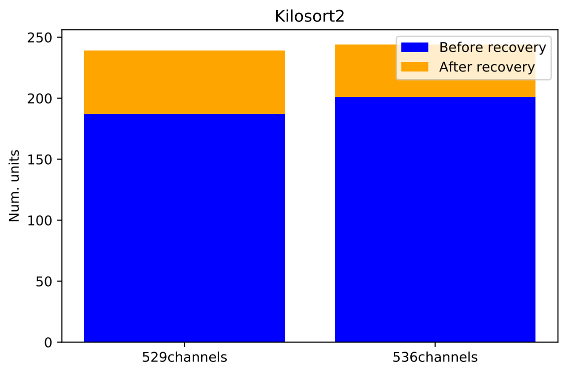
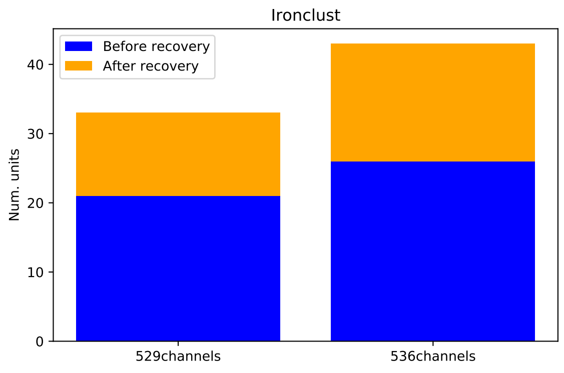


Figure 4: The plot compares the accuracy difference of *automatic curation* with respect to the ideal case (availability of groud truth) for `Kilosort2`, `Herdingspikes` and `Ironclust` algorithms on recordings simulated with Neuronexus (magenta), squared MEA (blue) and Neuropixels (orange) with increasing noise level. The accuracy decrease remains fairly constant for Neuropixels ($\approx 2\%$) while it is larger for squared MEA at low noise levels.



(A)



(B)

Figure 5: Performance of units recovery on real recordings. **(A)**: the results of `Kilosort2` that found 52 and 43 new units on the recording with 526 and 539 channels respectively. **(B)**: the results of `Ironclust` which found 12 and 17 new units.

sults of the algorithm using `Ironclust` as spike sorting algorithm which found 12 and 17 new units (after recovery) for a total of 33 and 43 neurons the 529-channels recording and the 536-channels recording respectively.

5. Conclusions

Independent Component Analysis is a blind source separation method which estimates the signal of sources that generated the recording. The application of ICA in spike sorting gained new interest because of the high spatial resolution of high-count channel probes, such as **High-Density Micro-Electrode Arrays**. Since ICA scales quadratically with the number of features, it is unfeasible its application to large recordings. The sparsity of neuron firing rates allows to subsample the input dataset. Keeping only the detected spike,s greatly decreases ICA running time without lowering the *goodness* of the ICs. Moreover, it is possible to use a subset of the detected spikes (up to 20% of them) without worsening ICs estimation.
The ability of ICA to increase the SNR of a signal [5] can be exploited to improve the performances of existing spike sorting algorithms. Indeed, after removing from the signal the spikes of *well detected* neurons, the recording has a low SNR as it carries small neurons only. Applying ICA to the residual recording and back-projecting the ICs to the recording space enhances the activities of neurons that were not removed from the signal, making it easier for sorting algorithms to detect them if they are run again. It is worth noting that the algorithm is influenced by the spatial resolution of the probe used and by the noise level of the signal. Indeed, the higher the spatial resolution of the probe and the lower the noise level, the better the ICs estimation which suggests to apply ICA on high-count channel probe recordings such as HD-MEAs which feature a large number of recording electrodes and a small noise level.

Future Developments The *goodness* of the ICs can be improved by *convolutive* ICA instead of using the classical model as it follows temporal patterns (which tackle the duplicate problem too) in the signal by estimating a set of *unmixing matrices* delayed in time. It is worth to mention that the computational time of cICA is even greater than the one of classical ICA, so the choice between classical and convolutive ICA is a trade-off between computational speed and *goodness* of the estimation. Understanding which units have been *well detected* is not trivial if ground-truth data are not available. We proposed a simple *automatic curation* approach which can be improved by using a set of more complex metrics to evaluate the output of a sorter (see `SpikeInterface` [2] paper for a full list).

References

[1] Alessio P. Buccino, Espen Hagen, Gaute T. Einevoll, Philipp D. Hafliger, and Gert Cauwenberghs. 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)*, pages 2627–2630. IEEE.

[2] Alessio P Buccino, Cole L Hurwitz, Samuel Garcia, Jeremy Magland, Joshua H Siegle, Roger Hurwitz, and Matthias H Hennig. SpikeInterface, a unified framework for spike sorting. 9:e61834.

[3] Alessio Paolo Buccino and Gaute Tomas Einevoll. MEArec: A fast and customizable testbench simulator for ground-truth extracellular spiking activity. 19(1):185–204.

[4] Daniel N. Hill, Samar B. Mehta, and David Kleinfeld. Quality metrics to accompany spike sorting of extracellular signals. 31(24):8699–8705.

[5] David Jäckel, Urs Frey, Michele Fiscella, Felix Franke, and Andreas Hierlemann. Applicability of independent component analysis on high-density microelectrode array recordings. 108(1):334–348.

Investigating Independent Component Analysis-based strategies to improve spike sorting performance for high-density micro-electrode arrays

Tesi di Laurea Magistrale in
Biomedical Engineering - Ingegneria Biomedica

Author: **Mattia Randazzo**

Student ID: 940678
Advisor: Dott. Alessio Paolo Buccino
Co-advisors: Prof. Alessandra Laura Giulia Pedrocchi,
Prof. Andreas Hierlemann
Academic Year: 2020-21

Abstract

Neuroscience is a multidisciplinary science studying the brain and the nervous system. One widely used approach to record neuronal electrical activity is extracellular electrophysiology where neuronal probes record the electrical potential of the extracellular medium. The recorded signal is made up by a summation of the activity of all neurons close to the recording device, meaning that the signals need to be processed to reconstruct single unit activities. That is the goal of *spike sorting* algorithms. Even though a huge number of algorithms have been published in the last years, none can perform well on recordings with large noise level and small neuronal activity, meaning small neurons may not be detected.

Independent Component Analysis is a blind source separation method that improve the quality of a signal, focusing it on the activity of its sources. Application of ICA has gained new interest in the latest years due to the increased spatial resolution of recording devices. The output of ICA is a set of signals, called Independent Components, that are, ideally, focused on the activity of a single neuron. That enhances the neuronal signals which, in turn, may improve spike sorting performances. Starting from that assumption, this work proposes new approaches for the application of ICA in spike sorting and it attempts to solve its computational limitations.

Keywords: ICA, spike sorting, neuroscience

Abstract in lingua italiana

Le neuroscienze sono l'insieme delle scienze che studiano il sistema nervoso. Un approccio comune per registrare i segnali elettrici prodotti dai neuroni è l'elettrofisiologia extracellulare, nella quale sonde neuronali misurano il potenziale elettrico del liquido extracellulare. Il segnale registrato è composto dalla somma dei segnali prodotti dai singoli neuroni, quindi è necessario processare il segnale registrato per avere accesso all'attività delle singole unità. Questo è l'obiettivo degli algoritmi di spike sorting che ad oggi, nonostante il grande numero di algoritmi esistenti, falliscono nell'analisi di segnali con un alto livello di rumore.

L'Analisi delle Componenti Indipendenti (ICA) è un metodo di elaborazione dei segnali che mira a scomporre un segnale nelle sorgenti che lo hanno generato. L'applicazione di ICA nell'ambito di spike sorting ha ricevuto nuova attenzione grazie all'aumento della risoluzione spaziale delle sonde neuronali. L'output di ICA sono i segnali delle singole sorgenti, anche chiamate Componenti Indipendenti (ICs). Questa caratteristica porta le ICs ad avere un rapporto segnale/rumore maggiore rispetto ai segnali registrati che a sua volta porta ad una migliore performance degli algoritmi di spike sorting. Questa è l'assunzione di base dietro alla nuova applicazione di ICA per spike sorting proposta in questo elaborato.

Parole chiave: ICA, spike sorting, neuroscienze

Contents

Introduction

Among all of mysteries of life, the brain is one the most fascinating and interesting. Many and many studies have been carried out to shed light on brain mechanisms, yet we have very little knowledge about it. The main players of brain activity are neurons; in the brain, approximately 86 billion of them are grouped in only 1260 cm^2, pretty amazing, isn't it? "*Phenomenal cosmic powers, itty bitty living space!*" to cite a well known feature-film. Neurons share information by means of electrical signals. Information is first gathered through dendrites which branch as if they were trees roots looking for nourishment. Signals are then processed in a central hub, the soma, and finally the output is sent to other neurons through the axon. Studying the response of a neuronal network to an external stimulus can provide important information about the properties of each cell as well as the characteristic behavior of the network. Neuronal activity can be evaluated either measuring intracellular or extracellular electric potential by means of probes. The main difference between the two types of recording lies in the number of neurons that can be analyzed at the same time. Intracellular recordings allow to directly measure the activity of a single neuron, but they cannot capture the activity of a neuronal network. Conversely, extracellular recordings allow us to record from many neurons. Unfortunately, if one records the electrical signal from a network of neurons, it will contain a mixture of the activity of each neuron. Hence, the recordings need to be processed to extract single-neuron activities. This essential processing step is called **spike sorting**.

Spike Sorting Spikes can be interpreted as the unitary information, like a single bit, sent by a neuron. At the beginning of spike sorting, clustering was mainly performed by hand, which suffers of several limitations. First of all, the process strongly suffers of subject variability. Second, it does not scale well with the number of probe channels. Recently, more and more semi-automatic and automatic algorithms have been developed. The first step of a spike sorting algorithm is *spike detection* which often leverages on a fixed threshold, applied to the signal amplitude, and a shadow period to estimate spike times [27]. After spikes have been detected, the data needs to be projected into new spaces with lower dimensions. The features for dimensionality reduction can be chosen empirically or automatically extrapolated from data, e.g. through PCA or wavelet features. Despite their wide use, clustering-based approaches have some limitations, such as oversplitting units and overlapping spikes [7]. Oversplitting means spikes from a unit are grouped in two or more clusters, while overlapping spikes means the algorithm cannot distinguish spikes overlapping both in time and space. Whereas oversplitting can be solved with either a manual or automated curation step, correcting for overlapping spikes is more tricky. Template matching approaches have been developed to ameliorate that issue. A template is considered to be the characteristic waveform of spikes coming from the same neuron. It is then assumed the signal is made up by a scaled sum of templates. They are first estimated by running a clustering step on detected spikes and then everytime a template is detected in the signal, for example by means of correlation, it is subtracted. However, two issues seem to be unsolved [25]: how to perform event detection for redundant data and which features should be used for unit separation. Independent Component Analysis (ICA), a third, less explored, solution may answer these questions.

Independent Component Analysis ICA is a blind source separation technique that aims at demixing a set of linearly mixed and statistically independent sources. When signals are recorded, either *in-vivo* or *in-vitro*, each electrode of the probe collects a mixture of neuronal signals, meaning that ICA may ideally draw out single activities [9]. However, ICA has three assumptions that need to be considered:

1. The number of sources is lower than or equal to the number of signals: thanks to *High-Density Microelectrode Arrays* (HD-MEAs) developed throughout the last years, the number of recording electrodes is getting closer and closer to the number of expected neurons.

2. The signals are an instantaneous and linear mixture of the sources: this condition is not fully met. Spikes recorded along the dendrites have a phase shift with respect to spikes recorded in the vicinity of the soma because of the filtering properties of dendrites. As a consequence, ICA may estimate some sources focused on the same neuron.

3. The sources need to be statistically independent: although neurons are not independent from each other, ICA requires an instantaneous independence. Luckily, due to randomness of spiking activity [9, 25] that is not the case.

The sources are estimated by maximizing their statistical independence, leading to an optimization problem which can be solved in many ways [21]. In terms of application to spike sorting, ICA suffers of two main problems: the huge computational effort and the generation of false sources. The former can be addressed by running the algorithm after -

wisely - subsampling the input dataset. The latter requires to detect and discard the false neuronal sources, which usually only carry noise. Conversely, ICA has advantages too. In particular, it sharply increases the *Signal to Noise Ratio* (SNR) of a signal [25]. The SNR is defined as the ratio of signal power to the noise power and it is used to compare the amplitude of the desired signal with respect to the noise level. Increasing the SNR in a spike sorting framework can improve spike detection and consequently the performance of spike sorting algorithms on recording with low SNR.

Aim of the work The objectives of the work were the following:

1. Improving ICA efficiency: ICA suffers of huge computational cost which makes its application to large dataset unfeasible. The first goal of this project is to find suitable and efficient ways to subsample the input dataset to minimize the computational cost without losing any crucial information for estimating the sources.
2. Improving the SNR of neuronal signals through ICA
3. Increasing accuracy of existing spike sorters: despite many spike sorting algorithms exist, they usually fail in detecting small neurons whose spikes have small amplitude. Given the ability of ICA to increase SNR [25], it is possible to use ICA as a post-processing tool to improve spike sorting performances on recording with small SNR.

1 | State Of The Art

1.1. Electrophysiology

Neurons communicate each other with electrical signals, generating on/off events called action potential or, equivalently, *spikes*. Every neuron, in resting conditions, express a difference between the electric potential of intra- and extra-cellular fluids, which is usually about -70 mV. External perturbations may induce a variation of the potential and when it crosses a threshold, conventionally -55 mV, a spike is generated. Electrophysiology studies electrical properties of neurons by recording and generating spikes under different conditions. Recordings can be classified as intracellular and extracellular recordings.

During intracellular recordings the recording electrode is placed inside the soma or the axon while the reference one is put outside the cell. That is why they focus on single cells addressing variations of membrane potential due to neurotransmitters or pharmacological agents. A widely-used intracellular recording is the patch-clamp techinque. A micropipette is used to "patch" a small portion of the membrane which is perforated to have direct access to the intracellular space. Now, ions can flow from the intracellular region into the pipette. The generated current is then sensed by an amplifier. On the contrary, extracellular recordings collect signals from several neurons.

1.1.1. Extracellular electrophysiology

During extracellular recordings the recording electrodes are placed outside the cell and the ground electrode in a far extracellular position, so now the probe records the electric potential of the extracellular medium. The recorded potential is generated by ion currents that flow across cells membrane, called *transmembrane currents*, and propagate from the cell throughout the extracellular medium. It acts as a low-pass filter that causes the presence of two components in the recorded signal: slow variations of the potential, called *Local Field Potentials* (LFPs) which represent the low frequency component due to the activity of neurons far from the recording site and the spiking activity of close neurons which is a high frequency component [7] (Figure 1.1).

Knowing the total current generated by a neuron and approximating it as a *point current source*, using **volume conduction theory** we can approximate the extracellular potential as [2]:

$$\mathbf{\Phi}(r,t) = \frac{1}{4\pi\sigma}\frac{\mathbf{I}(t)}{r} \tag{1.1}$$

where r is the distance of the electrode from the source, $\mathbf{\Phi}(r,t)$ is the extracellular potential at time instant t and distance r, $\boldsymbol{I}(t)$ is the transmembrane current at time t and σ is the conductivity of the extracellular fluid. Equation 1.1 provide the potential for a single source, but it can be easily extended to N *point current sources* as follows[2]:

$$\mathbf{\Phi}(r,t) = \frac{1}{4\pi\sigma}\sum_{i=1}^{N}\frac{\boldsymbol{I}_n(t)}{r_n} \tag{1.2}$$

From the above equation it can be seen that closer the source to the recording electrode, the larger its contribute while the more distant the source is, the more attenuated its signal.

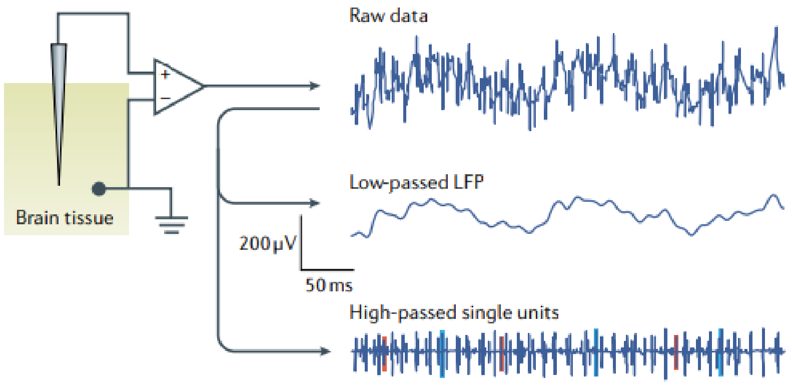
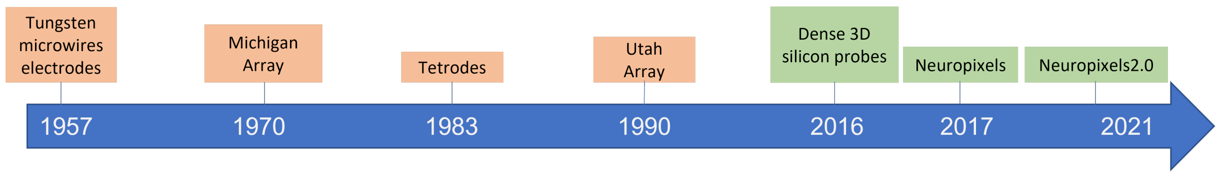


Figure 1.1: A schematic showing the signals obtained through an extracellular recording. At the top of the image the raw signal recorded by a single electrode. Below, the Local Field Potential obtained from the raw recording after low pass filtering it. At the bottom the spiking activity of the network obtained after applying a high pass filter to the raw recording. Image taken from [19].

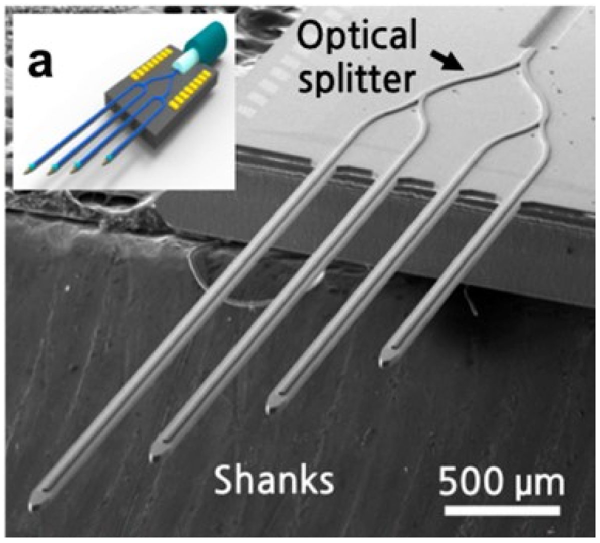
1.1.2. Recording Devices

Neuronal signals are measured by means of neuronal probes which have greatly changed and evolved since the beginning of neurophysiology (Figure 1.2.A). Classical recording devices can be broadly classified in three categories: tungsten microwires, tetrodes and silicon probes. Tungsten microwires have been the first conventional recording device, used because of tungsten high resistance to corrosion [14]. They feature very thin wires embedding a limited number of electrodes made of tungsten which provide a non-biological interface between the electrode and the neuron. Despite they can very well isolate activity of single neurons, extracellular electrophysiology requires to gather signals from many neurons. That need drove the development of tetrodes, which are bundles of low-impedance microwires twisted together. Thanks to greater size and lower impedance they can record signals up to almost 20 neurons [40]. The major drawback of tetrodes lies in their size as it makes them invasive for *in-vivo* recordings. Silicon probes managed to keep small dimensions with higher channel count exploiting micro-scale recording sites [39, 40]. They do not feature microwires anymore but metal or silicon shanks - also called needles - housing the recording lines, while the electrodes are placed either along the surface of the shank (Michigan silicon probes [3]) or at the tip of the shank (Utah silicon probes [30]). Development in microfabrication allowed to move from a single needle to 2D or 3D configurations (Figure 1.2.B and C). Moreover, the use of CMOS technology allowed a huge leap in the number of electrodes per needle, reaching 300 and more channels [39]. Application of CMOS technology represents a breakout point for neural probe manufacturing, providing high spatial integration thanks to higher number of electrodes, higher temporal stability and multifunctional integration. The high resolution of modern technology allows to get low-noise signals and to better isolate single units [24].

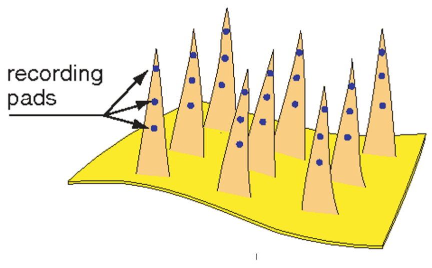
Micro-Electrode Arrays (MEAs) are chips for multi-site recording of extracellular signals. Signals are pre-processed on chip to provide signals of high quality. For example, dedicated circuitry provide on-site conditioning for weak neuron signals so that the recorded signal has an higher SNR. The new generation of CMOS-based MEAs (Figure 1.3) led to a huge increase in number of electrodes, reaching thousands on a single chip, which in turn allow for higher spatio-temporal resolution. It is now possible to selectively stimulate and record cells with a great precision [11, 15, 37]. Anyway, the higher dimensionality and redundancy of the recorded data lays new challenges for spike sorting algorithms, which, at the same time, have to be able to exploit the new information and have to scale well with high number of channels [7].



(A)



(B)



(C)

Figure 1.2: **(A)**: Timeline showing the evolution of neural recording technologies. Red indicates conventional technologies while green the modern ones. **(B)**: 2D probe for modulation of in-vivo signals. Image from [38]. **(C)**: 3D probe with multiple recording sites (black dots) on each needle. Image from [41].

1.2. Spike Sorting

Spike sorting is a technique whose goal is to reconstruct the spike trains of each neuron in the recording, i.e. to identify every time instant in which a neuron has fired an action potential. The general framework of a spike sorting algorithm is the following:

- Detect putative spikes
- Extract waveforms features
- Cluster waveforms based on features

Earlier spike sorting techniques required the manual intervention of an electrophysiologist to cut the putative spikes from the signal and cluster them based on explicit features like their shape. However, it is straightforward to understand how manual approaches do not scale well with high-channel count probes. (Semi-)automatic algorithms speed up

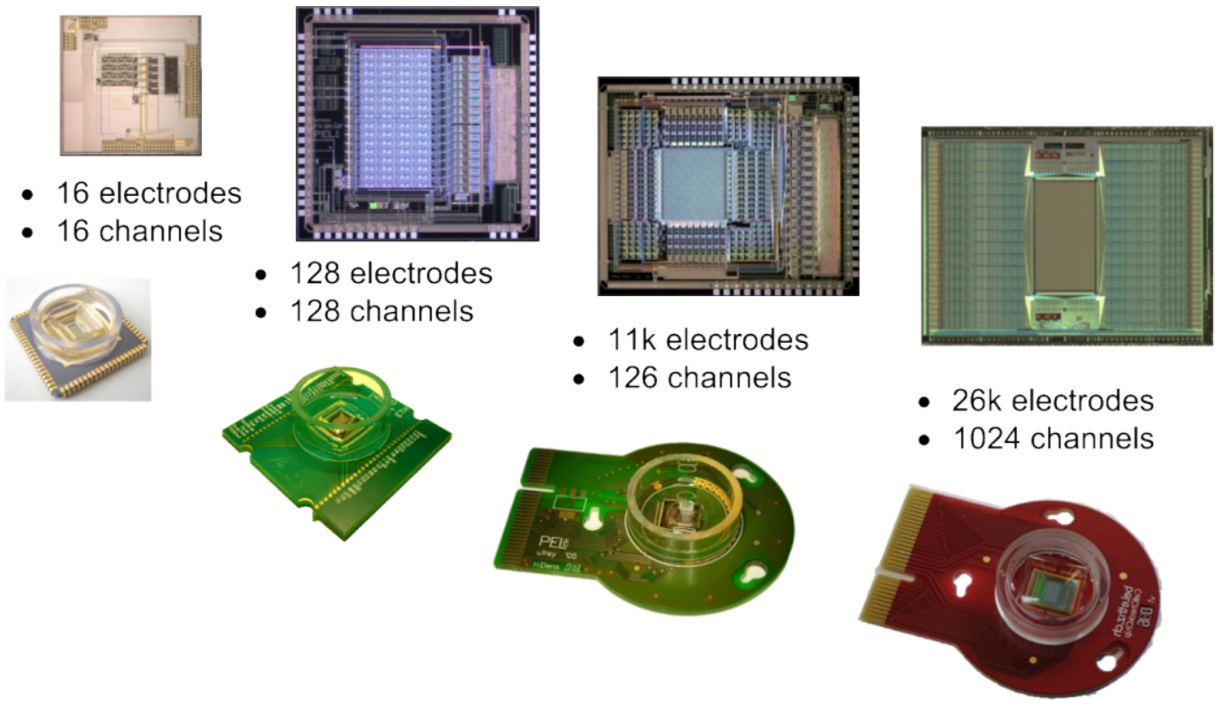


Figure 1.3: High-Density Micro-Electrode Arrays developed by ETH - Bio Engineering Laboratory (BEL-ETH) over the years. It shows how much the number of electrodes increased. Image from BEL-ETH website.

the entire process, remove user variability and outperform manual methods [16]. From the general framework we can identify 4 main steps in modern spike sorting algorithms: **spike detection**, **feature extraction**, **clustering** and **template matching**.

1.2.1. Spike Detection

Spike detection requires to distinguish neuronal activity from background noise. A *detection* function is applied on successive time windows of the signal returning a value that can be interpreted as the probability to have a spike in the analyzed time stamp. The most common method is to apply a voltage threshold to the signal and detect a spike whenever it crosses the threshold. The main problem of thresholding techniques is that they are not robust against noise [27], as the detection is not efficient with low $SNRs$. Classical solutions to improve efficiency leverage on the estimation of noise statistical model from data [31, 45]. However, all these techniques require both the estimation of model parameters, which need to be re-initialized periodically since biological signals are non-stationary, and of a threshold value. To counter this problem a template matching technique [27] has been published recently, which improves efficiency of spike detection with high noise levels at the cost of higher computational effort. Once the spike times

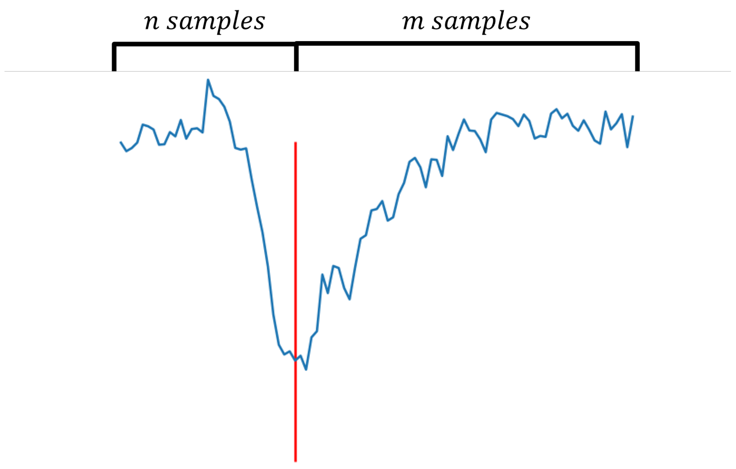


Figure 1.4: Spike waveform cut from a signal recorded at 32 kHz. The red line indicates the putative spike time. 32 and 64 samples have been selected respectively before and after the spike time.

have been estimated, spikes are cut from the signal selecting n samples before the spike time and m after it (Figure 1.4).

1.2.2. Feature Extraction

Each extracted spike waveform is composed by hundreds of samples per each channel, so it would be impossible to cluster in such a high-dimensional space. The approach is to decrease the dimensionality of the waveforms by extracting their most relevant features.

Due to heterogeneous morphology, electrical properties and positions with respect to the probe, spikes from different neurons normally exhibit different amplitudes and shapes [7]. These are useful characteristics to cluster them, but they have to be extracted from the spikes detected in the previous step. One common solution is the use of principal component analysis (PCA), which computes a new orthogonal basis that better describes the variability of the data. Data are first *standardized*:

$$z = \frac{(x_i - \bar{x})}{\sigma_x} \tag{1.3}$$

where $\bar{x}$ is the mean of the signal and σ_x its variance. Then the *covariance matrix* $\mathbf{C}$ is computed as:

$$\mathbf{C} = \frac{ZZ^T}{n-1} \tag{1.4}$$

where Z is the matrix of *standardized* variables and n is the number of samples. From $\boldsymbol{C}$, *eigenvalues* λ_i are computed and ranked from the highest to the lowest. The principal components are the *eigenvectors* $\boldsymbol{v}$ of $\boldsymbol{C}$ and the amount of variance explained by each v_i depends on its λ_i as $\frac{\lambda_i}{\sum_{j=1}^{k}\lambda_j}$. As a final step data are projected onto the new orthogonal basis as $\bar{Z} = \mathbf{v}^T Z^T$. The advantage of using PCA is that it provides an easy way to decrease the dimensionality of the input dataset. Indeed, one can select the first n components and the new dataset still explains a percentage of the original variance equal to:

$$explained\_variance = \frac{\sum_{i=1}^{n}\lambda_i}{\sum_{j=1}^{k}\lambda_j} \tag{1.5}$$

The perfect number of components to be chosen it is then a trade-off between not losing too much information and selecting few components to decrease the computational burden. However, the choice is often based on the so called *elbow rule*. By plotting the ratio of variance explained by each component vs their number (Figure 1.5), the line can ideally be approximated as an elbow. According to the rule, all the components after the *elbow*, that is the point at which the graph tends to be almost flat, are not significant and can be discarded.

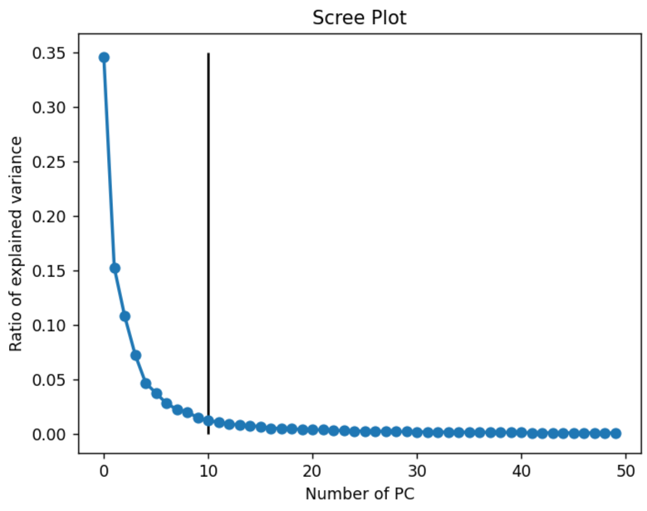


Figure 1.5: A Scree Plot for the first 50 Principal Components of a signal recorded by a 100-channel probe. The red line indicates the position of the elbow, meaning all components after the 10th can be discarded.

1.2.3. Clustering

During the clustering step, spikes are grouped based on similarity of their features. Many algortihms exists but they can be classified in seven groups [43].

- **Partitional**: Each waveform can be represented as a point in the feature-space. Partitional algorithms make clusters by dividing the feature space based on the density of points around centroids or medoids. Both are the center of a cluster, but a medoid must be an element of the dataset. Common partitional algorithms are K-means (which is based on centroids) and K-medoids (based on medoids as the name suggest). `Kilosort` [32] is a spike sorting algortihm which makes use of K-means for the clustering step.
- **Hierarchical**: They are based on dendograms or binary trees to separate points. The algorithm starts by considering all observations as separate clusters and iteratively merge the closest clusters until all of them are merged. At the end, it outputs a dendogram or a tree showing the relationship between each cluster. The definition of the closest clusters is based on a metric which may be the Euclidean distance [43]. A spike sorting algorithm exploiting hierarchical clustering is `Mountainsort` [12] which implements the ISO-SPLIT clustering algorithm.
- **Probabilistic**: Probabilistic algorithms aims at maximizing the posterior distribution as function of the probability that a data point belongs to a certain cluster. A common method is the *Expectation Maximization* (EM) which finds the *Maximum Likelihood* (ML) estimates of parameters of a distribution and it is implemented in `Klusta` sorting algorithm [36].
- **Density-Based**: Density-based algorithms do not need to know the number of clusters a priori and do not need any parameter. They are based on the assumption that clusters are contiguous space regions with high density of data points, separated by regions with low density of data points which are considered to be outliers. Common algorithms are `DBSCAN` and `OPTICS`. Density-based algorithms are used in `Ironclust` [24].
- **Graph-Based**: Data to be clustered can be represented as graphs where each data is a node and the distance is modeled by a weight on the edge linking two nodes. In graph-clustering, nodes are connected to other nodes within the same cluster but they have no link with nodes belonging to different clusters. A common graph-based clustering algorithm is *Minimal Spanning Tree* implemented in `Wave_clus` [33].
- **Fuzzy Logic**: It is also known as **soft-k-means**. Algorithms belonging to this

class allow data points to be members of more than one cluster.

- **Learning Based**: They are not a popular choice for spike sorting algorithms as they require either prior information about the data or ground-truth data. Indeed, before applying them a network needs to be trained with ground-truth data or parameters need to be set based on a prior information.

Each class of algorithms has its own advantages and disadvantages but all of them share three major limitations:

1. **High computational cost**: it is gaining more and more attention since probes manufacturing is moving towards higher and higher number of channels meaning the size of the input dataset is increasing as well.
2. **Over-split units**: the algorithm generates two or more clusters collecting spikes of the same unit and requires a curation step which can be either manual or automatic [12].
3. **Overlapping spikes**: spikes can overlap in space and/or time. Applying a spatial mask [36] or evaluating the location of the spike can help to distinguish temporal overlapping spikes but it cannot be applied to spatio-temporal overlapping spikes as those methods assume the sources to be distant from one to the other.

1.2.4. Template Matching

Template matching algorithms entered the game as an attempt to solve the problem laid by overlapping spikes. Templates are first estimated running a pre-clustering step on a subset of detected spikes. The templates are obtained as the average waveform in each cluster. Then, each template is matched with the recordings over time through a metric, for example by evaluating the cross correlation between the two [27], and when it exceeds a thresholds the templates are subtracted from the signal. Doing it recursively, allows to uncover overlapping spikes, which are clustered again. All template-matching algorithms have in common the assumption that the recorded signal *s(t)* is composed by a scaled sum of templates $\boldsymbol{T}$ with a superimposed noise *e(t)*:

$$s(t) = \sum_{ij} a_i \mathbf{T}_{ij}(t - t_i) + e(t) \tag{1.6}$$

the subscript j refers to neurons while subscript i refers to firing times. On the other hand they differ in the pre-clustering algorithm and the metric used for matching the templates and recordings. Based on the metric, conventional template-matching algorithms can be classified in three classes:

1. **Subtractive**: the Euclidean distance is computed between templates and recordings at each time step. The putative spike times are then identified as the time instants when the distance is small enough, i.e. lower than a threshold.
2. **Convolutional**: it is computed the convolution between templates and recording. The putative spike times are then identified as the time instants when the convolved signal is greater than a threshold.
3. **Matched Filtering**: it follows the same steps of convolutional algorithms but matched filters are used to compute the convolution instead of templates. Matched filters are the result of the correlation between templates and delayed recordings.

1.2.5. Challenges for modern spike sorters

In recent years, improvement in semiconductors manufacturing led to a huge increase of the number of electrodes on recording sensors, developing the so-called **High-Density Micro-Electrode Arrays** (HD-MEAs), featuring thousands of recording sites [25]. The larger number of recording channels increases the amount of spike collisions in the recording leading to a wider use of template matching approaches. At the same time, the modern devices laid the foundation for ICA-based spike sorting algorithms.

1.3. ICA

Independent Component Analysis is a blind source separation algorithm whose goal is to decompose the recorded signal into its sources. Since each output signal, called **Independent Components** (ICs), is focused on the activity of a source, ICA can also increase the SNR of the input dataset [25]. The hypothesis behind ICA is that signals are generated by an instantaneous linear combination of the sources, through a *mixing matrix* A:

$$\mathbf{r}(t) = A\mathbf{s}(t) \tag{1.7}$$

where $\boldsymbol{r}(t)$ are the recorded signals and $\boldsymbol{s}(t)$ are the sources. The *mixing matrix* is a matrix with shape NxN, with N features and N sources. Since the aim of ICA is to demix the signals, it estimates the *unmixing matrix* W which is the pseudo-inverse of A, such that:

$$\bar{\mathbf{s}}(t) = W\mathbf{r}(t) \tag{1.8}$$

where $\bar{\mathbf{s}}$(t) are the estimated sources. Since ICA algorithms output a square *unmixing matrix*, to retrieve all source signals it is needed that their number is at most equal to the number of features [22]. When dealing with neuronal recordings that constraint means that the number of neurons is lower than or equal to the number of recorded channels, that is why ICA is better suited for HD-MEAs. For the algorithms to be able to estimate $\bar{\mathbf{s}}(t)$, they must be **statistically independent**, meaning **non-Gaussian** [20].

1.3.1. Statistical Independence

Two events A and B are said to be independent if event A does not give any information about event B and vice-versa. Given P(A) and P(B) the probabilities of event A and event B respectively, the two events are independent if $P(A \cap B) = P(A)P(B)$. Extended to n events $A_i, ..., A_n$, it leads to $P(\cap_{i=1}^{n} A_i) = \prod_{i=1}^{n} P(A_i)$. A consequence of independence is that $E(AB) = E(A)E(B)$ meaning that A and B are uncorrelated too: $C(A, B) = E(AB) - E(A)E(B) = 0$ [44]. Hyvärinen shows that non-Gaussian variables are independent too [21]. The main idea is that from the Central Limit Theorem we know that the distribution of the sum of two or more independent random variables tends to be Gaussian, under certain conditions. Therefore, the single variables (the sources in the case of ICA) are less Gaussian than their sum (the recorded signal). Examples of non-Gaussianity measures are **kurtosis**, **skewness**, **negentropy** and **mutual information**.

Kurtosis is a measure of "peakedness" of a curve and it is computed as the ratio between the fourth central moment μ_4 and the fourth power of the standard deviation σ^4:

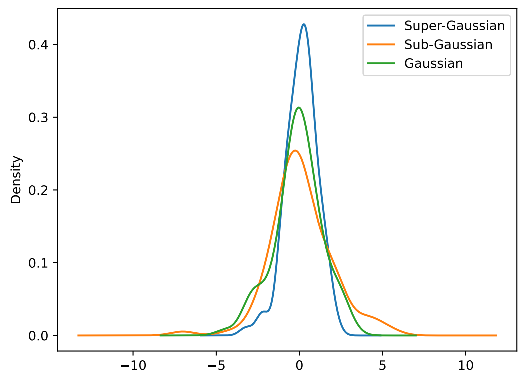
$$\mathbf{K} = \frac{\mu_4}{\sigma^4} = E[(\frac{X-\mu}{\sigma})^4] \tag{1.9}$$

Since for a normal distribution $\boldsymbol{K} \approx 3$, it is often evaluated as $\boldsymbol{K} = \frac{\mu_4}{\sigma^4} - 3$ to get a quicker comparison between the analyzed distribution and a Gaussian one. Depending on the value of $\boldsymbol{K}$ a distribution can be classified as *supergaussian* if $\boldsymbol{K} > 3$, *gaussian* if $\boldsymbol{K}$ $= 3$ and *subgaussian* if $\boldsymbol{K} < 3$ (Figure 1.6). In the former case the distribution is more "peaked" than a Gaussian one, while in the latter it is more flat.

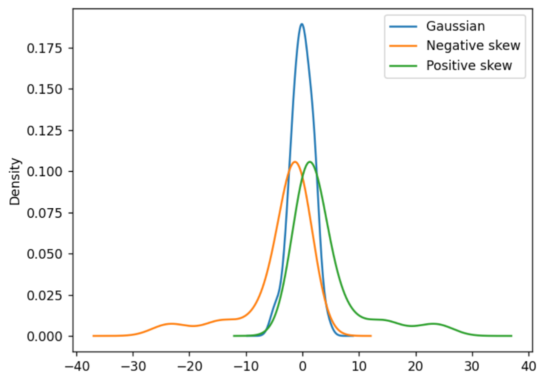
Skewness is a measure of the asymmetry of a distribution and it is computed as the ratio between the third central moment and the third power of the standard deviation:

$$\mathbf{S} = \frac{\mu_3}{\sigma^3} = E[(\frac{X-\mu}{\sigma})^3] \tag{1.10}$$

A distribution is said to be positively skewed when $\boldsymbol{S} > 0$, meaning the tail of the distribution tends towards positive values. On the contrary, it is negatively skewed when $\boldsymbol{S} < 0$ and its tail tends towards negative values (Figure 1.6).



(A)



(B)

Figure 1.6: In both images the *probability density function* of three distributions are plotted. **(A)**: In orange, a distribution with kurtosis lower than 3. In blue, a *supergaussian* distribution whose kurtosis is greater than 3. In green, a Gaussian distribution with a kurtosis equal to 3. **(B)**: In orange, an asymmetric left-tailed distribution. In blue, a symmetric Gaussian distribution. In green, an asymmetric right-tailed distribution.

Negentropy Negentropy is another direct measure of **non-Gaussianity** based on entropy. Entropy is a measure of randomness of data, the higher the entropy, the higher its uncertainty. The original definition of *Entropy* provided by Shannon follows:

$$H(X) = -\sum_{x} P(x) log P(x) \tag{1.11}$$

$$H(X,Y) = -\sum_{x,y} P(x,y) log P(x,y) \tag{1.12}$$

where $H(X)$ is the *entropy* for a random variable X and $H(X,Y)$ is the *joint entropy*. Now, we can define the *negentropy* $J(Y)$ as:

$$\mathbf{J}(Y) = H(Y_{gauss}) - H(Y) \tag{1.13}$$

where Y_{gauss} has a Gaussian distribution with same covariance of Y. The entropy of a Gaussian variable is large, while it is small for a non-Gaussian one. Hence, the *negentropy* of a non-Gaussian variable is large, that is why negentropy-based algorithms estimate the

Independent Components by maximizing $\boldsymbol{J}(Y)$. Unfortunately it is tricky to evaluate negentropy with equation 1.13, so Hyvärinen and Oja proposed a new evaluation of $\boldsymbol{J}(Y)$ for their **fastICA** algorithm [20]:

$$\mathbf{J}(\bar{Y}) = E[\phi(\bar{Y})] - E[\phi(U)]^2 \tag{1.14}$$

with $\bar{Y}$ as standardized Y, U as standardized Gaussian variable and $\phi(.)$ as non-quadratic function, e.g. *tanh(.)*. Now, the algorithm proceeds by estimating the unmixing matrix W through the following optimization problem:

$$\arg\max_{\mathbf{w}_i} \sum_{i=1}^{n} \boldsymbol{J}_G(\boldsymbol{w}_i x_i(t)) \; wrt. \; \boldsymbol{w}_i, \; i = 1, ..., n \tag{1.15}$$

with each $\boldsymbol{w}_i$ representing a row of W. The sources $\boldsymbol{s}_i(t)$ are then provided by equation 1.8.

Mutual Information Mutual Information is a measure of dependence between two variables. It is closely related to *negentropy* if the sources are *uncorrelated* and of unit variance [21]. The *mutual information* for n scalar random variables is defined as:

$$I(y_1, y_2, ..., y_n) = \sum_{i=1}^{n} H(y_i) - H(\boldsymbol{y}) \tag{1.16}$$

In the case of an invertible linear transformation, e.g. $\boldsymbol{y} = W\boldsymbol{x}$, the above equation change slightly:

$$I(y_1, y_2, ..., y_n) = \sum_{i=1}^{n} H(y_i) - H(\boldsymbol{y}) - log|detW| \tag{1.17}$$

It can be proved that $detW$ is constant, if y_i are *uncorrelated* and of unit variance [21]. Therefore, equation 1.17 can be rewritten as:

$$I(y_1, y_2, ..., y_n) = C - \sum_{i} J(y_i) \tag{1.18}$$

where $J(y_i)$ is the negentropy of the variable y_i.

1.3.2. Previous work

After the advent of HD-MEAs, the application of ICA for spike sorting has gained more attention leading to the publication of many works.

In Jackel et al. [25] ICA is applied recursively to the data, similarly to what happens in template matching algorithms. At each iteration ICA is applied to the input dataset and spikes are detected by a thresholding technique, then the detected waveforms are clustered to identify the template of each neuron and finally the detected templates are subtracted from the recordings. The dataset made up by the residual signal is the input of the next iteration. Thanks to the iterative subtraction it is possible to disclose overlapping spikes. Moreover, the authors propose the following metrics to evaluate the applicability of ICA to HD-MEA recordings:

- ***SNR* of neurons**: it is compared the SNR of neurons in the recordings (SNR^i_{el}) and in the ICs (SNR^i_{ic}) through the following equations:

$$SNR^i_{el} = \frac{\max[abs(\boldsymbol{f}^i_j)]}{\sigma_j} \tag{1.19}$$

$$SNR^i_{ic} = \max(|\boldsymbol{H}^i|) \tag{1.20}$$

  where $\boldsymbol{f}^i_j$ is the average spike waveform of the neuron i recorded by the channel j, σ_j is the standard deviation of the channel j and $\boldsymbol{H}^i$ is the average spike waveform of neuron i after demixing the signal through ICA.

- **Separability**: it measures how well ICA can demix the signals. The main idea behind it is that a neuron k is well demixed if its signal has a large peak on the IC i while the signals of all the other neurons have a small peaks on the same IC i.

- **Redundancy**: it counts the number of signals in which the signal of a neuron overcomes a threshold.

it is shown that ICA can increase both SNR and separability and decrease the redundancy of the data with respect to the original recordings.

Another application of ICA is provided by Buccino and colleagues [9]. Now after the estimation of the ICs, the algorithm attempts to correct the data to compensate for ICA limitations. First, ICs are *cleaned* by removing ICs with low skewness as they carry very few spikes. Second, spikes are detected from the *cleaned* ICs when the signal exceeds 4 times the Mean Absolute Deviation (MAD). Third, *mixture of Gaussians* are applied to each source separately to cluster their spikes. In the ideal case each IC should carry the spikes of a single source but due to the non-linearity of templates [25] it is possible that an IC carries some spurious spikes of other neurons. Clustering each source separately makes it possible to discard the spurious spikes. Finally, spike trains are compared to each other to identify duplicates due to the non instantaneous mixture of the signals. If two spike trains share more than 80% of spikes, the one with lower spikes is discarded.

Leibig et al. [28] suggest the use of *convolutive ICA* (cICA) to tackle the problem of duplicates due to the non-instantaneous mixture of the signals. cICA estimates a set of L mixing matrices, so the new formulation of equation 1.7 is the following:

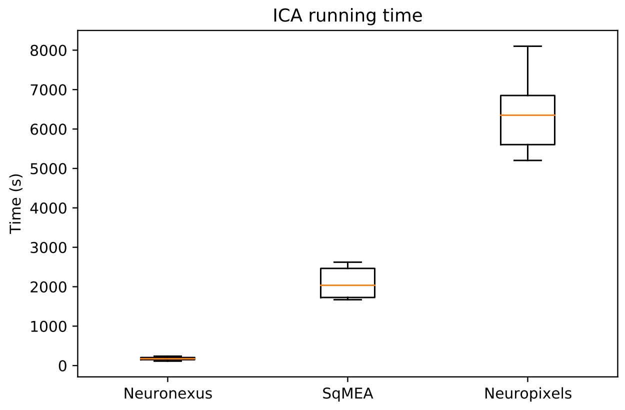
$$\boldsymbol{x}(t) = \sum_{\tau=0}^{L} A_{\tau} \boldsymbol{s}(t-\tau) \tag{1.21}$$

after the estimation of the ICs, spikes are detected directly on them by evaluating when the signals exceed 3 times the MAD and then spikes are clustered with `KlustaKwik` algorithm [36].

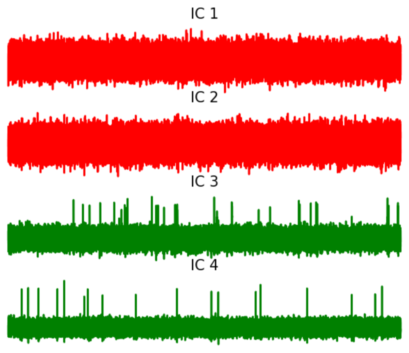
cICA estimates ICs with higher SNR and lower redundancy compared to classical ICA. Moreover, the spike sorting algorithm based on cICA proposed by Leibig et al. [28] achieves a lower error rate with respect to ICA-based algorithms. Nevertheless, cICA greatly increases the computational cost making it unfeasible applying cICA to large recordings. Indeed, the cited cICA-based algorithm with L=3 took ≈40 minutes on a simulated recordings of 5 seconds only.

1.3.3. ICA Constraints and Limitations

Recalling ICA assumptions about the data, neuronal recordings must be generated by **statistically independent** sources and by a **linear mix** of them. Although neurons do not behave independently with respect to other neurons in the network, ICA can still be applied as it requires an instantaneous independence which is satisfied assuming that sets of neurons do not fire *exactly* at the same time every time [9, 25]. On the contrary, the hypothesis of instantaneous **linearity** between sources and signals is not fully met because of the filtering behavior of dendrites. Indeed, the peaks of signals recorded close to the soma or along the dendrites are not at the same time instants due to the propagation time of the action potential that generated them. The ICA model may interpret the delayed spikes as belonging to different sources which may lead to the estimation of duplicate sources, i.e. ICs tuned to the same neuron, and requires a duplicate-rejection step [7]. The ICA algorithm used in this work is the above mentioned **fastICA**, available from `SCIKIT-LEARN` python package. The package allows to perform a *pre-whitening* of the data which is highly recommended to improve convergence [20]. Unfortunately, despite the "fast" in the name of the algorithm, when it comes to recordings made with large number of channels it has a high estimation time (Figure 1.7.A) which is further increased



(A)



(B)

Figure 1.7: **(A)**: Average time required by fastICA on recordings simulated with three different probes: Neuronexus (32 channels), squared MEA (100 channels) and Neuropixels (128 channels). For every probe the average has been computed on 5 simulated recordings which were 120 s long, sampled at 32 kHz (3.840.000 samples per channel) and 10 μV of noise level (see section 2.1 for a detailed description of the simulated datasets). From left to right the average computation time is: 175.48 ± 43.17 s, 2104.23 ± 382.93 s and 5251.59 ± 141.27 s. **(B)**: The 4 Independent Components estimated from a simulated 4 channel recording with 2 neurons. In red, ICs carrying only noise that must be discarded. In green, ICs focused on the 2 neurons that must not be discarded during the cleaning.

by the whitening phase, shown by the following equations [46]:

$$flops_{whitening} = 2K^2T \tag{1.22}$$

$$flops_{fastICA} = 2(K+1)T \; per \; iteration$$

where K is the number of channels and T the number of samples. None of the previous works exploiting ICA for spike sorting tackled this problem, so the first step of this project aimed at improving ICA efficiency to speed up sources estimation by decreasing the dimensionality of the input data without losing crucial information for the estimation. The second problem we had to deal with is related to the estimation of false sources. If we recall how the *unmixing matrix* W is computed, we can notice it is a square matrix, meaning ICA always outputs a number of estimated sources equal to the number of channels. So, if the recording probe has M channels and the signals are generated by $N < M$ neurons, ICA will provide $M - N$ false sources which are composed by noise only (Figure 1.7.B).

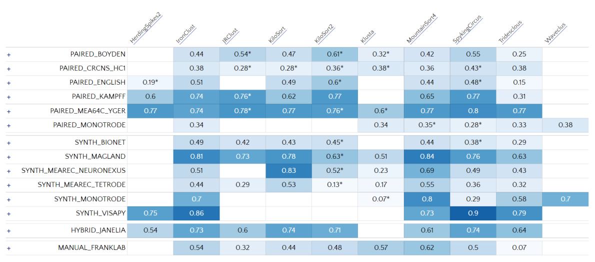
1.4. Aim of the work

Given the above-mentioned limitations of ICA for spike sorting, the overall goal of this work is to investigate and benchmark new strategies to use ICA to improve spike sorting performance.

In recent years the number of spike sorting algorithms - which I am going to address as sorters - has increased exponentially. Their performance on different recordings can be found on `SpikeForest` [29]. It is a useful platform to compare spike sorting algorithms by modifying many parameters like the minimum SNR of recordings used for testing. It has a huge dataset of recordings which, at the time of writing, total 1.3 TB of recordings and 34,773 of ground-truth units. Decreasing the minimum SNR shows how the performances of all sorters drop dramatically, especially when $SNR \leq 3$ as you can see from Figure 1.8. An interpretation of these result is that existing sorters usually fail to discover small units or equivalently fail to discover units within low-SNR recordings, but ICA can increase the SNR of neuronal signals [25]. That is why the last step of this work aimed at improving sorting performances by applying ICA, as a postprocessing phase, to increase the number of neurons detected by sorters.

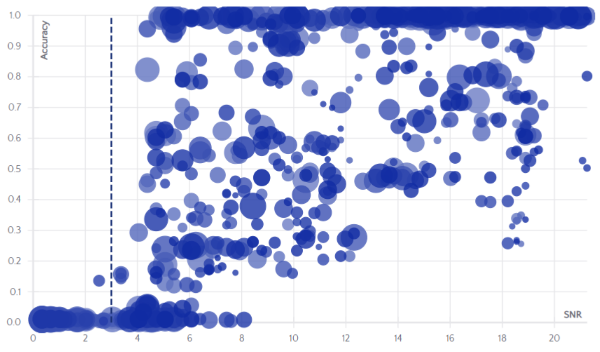
To summarize, the main objectives of the work are the following ones:

- Improving efficiency and decreasing running time of ICA.
- Increasing SNR of neuronal recordings through ICA.
- Improving performances of existing spike sorting algorithms on recording with small neurons.



		HerdingSpikes2	IronClust	JRClust	KiloSort	KiloSort2	Klusta	MountainSort4	SpykingCircus	Tridesclous	Waveclus
+	PAIRED_BOYDEN		0.44	0.54*	0.47	0.61*	0.32*	0.42	0.55	0.25	
+	PAIRED_CRCNS_HC1		0.38	0.28*	0.28*	0.36*	0.38*	0.36	0.43*	0.38	
+	PAIRED_ENGLISH	0.19*	0.51		0.49	0.6*		0.44	0.48*	0.15	
+	PAIRED_KAMPFF	0.6	0.74	0.76*	0.62	0.77		0.65	0.77	0.31	
+	PAIRED_MEA64C_YGER	0.77	0.74	0.78*	0.77	0.76*	0.6*	0.77	0.8	0.77	
+	PAIRED_MONOTRODE		0.34				0.34	0.35*	0.28*	0.33	0.38
+	SYNTH_BIONET		0.49	0.42	0.43	0.45*		0.44	0.38*	0.29	
+	SYNTH_MAGLAND		0.81	0.73	0.78	0.63*	0.51	0.84	0.76	0.63	
+	SYNTH_MEAREC_NEURONEXUS		0.51		0.83	0.52*	0.23	0.69	0.49	0.43	
+	SYNTH_MEAREC_TETRODE		0.44	0.29	0.53	0.13*	0.17	0.55	0.36	0.32	
+	SYNTH_MONOTRODE		0.7				0.07*	0.8	0.29	0.58	0.7
+	SYNTH_VISAPY	0.75	0.86					0.73	0.9	0.79	
+	HYBRID_JANELIA	0.54	0.73	0.6	0.74	0.71		0.61	0.74	0.64	
+	MANUAL_FRANKLAB		0.54	0.32	0.44	0.48	0.57	0.62	0.5	0.07	

(A)



(B)

Figure 1.8: **(A)**: Dataframe from **Spikeforest** showing accuracy of sorters on different datasets with minimum $SNR = 3$. The rows refers to different datasets while the columns refers to different sorters. The star near accuracy values indicates the algorithm failed to converge. **(B)**: Scatter plot showing *unit accuracy* vs SNR for `Kilosort2` on **HYBRID_JANELIA** dataset. From the plot it can be seen that the *unit accuracy* dramatically drops when the $SNR \leq 3$.

2 | Materials and Methods

The chapter is organized as follows: first I cover how simulated and real recordings are obtained. Second, I present the spike sorting framework used throughout the work. Finally, I outline how ICA can be used for spike sorting purposes.

2.1. Datasets

Recordings of large networks lack ground truth information [8] making validation of spike sorting algorithms tricky. Hence, for the validation of the work recordings have been simulated in `MEArec` [8] [`https://github.com/alejoe91/MEArec`] framework which provides simulated neuronal recordings with ground-truth information.

2.1.1. Simulation of Ground-Truth recordings

Simulated recordings are obtained using the `MEArec` [8] Python package. The package allows to simulate neuronal recordings starting from probe and cell models which can be loaded from existing databases [1, 34]. The simulation consists in two steps: **Template Generation** and **Recording Generation**.

Template Generation The template generation step is split in two phases: *intracellular stimulation* and *extracellular stimulation*. During the *intracellular stimulation*, an intracellular stimulation is simulated, using the software NEURON, for each available cell model. This step generates a set of transmembrane currents which are saved and reloaded for the actual generation of the templates. During the *extracellular stimulation* every cell is randomly shifted and rotated around the selected probe n times, chosen by the user. For every new position the transmembrane currents are loaded and EAPs are computed through the LFPy package [2] (Figure 2.1). The higher the number of templates, the higher the number of different recordings that can be simulated for the same set of cells and probe. Cell models can be loaded in the package from online datasets; for this project 13 cortical cell models of layer 5 from Neocortical Microcircuit Portal (NPC) [34] have been used. Moreover, three sets of templates have been simulated for the work. For each

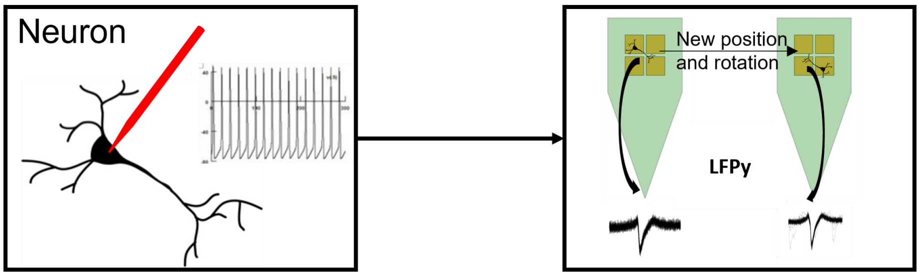


Figure 2.1: MEArec template generation pipeline. First transmembrane currents are simulated through NEURON (left), then EAPs are computed by LFPy package using results from intracellular stimulation (right).

set, all 13 cell models have been used and 100 EAPs with amplitude $\in [5, 300]$ μV have been simulated per cell. The three sets differ in the probe model selected for the simulation. The three selected models (provided by `MEArec`) are: the model of Neuronexus A1x32-Poly3-5 mm-25s-177-CM32 probe with 32 channels (Figure 2.2.A), the model of a squared MEA featuring 10x10 channels with 15 μm pitch (Figure 2.2.B) and the model of a Neuropixels probe with 128 channels [39] (Figure 2.2.C).

Recording Generation In the second phase, the templates and simulated spiketrains are combined to generate the recordings (Figure 2.3). The process can be finely controlled by many parameters which allow to characterize the signal in different ways, from modulating the drift of the signal to choosing the amount of bursting neurons. From the cell models used in phase one, it is possible to select how many excitatory (n_{ex}) and inhibitory (n_{in}) cells are used during the simulation. The simulation starts with the generation of $n_{ex}+n_{in}$ spike trains, which are modeled either as Poisson or Gamma distributions. Now, based on a set of parameters defined by the user, e.g. the minimum and maximum amplitude, templates are selected and convolved with spike trains using a modulated (or customized) convolution. Finally, noise is added to the convolved signal and, if selected, recordings are filtered.

For each one of the three probes used for **template generation** we simulated four groups of recordings. Each recording is 120 s long, it is sampled at 32 kHz and it has a colored background noise. They differ in the following parameters:

- **Number of cells**: The number of electrodes in HD-MEAs is assumed to be greater than the number of neurons recorded. To keep the simulated recordings consistent with the real ones, we chose to use a number of cells equal to half of the number of channels. Therefore, we simulated recordings with 16 cells for Neuronexus tem-

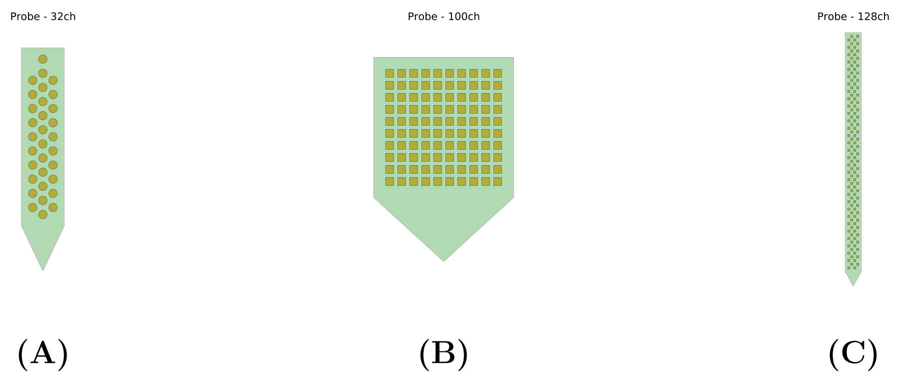


Figure 2.2: The three probe models used for the template generation step. From left to right: a Neuronexus probe with 32 channels, an high-density MEA with 100 channels and the Neuropixels probe with 128 channels.

plates, with 50 cells for squared MEA templates and with 64 cells for Neuropixels templates.

- **Noise level**: The noise level is the standard deviation of the signal measured in μV. Within each group we kept fixed the number of cells and varied the noise level. So, for each probe we selected four noise levels equal to 5, 10, 20 and 30. For each noise level, 5 recordings have been simulated with different seeds to test the generalization capability of the algorithms.

To summarize, we have 3 sets of recordings with 16, 50 and 64 cells. Within each set the recordings differ in the noise level which increases from 5 to 30 and for each noise level there are 5 different recordings with randomly selected templates, spike trains and noise realizations.

2.1.2. Real Recordings

Biological preparations Cells are extracted from embryons of 18-day Wistar rats following a standard protocol [35]. The cultures are kept in a humidified cell-culture incubator at 37°C and 5% CO2/95% air and plated on a CMOS HD-MEA. All experimental protocols were approved by Basel-Stadt veterinary office and follows Swiss federal laws on animal welfare and a detailed explanation of the protocol, developed by ETH Bio-Engineering Laboratory, can be found at Protocol Exchange [4].

CMOS HD-MEA The probe used for recording is a CMOS-based HD-MEA developed by the Bionegineering Laboratory (BEL) of ETH in Basel. It features 26'400 electrodes with 17.5 μm pitch and up to 1'024 can be routed and recorded simultaneously.

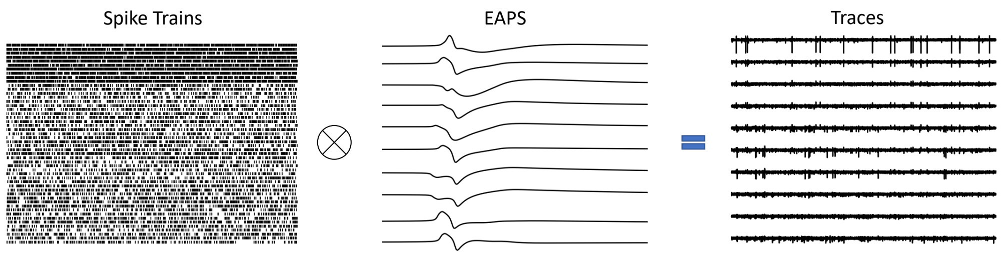


Figure 2.3: MEArec recording generation. Spike trains are generated as Poisson or Gamma distributions. Templates are then convolved with spike trains and background noise is added as final step.

Neurons tend to cluster together [26] leading to a very low activity in some region of the culture. To remove *silent* channels from the acquisition, an *activity scan* is performed before recording the signal. During the scan, the acquisition software (MaxWell Lab) divides the array in 7 columns which are analyzed independently. After the analysis of all the columns, the software records the signal from the most active channels among all the columns. We recorded from two chips for 5 minutes. The number of channels selected by the *activity scan* were 534 and 526 for the first and second chip respectively.

2.2. Spike sorting framework

To handle both simulated and real recordings, to run spike sorting algorithms and to validate their outputs we used `SpikeInterface` [10] [`https://github.com/SpikeInterface-/spikeinterface`], a Python package collecting tools to ease processing and spike sorting for different format of recordings. It provides: tools to read recordings and output of spike sorting algorithms of different formats; many existing spike sorting algorithms; a framework to compare them against each other or against ground-truth data (if available). All the tools and methods developed in this work are highly integrated with the `SpikeInterface` framework.

I am going to describe more in detail only the modules heavily used in this work, for a complete explanation of the package I refer the reader to `SpikeInterface` paper [10].

2.2.1. Spike sorters used

The work makes use of three spike sorting algorithms that are integrated in `SpikeInterface`: `Herdingspikes` [17], `Ironclust` [23] and `Kilosort2` [32]. They are all designed for HD-MEAs.

Herdingspikes algorithm is split in two phases: in the first one spikes are detected by applying a threshold to the recorded signal, then the location of each detected event is estimated by evaluating their spatial spread over the recording electrodes and finally the locations are used to train a *support vector machine* to classify events as true spikes or noise. In the second step spikes are clustered with the *mean shift* algorithm [13]. It depends on a single parameter (the `bandwidth` h) which is related to the expected size of each cluster, so it does not require prior knowledge about the final number of clusters. It is possible to parallelize this step to speed up the process.

Ironclust spike sorting pipeline starts by high-pass filtering the signal to remove local field potentials (LFPs). Additionally, notch filters can be used to remove high-frequency components that overlap with spikes spectrum. Next, spikes are detected when the signal exceeds six times its Mean Absolute Deviation (MAD). Then spikes are used both to estimate the location of each spike and to track probe drift (it is based on the variation of spike amplitudes over time). Tracking the probe drift is an important feature as changing the relative position between neurons and electrodes modifies the amplitude of recorded spikes which can lower the performance of the clustering step. Before running the clustering step, the two most relevant features are extracted from the detected spike waveforms, through PCA. The clustering algorithm is `DPCLUS` [5], a density-based algorithm.

Kilosort2 filters data with an high-pass filter at 300 Hz to remove LFPs and it whiten the data to remove correlated noise. Now, spikes are detected via template matching. The templates are first initialized and at each iteration they are recomputed (the authors call this step *templates optimization*) using the detected spikes. The loop, alternating template optimization and spike detection goes on until a cost function reaches convergence. A final clustering step is run to merge oversplitted neurons.

2.2.2. Comparing spike sorting outputs

`SpikeInterface` [10] allows to compare sorting outputs with a ground truth, if available, or to compare multiple sorting objects. The output of a spike sorting algorithm is made up by N vectors of spike times, with N the number of neurons found by the algorithm.

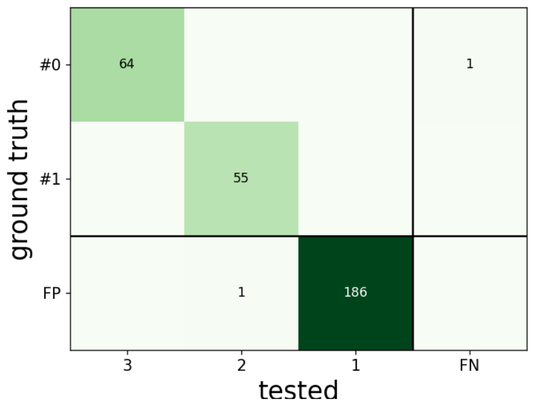
The spike times are the time instants in which a spike has been detected. The comparison is made by counting the matching events, i.e. spike times, between the compared objects. For the sake of clarity I provide an example between a sorting object and a ground truth one (GT), but it can be extended to comparison of two or more sorting outputs. Say that from `MEArec` we simulate a recording with N units while the sorter estimates M units. From the ground-truth data (GT) we can access N spike trains, i.e. sequence of spike times of a single unit, while from the sorting we can access M spike trains. For every pair of GT and sorted units we first count the number of matching events, i.e. how many elements of the vectors are within a small window (0.4 ms). The results make up a matrix, that we can call `matching_matrix`, of shape NxM whose values are the number of matched spikes for every pair of GT unit i and tested unit j. From it, the `confusion_matrix` can be evaluated as:

1. *True positive*[i, j] $= matching\_matrix[i, j]$
2. *False negative*[i, j] $= n_{GT}[i] - matching\_matrix[i, j]$
3. *False positive*[i, j] $= n_{sorting}[j] - matching\_matrix[i, j]$

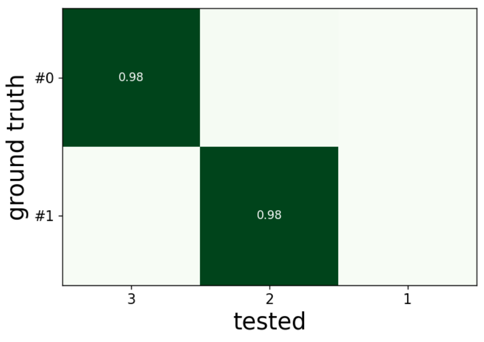
where i index refers to the i^{th} unit from GT and j the j^{th} unit from the sorting; n_{GT} is a vector of length N whose values are the number of spikes of each unit; $n_{sorting}$ is a vector of length M whose values are the number of detected spikes of each unit. From the `confusion_matrix` (Figure 2.4.A), the performance of the sorter is evaluated by means of the following quantities:

1. $Accuracy = \frac{tp}{tp+fp+fn}$
2. $Recall = \frac{tp}{tp+fn}$
3. $Precision = \frac{tp}{tp+fp}$
4. $False\ Discovery\ Rate = \frac{fp}{tp+fp}$
5. $Miss\ Rate = \frac{fn}{N}$

with $tp = True\ positive, fp = False\ positive, fn = False\ negative$. Using the *Accuracy* values of every pair of ground-truth and sorted units (i, j), we can build the `agreement_matrix` (Figure 2.4.B) which is again a matrix of shape NxM. Exploiting the values of the agreement matrix, each GT unit is associated to a sorted unit and vice versa by the Hungarian matching algorithm so that a sorted unit is associated, at most, to only one GT unit. After the unit matching, a sorted unit j can be classified as *well detected* if the **agreement score** with its matched GT unit i ((i, j) value of the `agreement_matrix`) is higher than a threshold, which by default is set to 0.8.



(A)



(B)

Figure 2.4: **(A)**: matrix showing *true positives*, *false positives* and *false negatives*. The three values are separated by the two thick black lines. The last column shows the *false negatives*, i.e. missed spikes, for each GT unit (vertical axis). The last row shows the *false positives*, i.e. spikes that are missing from GT spike trains, for each tested unit (horizontal axis). The upper left block shows the *true positives* for each pair of tested and GT unit. **(B)**: Agreement matrix. The (i, j) value is the accuracy of the sorted unit j with respect to the GT unit i.

2.3. ICA for spike sorting

The current section shows pre- and post-processing steps to tackle ICA limitations and possible ways to use ICA for spike sorting. It is organized as follows: first, I introduce a pre-processing step to increase ICA efficiency; second, I present a post-processing step to remove *false* neuronal sources (see subsection 1.3.3); next, two different approaches show how to increase the SNR of neuronal recordings using ICA as a pre-processing tool; finally, I show how to improve spike sorting algorithms performances using ICA as a post-processing tool.

2.3.1. Peak Selection

As reported in subsection 1.3.3 and Figure 1.7.A the computational burden of ICA is significantly high. To speed up the independent components estimation, we attempted to decrease the dimension of the input dataset without decreasing the *goodness* of the ICs. Neurons respond to stimuli in a sparse way, meaning they rarely fire [6, 42]. For example, considering a sensory cortex neuron with a firing rate range of [4.94; 7.22] Hz [6], we can say that it emits a spike every 0.14 s at most. If the signal is sampled at 20 kHz, 2800 samples (0.14 s) are recorded without any spike from that neuron. Since ICA does not follow temporal patterns in the signal but it assumes the recordings to be an instantaneous mixture (see equation 1.7) it is possible to run the model on a subsampled version of the original dataset. First, a spike is detected when the signal **Mean Absolute Deviation** crosses a threshold, which can be selected externally and equals to 5 by default. The detection is *locally exclusive* meaning if a spike is detected on n channels, it is selected only on the channel with highest amplitude but not on neighboring channels which are all the channels within a radius r from the peak channel. By default $r = 100$ μm. Then, for each detected spike a window is selected across all channels as:

$$w = [spike\_time - n_{before};\ spike\_time + n_{after}] \tag{2.1}$$

where n_{before} is the number of samples selected before the spike time and n_{after} the number of subsequent ones. The waveforms obtained are finally concatenated channel by channel to reconstruct the time sequence (Figure 2.5). The resulting dataset has a smaller dimensionality due to the removed samples and it is more focused on spikes which carry the important information for IC estimation for spike sorting purposes. The size of the window can be chosen externally by providing the amount of ms to select before and after the spike time, then based on the sampling frequency of the recording, the number of samples is simply computed as:

$$n_{after} = \frac{ms_{after} * fs}{1000} \tag{2.2}$$

$$n_{before} = \frac{ms_{before} * fs}{1000} \tag{2.3}$$

If the number of neurons in the recording is very high, the size of the dataset may still be very large. That is why we decided to give the possibility to use a percentage of the selected spikes, meaning after spike detection (Figure 2.5.A), waveforms are selected for a sub-set of detected peaks. The spikes can be either selected randomly or by balancing them across the channels. In the latter case, the algorithm picks the selected percentage of samples with the following steps channel by channel:

1. Count how many spikes are detected in channel i
2. Select $n_{selected,i} = n_{spikes,i} * p$

with $n_{spikes,i}$ the total number of detected spikes in channel i and p the selected percentage of spikes to be used. The suggested option is to balance the spikes as a pure random sampling may select too many spikes from few channels. In that case, the ICA estimation

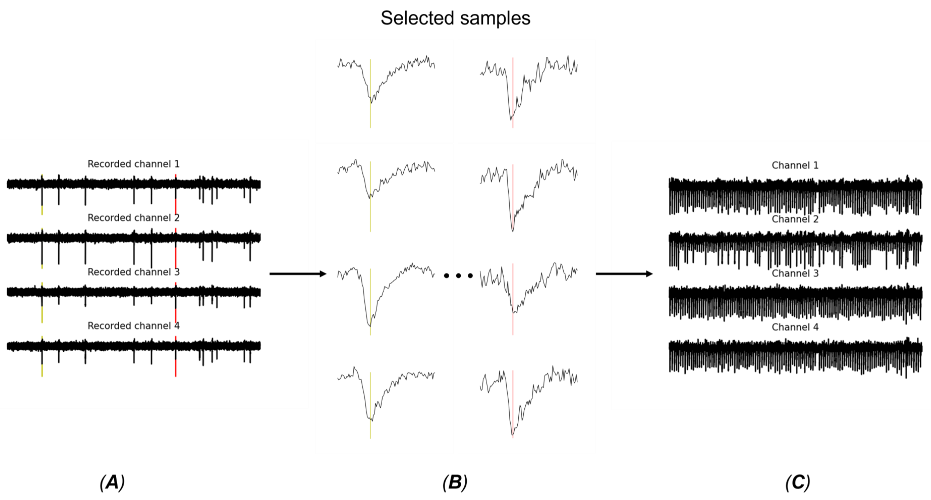


Figure 2.5: Full pipeline of subsampling. **(A)**: spikes are first detected channel by channel. **(B)**: a window of $n_{before} + n_{after}$ samples is selected across each detected spike. **(C)**: waveforms obtained at previous step are then concatenated to reconstruct the signal. In the first two images, I marked only two of the detected spikes for sake of clarity.

might not exploit the full variability of spiking activity in the original recording.

2.3.2. Source Cleaning

ICA estimates a square *unmixing matrix* W, whose dimension is NxN with N being the number of channels. As the sources are obtained by equation 1.8, the algorithm estimates N sources. If N is greater than the number of neurons, some ICs are not focused on any neuronal source, i.e. they carry noise only. In *true* neuronal sources, the presence of spikes makes the signal asymmetric. The distribution of additive noise, instead, can be assumed to be more symmetric. That is why a possible solution to distinguish between *true* or *false* neuronal sources lies in evaluating their symmetry.

The most common statistic measure of symmetry is the **skewness**, which is the third central moment and it is evaluated as:

$$\bar{\mu_3} = \frac{\mu_3}{\sigma_3} = E[(\frac{X-\mu}{\sigma})^3] \tag{2.4}$$

where μ is the mean and σ the standard deviation. The **skewness** of a symmetric distribution equals zero, while it increases the more asymmetric the distribution is. So, by setting a threshold it is possible to discard *false* sources (Figure 2.6). The threshold can be set externally and by default is 0.1.

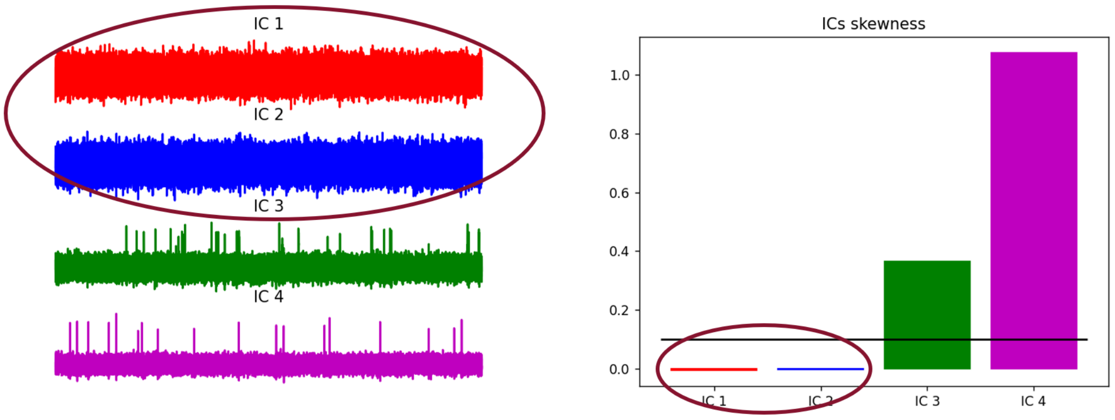


Figure 2.6: On the left, estimated ICs from a simulated recording with 2 neurons and a tetrode as recording probe. The ellipse highlight the two *false* sources. On the right, a bar plot representing the skewness of each component. The color code represents different ICs.

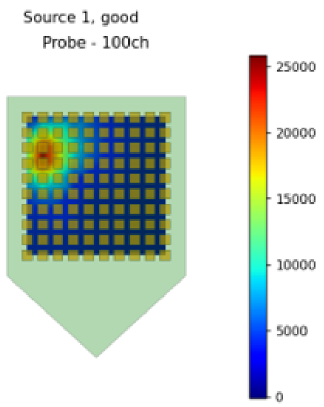
The process to remove *false* neuronal sources is taken from the work of Buccino et al. [9] and as they do I refer to it as *source cleaning*. Nevertheless, computing the skewness for each source can take a long time if the number of channels is high, even higher for longer recordings. That is why the *cleaning* process was improved by estimating the skewness on chunks of the recording in parallel. Given N channels, M samples, K parallel processes and L chunks, the pipeline is the following:

1. The recording is splitted in L time chunks of shape $\frac{M}{L}xN$
2. Each chunk is assigned to an available process k which performs cleaning on the input data
3. After all chunks have been processed, only sources selected by all processes are chosen as final output

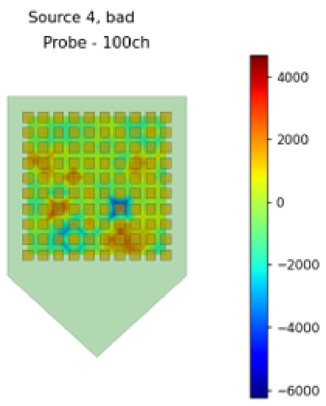
2.3.3. Sources Localization

The *mixing matrix* of a *true* neuronal source maps the source signal to the recording space, hence it is focused on few electrodes of the probe (Figure 2.7). We exploited that spatial information to estimate the position of the neuronal sources by computing the center of mass of their *mixing matrices*. The position of a source i is computed as:

$$com_i = \frac{mixing\_matrix_i * channel\_positions}{\sum_{j=1}^{N} m_{ij}} \tag{2.5}$$



(A)



(B)

Figure 2.7: Amplitudes distribution of two mixing matrices over a 100 channels recording simulated with a sqaured MEA probe. **(A)**: Mixing matrix of a *true* source. The matrix is focused on the upper left corner of the image **(B)**: Mixing matrix of a *false* source. Many peaks can be seen across the channels, meaning the matrix is not focused on any of them.

where $mixing\_matrix_i$ is the *mixing matrix* of the source i with shape $1xN$, $channel\_positions$ is a matrix of shape $Nx2$ whose columns are the x and y coordinates of the recording electrodes, m_{ij} is the j^{th} element of $mixing\_matrix_i$ and N is the number of channels.

2.3.4. ICA for SNR improvement

Event detection on the ICs As shown by Jackel et al. [25], ICA can increase the SNR of neuronal signals as each IC is ideally focused on a single unit. That is why spike detection should perform better if applied on the IC rather than applying it on the raw signal. Moreover, ICA-based spike sorting algorithms applies event detection and clustering directly on the ICs [9, 25, 28]. Therefore, spike sorting algorithms should provide better results if they are run on IC signals.

To run the algorithms on the ICs, the *mixing matrix* is first estimated running ICA on the input dataset after **peak selection**. Then, the original dataset is projected into the IC space by equation 1.7. It is necessary to project the whole dataset because during **peak selection** the time structure of the data is altered. It is not needed by ICA which assumes the data to be an instantaneous mixture of the sources, but it is needed by spike sorting algorithms which have to reconstruct the firing times of each neuron. Finally, the positions of IC sources are estimated as explained in subsection 2.3.3 and provided to the sorter. Spike sorting algorithms often use the channel positions throughout their code (see subsection 2.2.1), but after projecting the data into the IC space the processed signals do not originate from the electrodes anymore. That is why we have to estimate the position of the IC sources.

Backprojection The ICA model assumes the signal to be an instantaneous mix of the sources. Unfortunately, dendrites act as filters delaying the recorded spikes, as a consequence more than one IC can be tuned to the same neuron and an IC may be tuned to more than one neuron (Figure 2.8). Since each IC should represent the spiking activity of the neuron it is focused on and `MEArec` [8] provides the spike trace (signals with spikes only) of each neuron after a simulation, it is possible to match each IC to a spike trace and viceversa by means of the correlation between the two. An IC i is matched to the spike trace j as $argmax(c_i)$, where c_i is a row of the correlation matrix and each element $c_{i,j}$ is the correlation between the IC i and the spike trace j.

Although **cleaning** can be interpreted as removing some noise from the signal, it cannot reject duplicates as they carry spikes as well. A solution to the problem is to project the *cleaned* sources back to the original space thanks to the *mixing matrix*. After **cleaning** the dimensionality of A is decreased by keeping the rows that correspond to a *true* source. For example, if the IC number 4 is marked as *true*, the 4^{th} row of A is kept (Figure 2.9). Now the **backprojection** can be computed as:

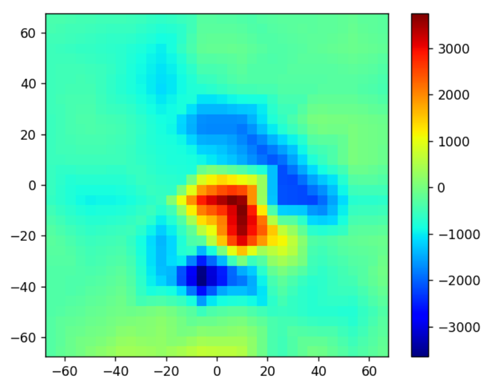
$$\bar{V} = A_c^T S_c \tag{2.6}$$

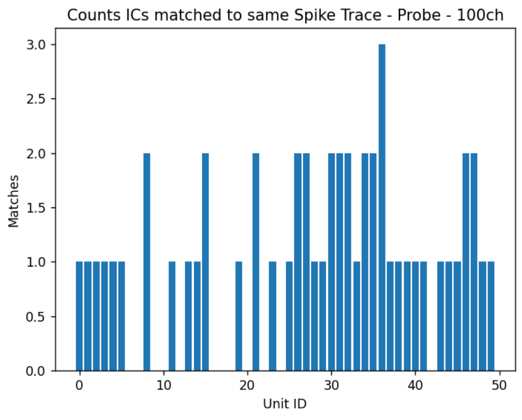
where A_c and S_c are the cleaned *mixing matrix* and *IC signals* respectively (Figure 2.10). Thanks to the **backprojection** we can tackle both the issue arisen from dendrites filtering and increase the SNR of the signals. The first point is solved because we reconstruct the original structure of the signals meaning there are not duplicates anymore. In the worst case, the activity of neurons that led to duplicates is enhanced. The increase of SNR is due to the **cleaning** step that removes a portion of the noise.

2.3.5. Small-Units Recovery

As shown in section 1.4, existing spike sorting algorithms perform bad when the SNR is low, meaning they fail to discover neurons whose spike amplitude is small compared to the noise. ICA has the ability to increase the SNR of a signal [25] which is further increased after **cleaning** and **backprojection**, so it may address the limitation of modern sorters.

First, a spike sorting algorithm is run on a recording. The output provides the spike trains





(A) **(B)**

Figure 2.8: **(A)**: Amplitude distribution of mixing matrix of a sample IC. It can be seen that instead of being focused on a single region it is focused on at least two region, thus that IC may be focused on more than one neuron. **(B)**: Bar plot showing how many ICs are focused on the same neuron. On the horizontal axis the ID of the unit in the simulated recording; on the vertical axis the number of ICs focused to the same unit.

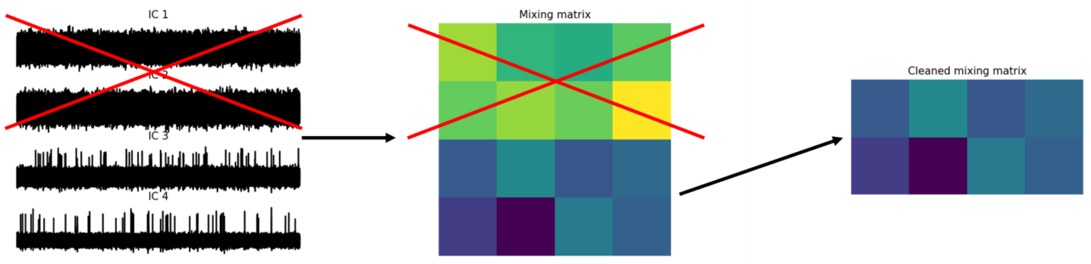


Figure 2.9: Cleaning of the mixing matrix. Example for a recording with 2 cells and a tetrode as recording probe. On the left the 4 ICs estimated by ICA; the first two are detected as *false*. At the center the full mixing matrix; as the first two ICs has been marked as *false*, the first two rows are removed. On the right the cleaned mixing matrix, keeping only the last two rows.

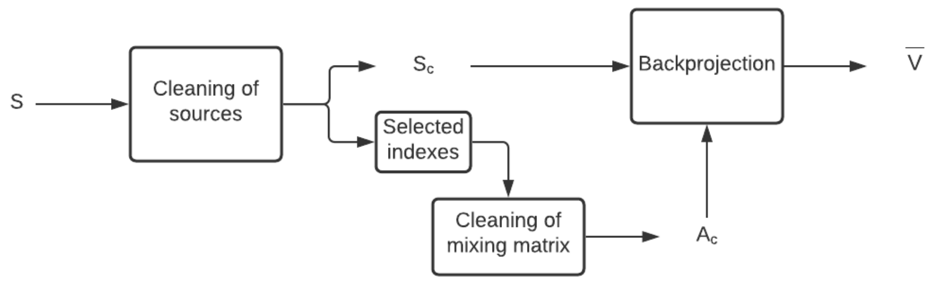


Figure 2.10: Flowchart showing the steps from **cleaning** to **backprojection**. S is the ICs matrix with shape $N_{sources}xN_{samples}$, whose rows are the ICs. The **cleaning** selects some rows of S and outputs the new ICs matrix with same number of samples but lower number of rows and the indexes of selected rows. The indexes are then used to select the rows of A obtaining A_c whose dimension is $N_{selected}xN_{channels}$. Finally A_c and S_c are multiplied to project S_c back to the original space.

of each detected neuron that are the time instants in which the neuron fired. Second, neuron templates are extracted from the recording. More in detail, given a spike t_i of a neuron i, from each channel the waveform is extracted by selecting n samples before t_i and m after. The resulting vector of samples $\mathbf{w}_{ij}$ is the waveform of spike t_i recorded by channel j. After the extraction of all the spikes, the vectors $\mathbf{w}_{ij}$ are used to build a matrix of size $N_{spikes}xN_{samples}xN_{channels}$ for each neuron. The waveforms $\mathbf{w}_{ij}$ are the rows of the submatrix. The template T_{ij} is the average of the columns of the submatrix. Now, for each neuron we have a matrix of templates $\mathbf{T}_i$ whose shape is $N_{samples}xN_{channels}$. Finally, if the ground truth is available, we compare it with the sorting output to extract the *well detected* units (see subsection 2.2.2). If the ground truth is not available, the *well detected* units need to be inferred from the spike sorting output using *ad-hoc* quality metrics [18] (a process we defined as *automatic curation*). We considered both the *firing rate* and the *Inter Spike Interval* (ISI) *violation ratio* of a sorted unit. The *firing rate* is the average spiking frequency of a unit, while the ISI violation ratio is the portion of spikes of a unit that violate the refractory period (time interval in which a neuron is "recovering" from a spike, meaning that it should not emit a spike within that time). The length of the refractory period can be set by the user (equal to 1 ms by default). So, a unit i is marked as *well detected* if:

1. **ISI violation ratio**$_i \leq 0.3$

2. **Firing rate**$_i \geq 0.1\ Hz$

The values shown here are the default ones and taken from Hill et al. [18], but they can be adjusted by the user to provide a more or less strict *automatic curation.* Once the *well detected* units are identified, their spike trains are extracted from the sorting output and templates are obtained from the template matrices $\mathbf{T}_i$. All the spikes of the detected units are removed from the signal by subtracting the templates channel by channel (Figure 2.11). The pseudo-code is the following:

```
**for** $i = 1, ..., N_{units}$ **do**
    $spike\_train \leftarrow all\_spike\_trains[i]$
    $templates \leftarrow all\_templates[i]$
    **for** $i = 1, ..., N_{channels}$ **do**
        **for all** $t \in spike\_train$ **do**
            $traces[t - n_{before} : t + n_{after}, i] \leftarrow traces[t - n_{before} : t + n_{after}, i] - templates[:, i]$
        **end for**
    **end for**
**end for**
```

Once all the detected spikes are removed, only undiscovered or *badly* discovered units remain in the signal. Now, we can apply all the methods explained in the previous sections of this chapter to enhance the SNR of undiscovered units that are still in the signal after template subtraction. So, first we subsample the *subtracted traces* and run ICA on the output. Then we apply **cleaning** and **backprojection** so that the back-projected traces have a higher SNR, meaning spikes are enhanced with respect to the noise. Finally, we run again the same spike sorting algorithm we ran at the beginning to find new neurons. See Figure 2.12 for a simplified flowchart of the full small-unit recovery algorithm.

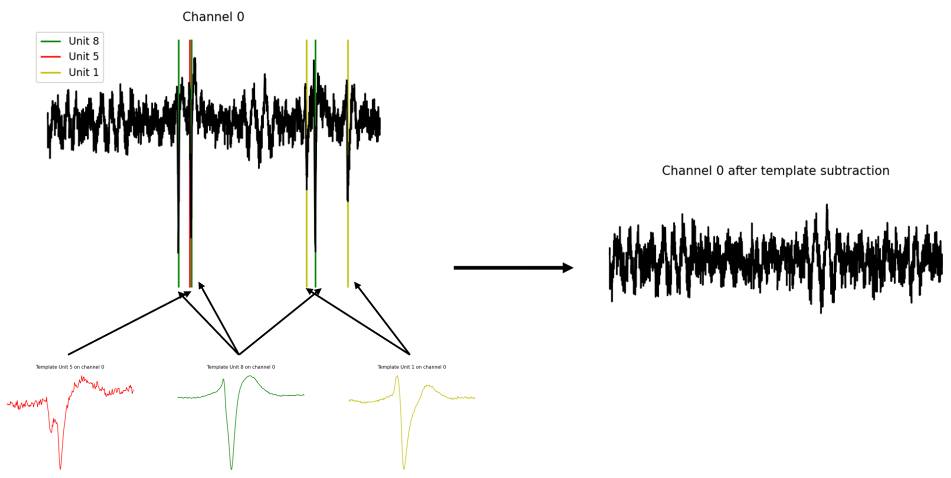


Figure 2.11: Template subtraction. On the left the trace recorded by a channel (vertical lines shows the spike times) and templates of three units. The color code shows templates and spike times of different units. On the right, the same time interval of the recording after template subtraction.

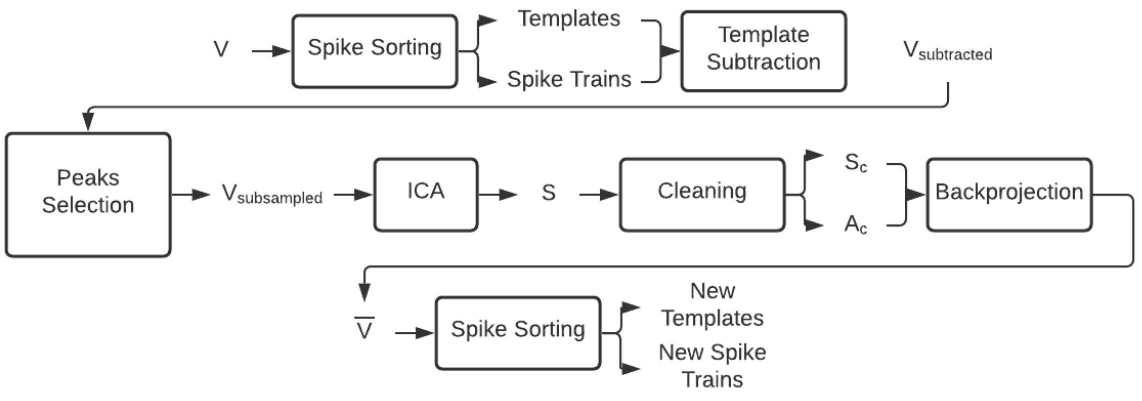


Figure 2.12: The whole pipeline for recovery of small units

3 | Results

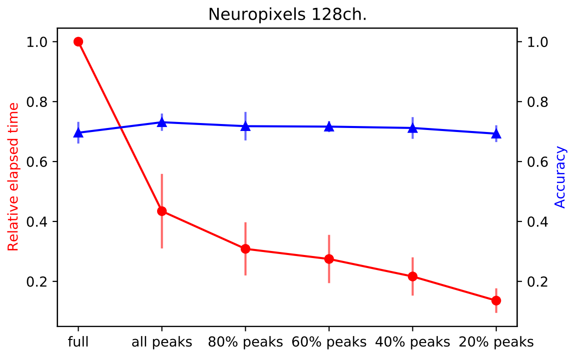
The results are organized as follows: the first section shows ICA efficiency improvement; the second section shows the performances of spike sorting algorithms run on the IC signals; the third section shows the SNR improvement after **backprojection** and **cleaning**; the last section shows the performances of **units recovery** algorithm.

The results shown throughout the chapter are obtained from simulated recordings with a duration of 120 s and a sampling rate of 32 kHz, for a total of 3’840’000 samples per channel. The number of neurons is half the number of channels of the probe used to simulate the recordings and the noise levels used are equal to 5, 10, 20, and 30 μV. The last section also shows results from two real recordings taken from embryonic cortical neurons of Wistar rats. The recordings are 5 minutes long and feature 534 and 529 channels.

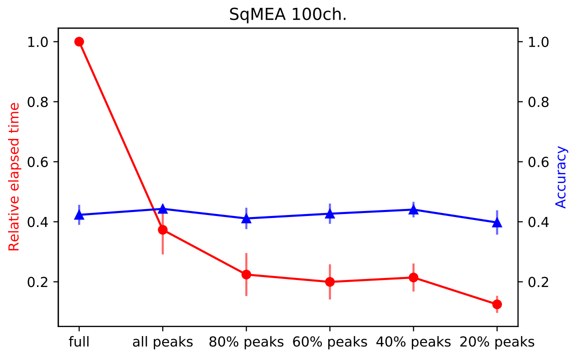
3.1. Efficiency Improvement

Figure 3.1 compares the average time required by ICA to estimate the independent components (red) and the average accuracy of `Ironclust` [23] (blue), as a proxy of the goodness of the ICA model, run on the ICs as function of the percentage of detected peaks, for the three probes used to simulate the recordings. Five recordings with `noise_level=10` have been averaged for each probe. The time values in the plots were normalized by the maximum elapsed time to provide an easier understanding of the percentage of time saved as function of the number of detected peaks selected. The accuracy was only evaluated on units whose $SNR > 5$ (see equation 3.2). The average number of peaks detected were $57'222 \pm 6'058$, $33'235 \pm 2'658$ and $19'935 \pm 2'992$ for Neuropixels, squared MEA and Neuronexus respectively. For each spike, a window of 1 ms before and 2 ms after the spikes has been used, thus 96 samples have been selected per spike. From 3’840’000 samples per channel, using all the detected spikes the dataset dimension was decreased to $2'458'775 \pm 119'186$ (35.97% reduction), $1'801'420 \pm 97'566$ (53.1% reduction), $1'118'709 \pm 84'511$ (70.8% reduction) samples per channel, for Neuropixels, squared MEA and Neuronexus respectively. That dimensionality reduction led to a decrease of the running time from $5'251.59 \pm 141.27$ s to $2'562.60 \pm 125.79$ s (51.2% reduction), from $2'104.23 \pm 382.93$ s to 762.51 ± 117.75 s (63.8% reduction) and from 175.48 ± 43.17 s to 54.66 ± 15.95 s (58.34% reduction) for Neuropixels, squared MEA and Neuronexus respectively. Conversely, the accuracy increased from $69.62 \pm 3.59\%$ to $73.12 \pm 3.74\%$, from $42.3 \pm 3.29\%$ to $44.3 \pm 1.8\%$ and from $54.3 \pm 11.29\%$ to $64.4 \pm 4.31\%$ for Neuropixels, squared MEA and Neuronexus. Decreasing the percentage of spikes used decreases both the estimation time and the accuracy. For the lowest amount of spikes tested (20%) for Neuropixels, squared MEA and Neuronexus the dataset dimension was reduced to $826'128 \pm 70'449$ (88.5% reduction), $513'235 \pm 3'6854$ (96.64% reduction) and $299'260 \pm 40'834$ (92.21% reduction) respectively and the elapsed time decreased to 802.32 ± 91.01 s, 255.06 ± 51.37 s, 16.28 ± 5.47 s, while the accuracy decreased to $69.3 \pm 3.17\%$, $39.76 \pm 2.13\%$, $47.8 \pm 12.18\%$ for Neuropixels, squared MEA and Neuronexus respectively. The algorithms were run on an Ubuntu server with 32 cores (Intel(R) Xeon(R) CPU E7- 4870 @ 2.40 GHz) and 128 GB of RAM.

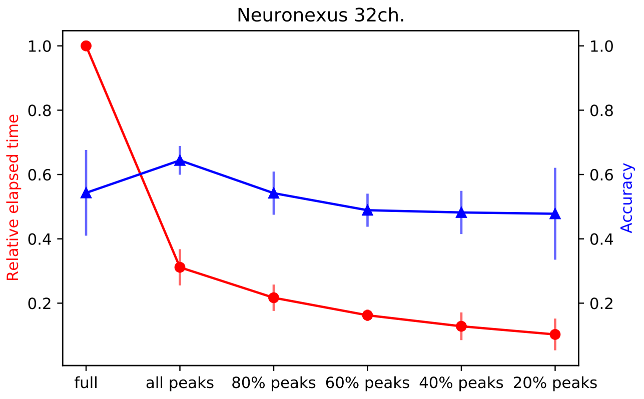
From the results we can say that it is possible to decrease the dimensionality of the input dataset without lowering the *goodness* of the estimated ICs. Indeed, the largest Δs of accuracy equal 3.82% and 4.54% for Neuropixels and squared MEA respectively. The accuracy values for Neuronexus have larger oscillations as the largest Δ equals 16.4%. That may be due to the lower spatial information provided by the Neuronexus probe which makes the results more sensitive to the number of peaks. It may also be confirmed by the larger standard deviation of the accuracy which suggests that the IC estimation on Neuronexus recordings is more influenced by neurons position with respect to the IC estimation on Neuropixels and squared MEA recordings.



(A)



(B)



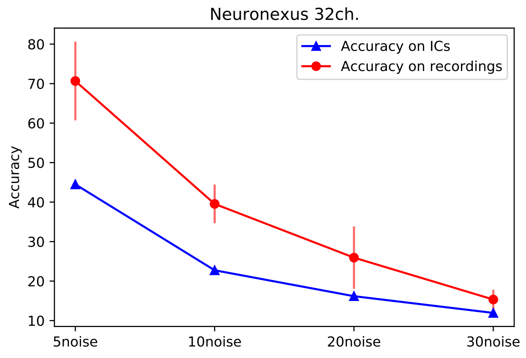
(C)

Figure 3.1: The plots show the accuracy of `Ironclust` (blue) and ICA running time (red) as function of the number of samples. The time values were normalized by ICA running time on the *full* dataset. For each probe - Neuropixels probe (128 channels, figure **(A)**), a squared MEA probe (100 channels, figure **(B)**) and a Neuronexus probe (32 channels, figure **(C)**) - 5 simulated recordings, 120 s long and with noise level equal to 10 μV, have been averaged for the plots. *full* refers to the full and original dataset, *all peaks* means all detected peaks have been used for subsampling and *x% peaks* means x% of all detected peaks have been used. The time required by ICA decreases by decreasing the number of detected peaks used. Conversely, the accuracy slightly improves when *all peaks* are used, but overall we can say it is fairly constant for Neuropixels and squared MEA. On the contrary, the accuracy on Neuronexus has larger oscillations and standard deviation which may be due to the lower spatial resolution of Neuronexus probe with respect to the spatial resolution provided by Neuropixels and squared MEA.

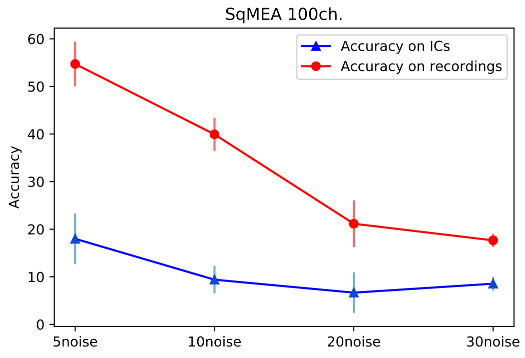
3.2. Event detection on the ICs

The Independent Components estimated by ICA are focused on the neuronal sources which increases the SNR of neurons [25]. Therefore, the event detection should perform better on the ICs than on the raw recording. To test that assumption, the accuracy of `Ironclust` [23] run on both IC signals and raw recordings has been compared. Figure 3.2 shows the results for Neuronexus, squared MEA and Neuropixels probes. The red lines refer to the accuracy on the recording, while the blue ones to the accuracy on the ICs. `Ironclust` [23] with Neuronexus probe (Figure 3.2.A) achieves $70.67 \pm 9.69\%$ of accuracy on recordings with `noise_level` $= 5$ and $44.5 \pm 6.08\%$ on IC signals estimated from the same recordings. The accuracy in both cases decreases as the noise level increases. The lowest values at `noise_level` $= 30$ are $15.34 \pm 2.45\%$ on recordings and $11.94 \pm 3.98\%$ on IC signals. Regarding squared MEA (Figure 3.2.B) the performances of the sorters are slightly lower than the previous case on both signals. Moreover, the accuracy difference between recordings and IC signals is greater. At `noise_level` $= 5$ the accuracy on recordings is $54.72 \pm 4.69\%$ while on the IC signals is $17.98 \pm 5.33\%$. At `noise_level` $= 30$ the accuracy on recordings drops to $17.65 \pm 1.41\%$ and to $8.55 \pm 1.44\%$ on IC signals. The same trend is shown by performances of `Ironclust` [23] with Neuropixels probe (Figure 3.2.C) with $72.6 \pm 2.91\%$ and $37.31 \pm 3.8\%$ accuracy on recordings and IC signals with `noise_level` $= 5$ and with $15.65 \pm 1.32\%$ and $3.8 \pm 1.51\%$ accuracy on recordings and IC signals with `noise_level` $= 30$. The same comparison was made running `Herdingspikes` [17] and `Kilosort2` [32] but both failed when run on the IC signals.

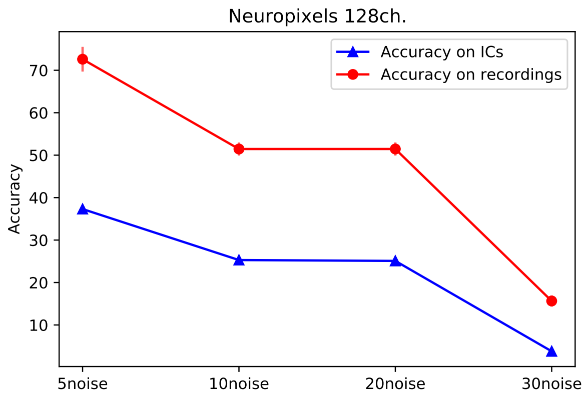
The results may point out that existing spike sorting algorithms should not be run directly on the ICs. After projecting the data into the IC space, the structure of the recorded signals changes. Before the projection a spike is recorded by a group of close channels, while after the projection a spike should be present in one IC only. Therefore, the high spatial information provided by high-density probes may be condensed by ICA. Moreover, since spikes in the ICs may be shifted in time an *alignment* step should be included to correct for the shifting as done by Buccino and colleagues [9]. That is why structural and temporal changes may undermine the performance of the sorter.



(A)



(B)



(C)

Figure 3.2: The plots show the average accuracy of `Ironclust` on 4 sets of 5 recordings simulated with Neuronexus (32 channels, **(A)**), squared MEA (100 channels, **(B)**) and Neuropixels probe (128 channels, **(C)**). The noise level increases across the sets from 5 to 30. In blue the accuracy of the algorithm on IC signals, in red its accuracy on the raw recordings. For every probe as the noise level increases, the accuracy decreases in both cases. For all the three plots the performance of `Ironclust` is higher when it is run on the raw signals rather than running it on the ICs. The greatest difference is on the squared MEA probe (figure **(B)**) at 5 μV of noise level with more than 30% of difference. The disparity tends to decrease increasing the noise level, very likely because of a lower performance of the sorter at large noise levels.

3.3. SNR improvement

Dendrites filtering properties delay the propagation of APs within a neuron. The consequence is that spikes generated close to the soma and the ones generated along dendrites by the same AP are recorded at different time instants. That induces ICA to interpret spikes coming from the soma and the ones coming from the dendrites as belonging to different sources.

Backprojecting the ICs to the recording space reconstructs the original structure of the data which tackles the problem of duplicate sources. Moreover, thanks to the cleaning process, **backprojecting** the IC signals decreases the noise level of the traces increasing their SNR which is computed as:

$$traces\_snr_i = \frac{max(traces_i)}{noise\_level_i} \tag{3.1}$$

where $noise\_level_i$ is computed internally by `SpikeInterface` as the Median Absolute Deviation of the trace. Figure 3.3.A compares a snippet of a trace after **backprojection** (orange) and the same one before **backprojection** (blue). After **backprojection** the trace has a lower noise level while the amplitude of the spikes is preserved. This sentence is confirmed in Figures 3.3.B, 3.3.C and 3.3.D, where the left panel in each figure compares the SNR of all traces before and after **backprojection** for the three probes used during the simulation, while the right one shows the count of traces whose SNR increased (green) or decreased (red). For Neuronexus (32 channels), squared MEA (100 channels) and Neuropixels (128 channels) the number of traces whose SNR increased are 27 out of 32, 99 out of 100 and 124 out of 128.

Thanks to `SpikeInterface` again and the availability of ground-truth data, it is possible to compare how well defined the spikes of each unit are by computing what in the package is defined as `units_snr`. It is computed as:

$$units\_snr_j = \frac{max(waveforms_j)}{noise\_level_i} \tag{3.2}$$

where $waveforms_j$ is a vector collecting all spike waveforms from unit j and $noise\_level_i$ is the noise level of the trace from which the spike given by $max(waveforms_j)$ is extracted. Figures 3.4.A and 3.4.B compares two couples of waveforms before and after **backprojection**, showing that the waveform are well preserved. Figures 3.4.C, 3.4.D and 3.4.E compare `units_snr` before and after **backprojection** for Neuronexus (simulated recordings with 16 neurons), squared MEA (simulated recordings with 50 neurons) and Neuropixels (simulated recordings with 64 neurons) probes respectively. The bar plots count the number of units whose SNR increased (green) or decreased (red). The SNR of neurons improved too, even though the improvement is lower than the one achieved for the traces. The best improvement was received by Neuropixels for which 41 out of 64 neurons got an improvement in the SNR. Regarding squared MEA, the SNR improved for 30 out of 50 neurons, while Neuronexus got the lowest improvement as the SNR increased for 9 out of 16 neurons.

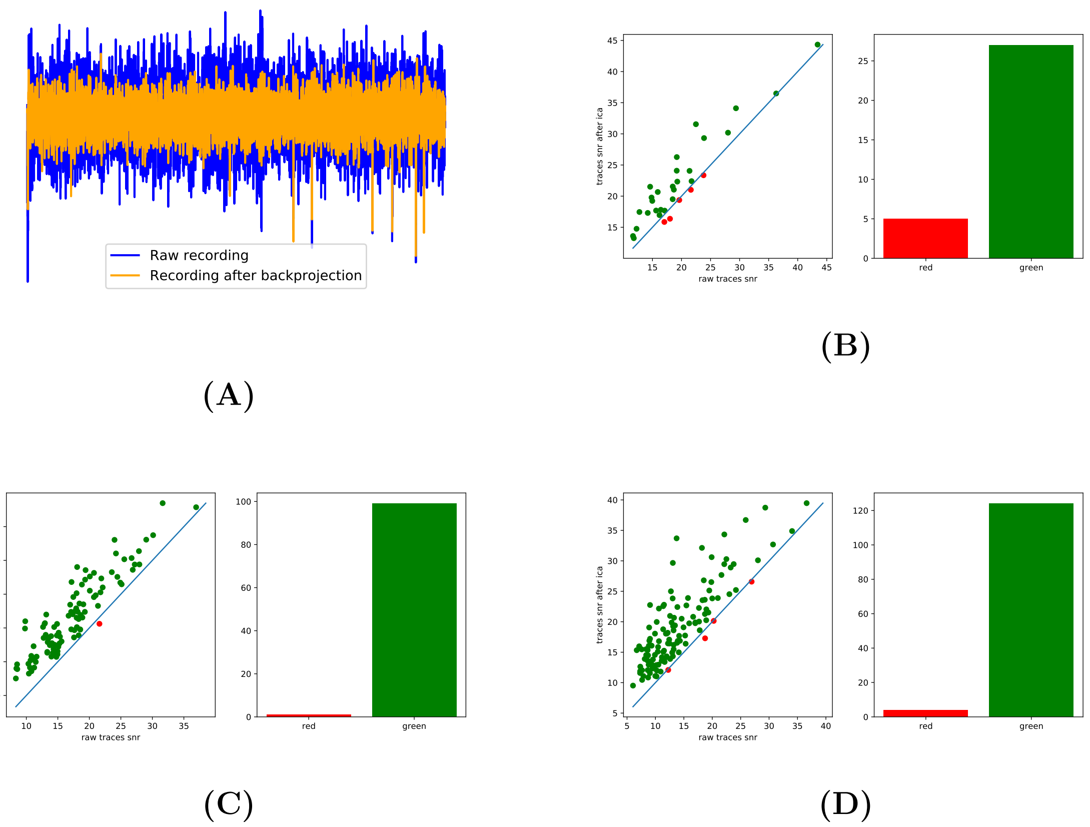
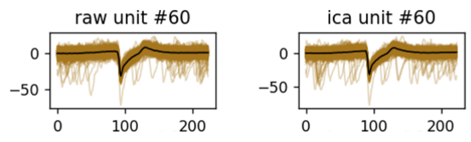
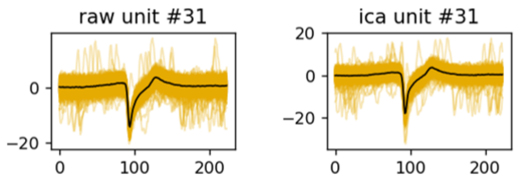


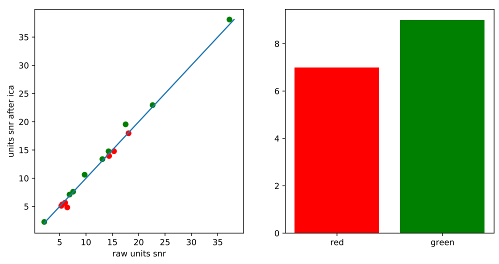
Figure 3.3: **(A)**: Snippet of signal of a channel from squared MEA simulated recording before and after **backprojection**. The noise level of the backprojected signal (in orange) is smaller than the noise of the original one, while the amplitude of the spikes is preserved. **(B)**, **(C)** and **(D)** are scatterplots comparing the traces SNR before **backprojection** (horizontal axis) and after (vertical axis). Each point represents a channel and it is colored red if its SNR decreased after **backprojection**, while it is green if its SNR increased. The bar plots show the count of red and green dots in each scatter plot. Figure **(B)** shows results for a 32 channel recording for which the SNR of 27 out of 32 channels improved. Figure **(C)** shows results for a 100 channel recording for which the SNR of 99 out of 100 channels improved. Figure **(D)** shows the results for a 128 channel recording for which the SNR of 124 out of 128 channels improved.



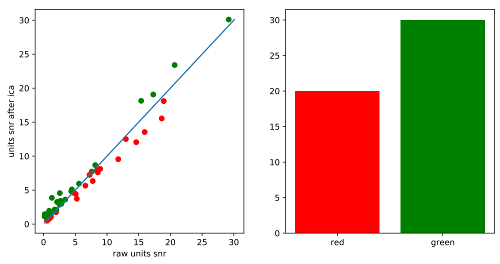
(A)



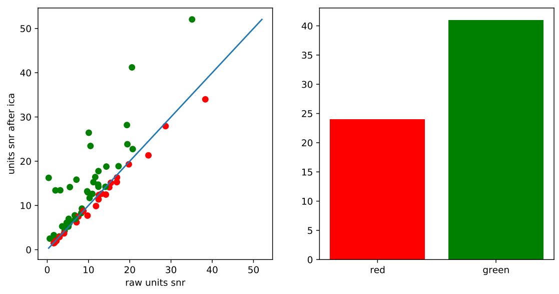
(B)



(C)



(D)



(E)

Figure 3.4: Figures **(A)** and **(B)** show the waveforms of two units before (left of each image) and after (right of each image) **backprojection** for a simulated recording with 64 neurons and 128 channels. It can bee seen the shapes of the waveforms are not distorted by the **backprojection**. Figures **(C)** (Neuronexus), **(D)** (squared MEA) and **(E)** (Neuropixels) show the same information of the previous figure but in this case the values on the horizontal and vertical axis are the neuron SNRs before and after **backprojection** respectively. The number of units whose SNR improved is 9 out of 16 for Neuronexus, 30 out of 50 for squared MEA and 41 out of 64 for Neuropixels (it received the best improvement).

3.4. Units Recovery

Bad performances of spike sorting algorithms on low-SNR recordings cause them to miss units of small dimensions. The ability of ICA to increase the SNR of a signal can solve that problem, but applying event detection directly on the IC signals can cause new issues as ICA modifies the structure of the data (see section 3.2). Backprojecting the ICs to the original space after proper *cleaning* solves that problem and still increases the SNR of the recording. Therefore, once the large units (detected during a first round of spike sorting) are removed from the signal (decreasing the SNR of the traces as shown in Figure 3.5), ICA can enhance the activity of units remained in the residual recording. Figures 3.6.A, 3.6.B and 3.6.C compare the average accuracy of `Kilosort2` before (blue) and after (orange) small-units recovery as function of the noise level in the recordings for the three probes used during simulations: Neuronexus (32 channels, figure **(A)**), squared MEA (100 channels, figure **(B)**) and Neuropixels (128 channels, figure **(C)**). For every probe the best improvement is achieved at `noise_level` $= 5$. The starting accuracy improved from $74.53 \pm 10.34\%$ to $82.27 \pm 2.9\%$, from $47.1 \pm 1.0\%$ to $64.88 \pm 4.71\%$ and from $59.15 \pm 1.13\%$ to $80.75 \pm 0.74\%$ for Neuronexus, squared MEA and Neuropixels respectively. If the noise level increases, the performance of `Kilosort2` decreases but the accuracy still improves after the recovery, even though by a lower amount, for squared MEA and Neuropixels. More in detail, at `noise_level=30` (the highest noise level simulated) the starting accuracy improves from $14.91 \pm 1.77\%$ to $19.76 \pm 2.42\%$ and from $10.53 \pm 1.95\%$ to $17.56 \pm 1.96\%$ for squared MEA and Neuropixels respectively. A different behavior is shown for Neuronexus probe as at the same noise level the accuracy remained still at $12.15 \pm 3.73\%$.

Figure 3.6 summarizes the average improvement after units recovery for three different spike sorting algorithms: `Herdingspikes` [17], `Ironclust` [23] and `Kilosort2` [32]. The results show that the improvement is strongly influenced by the noise level of the signal as at `noise_level=5` the Δaccuracy values are $3.87 \pm 0.2\%$, $9.12 \pm 0.1\%$ and $10.8 \pm 0.2\%$ for Neuronexus, squared MEA and Neuropixels respectively, while at `noise_level=30` the improvement drops to 0%, $2.43 \pm 0.32\%$ and $3.52 \pm 0.23\%$ for Neuronexus, squared MEA and Neuropixels respectively. Moreover, the improvement on Neuronexus recordings is much lower ($\approx 4\%$) than the one on squared MEA and Neuropixels recordings ($\approx 9\%$ and $\approx 11\%$) suggesting that the higher spatial resolution of the last two probes may improve the output of ICA. Finally, Figure 3.6.D shows also a low variance of the average improvement (almost null for every point apart from Neuronexus recordings at `noise_level=10` and `noise_level=20` when the standard deviation is 0.8% and 1.3%

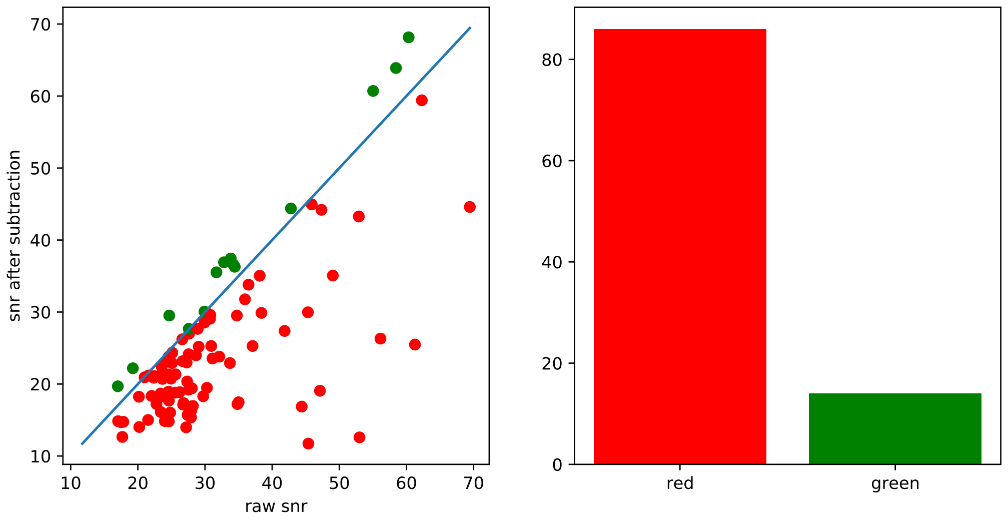
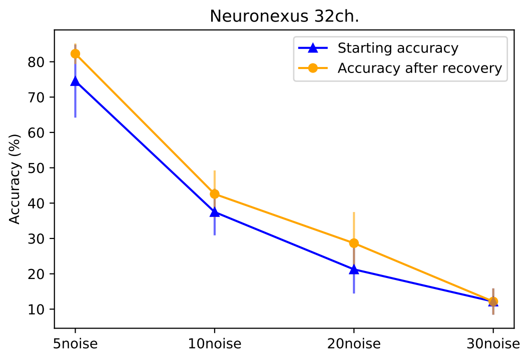


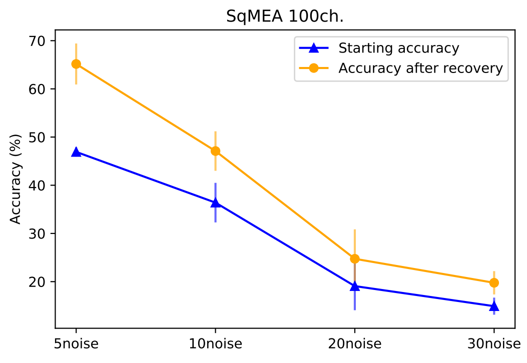
Figure 3.5: Example to show how the channel SNRs change after removing *well detected* units from the recording. The example shows the result for a recording simulated with squared MEA (50 neurons, 100 channels). It can be seen that the SNR of 87 out 100 channels decreased after template subtraction.

respectively), meaning the variability of results shown by Figure 3.6 is influenced more by the sorting algorithm rather than being influenced by ICA.

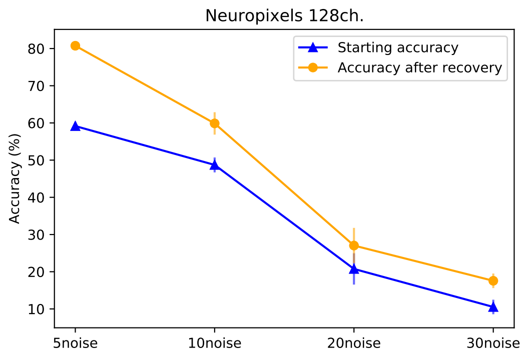
How to recognize the *well detected* units may not be trivial if ground-truth data are not available. The approach suggested by the present work, defined by us *automatic curation*, consists in removing units with very low firing rate and with high ISI violation ratio. The first three plots of Figure 3.7 compare the accuracy of `Kilosort2` after recovery when the *well detected* units are removed based on ground-truth data (blue) and when they are removed with *automatic curation* (orange). Regarding Neuronexus (**A**) and Neuropixels (**C**) the difference in accuracy is almost null. The highest difference for Neuropixels is at `noise_level=10` when the accuracy decreases from $59.87 \pm 3.01\%$ to $56.91 \pm 3.7\%$, while for Neuronexus it is at `noise_level=30` when the accuracy, counter intuitively, increases from $12.15 \pm 3.73\%$ to $16.06 \pm 3.78\%$. Regarding squared MEA (**B**) the difference is larger especially at `noise_level=5` when it decreases from 65.41 ± 4.35 to $54.9 \pm 5.68\%$. To evaluate the average accuracy difference ($accuracy_{well\_detected} - accuracy_{auto\_curation}$) the results of `Herdingspikes`[17], `Ironclust`[23] and `Kilosort2` [32] have been averaged and shown by Figure 3.7.D. The plot shows the same tendency of Figures 3.7.A, 3.7.B and 3.7.C as the highest Δaccuracy is related to squared MEA at `noise_level=5` and equal to 7.11% while the Δaccuracy on Neuropixels and Neuronexus is much lower and equal to 1.92% and 1.62% at the same noise level. Moreover, it is interesting to note that the difference on Neuropixels remains ≈ 2% and that the accuracy on Neuronexus for `noise_level=20` and `noise_level=30` becomes negative and equal to −1.02% and −0.69%, meaning it increases with *automatic curation*.



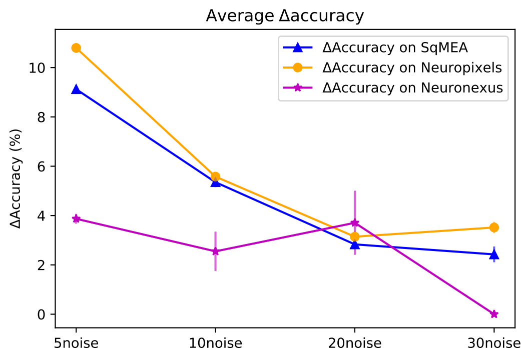
(A)



(B)

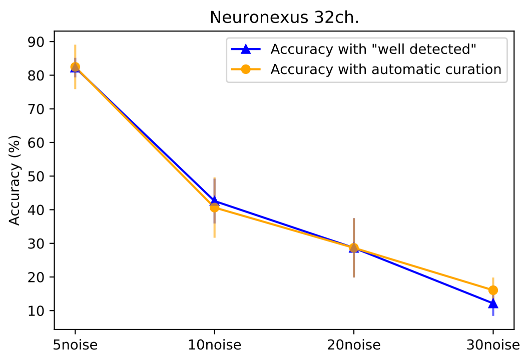


(C)

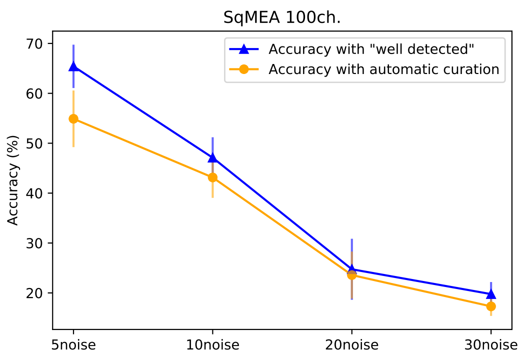


(D)

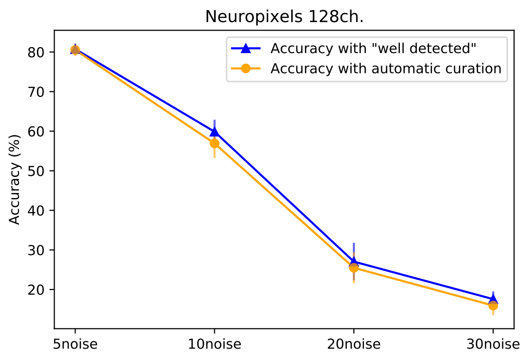
Figure 3.6: The plots **(A)**, **(B)** and **(C)** show the average accuracy of `Kilosort2` before (blue) and after (orange) *units recovery* on recordings simulated with three different probes: Neuronexus (Figure **(A)**), squared MEA (Figure **(B)**) and Neuropixels (Figure **(C)**). The best improvement is achieved ad 5 μV of noise level when the accuracy improves by 7.16%(**A**), 17.78%(**B**) and 21.6%(**C**). The improvement decreases by increasing the noise level. At 30 μV of noise level the accuracy remained still at $12.15 \pm 3.73\%$ for Neuronexus while the improvement drops to 4.85% and 7.03% for squared MEA and Neuropixels respectively. The plot in figure **(D)** shows the average accuracy improvement for `Kilosort2`, `Herdingspikes` and `Ironclust` algorithms on recordings simulated with Neuronexus (magenta), squared MEA (blue) and Neuropixels (orange) with increasing noise level. The algorithm performs better, on average, on squared MEA and Neuropixels recordings than on Neuronexus recordings, with an exception at 20 μV of noise level.



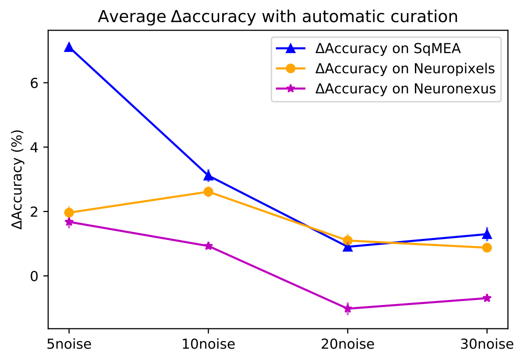
(A)



(B)



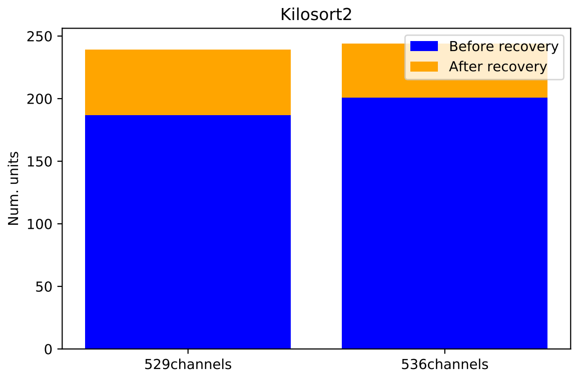
(C)



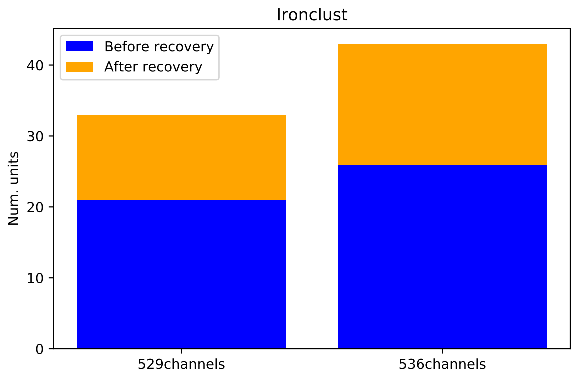
(D)

Figure 3.7: The plots in figures **(A)**, **(B)** and **(C)** compare the accuracy of `Kilosort2` after recovery when *well detected* units are found exploiting ground truth data and when they are removed by *automatic curation*. The performance on Neuronexus (**A**) and Neuropixels (**C**) is similar to the ideal case, as the largest decrease is on Neuropixels at 10 μV of noise level when it decreases by only 1.96%. The performance even increase by 3.91% for Neuronexus at 30 μV of noise level. On squared MEA (**B**) the algorithm behaves differently as the accuracy decreases by a larger amount equal to 10.51% at 5 μV of noise level. The plot of figure **(D)** shows the average accuracy difference for `Kilosort2`, `Herdingspikes` and `Ironclust` algorithms on recordings simulated with Neuronexus (magenta), squared MEA (blue) and Neuropixels (orange) with increasing noise level. The accuracy decrease is fairly constant for Neuropixels and $\approx$ 2% while it is larger for squared MEA at low noise levels. Indeed, it is equal to 7.11% at 5 μV of noise level. For large noise levels (20 μV and 30 μV) the accuracy does not decrease, but improves by 1.02% and 0.69%.

Real recordings Figure 3.8 shows the results for units recovery on two real recordings with 529 and 536 channels running `Kilosort2` [32] and `Ironclust` [23]. The blue bar refers to the amount of new units found during the first run, the orange one refers to the amount of units found after recovery using *automatic curation* to identify the *well detected* units. We attempted to run `Herdingspikes` [17] as well but it did not converge even during the first round of spike sorting. The performances of the two algorithms greatly differ as `Kilosort2` [32] found 187 and 201 neurons during its first run, while `Ironclust` [23] found only 21 and 26 units. After units recovery both algorithms found new units. Indeed, `Kilosort2` [32] found 52 and 43 new units on 526 channels and 539 channels recording respectively, while `Ironclust` [23] found 12 and 17 new units on the two recordings. Nevertheless, it is not possible to tell how well (or bad) the sorters performed and if the new units found after units recovery correspond to real neurons or if they are false positives.



(A)



(B)

Figure 3.8: Performance of units recovery on real recordings. **(A)**: the results of `Kilosort2` that found 52 and 43 new units on the recording with 526 and 539 channels respectively. **(B)**: the results of `Ironclust` which found 12 and 17 new units.

4 | Discussion

Independent Component Analysis is a blind source separation technique aiming to unmix a signal into its sources, increasing its SNR. This is why it can be a useful tool to analyze neuronal recordings. The application of ICA in spike sorting has gained new interest due to increasing number of electrodes in the recording devices. Indeed, ICA requires that the number of sources generating the signal is at most equal to the number of recording channels. This assumption is closer to be satisfied when neuronal activity is recorded by **High-Density Micro-Electrode Arrays** (HD-MEAs), given their very high spatial resolution. ICA-based spike sorting algorithms apply event detection and clustering directly on the ICs as they have an higher SNR [25]. Therefore, existing spike sorting algorithms have been tested on IC signals after improving the computational efficiency of ICA. Next, a new approach for the application of ICA has been proposed.

Efficiency Improvement Due to the sparsity of firing rates [42], the dimensionality of the input dataset can be reduced by selecting the spikes only. The new dataset is more focused on the spiking activity which can improve the estimation of the Independent Components. Indeed, after subsampling, the time required by ICA for the estimation of the *mixing matrix* decreased by 63.8% in the best case (squared MEA probe) while the accuracy increased by 2% in the same case. The increase in accuracy is not high, but remember here the goal was to decrease the efficiency. The accuracy improvement, even if small, is a well appreciated side effect. Moreover, the running time can be further decreased by sub-sampling the dataset using an user-defined percentage of the detected spikes. In this case, even if some important information for the estimation of the ICs is removed, the accuracy is fairly stable. Indeed, for the lowest percentage of spikes tested (20%) the accuracy only decreased by 2.54% with respect to the highest value, meaning that it was almost equal to the performance obtained on the full dataset.

For all the three probes the elapsed time decrease with the amount of spikes used. On the contrary, the accuracy has a different behavior for squared MEA and Neuropixels with respect to Neuronexus. For squared MEA and Neuropixels the accuracy can be considered fairly independent from the percentage of spikes, while for Neuronexus the accuracy has larger variations. The different behavior may be due to the lower spatial resolution of Neuronexus that makes the output of ICA more dependent on the position of the neurons. It is also worth to notice that the accuracy values for the Neuronexus probe have a far greater standard deviation than the accuracy values of squared MEA and Neuropixels, making the results less reliable.

Event detection on the ICs ICA-based spike sorting algorithms apply event detection and clustering directly on the IC signals [9, 25, 28] with good results thanks to the improved SNR of neuronal signals [25]; following the same approach three spike sorting algorithms have been run on the estimated Independent Components. Despite the good premise the results did not replicate the expectations. Two out of three applied algorithms (`Herdingspikes` and `Kilosort2`) failed. `Ironclust` did not fail, but for every tested recording the accuracy on the ICs was lower than the accuracy on the raw recording.

We speculate that the spatially redundant information of high-density configurations, which is *condensed* in the ICA space, might be used by the spike sorters and this could explain the decrease in accuracy. In addition to that, when the recordings are projected into the IC space the time structure may change slightly and then the estimated spike times do not match the ground-truth information. A possible solution is to align the IC signals to the original recording as shown by Buccino and colleagues [9].

SNR improvement Due to the filtering properties of the dendrites, spikes happening close to the soma are recorded at different time instants with respect to spikes happening at the dendrites, and this violates the instantaneous linearity assumption of the ICA formulation. That may result in the estimation of duplicate sources, which can be removed by *ad-hoc* processes [9]. Another possibility is to project the IC signals back to the original space so that the original data structure is reconstructed. Moreover, after the cleaning process a portion of the noise is removed from the data. That increases both the SNR of the signal in each channel and the SNR of neurons, even though the increase is lower in the latter case. Indeed, the SNR of some units decrease after **backprojection**. It is due to a non-excellent performance of the cleaning process which removed ICs that were focused on neurons instead of carrying noise only, so the spikes of those neurons have been removed from the signal.

Increasing the SNR of recordings is very important in signal processing, especially for spike sorting purposes as spikes are enhanced with respect to the noise. That result carries two main consequences: spike detection, which is still strongly dependent on the noise level of the recording [27], is improved; the activity of small neurons, difficult to detect (see section 1.4), is enhanced. Those results are what inspired the following approach.

Units recovery Many spike sorting algorithms exist, but, at the moment of writing, none can perform well on small SNR recordings. Low firing rates, small neurons or a noisy probe are only some of the factors that can lead to low SNR recordings. Therefore, it can be assumed that sorters can correctly identify only relatively large neurons. If we remove from the original recording the spikes from neurons that were identified by spike sorters, we obtain a recording with a low SNR which carry the activity of small neurons. Running ICA on the residual recording, *cleaning* the estimated sources and *backprojecting* the signals to the original space can enhance the SNR of residual units. Now, if the sorter is run again it is able to detect new neurons that were not detected during its first application. Understanding which neurons have been well detected after the first round is not trivial without ground-truth information. We implemented a relatively simple solution which removes the units with very low firing rate (below 0.1 Hz) and with high ISI violation ratio (higher than 0.3). The proposed approach achieves results comparable to the ideal case (availability of ground-truth data) for Neuronexus and Neuropixels probes, but it achieves a lower performance on the squared MEA probe (see Figure 3.7.B). This means that the performance of our *automatic curation* may vary across different probes.

The small-unit recovery algorithm is strongly influenced by both the probe used and by the noise level of the signal. Indeed, the higher the spatial resolution and the lower the noise level, the higher the improvement after `units_recovery`; that strengthen the assumption of applying ICA on HD-MEA recordings.

Future Development The problem of duplicate sources has been solved by backprojecting the ICs back to the original space after **cleaning** but the estimation can be improved further using *convolutive ICA* (cICA) instead of the classical one [28]. It estimates a set of mixing matrices delayed in time, which can help to solve the problem of duplicates due to phase shift introduced by dendrites. Moreover, the *cleaning* process may be improved by fusing the current method based on skewness with the spatial information provided by the mixing matrices, i.e. if they are focused on a specific region of the probe or not (see Figure 2.7).

The main limitation of small-units recovery algorithm lies in removing all units that have been well detected during the first run of spike sorting without removing any non-well detected unit. That is trivial if ground-truth data are available, but it is tricky if they are not. As of now we implemented in the algorithm a relatively simple *automatic curation* which marks as *well detected* the units with an ISI violation ratio lower than 0.3 and a firing rate in an interval set by the user and by default larger than 0.1 Hz. Despite the results are quite close to the ideal case (availability of ground-truth data), sometimes the algorithm may fail and the performance steeply decreases, meaning it could be improved by using a more complex set of quality metrics (see `SpikeInterface` paper [10] for an exhaustive list).

In conclusion, this thesis presented possible solutions to cope with major ICA limitations for spike sorting purposes and it explored different applications of ICA for spike sorting. Starting from applications suggested by the literature, this work proposed a new approach of ICA to improve the performance of spike sorting algorithms on low-SNR recordings.

Bibliography

[1] Allen institute shares first open database of live human brain cells, . URL https://alleninstitute.org/what-we-do/brain-science/news-press/press-releases/allen-institute-shares-first-open-database-live-human-brain-cells.

[2] LFPy: a tool for biophysical simulation of extracellular potentials generated by detailed model neurons, . URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3893572/.

[3] The michigan probe: Changing the course of brain research, . URL https://ece.engin.umich.edu/stories/the-michigan-probe-changing-the-course-of-brain-research.

[4] Preparation of biological samples for the recording of electrophysiological signals using high-density microelectrode arrays, . URL https://www.researchsquare.com.

[5] A. Amin, H. Tsuji, K. Kurokawa, H. Ashahi, Y. Shinbo, and S. Kanaya. DPClus a density-periphery based graph clustering software mainly focused on detection of protein complexes in interaction networks. volume 7, pages 37–42. ISBN 978-984-32-3394-3. doi: 10.1109/ICICT.2007.375338.

[6] A. L. Barth and J. F. Poulet. Experimental evidence for sparse firing in the neocortex. 35(6):345–355. ISSN 01662236. doi: 10.1016/j.tins.2012.03.008. URL https://linkinghub.elsevier.com/retrieve/pii/S0166223612000513.

[7] A. P. Buccino. A computationally-assisted approach to extracellular neural electrophysiology with multi-electrode arrays. URL https://www.duo.uio.no/handle/10852/72480. Accepted: 2020-01-23T11:33:43Z.

[8] A. P. Buccino and G. T. Einevoll. MEArec: A fast and customizable testbench simulator for ground-truth extracellular spiking activity. 19(1):185–204. ISSN 1539-2791, 1559-0089. doi: 10.1007/s12021-020-09467-7. URL https://link.springer.com/10.1007/s12021-020-09467-7.

[9] A. P. Buccino, E. Hagen, G. T. Einevoll, P. D. Hafliger, and G. Cauwenberghs. 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)*, pages 2627–2630. IEEE, . ISBN 978-1-5386-3646-6. doi: 10.1109/EMBC.2018.8512788. URL `https://ieeexplore.ieee.org/document/8512788/`.

[10] A. P. Buccino, C. L. Hurwitz, S. Garcia, J. Magland, J. H. Siegle, R. Hurwitz, and M. H. Hennig. SpikeInterface, a unified framework for spike sorting. 9:e61834, . ISSN 2050-084X. doi: 10.7554/eLife.61834. URL `https://elifesciences.org/articles/61834`.

[11] A. P. Buccino, T. Stöber, S. Næss, G. Cauwenberghs, and P. Häfliger. Extracellular single neuron stimulation with high-density multi-electrode array. In *2016 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, pages 520–523, . doi: 10.1109/BioCAS.2016.7833846.

[12] J. E. Chung, J. F. Magland, A. H. Barnett, V. M. Tolosa, A. C. Tooker, K. Y. Lee, K. G. Shah, S. H. Felix, L. M. Frank, and L. F. Greengard. A fully automated approach to spike sorting. 95(6):1381–1394.e6. ISSN 08966273. doi: 10.1016/j.neuron.2017.08.030. URL `https://linkinghub.elsevier.com/retrieve/pii/S0896627317307456`.

[13] D. Comaniciu and P. Meer. Mean shift: a robust approach toward feature space analysis. 24(5):603–619. ISSN 1939-3539. doi: 10.1109/34.1000236. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence.

[14] C. Dang, W. Lin, F. Meng, H. Zhang, S. Fan, X. Li, K. Cao, H. Yang, W. Zhou, Z. Fan, J.-j. Kai, and Y. Lu. Enhanced tensile ductility of tungsten microwires via high-density dislocations and reduced grain boundaries. 95:193–202. ISSN 1005-0302. doi: 10.1016/j.jmst.2021.04.021. URL `https://www.sciencedirect.com/science/article/pii/S1005030221004497`.

[15] I. Delgado Ruz and S. R. Schultz. Localising and classifying neurons from high density MEA recordings. 233:115–128. ISSN 0165-0270. doi: 10.1016/j.jneumeth.2014.05.037. URL `https://www.sciencedirect.com/science/article/pii/S0165027014002052`.

[16] K. D. Harris, D. A. Henze, J. Csicsvari, H. Hirase, and G. Buzsáki. Accuracy of tetrode spike separation as determined by simultaneous intracellular and extracellular measurements. 84(1):401–414. ISSN 0022-3077. doi: 10.1152/jn.2000.84.1.401.

[17] G. Hilgen, M. Sorbaro, S. Pirmoradian, J.-O. Muthmann, I. E. Kepiro, S. Ullo, C. J. Ramirez, A. Puente Encinas, A. Maccione, L. Berdondini, V. Murino, D. Sona, F. Cella Zanacchi, E. Sernagor, and M. H. Hennig. Unsupervised spike sorting for large-scale, high-density multielectrode arrays. 18(10):2521–2532. ISSN 22111247. doi: 10.1016/j.celrep.2017.02.038. URL `https://linkinghub.elsevier.com/retrieve/pii/S221112471730236X`.

[18] D. N. Hill, S. B. Mehta, and D. Kleinfeld. Quality metrics to accompany spike sorting of extracellular signals. 31(24):8699–8705. ISSN 0270-6474. doi: 10.1523/JNEUROSCI.0971-11.2011. URL `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3123734/`.

[19] G. Hong and C. M. Lieber. Novel electrode technologies for neural recordings. 20(6):330–345. ISSN 1471-0048. doi: 10.1038/s41583-019-0140-6. URL `https://www.nature.com/articles/s41583-019-0140-6`. Bandiera_abtest: a Cg_type: Nature Research Journals Number: 6 Primary_atype: Reviews Publisher: Nature Publishing Group Subject_term: Extracellular recording;Nanobiotechnology;Tetrode recording Subject_term_id: extracellular-recording;nanobiotechnology;tetrode-recording.

[20] A. Hyvarinen. Fast and robust fixed-point algorithms for independent component analysis. 10(3):626–634. ISSN 1941-0093. doi: 10.1109/72.761722. Conference Name: IEEE Transactions on Neural Networks.

[21] A. Hyvärinen and E. Oja. Independent component analysis: algorithms and applications. 13(4):411–430. ISSN 08936080. doi: 10.1016/S0893-6080(00)00026-5. URL `https://linkinghub.elsevier.com/retrieve/pii/S0893608000000265`.

[22] A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*, volume 26. ISBN 978-0-471-40540-5. doi: 10.1002/0471221317. Journal Abbreviation: Independent Component Analysis Publication Title: Independent Component Analysis.

[23] J. J. Jun, C. Mitelut, C. Lai, S. L. Gratiy, C. A. Anastassiou, and T. D. Harris. Real-time spike sorting platform for high-density extracellular probes with ground-truth validation and drift correction, . URL `https://www.biorxiv.org/content/10.1101/101030v2`. Company: Cold Spring Harbor Laboratory Distributor: Cold Spring Harbor Laboratory Label: Cold Spring Harbor Laboratory Section: New Results Type: article.

[24] J. J. Jun, N. A. Steinmetz, J. H. Siegle, D. J. Denman, M. Bauza, B. Bar-

barits, A. K. Lee, C. A. Anastassiou, A. Andrei, Aydın, M. Barbic, T. J. Blanche, V. Bonin, J. Couto, B. Dutta, S. L. Gratiy, D. A. Gutnisky, M. Häusser, B. Karsh, P. Ledochowitsch, C. M. Lopez, C. Mitelut, S. Musa, M. Okun, M. Pachitariu, J. Putzeys, P. D. Rich, C. Rossant, W.-l. Sun, K. Svoboda, M. Carandini, K. D. Harris, C. Koch, J. O'Keefe, and T. D. Harris. Fully integrated silicon probes for high-density recording of neural activity. 551(7679):232–236, . ISSN 1476-4687. doi: 10.1038/nature24636. URL https://www.nature.com/articles/nature24636. Bandiera_abtest: a Cg_type: Nature Research Journals Number: 7679 Primary_atype: Research Publisher: Nature Publishing Group Subject_term: Extracellular recording;Neural circuits;Pattern vision Subject_term_id: extracellular-recording;neural-circuit;pattern-vision.

[25] D. Jäckel, U. Frey, M. Fiscella, F. Franke, and A. Hierlemann. Applicability of independent component analysis on high-density microelectrode array recordings. 108 (1):334–348. ISSN 0022-3077, 1522-1598. doi: 10.1152/jn.01106.2011. URL https://www.physiology.org/doi/10.1152/jn.01106.2011.

[26] J. W. Kamande, T. Nagendran, J. Harris, and A. M. Taylor. Multi-compartment microfluidic device geometry and covalently bound poly-d-lysine influence neuronal maturation. 7:84. ISSN 2296-4185. doi: 10.3389/fbioe.2019.00084. URL https://www.frontiersin.org/article/10.3389/fbioe.2019.00084.

[27] K. J. Laboy-Juárez, S. Ahn, and D. E. Feldman. A normalized template matching method for improving spike detection in extracellular voltage recordings. 9(1):12087. ISSN 2045-2322. doi: 10.1038/s41598-019-48456-y. URL https://www.nature.com/articles/s41598-019-48456-y. Bandiera_abtest: a Cc_license_type: cc_by Cg_type: Nature Research Journals Number: 1 Primary_atype: Research Publisher: Nature Publishing Group Subject_term: Neurophysiology;Sensory processing Subject_term_id: neurophysiology;sensory-processing.

[28] C. Leibig, T. Wachtler, and G. Zeck. Unsupervised neural spike sorting for high-density microelectrode arrays with convolutive independent component analysis. 271: 1–13. ISSN 1872-678X. doi: 10.1016/j.jneumeth.2016.06.006.

[29] J. Magland, J. J. Jun, E. Lovero, A. J. Morley, C. L. Hurwitz, A. P. Buccino, S. Garcia, and A. H. Barnett. SpikeForest, reproducible web-facing ground-truth validation of automated neural spike sorters. 9:e55167. ISSN 2050-084X. doi: 10.7554/eLife.55167. URL https://doi.org/10.7554/eLife.55167. Publisher: eLife Sciences Publications, Ltd.

[30] E. M. Maynard, C. T. Nordhausen, and R. A. Normann. The utah intracortical electrode array: A recording structure for potential brain-computer interfaces. 102 (3):228–239. ISSN 0013-4694. doi: 10.1016/S0013-4694(96)95176-0. URL `https://www.sciencedirect.com/science/article/pii/S0013469496951760`.

[31] M. A. L. Nicolelis, D. Dimitrov, J. M. Carmena, R. Crist, G. Lehew, J. D. Kralik, and S. P. Wise. Chronic, multisite, multielectrode recordings in macaque monkeys. 100 (19):11041–11046. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.1934665100. URL `https://www.pnas.org/content/100/19/11041`. Publisher: National Academy of Sciences Section: Biological Sciences.

[32] M. Pachitariu, N. Steinmetz, S. Kadir, M. Carandini, and H. K. D. Kilosort: realtime spike-sorting for extracellular electrophysiology with hundreds of channels. URL `https://www.biorxiv.org/content/10.1101/061481v1`. Company: Cold Spring Harbor Laboratory Distributor: Cold Spring Harbor Laboratory Label: Cold Spring Harbor Laboratory Section: New Results Type: article.

[33] R. Q. Quiroga, Z. Nadasdy, and Y. Ben-Shaul. Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering. 16(8):1661–1687. ISSN 0899-7667, 1530-888X. doi: 10.1162/089976604774201631. URL `https://direct.mit.edu/neco/article/16/8/1661-1687/6903`.

[34] S. Ramaswamy, J.-D. Courcol, M. Abdellah, S. R. Adaszewski, N. Antille, S. Arsever, G. Atenekeng, A. Bilgili, Y. Brukau, A. Chalimourda, G. Chindemi, F. Delalondre, R. Dumusc, S. Eilemann, M. E. Gevaert, P. Gleeson, J. W. Graham, J. B. Hernando, L. Kanari, Y. Katkov, D. Keller, J. G. King, R. Ranjan, M. W. Reimann, C. Rössert, Y. Shi, J. C. Shillcock, M. Telefont, W. Van Geit, J. Villafranca Diaz, R. Walker, Y. Wang, S. M. Zaninetta, J. DeFelipe, S. L. Hill, J. Muller, I. Segev, F. Schürmann, E. B. Muller, and H. Markram. The neocortical microcircuit collaboration portal: a resource for rat somatosensory cortex. 9:44. ISSN 1662-5110. doi: 10.3389/fncir.2015.00044. URL `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4597797/`.

[35] M. N. Rasband. The axon initial segment and the maintenance of neuronal polarity. 11(8):552–562. ISSN 1471-0048. doi: 10.1038/nrn2852. URL `https://www.nature.com/articles/nrn2852`. Bandiera_abtest: a Cg_type: Nature Research Journals Number: 8 Primary_atype: Reviews Publisher: Nature Publishing Group Subject_term: Diseases of the nervous system;Ion channels;Neuronal physiology;Synaptic plasticity Subject_term_id: diseases-of-the-nervous-system;ion-channels;neuronal-physiology;synaptic-plasticity.

[36] C. Rossant, S. N. Kadir, D. F. M. Goodman, J. Schulman, M. L. D. Hunter, A. B. Saleem, A. Grosmark, M. Belluscio, G. H. Denfield, A. S. Ecker, A. S. Tolias, S. Solomon, G. Buzsáki, M. Carandini, and K. D. Harris. Spike sorting for large, dense electrode arrays. 19(4):634–641. ISSN 1546-1726. doi: 10.1038/nn.4268. URL `https://www.nature.com/articles/nn.4268`. Bandiera_abtest: a Cg_type: Nature Research Journals Number: 4 Primary_atype: Research Publisher: Nature Publishing Group Subject_term: Neural decoding;Software Subject_term_id: neural-decoding;software.

[37] H. Shin, S. Jeong, J.-H. Lee, W. Sun, N. Choi, and I.-J. Cho. 3d high-density microelectrode array with optical stimulation and drug delivery for investigating neural circuit dynamics. 12(1):492. ISSN 2041-1723. doi: 10.1038/s41467-020-20763-3. URL `https://www.nature.com/articles/s41467-020-20763-3`. Bandiera_abtest: a Cc_license_type: cc_by Cg_type: Nature Research Journals Number: 1 Primary_atype: Research Publisher: Nature Publishing Group Subject_term: Biomedical engineering;Extracellular recording;Neural circuits;Optogenetics Subject_term_id: biomedical-engineering;extracellular-recording;neural-circuit;optogenetics.

[38] Y. Son, H. Jenny Lee, J. Kim, H. Shin, N. Choi, C. Justin Lee, E.-S. Yoon, E. Yoon, K. D. Wise, T. Geun Kim, and I.-J. Cho. In vivo optical modulation of neural signals using monolithically integrated two-dimensional neural probe arrays. 5:15466. ISSN 2045-2322. doi: 10.1038/srep15466. URL `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4616027/`.

[39] N. A. Steinmetz, C. Aydin, A. Lebedeva, M. Okun, M. Pachitariu, M. Bauza, M. Beau, J. Bhagat, C. Böhm, M. Broux, S. Chen, J. Colonell, R. J. Gardner, B. Karsh, D. Kostadinov, C. Mora-Lopez, J. Park, J. Putzeys, B. Sauerbrei, R. J. J. v. Daal, A. Z. Vollan, M. Welkenhuysen, Z. Ye, J. Dudman, B. Dutta, A. W. Hantman, K. D. Harris, A. K. Lee, E. I. Moser, J. O'Keefe, A. Renart, K. Svoboda, M. Häusser, S. Haesler, M. Carandini, and T. D. Harris. Neuropixels 2.0: A miniaturized high-density probe for stable, long-term brain recordings, . URL `https://www.biorxiv.org/content/10.1101/2020.10.27.358291v1`. Company: Cold Spring Harbor Laboratory Distributor: Cold Spring Harbor Laboratory Label: Cold Spring Harbor Laboratory Section: New Results Type: article.

[40] N. A. Steinmetz, C. Koch, K. D. Harris, and M. Carandini. Challenges and opportunities for large-scale electrophysiology with neuropixels probes. 50:92–100, . ISSN 0959-4388. doi: 10.1016/j.conb.2018.01.009. URL `https://www.sciencedirect.`

com/science/article/pii/S0959438817303161.

[41] S. Takeuchi, T. Suzuki, K. Mabuchi, and H. Fujita. 3d flexible multichannel neural probe array. 14(1):104–107. ISSN 0960-1317. doi: 10.1088/0960-1317/14/1/014. URL https://doi.org/10.1088/0960-1317/14/1/014. Publisher: IOP Publishing.

[42] D. J. Tolhurst, D. Smyth, and I. D. Thompson. The sparseness of neuronal responses in ferret primary visual cortex. 29(8):2355. doi: 10.1523/JNEUROSCI.3869-08.2009. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6666250/. Publisher: Society for Neuroscience.

[43] R. Veerabhadrappa, M. Ul Hassan, J. Zhang, and A. Bhatti. Compatibility evaluation of clustering algorithms for contemporary extracellular neural spike sorting. 14:34. ISSN 1662-5137. doi: 10.3389/fnsys.2020.00034. URL https://www.frontiersin.org/article/10.3389/fnsys.2020.00034/full.

[44] Y. N. Wu. Statistical independence. In K. Ikeuchi, editor, *Computer Vision: A Reference Guide*, pages 759–760. Springer US. ISBN 978-0-387-31439-6. doi: 10.1007/978-0-387-31439-6_744. URL https://doi.org/10.1007/978-0-387-31439-6_744.

[45] Z. Yang, Q. Zhao, E. Keefer, and W. Liu. UT southwestern medical center,.

[46] V. Zarzoso and P. Comon. Comparative speed analysis of FastICA. In M. E. Davies, C. J. James, S. A. Abdallah, and M. D. Plumbley, editors, *Independent Component Analysis and Signal Separation*, volume 4666, pages 293–300. Springer Berlin Heidelberg. ISBN 978-3-540-74493-1 978-3-540-74494-8. doi: 10.1007/978-3-540-74494-8_37. URL http://link.springer.com/10.1007/978-3-540-74494-8_37. Series Title: Lecture Notes in Computer Science.

List of Figures

Acknowledgements

The work presented by this thesis was conducted at the ETH Bioengineering Laboratory (BEL) in Basel. First and foremost, I would like to thank Prof. Pedrocchi who made this great experience possible.

Huge thanks go to the whole BEL team/staff for you warm welcome. Thanks to Mario for cheering me up when I needed it and thank you all Chloe, Ebrahim, Isabel, Miral, Murezi, Neethu and Svenja for the wonderful time I had at the lab. A special thank you note goes to Alessio who supervised the whole work. Thank you for supporting me, encouraging me and keeping me on track during these past five months. Thank you for all the new knowledge you gave me: I close this mind-opening experience with a whole new perspective and interest in programming and neuroscience.

Last but not least, thank you Jannick, Kathrine(s), Lorenzo, Philippe, Sacha and Severin for the fantastic free time we spent together and for showing me the wonders of Basel.