Sliding window strategy for convolutional spike sorting with Lasso Algorithm, theoretical guarantees and complexity

Laurent Dragoni, Rémi Flamary, Karim Lounici, Patricia Reynaud-Bouret

Sliding window strategy for convolutional spike sorting with Lasso

Algorithm, theoretical guarantees and complexity

Laurent Dragoni · **Rémi Flamary** · **Karim Lounici** · **Patricia Reynaud-Bouret**

Abstract We present a fast algorithm for the resolution of the Lasso for convolutional models in high dimension, with a particular focus on the problem of spike sorting in neuroscience. Making use of biological properties related to neurons, we explain how the particular structure of the problem allows several optimizations, leading to an algorithm with a temporal complexity which grows linearly with respect to the size of the recorded signal and can be performed online. Moreover the spatial separability of the initial problem allows to break it into subproblems, further reducing the complexity and making possible its application on the latest recording devices which comprise a large number of sensors. We provide several mathematical results: the size and numerical complexity of the subproblems can be estimated mathematically by using percolation theory. We also show under reasonable assumptions that the Lasso estimator retrieves the true support with large probability. Finally the theoretical time complexity of the algorithm is given. Numerical simulations are also provided in order to illustrate the efficiency of our approach.

Keywords Sparsity · Lasso · Optimization · Neuroscience · Spike sorting

Laurent Dragoni
Université Côte d'Azur, CNRS, Laboratoire J.A. Dieudonné, 06108 Nice, France
Tel.: +33 (0)4 89 15 62 78
E-mail: laurent.dragoni@univ-cotedazur.fr

Rémi Flamary
École Polytechnique, Centre de Mathématiques Appliquées, 91128 Palaiseau, France
Tel.: +33 (0)1 69 33 46 25
E-mail: remi.flamary@polytechnique.edu

Karim Lounici
École Polytechnique, Centre de Mathématiques Appliquées, 91128 Palaiseau, France
Tel.: +33 (0)1 69 33 46 48
E-mail: karim.lounici@polytechnique.edu

Patricia Reynaud-Bouret
Université Côte d'Azur, CNRS, Laboratoire J.A. Dieudonné, 06108 Nice, France
Tel.: +33 (0)4 89 15 04 96
E-mail: patricia.reynaud-bouret@univ-cotedazur.fr

1 Introduction

The main focus of the field of neuroscience consists in better understanding how the brain works. From brain activity recordings, obtained for instance by extracellular electrodes, the goal of the experimenter is to analyze the recorded signals in order to gain insights about important aspects of the neural code, for instance the firing rate coding or the synchronization between neurons [1,19,13]. It is well established that neurons communicate by emitting shorts bursts of current, called action potentials. Unfortunately, in practice the recorded signals are often a mixture of these action potentials from the neuronal population. A fundamental pre-processing step allowing further analyses aims at extracting from the recorded signals the mixture and spiking activity of each neurons. This step is called *spike sorting*. The main idea relies on a important property: since the shapes of the action potentials emitted by the individual neurons do not vary too much along time, these shapes can be viewed as the signature of a particular neuron. Therefore the goal of spike sorting is to gather the shapes of the action potentials, to associate them to their respective neuron and to detect at which times each particular neuron has emitted an action potential (i.e. to determine the spike train of each neuron).

Numerous works have focused on designing efficient spike sorting procedures lately [21, 11,12]. The estimation of the shapes of the action potentials usually involves dimensionality reduction techniques (such as PCA) and clustering. Assuming all the spike shapes are known, a template matching procedure associates each detected spike to the neuron which has the closest shape [28]. Despite their popularity in the neuroscience community, these methods have some severe limitations. A large number of manual calibrations from the experimenter is often required in order to recover both the shapes and the spike trains. These tasks become even harder when the number of spike synchronizations increases, because the shapes of action potentials are mixed. Therefore, the outcomes of these methods heavily depend on the person using them [37,15]. These shortcomings may also limit the usability of the most recent acquisition devices called *MultiElectrode Arrays* (MEA) that can contain up to several thousand electrodes, whereas the classical recordings use only 4 electrodes, with devices called *tetrodes*. These MEA devices tend to produce large datasets in which the spike synchronization phenomenon worsens [10], making the manual sorting mentioned above infeasible.

In this paper, we propose a spike sorting procedure that is able to analyze large datasets without the need for a large number of manual calibrations. More specifically, our procedure relies on a convolutional model in order to link the activity of the neurons and the recorded signals. This model aims at reconstructing the recorded signal as a temporal convolution between the shapes of the action potentials and the sparse neurons activations. This type of model, originally proposed in [33] and applied for instance to speech signals [31], has also been used with success for spike sorting [11,12]. A nice characteristic of this model is based on the linearity of the convolution operator. This permits to treat synchronizations as additive superpositions of the shapes, which is not possible with traditional spike sorting relying on clustering [21]. We focus here on the estimation of the activations of the neurons by assuming that the shapes of action potentials are known, a problem commonly refered to convolutional sparse coding. We explain in the next section why the hard part is indeed the estimation of the activations. We also see that this estimation leads to a large scale Lasso convex optimization problem, for which well-established strategies exist. The basis pursuit estimator proposed in [11], which is equivalent to the Lasso estimator, verifies important statistical properties. For example, under specific conditions [4,7,23], it can be shown that it is able to recover the support of the activations (i.e. the activation times and the active

neurons). Unfortunately computing this estimator is hard in practice since its computational complexity increases cubically with the number of variables of the optimization problem, which is here proportional to the length of the signal. In the more general context of sparse coding problems, [20] proposed the Feature Sign Search (FSS) algorithm, which essentially consists in a working set strategy paired with the resolution of a QP problem. They notably showed that this strategy grants better results than the LARS on practical applications. [14] proposed an extension of FSS to convolutional sparse coding applied to audio classification, by splitting the signal into smaller temporal windows. Although they managed to improve the performances of the original FSS strategy on large signals, the fact that these windows were fixed a priori required to consider multiple passes on the whole signal. Another similar approach, using an a priori partition of the signal proposed by [25], aimed at solving smaller supbroblems with a local coordinate descent algorithm. But since the structure of this strategy imposes to wait for the convergence on every windows to reach global convergence, its domains of applications remain limited to offline analyses.

After introducing the convolutional model, we present an efficient algorithm for the computation of the Lasso. This algorithm refines the working set strategy by using temporal sliding windows, allowing it to scale in high dimension. In contrast with [14, 25], our method requires a single pass on the whole signal and is fast enough to allow an online analysis.

Then we explain how we can take advantage of some biological facts in order to prove that the Lasso enjoys nice statistical properties. Furthermore we derive theoretical time complexity of the algorithm. In particular, we note that it grows linearly with respect to the size of the recorded signal. Other works have tackled similar optimization problems [17, 18], but to the best of our knowledge the approach that we present is the first to fully take advantage of the structure of the problem and to attain linear complexity. Interestingly since we only perform linear operations such as convolutions, this algorithm can be adapted to GPU architectures, allowing a very efficient scaling. Finally we present some numerical results illustrating both the theoretical results and the performance of our approach. Note that this algorithm is designed to solve an estimation problem associated to a convolutional model, and it can potentially be used to other domains, as long as the quantities of interest verifies similar sparsity properties as in the spike sorting problem (for instance the recognition of musical notes).

2 Physical model & Optimization

2.1 Convolutional model

During an experiment, E electrodes record the activity of N neurons. Each electrode records a signal of size T (number of time steps). We propose to model the link between the activity of the neurons and the recorded signals using a convolutional model, introduced by [11]. This model is written as

$$\mathbf{Y} = \sum_{n=1}^{N} \mathbf{W}_n * \mathbf{a}_n^\star + \boldsymbol{\Xi}, \tag{1}$$

where $\mathbf{Y} \in \mathbb{R}^{E\times T}$ is the matrix of the observations containing the E recorded signals of size T, $\boldsymbol{\Xi} \in \mathbb{R}^{E\times T}$ is a random noise matrix, and $*$ is the *convolution* operator along time defined for any two vectors $\mathbf{x} \in \mathbb{R}^\ell$ and $\mathbf{y} \in \mathbb{R}^T$ as

$$(\mathbf{x} * \mathbf{y})_n = \sum_{i=1}^{\min(\ell,n)} x_i \, y_{n-i+1}.$$

Quantity	Notation	Orders of magnitude
Number of electrodes	E	4-4000
Number of neurons	N	1-1000
Number of time steps	T	10^8
Shape length	ℓ	30-150

Table 1 Orders of magnitude for the quantities of the problem. This table illustrates a typical situation for a recording of one hour at the sampling rate of 30kHz. Taking a realistic neuron spiking rate of 30Hz, we then can expect that the number of non zero coordinates in each $\mathbf{a}_n$ is about 10^5, which is very small compared to their size of 10^8.

Note that we suppose 0 values for $\mathbf{y}$ outside of its support as is classical in signal processing [1]. An illustration of the model for given parameters $\mathbf{W}_n$ and $\mathbf{a}_n$ is provided in Figure 2.1. The matrix $\mathbf{W}_n = [\mathbf{w}_{n,1}, \ldots, \mathbf{w}_{n,E}]^\top \in \mathbb{R}^{E\times\ell}$ contains the shapes $\mathbf{w}_{n,e}$ of the action potentials of neuron n on every electrodes e. Note that each shape is described by $\ell \ll T$ points. This model assumes that for any neuron/electrode couple, its shape does not change along time (stationarity of the shape). The vector $\mathbf{a}_n^\star \in \mathbb{R}^T$ is called the activation vector of neuron n. The non zero entries of $\mathbf{a}_n^\star$ correspond to the activation times of this neuron. We recall in table 1 the main quantities from the model and data and provide some orders of magnitude. Note that while a natural assumption would be that vector $\mathbf{a}_n^\star$ is binary (0/1 values), the resulting optimization problem becomes NP-complete. In addition this does not allow to model the change of amplitude for the action potentials, which can occur for neurons [21], typically after a burst of activity.

Let us write $\mathbf{A}^\star$ the matrix in $\mathbb{R}^{N\times T}$ which contains all the $\mathbf{a}_n^\star$ in its columns. Estimating $\mathbf{A}^\star$ when the number of neurons is greater that the number of electrodes would be impossible to tackle without additional structural assumptions. Since the firing rates of the neurons are very small compared to the sampling rate of the signal, $\mathbf{a}_n^\star$ is clearly a sparse vector. This encourages us to consider an estimator of $\mathbf{A}^\star$ promoting sparsity. We choose the well-known Lasso estimator, originally proposed by [34] and applied to spike sorting problems in [11]. The problem writes as:

$$\min_{\mathbf{A}} \quad \left\| \mathbf{Y} - \sum_{n=1}^{N} \mathbf{W}_n * \mathbf{a}_n \right\|_2^2 + 2\lambda \sum_{n=1}^{N} \|\mathbf{a}_n\|_1, \tag{2}$$

where $\lambda > 0$ is the regularization parameter of the Lasso and the norm are defined as the Frobenius norm $\|S\|_2 = \sum_{i,j} S_{ij}^2$ and ℓ_1 norm $\|\mathbf{a}\|_1 = \sum_i |a_i|$. It is the only tuning parameter of the method and its value depends on the signal-to-noise ratio.

The activations $\mathbf{a}_n$ and the shapes $\mathbf{W}_n$ of the model (1) could be estimated by alternative optimization. However, due to the orders of magnitude presented in table 1, the hardest step would be the update of the $\mathbf{a}_n$: indeed, their number of variables growths linearly with T, while ℓ is fixed and small. Therefore, as announced in the introduction, we decide to focus on the estimation of the activations $\mathbf{a}_n$ while the shapes $\mathbf{W}_n$ are assumed to be known. In practice, we can obtain the shapes and the number of neurons from a classical spike sorting algorithm. Our estimation of the activations $\mathbf{a}_n$ then allows to handle correctly the synchronizations and all the spikes that were not sorted by the first algorithm.

[1] For a Python implementation where the handling of the convolution on the borders can be passed as parameter, see for instance `https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.convolve.html`

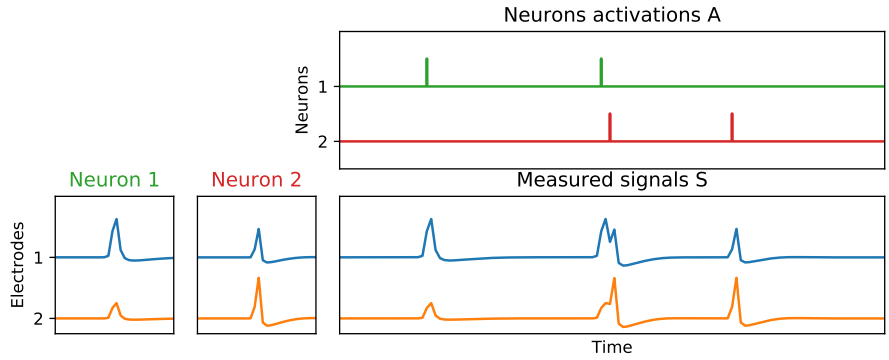

Fig. 1 Convolutional model illustration in the simple setting with $E = 2$ electrodes and $N = 2$ neurons. The shapes of the action potentials are represented in the bottom left part. Remark that for each activation in the top right corner, the corresponding shape appears on the recorded signal in the bottom right corner. Also note the unusual shapes recorded around the center of the graph. This phenomenon may appear when two neurons activate at almost the same time (synchronization). This causes a superposition of the action potentials that creates shapes which are not in the model.

2.2 Vectorizing the Lasso problem and optimality condition

Vectorized model Since the convolution is a linear operator, the convolutional model (1) can be translated as a linear model, doing a vectorization step. We define the vectorization as the concatenation of the temporal signals, *i.e.* the lines of the multivariate signals, with $\mathbf{y} = (Y_{1,1}, Y_{1,2}, \dots, Y_{E,T-1}, Y_{E,T})^\top$ the vectorization of the recorded signals in $\mathbf{Y}$ and $\boldsymbol{\xi} = (\Xi_{1,1}, \Xi_{1,2}, \dots, \Xi_{E,T-1}, \Xi_{E,T})^\top$ the concatenation of the noise signals in $\boldsymbol{\Xi}$ and $\mathbf{a}^\star = (A^\star_{1,1}, A^\star_{1,2}, \dots, A^\star_{N,T-1}, A^\star_{N,T})^\top = (\mathbf{a}_1^{\star\top}, \dots, \mathbf{a}_N^{\star\top})^\top$ the concatenation of the activations in $\mathbf{A}^\star$. Note that in the following we will sometime index vectors $\mathbf{a}$ with doubles indices $a_{n,t} = A_{n,t}$ for readability reason. The linear convolutional model in (1) can be expressed in vectorized format as

$$\mathbf{y} = \mathbf{H}\mathbf{a}^\star + \boldsymbol{\xi} \tag{3}$$

where the matrix $\mathbf{H} \in \mathbb{R}^{ET \times NT}$ is a very structured block Toeplitz matrix (due to the convolutions by the different shapes in $\mathbf{W}$). The columns of $\mathbf{H}$ will be indexed here by a time t and neuron index n for a better readability. The matrix $\mathbf{H} = (\mathbf{h}_{1,1}, \mathbf{h}_{1,2}, \dots, \mathbf{h}_{N,T-1}, \mathbf{h}_{N,T})$ is the concatenation of columns $\mathbf{h}_{n,t}$ corresponding to an activation of neuron n at time t. The column $\mathbf{h}_{n,t}$ can be recovered from the shapes $\mathbf{w}_{n,e}$ with $\mathbf{h}_{n,t} = ((\mathbf{w}_{n,1}^{\to t})^\top, \dots, (\mathbf{w}_{n,E}^{\to t})^\top)^\top$ where $\mathbf{w}^{\to t}$ is a vector of size T where the shape $\mathbf{w}$ has been pushed at position $t > 0$ and its remaining components have value 0.

Optimization problem and KKT The sparse estimation of the activations with the Lasso problem (2) reformulates after vectorization as

$$\hat{\mathbf{a}} = \underset{\mathbf{a} \in \mathbb{R}^{NT}}{\operatorname{argmin}} \quad \|\mathbf{y} - \mathbf{H}\mathbf{a}\|_2^2 + 2\lambda \|\mathbf{a}\|_1 . \tag{4}$$

An essential property of the Lasso are the following necessary and sufficient optimality conditions. $\hat{\mathbf{a}} \in \mathbb{R}^{NT}$ is a Lasso solution, that is a solution of problem (4), if and only if

$$\begin{cases} \mathbf{h}_{n,t}^\top(\mathbf{y} - \mathbf{H}\hat{\mathbf{a}}) = \lambda \operatorname{sign}(\hat{a}_{n,t}) & , \text{if } \hat{a}_{n,t} \neq 0, \\ |\mathbf{h}_{n,t}^\top(\mathbf{y} - \mathbf{H}\hat{\mathbf{a}})| \leq \lambda & , \text{if } \hat{a}_{n,t} = 0. \end{cases} \tag{5}$$

In particular for a Lasso solution $\hat{\mathbf{a}}$, we have for any $1 \leq t \leq T$ and $1 \leq n \leq N$

$$\text{if} \quad |\mathbf{h}_{n,t}^{\top}(\mathbf{y} - \mathbf{H}\hat{\mathbf{a}})| < \lambda, \quad \text{then} \quad \hat{a}_{n,t} = 0. \tag{6}$$

Condition (6) is simple to test and really useful in a context of high sparsity where the majority of coordinates in $\hat{\mathbf{a}}$ are optimal in 0. This suggests the use of an iterative scheme for the resolution of the Lasso problem: starting from the null vector, we can activate iteratively the coordinates of $\hat{\mathbf{a}}$. This strategy, called working set, is presented in more details in the next subsection 2.3.

To describe this algorithm, we need the following notation. For any $J \subset \{1, \dots, NT\}$ and any $\mathbf{a} \in \mathbb{R}^{NT}$, we define $\mathbf{a}_J$ as the vector obtained by keeping only the coordinates from $\mathbf{a}$ which are in J. We also define $\mathbf{H}_J$ as the matrix obtained by keeping only the columns from $\mathbf{H}$ which are in J.

2.3 Generic working set algorithm

From a computational point of view, computing straightforwardly a Lasso estimator, ie a solution of the problem (4), can be very expensive in high dimension even when efficient convolution can be implemented instead of the full matrix product in (4). In this work, we harness the working set strategy (also known as active set) [20,32,6] in order to compute it more efficiently. The main idea of the working set is to activate sequentially the coordinates of the solution using the optimality condition (6) as a criterion.

Principle of the algorithm We call working set and we note J the set of the active coordinates in $1, \dots, NT$ of the solution. We provide a generic formulation of the algorithm in Algorithm 1. Initializing the solution as the null vector, we proceed iteratively as follows: as long as the optimality condition is not verified, we activate the coordinate j_0 that violates the most the KKT condition (6) (line 4 of algorithm 1), we add it to J (line 5), and we update the solution by solving the Lasso problem on the current working set with $\mathbf{H}_J$ (line 6). Thanks to the sparsity of $\mathbf{a}^*$, we expect to solve several Lasso problems of size $\leq |J|$, which remains small compared to ET. In the worst case, the working set algorithm would activate every possible coordinates, thus it ends in finite time. In practice, the computation of the optimality condition vector (line 3) and the update of the solution (ligne 6) are the most expensive steps of the algorithm.

Efficient implementation on convolutional models One simple approach to solve problem (2) would be to pre-compute the matrix $\mathbf{H}$ and use it for the KKT line 3 and Lasso solvers line 6 in Algo. 1. But this approach does not scale properly in memory since the memory complexity of storing $\mathbf{H}$ is $O(ENT^2)$ where T is typically very large. The computational complexity of line 3 for a dense matrix $\mathbf{H}$ is also $O(ENT^2)$ which does not scale well with the problem dimensionality.

But in practice $\mathbf{H}$ is very sparse, completely defined through $\mathbf{W}$, and the convolutional operator can be computed exactly with a much smaller memory footprint. Using direct convolution instead of a general matrix product, one can compute the gradient of the quadratic loss in line 3 for a computational complexity of $O(ENT\ell)$ and a memory complexity of $O((E+N)T)$. This means that this operation can be used to compute efficiently the KKT condition in the working set and to compute the gradient in the inner lasso solver line 6. Also note that in addition to the efficient implementation for the convolution operator, the matrix

Algorithm 1 Generic working set algorithm

Require: $\mathbf{y}, \mathbf{H}, \lambda > 0, \varepsilon > 0$
$J \leftarrow \emptyset$, $\hat{\mathbf{a}} \leftarrow \mathbf{0}$
repeat
  $\mathbf{g} \leftarrow \mathbf{H}^\top(\mathbf{y} - \mathbf{H}\hat{\mathbf{a}})$
  $j_0 \leftarrow \text{argmax}_{l \in J^c} |g_l|$
  $J \leftarrow J \cup \{j_0\}$
  $\hat{\mathbf{a}}_J \leftarrow$ Solve Lasso (4) for sub-problem $(\mathbf{H}_J, \mathbf{y})$
until $g_{j_0} < \lambda + \varepsilon$
return $\hat{\mathbf{a}}, J$

$\mathbf{H}_J$ in the working set is still very sparse with only an order of $O(E|J|\ell)$ non-sparse lines, which means that the inner solver can be solved exactly on a much smaller subproblem with a matrix of size $O(E|J|\ell) \times O(|J|)$.

The efficient implementation discussed above allows to solve larger problems but several computational bottleneck persists: at each iteration in the working set one needs to perform $O(ENT\ell)$ operations and since the number of iterations in the working set will be proportional to T it leads to at least a quadratic complexity *w.r.t.* T, which again does not scale well and does not allow to provide real time spike sorting. We propose in section 3 the idea of sliding window working set, which takes advantage of the temporal structure of the problem to solve it more efficiently.

2.4 Biologically based assumptions

We present here some biological properties about neurons and action potentials and explain how these properties translate into mathematical assumptions related to our model. Taking advantage of these properties in later sections, we demonstrate that our estimator of $\mathbf{a}^*$ verifies nice statistical and computational properties. Moreover it also allows us to derive the theoretical temporal complexity of our algorithm.

First we present the following assumption about the support $S^* = Supp(\mathbf{a}^\star)$ of the true model parameter $\mathbf{a}^\star$.

Assumption 1 *(Absolute refractory period) All indices in the support S^* are at least $\ell + 1$ apart.*

This assumption is a mathematical reformulation of the idea of refractory period of a neuron. Right after a neuron fired an action potential, there is a short period of time during which the neuron can not fire again. In the following, we assume that every neuron in the model shares the same refractory period. As stated in page 4 of [8], this refractory period length is several times larger than the action potential length. For simplicity reason we suppose that this period is ℓ, which is the length of the potential shape window. This assumption is of particular importance in our case: it means that the activation of a particular neuron are far away temporally which makes them easier to identify statistically.

After making some assumption on the support of the true model, we make some assumptions on the shapes of the action potentials of the individual neurons. These shape assumptions are better described as properties of the Gram matrix $\mathbf{G} = \mathbf{H}^\top\mathbf{H}$. From the definition of the columns of $\mathbf{H}$ (see below Equation 3), we can recover the following Lemma

Lemma 1 *For all t and t' in $\{1,...,T\}$ and for all n and n' in $\{1,...,N\}$, we have:*

$$G_{(n,t),(n',t')} = \mathbf{h}_{n,t}^{\top}\mathbf{h}_{n',t'} = \sum_{e=1}^{E} (\mathbf{w}_{n,e}^{\rightarrow t})^{\top}\mathbf{w}_{n',e}^{\rightarrow t'} \tag{7}$$

In particular it is null when $|t-t'| > \ell$.

Note from the Lemma above that similarly to the columns $\mathbf{h}_{n,t}$ that are indexed by neuron n and time t we will index the components of $\mathbf{G}$ such as $G_{(n,t),(n',t')}$. We now define below several correlation assumptions on the shapes.

Assumption 2
2.a *(Neurons are recognizable) There exists $\varepsilon > 0$ such that for all $n \neq n'$ and $|t-t'| \leq \ell$*

$$|G_{(n,t),(n',t')}| \leq \varepsilon. \tag{2.a}$$

2.b *(Spikes are peaky) There exists $\rho \in (0,1)$, such that for all n and $0 < |t-t'| \leq \ell$*

$$|G_{(n,t),(n,t')}| \leq \rho. \tag{2.b}$$

2.c *(Shapes have bounded energy). There exist $\overline{c} > \underline{c} > 0$ such that for all n and t,*

$$\underline{c} \leq |G_{(n,t),(n,t)}| \leq \overline{c}. \tag{2.c}$$

Assumption **2.a** is of great importance for the statistical analysis of our methodology. It essentially means that the shapes of two distinct neurons are distinguishable, allowing to attribute each spike to the correct neuron. This is reasonable due to the difference between neurons but also due to their spatial localization that will means different impact on different electrodes [5]. Assumption **2.b** is also reasonable due to the fact that action potentials are also called spikes that will definitely diminish their autocorrelation in the presence of a temporal delay (see classical shapes for instance in [27]). Finally Assumption **2.c** is also physically plausible since an action potential with too small energy would be indistinguishable from recording noise and the potential obviously has a bounded energy [27].

3 Sliding window working set algorithm

In this section we present our novel working set algorithm. This algorithm builds on the fact that thanks to the structure of the problem and of its solution, it can be decomposed into smaller problems. This will be illustrated and discussed next in 3.1. The algorithm is then introduced and illustrated in subsection 3.2 and discussed more in detail in subsection 3.2.

In the sequel we need the following notation. We define a temporal window $\omega = [\![\omega_1, \omega_2]\!]$ where $\omega_1 \leq \omega_2$ and $\omega_1, \omega_2 \in \{1,\ldots,T\}$ that contain all samples whose temporal indice $\omega_1 \leq t \leq \omega_2$. This temporal window will be used in the following to index vectors with $\mathbf{a}_\omega$, that contains the temporals samples $\omega_1 \leq t \leq \omega_2$ for all neurons n, and matrix $\mathbf{H}_\omega$ where are selected only the columns $\mathbf{h}_{n,t}$ where $\omega_1 \leq t \leq \omega_2$, for all neurons n.

3.1 Overlaps and independent problems

We introduce here the notions of spatial and temporal overlaps that will be useful in the remaining. These overlaps will allow us to split the large optimization problem 2 into several smaller scale problems that are individually easier to solve. We first discuss the notion of spatial overlap that will be important in the MEA case and is related to the physical position of the neurons. Next we discuss the temporal overlaps that are related to the temporal activations of the individual neurons.

3.1.1 Spatial overlaps

In the MEA case, solving the problem on the full set of N neurons has a heavy computational cost. In this section, we want to take advantage of an important property of the problem: the spatial distribution of the neurons. Simply put, we harness the fact that two physically distant neurons are not recorded by the same electrodes. Thus their respective spikes should not overlap on any electrode, even if these neurons emit simultaneous spikes.

The problem is in fact a bit more complex than that because there might be transitive effect. Indeed in Figure 2, neuron N_1 and N_3 are recorded by disjoint sets of electrodes, but still it is not possible to speak of disjoint independent Lasso problems, because their spikes might be mixed with the ones of N_2. Hence we need to access the spatial overlaps that they form. We hope that these overlaps will form much smaller sets that the complete set of neurons and this is linked to the spatial distribution of the neurons. So let us first precise why such a phenomenon might appear from a biological point of view in the MEA case. In the tetrode case, the number of electrodes is so small that such phenomenon is not relevant.

Neuron density and localization Typical studies of *in vitro* cultures report roughly 1000 neurons per mm^2 [5]. Note that these cultures usually provide a higher density of cells than in *ex vivo* experiments where slices of brain are used. The range between electrodes in a MEA depends on the type of MEA and might range from 200 μm [5] to about $30\mu m$ [26]. Finally the electrical signal that is emitted by a neuron suffers from various kind of attenuation and people analyzing MEA signals usually think that a neuron is recorded by very few nearby electrodes (for instance only 5 electrodes in the MEA are used by [26], which corresponds to a range of about 200 μm). These orders of magnitude mean that in practice the impact of the activation for a given neuron will be very localized between a few electrodes, which introduces nice properties discussed below. More precisely, in Section 4.1.1, we will leverage percolation results to upper bound the size of the spatial overlaps with large probability.

To fix the configuration, from now on, the MEA case corresponds to E electrodes placed on a square lattice.

Spatial clustering of the neurons Let us now formalize the concept of spatial overlaps to explain the algorithm. Two neurons n an n' are independent when we have:

$$\mathbf{w}_{n,e}^\top \mathbf{w}_{n',e} = 0 \qquad \forall e \tag{8}$$

This condition is true when one of the two shapes is the null vector which happens when the two neurons do not share any electrode where they are both recorded (they do not overlap). Using this pairwise independence, one can easily construct a clustering of the neurons as illustrated in Figure 2.left where 3 independent spatial overlaps are recovered. Note

that (8) implies that between two neurons n and n' in two independent clusters we have $\mathbf{h}_{n,t}^\top \mathbf{h}_{n',t'} = 0, \quad \forall t,t'$, which means that both the quadratic term and the Lasso regularizer are separable in several Lasso subproblems (one by spatial overlap) and that they can be solved independently. This means that by performing beforehand a clustering of the neurons based on the shapes of their action potentials, we can greatly decrease the complexity of the problem. In the following we will suppose that this clustering has been done and that the problem is solved on a subset of neurons (in a spatial overlap) and of electrodes (the ones active inside the spatial overlap).

3.1.2 Temporal overlaps

In addition to splitting the optimization problem thanks to the spatial overlaps of the neurons, one can also use the structure of the problem to split the problem into almost independent temporal windows. Let us first review the biological phenomenon which explains such reduction.

Neuron activations and refractory period Neurons fires quite scarcely and some classical models are either Poisson processes in continuous time with a frequency of usually 10Hz (max 100Hz) or their discrete counterpart that are Bernoulli process (see for instance [35,30] and the reference therein). Both models have been adapted to encode the refractory period, for instance using Poisson with dead time, which basically consists in erasing the spikes that are too close. These more precise variations can only decrease the number of spikes. In case of synchronization, the synchronizations between neurons are usually simulated by joint Poisson or Bernoulli process [35], so that the firing pattern of the whole system remains globally Poisson (or Bernoulli).

Temporal overlap and independent windows Similarly to spatial overlaps we can find independent temporal clusters (temporal windows) of activations. We define two activation at times t and t' as independent if $|t-t'| > \ell$. Indeed in this case the supports of the convolution (of size ℓ) do not overlap and it is easy to show that $\mathbf{h}_{n,t}^\top \mathbf{h}_{n',t'} = 0, \quad \forall n,n'$. This is interesting because it means that for any windows ω and ω' such that $\omega_2 < \omega'_1 + \ell$ we have $\mathbf{H}_\omega^\top \mathbf{H}_{\omega'} = \mathbf{0}$ where $\mathbf{0}$ is the null matrix. This implies again that the optimization problem can be solved independently on ω and ω'.

Similarly to spatial overlap, one can find independent windows that contain the activations of the neurons, as illustrated in Figure 2.right. But note that this time the temporal overlaps cannot be found *A priori* since the actual support of the temporal activations is not known. This means that despite this nice separability of the problem, one cannot use it to speedup the optimization until the support of the solution is known. The main motivation for the sliding window working set algorithm introduced below is to find this support and independent windows in an efficient and online way. Mathematically, the size of the temporal overlaps themselves is also controlled with large probability (see Section 4.1.1) and this will impact the overall complexity of the algorithm.

3.2 Sliding window working set for the global problem

We present here our sliding window working set algorithm. The main idea of the algorithm is to work only on a small temporal window and use the working set principle to simultaneously solve the optimization problem in the window and find the window that is independent

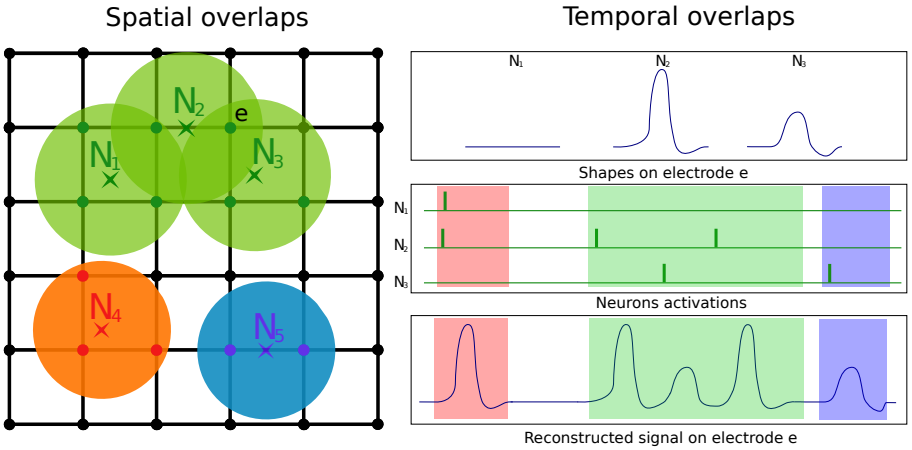

Fig. 2 Illustration of the spatial and temporal overlaps. At the left hand side, we present an example of 3 spatial overlaps in the case of 5 neurons, on a regular grid of 36 electrodes. The position N_j of neuron j is represented by a check, and the reach of its spikes by a disc of radius r. On the right hand side, we provide an example of temporal overlaps for the neurons 1, 2 and 3. We provide the shapes of each neuron and the reconstructed signal on the electrode e. Remark that since neuron 1 is far away from e, its shape on e remains at 0. The independent spatial and temporal overlaps are illustrated with different colors.

from the rest of the signal. Note that this algorithm is used on each independent spatial overlap so in fact for small values of N and E in the MEA case.

Principle of the algorithm The main algorithm is detailed in Algorithm 3.2 where line 3 denotes an update of the large vector $\hat{\mathbf{a}}$ on the current window. The idea is to solve the Lasso problem on small windows ω starting with the beginning of the signal $\omega = [\![1, 4\ell]\!]$ and perform the following operations until the end of the signal is reached:

1. Computing the Lasso solution $\hat{\mathbf{a}}_\omega$, on the window ω with the working set algorithm.
2. Updating the window ω depending on the support of $\hat{\mathbf{a}}_\omega$:
   (a) If the support $Supp(\hat{\mathbf{a}}_\omega) \in [\![\omega_1 + \ell, \omega_2 - 2\ell]\!]$ then the current problem is independent from the rest of the signal and the widow is updated as $\omega = [\![\omega_2 + 1 - \ell, \omega_2 + 3\ell]\!]$.
   (b) If the support $Supp(\hat{\mathbf{a}}_\omega) \cap [\![\omega_1, \omega_1 + \ell - 1]\!] \neq \emptyset$ has components in the first ℓ samples of the widows then we merge the current window with the last (because the KKT conditions on the last ℓ samples in the previous window have changed.)
   (c) Else the window needs to be extended as $\omega = [\![\omega_1, \omega_2 + \ell]\!]$

Once the Lasso is solved on the window ω in step 1, the optimality conditions are verified on the window. If the support of the activation $Supp(\hat{\mathbf{a}}_\omega) \subset [\![\omega_1 + \ell, \omega_2 - 2\ell]\!]$, it means that the reconstructed signal after convolution signal is entirely contained into ω. This means that the solution on this window is independent from the previous one and probably independent from the next one (since there is no activations in the last ℓ samples of the window). In this case, we have found the Lasso solution for the previous window, and then we can work on a new window immediately after it (step 2.a). If there are activations at the beginning of the window however (first ℓ samples), it means that the residual and the KKT conditions have changed on the last ℓ samples of the previous window and we need to potentially update the model there, so we merge the current and previous windows. Else, if there are activations in the last ℓ samples of the window, we extend it in order to

Algorithm 2 Sliding window working set

Require: $\mathbf{y}, \mathbf{H}, \lambda > 0$
1: $\hat{\mathbf{a}} \leftarrow \mathbf{0}$, $\omega = [\![\omega_1, \omega_2]\!] \leftarrow [\![1, 4\ell]\!]$, Empty list of windows $\Omega = []$
2: **repeat**
3: $\hat{\mathbf{a}}_\omega \leftarrow$ Solve Lasso with algo. 1 for sub-problem $(\mathbf{H}_\omega, \mathbf{y})$ using warm-start $\hat{\mathbf{a}}_\omega$
4: **if** $Supp(\hat{\mathbf{a}}_\omega) \subset [\![\omega_1 + \ell, \omega_2 - 2\ell]\!]$ **then**
5: $\Omega \leftarrow [\Omega, \omega]$ //Insert current window ω at the end of list Ω
6: $\omega \leftarrow [\![\omega_2 + 1 - \ell, \omega_2 + 3\ell]\!]$ //Independent problem solved so move window to next time segment
7: **else if** $Supp(\hat{\mathbf{a}}_\omega) \cap [\![\omega_1, \omega_1 + \ell - 1]\!] \neq \emptyset$ **then**
8: $\tilde{\omega}, \Omega \leftarrow$ Return last window $\tilde{\omega} = [\![\tilde{\omega}_1, \tilde{\omega}_2]\!]$ in Ω and remove it from the list Ω.
9: $\omega \leftarrow [\![\tilde{\omega}_1, \omega_2]\!]$ // merge current window with last window
10: **else**
11: $\omega \leftarrow [\![\omega_1, \omega_2 + \ell]\!]$ // Extend window to find the independent temporal overlap
12: **end if**
13: **until** $\omega_2^{(m)} \geq T$
14: **return** $\hat{\mathbf{a}}$, $\Omega = [\omega_1, \omega_2, \dots]$

ensure that at least ℓ samples are not activated at the end. Finally for each iteration after updating the current window, we solve again the Lasso on this window efficiently thanks to the working set strategy. For illustration, one execution of the algorithm with one electrode and two neurons is provided in Figure 3. It shows both configurations: when the window is extended and when the window is shifted to the right.

Algorithm solution w.r.t. *the original Lasso* We now address the following question: is the proposed algorithm actually solving the global optimization problem (2)? We provide to this end the following theorem.

Theorem 1 *The solution* $\hat{\mathbf{a}}$ *computed by the sliding window working set is a solution of the initial Lasso problem* (4).

Proof By construction of the algorithm, line 4, the algorithm will return a list of windows Ω such that $\forall \omega \in \Omega$ the support $Supp(\hat{\mathbf{a}}_\omega) \in [\![\omega_1 + \ell, \omega_2 - 2\ell]\!]$ which means that the current model will have an effect only inside ω and that for two consecutive windows ω and ω' in Ω the temporal indexes of all active variables $\hat{a}_t \neq 0$ with $t \in \omega$ and $\hat{a}_{t'} \neq 0$ with $t' \in \omega'$ are by construction $|t' - t| > 2\ell$. This means that as discussed in subsection 3.1.2, the Lasso problems can be split as two independent problems on the support and all the other components that are not active respect the KKT meaning that they will be 0. The solution $\hat{\mathbf{a}}$ is obtained by successive juxtaposition of the solutions of independent problems on disjoint windows that are estimated line 3.

Numerical complexity and efficient implementation We now discuss why the proposed algorithm is more efficient than the convolutional working set discussed in section 2.3. In the standard working set, we recall that the computational cost for the optimality condition is of order $O(ENT\ell)$ at each step because of the multiple convolutions necessary to compute the KKT conditions. In the sliding window working set, the KKT are computed only on the window ω, and their complexity is reduced to $O(|\omega|NE\ell)$ with $|\omega| << T$. Proving mathematically this reduced computational complexity is one of the main focus of section 4. Note that the complexities above are given on the whole problem, but as discussed above, in the MEA case, the spatial clustering of the neurons means the the complexity depends on the size E_c and N_c of the spatial overlap instead of E and N.

Comparison with a similar approach We can find in [25] a similar strategy, named Distributed Convolution Coordinate Descent (DICOD), for the efficient resolution of large scale convolutional sparse coding formulated as Lasso problems. By considering a temporal partition of the whole signal, their approach aims at solving small local problems using coordinate descent algorithm. Since the update of a coordinate only influences its vicinity, these local problems can be treated in parallel, almost as independent problems. In the same manner as theorem 1, they could prove under mild assumptions that the computed solution is indeed a Lasso solution. Although they demonstrated an important speedup with respect to the global coordinate descent, they do not provide the theoretical complexity of their algorithm with respect to the sizes of the problem. Taking advantage of the biological assumptions presented in section 2.4, we prove in the following sections not only that the Lasso estimator retrieves the true support, but also ascertain the theorical complexity of the sliding window working set algorithm. Moreover, in our approach, the temporal exploration of the signal with a window allows to treat the problem in an online manner.

4 Mathematical results

4.1 Control of the spatial and temporal overlaps

4.1.1 Spatial overlaps

In the MEA case, the E electrodes are placed on a square lattice with fixed distance δ between electrodes. The range of detection of a neuron by an electrode is r_0. We classically approximate the spatial distribution of the neurons on the lattice by a Poisson process of constant intensity γ.

An electrode detects a neuron if it is at distance less than r_0. Therefore two neurons can be detected by the same electrode if their distance is less than r_0. If this is the case, we say that these neurons are "connected". Spatial clusters are given by maximal sets of neurons that are connected together, or for which there exists a path in between of "connected neurons".

This framework is known in probability as a particular case of the Poisson-Boolean percolation (see [9] and references therein). Thanks to this, we can prove the following proposition.

Proposition 1 *There exists a critical value $\gamma_c > 0$ which only depends on r_0, such that, if $\gamma < \gamma_c$, then for all α in $(0,1)$, such that $E \geq \tau \log(3/\alpha)$ for some positive constant τ, there exists an event $\Omega_{\alpha,s}$ of probability larger than $1-\alpha$, such that on $\Omega_{\alpha,s}$, any spatial overlap of neurons c, with cardinality N_c, satisfies*

$$N_c \leq \kappa [\log(E/\alpha)]^2,$$

with $\kappa > 0$, which only depends on γ, δ, τ and r_0.

The event $\Omega_{\alpha,s}$ only depends on the position of the neurons on the lattice representing the MEA.

The proof is given in Section 6.1.

Let us comment this result qualitatively. First of all γ_c is a critical parameter of the percolation theory. When the parameter γ is small with respect to γ_c, as usual for crital percolation parameters, clusters cannot reach infinity, whereas they can if γ is too big. As

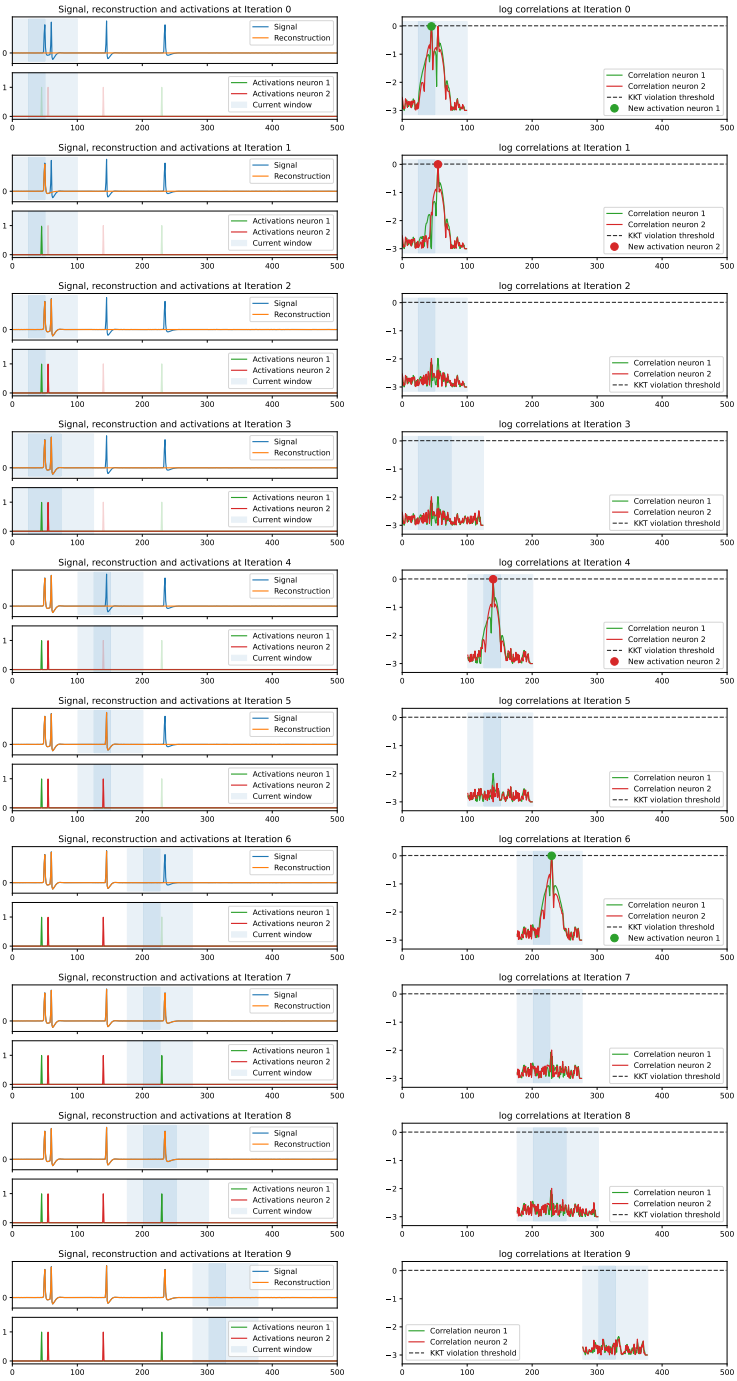

Fig. 3 Illustration of the different steps in the proposed algorithm. (left) Observed signal $\mathbf{S}$ with model reconstruction and sparse model $\mathbf{a}_i$ The current widows is illustrated as a light blue background. True activation are illustrated with transparency (right) KKT violation at the current step. temporal instant and neuron violating the KKT are over the black line.

far as we know, precise knowledge of γ_c is unknown, but one can still have the following heuristic reasoning : if a neuron can be detected at a range of r_0 and if the intensity γ (that is, informally, the density of neurons) is very low, it will be quite rare to have two connected neurons, and the cluster size will be roughly one. On the other hand if γ is too large, the distance between neighbors will be very small and eventually all neurons will belong to one giant cluster.

Note that when the shape of the action potential as perceived by the various electrodes are known, it is very easy to find these spatial overlaps before hand and we will easily know if we are in a subcritical regime where the size of the spatial overlaps vary logarithmically with E or not.

In the rest of the paper,

(i) either we focus on a tetrode like case where E is small, so that we discard totally the dependence in E,
(ii) or on a supercritical regime for a lattice MEA, and then N_c is roughly of the size of N, that is the total number of neurons and we can as well solve the big system,
(iii) or we are in a subcritical regime for a lattice MEA and once restricted to the event $\Omega_{\alpha,s}$, we can solve independently the Lasso problems for each of the spatial overlaps c. In this case, the number of neurons is roughly of the order $(\log E)^2$.

Note that in (i) or (ii), N is thought to be fixed and a parameter of the problem whereas in (iii) the number of neurons is a random variable and the event $\Omega_{\alpha,s}$ to which we restrict ourselves depends on it.

4.1.2 Temporal overlaps

Here we assume that the activations $\mathbf{a}^*$ are the realisation of a given random process. More specifically, and as explained in Section 3.1.2, we do not need to model each neuron individually and we do not need to model the amplitude of $\mathbf{a}^*$. Hence the following formalism can be seen as a worst case scenario.

We denote A the joint process of length T with values 0 or 1, 1 meaning that at least one of the recorded neurons has fired. Note if we are in the subcritical regime, A is restricted to the neurons in a given spatial overlap.

We model A by a Bernoulli process of rate $p = Nm\Delta$, with m the average firing rate and N the number of neurons (of the spatial overlap possibly) (that is the A_i's are i.i.d. Bernoulli with parameter p). Note that in this set up, we force $\Delta << 1/N$ so that we cannot analyze too much neurons at the same time. Another way to see this is to say that p should be small and to fix ideas we assume that $p \leq 1/2$.

We are saying that two successive spikes t and t' in A are overlapping if $|t-t'| \leq \eta = 4\ell$.

As from a spatial point of view, from a temporal point of view, the spikes in A that includes all the activation times of all the neurons (of the spatial overlap) are therefore partitioned in overlaps. We can, as for the spatial overlaps, control their size.

Proposition 2 *The temporal overlaps are controlled as follows.*

– *In the non-subcritical MEA case or in the tetrode case, with a global activation rate $p = Nm\Delta \leq 1/2$, there exists an event $\Omega_{\alpha,A}$ of probability larger than $1-\alpha$ such that on $\Omega_{\alpha,A}$, each temporal overlaps has a length bounded by*

$$\mathscr{W} = c' \log(T/\alpha),$$

with $c' > 0$ depending only on η.

– *In the subcritical MEA case, with an activation rate per cluster $p_c = N_c m\Delta \leq 1/2$, there exists an $\Omega_{\alpha,A,s}$ such that, on $\Omega_{\alpha,A,s}$, for each spatial overlap and for each temporal overlap inside a spatial overlap, the size of this temporal overlap is bounded by*

$$\mathscr{W} = c'' \log(ET/\alpha),$$

with c" > 0 depending only on $\eta, \delta, \gamma, \tau$ and r_0.

The proof is given in Section 6.2.

In the subcritical MEA Case, the condition is on the activation rate per cluster ($p_c \leq 1/2$), which means that the result holds valid even when the global activation rate is $p > 1/2$. In this sense, if the problem is subcritical, even if the MEA is very large and record a huge number of neurons, the size of the temporal overlaps, which governs the numerical complexity, will still be reasonable.

Note that in the first case, the event $\Omega_{\alpha,A}$ depends only on the distribution of the spikes, whereas in the second case, $\Omega_{\alpha,A,s}$ depends both on the spiking distribution but also on the spatial distribution of the neurons.

4.2 Control of the noise

Lemma 2 *Assume that the noises $((\xi_{e,t})_{e,t})$ are i.i.d. normal random variables with zero mean and finite variance σ^2. For $\alpha \in (0,1)$, define*

$$z_\alpha = \sqrt{2\sigma^2 \bar{c} \log\left(\frac{2NT}{\alpha}\right)},$$

where $\bar{c}$ is given in (2.c) and the event

$$\Omega_{\alpha,\xi} = \bigcap_{n,t} \left\{ \left| h_{n,t}^\top \xi \right| \leq z_\alpha \right\}. \tag{9}$$

Then we have

$$\mathbb{P}\left(\Omega_{\alpha,\xi}\right) \geq 1 - \alpha.$$

The proof is given in Section 6.3.

This lemma is very classical and help us to control the level of the noise. From now on, Ω_α refers to the event of probability controlled by $1-\alpha$ which is either $\Omega_{\alpha/2,A} \cap \Omega_{\alpha/2,\xi}$ in the tetrode or non-subcritical MEA case, or $\Omega_{\alpha/2,A,s} \cap \Omega_{\alpha/2,\xi}$ in the subcritical MEA case.

4.3 Theoretical properties of the Lasso estimator

Theorem 2 *Fix $\alpha \in (0,1/2)$. Let Assumptions 1 and 2 be satisfied and let us assume that the noises are i.i.d. Gaussian. With the notation of Propositions 1, 2 and Lemma 2, there exists an event Ω_α of probability larger than $1-\alpha$ such that, on Ω_α, for all temporal window ω and any solution $\hat{\mathbf{a}}_\omega$ of the Lasso problem $(\mathbf{H}_\omega, \mathbf{y})$ with regularization parameter λ (possibly restricted in the subcritical MEA case to any spatial overlap), the following holds.*

1. *No spurious activation is discovered, that is*

$$Supp(\hat{\mathbf{a}}_\omega) \subset S^* \cap \omega,$$

where $S^* = Supp(a^*)$ *is the true set of activations, as long as*

$$\lambda > \frac{\underline{c}+2\rho}{\underline{c}-2\rho-4\varepsilon\mathscr{N}}\left(z_\alpha + 2(\rho+\varepsilon\mathscr{N})\|\mathbf{a}^*\|_{\infty,\partial\omega}\right) \quad \textit{and} \quad \underline{c} > 2\rho+4\varepsilon\mathscr{N} \tag{10}$$

with the convention that

$$\|\mathbf{a}^*\|_{\infty,\partial\omega} = \sup_{n,t\in\partial\omega} |\mathbf{a}^*_{n,t}|,$$

where the boundary $\partial\omega = \{t \notin \omega / \exists s \in \omega, |t-s| < \ell\}$ *and with*

$$\mathscr{N} = \begin{cases} N \textit{ in the tetrode or non subcritical MEA case} \\ \kappa[\log(E/\alpha)]^2 \textit{ in the subcritical MEA case} \end{cases}.$$

2. *Moreover, if*

$$\inf_{(n,t)\in S^*\cap\omega} |\mathbf{a}^*_{n,t}| > \frac{z_\alpha + \lambda + 2\|\mathbf{a}^*\|_{\infty,\partial\omega}(\rho+\varepsilon\mathscr{N})}{\underline{c}-2\varepsilon\mathscr{N}}, \tag{11}$$

then

$$Supp(\hat{a}_\omega) = S^* \cap \omega.$$

See Section 6.4 for the proof.

This theorem is stronger than the usual retrieval of support for Lasso estimator. Indeed it first applies to all possible subwindows at the same time, including the whole Lasso estimator itself. Next it does not calibrate λ by the level of sparsity, that is $|S^*|$. Indeed in our problem even if $|S^*|$ is small compared to T, this grows linearly with T since in expectation, under the assumptions of Proposition 2, it is roughly pT.

Let us now discuss a bit more the choice of λ and the calibration conditions. The first stringent condition is

$$\underline{c} > 2\rho + 4\varepsilon\mathscr{N}.$$

Note that by Cauchy Schwarz , we already have that $\underline{c} > \rho$, so what we ask here is a little bit stronger. The shape of the action potentials need to be picky enough to have ρ small. In the same way, we need action potential shapes that are sufficiently different to have ε small. The multiplication by $\mathscr{N}$ is in fact very large and a conservative upper-bound to the phenomenon taking place here. By looking at the proof, we can see that this is in fact the number of neurons (in a spatial overlap) that synchronizes with a lag less than ℓ, that is a few milliseconds. In practice, if this phenomenon is important for the neural coding [36], it usually involves a few neurons, except during epileptic crisis.

Next, $\mathbf{a}^*$ is usually assumed to be a binary 0/1 vector in the classical problem. Therefore (10) means that we need

$$\lambda > \frac{\underline{c}+2\rho}{\underline{c}-2\rho-4\varepsilon\mathscr{N}}\left(\sqrt{2\sigma^2\bar{c}\log(2NT/\alpha)} + 2(\rho+\varepsilon\mathscr{N})\right),$$

On the other hand, Condition (11) becomes

$$\lambda < \underline{c} - 2\rho - 4\varepsilon\mathscr{N} - \sqrt{2\sigma^2\bar{c}\log(2NT/\alpha)}.$$

So there is room to find such a λ if typically

$$\underline{c} > \max\left(10\rho + 8\varepsilon_{\mathscr{N}} \quad , \quad 3\sqrt{2\sigma^2\overline{c}\log(2NT/\alpha)} + 4\rho + 6\varepsilon_{\mathscr{N}}\right).$$

This condition is reasonable in the context of spike sorting with high energy and peaky spike shapes, weak correlations between different neurons and normal neural activity.

Moreover the windows generated by Algorithm 3.2 are built to guarantee that

$$\|\hat{\mathbf{a}}_\omega\|_{\infty,\partial\omega} = 0,$$

which would imply that

$$\|\mathbf{a}^*_\omega\|_{\infty,\partial\omega} = 0$$

So in practice,

$$\lambda > \frac{\underline{c} + 2\rho}{\underline{c} - 2\rho - 4\varepsilon_{\mathscr{N}}}\sqrt{2\sigma^2\overline{c}\log(2NT/\alpha)}$$

should be sufficient.

Also the windows which are generated by the algorithm can be controlled, as we can see in the following result.

Corollary 1 *Fix $\alpha \in (0,1/2)$. Let Assumptions 1 and 2 be satisfied and let us assume that the noise variables are i.i.d. Gaussian. With the notations of Propositions 1, 2 and Lemma 2, on the same event Ω_α of probability larger than $1-\alpha$, if λ is chosen so that* (10) *and* (11) *are satisfied, then all the windows $\omega \in \Omega$ constructed by Algorithm 3.2 have a length controlled by*

$$\mathscr{W} = \begin{cases} c'\log(T/\alpha) & \text{in the non-subcritical MEA or the tetrode case} \\ & \quad \text{as soon as } p = Nm\Delta \leq 1/2, \\ c"\log(ET/\alpha) & \text{in the subcritical MEA case} \\ & \quad \text{as soon as the rate per cluster } p_c = N_c m\Delta \leq 1/2. \end{cases}$$

Also on the same event, steps 7,8 and 9 of the algorithm never occur.

4.4 Complexity of the sliding window working set algorithm

We recall first an important existing result which gives the general (approximate) complexity of solving the Lasso with a working set algorithm.

Theorem 3 ([22], Section 2.4) *Consider the Lasso problem* (3) *with n observations and p features. Then in order to compute a Lasso solution which selects k features out of p, the working set algorithm has an approximate complexity of*

$$C_{ws}(n,p,k) = O(n^2pk + nk^3 + k^4). \tag{12}$$

Therefore applying Theorem 3 to the resolution of the global Lasso problem with the working set strategy gives the following complexity.

Proposition 3 *Under the hypothesis of Theorem 2, the algorithm 1 solves a problem of TE observations with TN features and a number of true activations $O(TN)$, an application of Theorem 3 recovers an approximate complexity of*

$$C_{ws}(TE,TN,TN) = O(T^4(E^2N^2 + EN^3 + N^4)). \tag{13}$$

This result informs us that the naive global working set strategy cannot solve in an efficient manner our problem. For a multi electrode array, the constants E and N are expected to be large. But even in the tetrode case, for which the constants E and N remains small, the length of the signal T might arbitrarily increase depending on the duration of the experiment. The quartic complexity in T and N makes it impossible to apply this algorithm in practical situations.

By contrast, we now state our result regarding the complexity of the sliding window working set.

Theorem 4 *Under the hypothesis of Theorem 2 and Corollary 1, there exists an event Ω_α of probability larger than $1-\alpha$ such that, on Ω_α, the sliding window working set algorithm 3.2 has the following approximate complexity*

- *In the non-subcritical MEA case or in the tetrode case*

$$O(T\log^4(T/\alpha)(E^2N^2 + N^4)) \tag{14}$$

- *In the subcritical MEA case*

$$O(ET\log^4(ET/\alpha)\log(E/\alpha)^8) \tag{15}$$

The above result reveals that our sliding window working set can avoid the high computational quartic costs from Proposition 3 of the naive working set method thanks to the structure of the convolution. In addition our method achieves with high probability a very impressive quasi-linear time complexity $(T\log(T)^4)$ in T for both tetrode and MEA. The complexity quadratic in E and quartic in N is still quartic in the non-subcritical MEA case or in the tetrode case but becomes quasi-linear with $O(E\log(E)^{12})$ in the subcritical MEA case where the spatial overlaps limit the increase in size for the independent sub-problems.

To the best of our knowledge, this is the first proof of complexity with high probability that recovers a quasi-linear complexity in the dimensionality of the data for solving the Lasso.

5 Numerical experiments

In this section we show with numerical simulations the performance of the sliding window working set presented in section 3. First we compare its computation time with other approaches. As all theses approaches require to solve efficiently Lasso problems, we used the parallel implementation of the accelerated proximal gradient FISTA [2] from [24]. Then we show the accuracy of the support of the Lasso estimator for several values of the regularization parameter and noise level, and also when the number of synchronization between neurons grow. All experiments were performed of a simple notebook having 8GB memory and a CPU Intel(R) Core(TM) i7-4810MQ CPU @ 2.80GHz. The Python code from the experiments will be made available on Github upon publication.

5.1 Computational complexity

In order to illustrate the performances of the sliding window working set, we present here a comparison of the computation times of four different approaches detailed below:

- **Global solver**: This is a generic Lasso solver of [24] using the accelerated proximal gradient FISTA to solve the global problem (4) with a pre-computed matrix $\mathbf{H}$. Since the size of the design matrix $\mathbf{H}$ growths as $O(T^2)$, this approach rapidly suffers from the growth of T, both in terms of computation times and memory usage.
- **Working set (naive)**: This is a straightforward implementation of working set algorithm 1 using FISTA as the inner Lasso solver. As previously stated, this approach does not scale properly in memory. Moreover, the computational complexity of the computation of the KKT is $O(ENT^2)$, which also does not scale well with the problem dimensionality.
- **Working set with convolution**: As discussed above using standard solvers with a pre-computed matrix $\mathbf{H}$ is not scalable with the signal length T. In this method we adapt the standard working set algorithm Algorithm 1 to take into account the structure of $\mathbf{H}$. The computational bottleneck comes from the KKT violation (line 3 of Algorithm 1. But in practice those conditions can be computed efficiently using a convolution by the shapes $\mathbf{W}$ instead of expensive matrix products. The residual $\mathbf{y} - \mathbf{H}\hat{\mathbf{a}}$ and the correlation $\mathbf{H}^T(\mathbf{y} - \mathbf{H}\hat{\mathbf{a}})$ (a convolution with reversed shapes) can be computed with complexity $O(ENT\ell)$ hence linear in T. The use of an active set also means that the storage of $\mathbf{H}$ is not necessary anymore since we solve the Lasso on the much smaller $\mathbf{H}_J$. We use the FISTA solver from [24] to solve the sub-problems at each iteration. Finally note that $\mathbf{H}_J$ will be very sparse due to the convolutional model and the sub-problem can be solved on a matrix $\tilde{\mathbf{H}}_J$ of $O(E|J|\ell)$ lines instead of $O(ET)$.
- **Sliding window working set**: This is the method proposed in section 3 and described in Algorithm 3.2 . It focuses only in a small temporal widows and slides the windows when the problem is solved locally. temporal group contained by the current window (similarly to teh previous working set). On the other hand, since the research of the new activation is only carried out on the current window ω and not on the full signal, we computation cost of the KKT condition is greatly reduced from $O(ENT\ell)$ to $O(EN|\omega|\ell)$.

We have simulated our dataset realistically by using the classical model from [16] for the description of the shape of the action potentials, and implemented by [27]. In order to focus our study on the influence of T, we limited ourselves to reasonable values for the number of neurons ($N = 5$) and the number of electrodes ($E = 4$). Note that these small sizes for the parameters would actually correspond to the resolution of the problem on a single spatial group or to the tetrode case.

We present in figure 5.1 the computation times of teh different methods and their 20/80 percentiles for different values of T (each simulation is performed 40 times). It is clear from the Figure that the proposed algorithm is tye most efficient and is actually the only one that can solve problems with $T = 10^6$ temporal samples. The slope of the different methods in the log-log space also show the difference in computational complexity with a slope near 1 for the proposed algorithm that correspond to the $O(T \log T)$ obtained in the theoretical results.

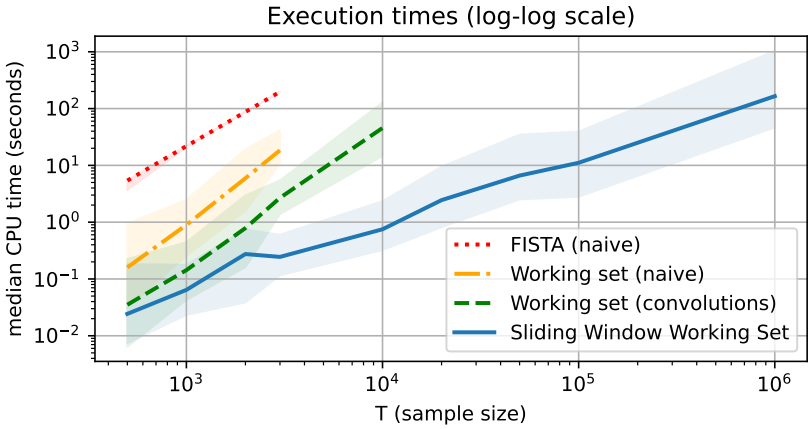

Fig. 4 Comparison of the execution times for three different algorithms, when the size of the signals n growths. We represented the median execution times over 40 simulations as the dotted lines. The bands represent the execution times between the 20 and 80 percentiles.

5.2 Influence of the noise and the regularization parameter

Proper calibration of the regularization parameter λ is crucial for the success of the estimation. We want to visualize the influence of this choice, especially for various noise levels. Using real shapes of action potentials recorded in [3] and that have been already spike sorted by classical algorithms, we simulate signals of size $T = 500$ for different noise levels, with $N = 2$ neurons firing at $50Hz$ and recorded by $E = 4$ electrodes.

In order to measure the performance of the algorithm to recover the true support we consider fist the classical F-measure used for binary classification, which estimates a balance between false positive and false negative rates. More precisely, we define the precision as $PRE = \frac{TP}{TP+FP}$ and the recall as $REC = \frac{TP}{TP+FN}$, where TP, FP and FN are respectively the numbers of true positives, false positives and false negatives. Then the F-measure is computed as $2\frac{PRE.REC}{PRE+REC}$. This measure tends to be pessimistic as it penalizes equally short and long temporal deviations in the recovery. We introduce a softer measure of performance: $CP(\mathbf{x},\mathbf{y}) = 1 - \|K * (\mathbf{x}-\mathbf{y})\|_1 / (\|\mathbf{x}\|_1 + \|\mathbf{y}\|_1)$, where K is a normalized rectangular function. Depending on the size of the support of K, this measure allows us to penalize less small time deviations than large time deviation. Here we took the size of its support equal to 10, which corresponds to an accepted error of the order of 1ms.

We provide on figure 5.2 the performance in F1 score (left) and the proposed convolutional performance (right). We can see that the support recovery is very good in a large interval of values for the large SNR but becomes narrow for low SNR where the support is harder to recover. High SNR recordings constitute an ideal setting for performing spike sorting, therefore the extracellular recording devices should be placed so that this SNR is high enough. Unfortunately this ideal environment may not always be guaranteed, especially in presence of bursting neurons, which action potential amplitudes may decrease down to the noise level [21]. Therefore these experiments show that our method is robust enough to even treat low SNR recordings.

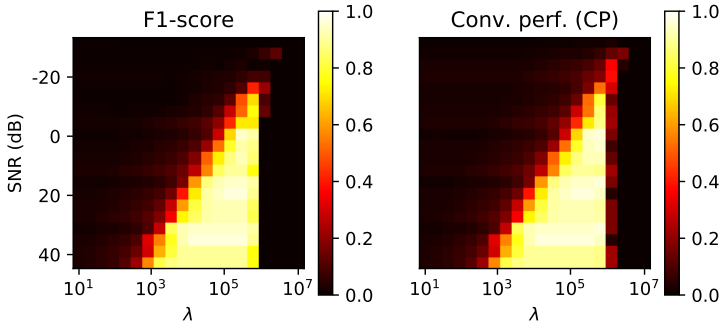

Fig. 5 Influence of λ and the signal-to-noise ratio on the performances of the Lasso estimator. Results are averaged over 5 draws.

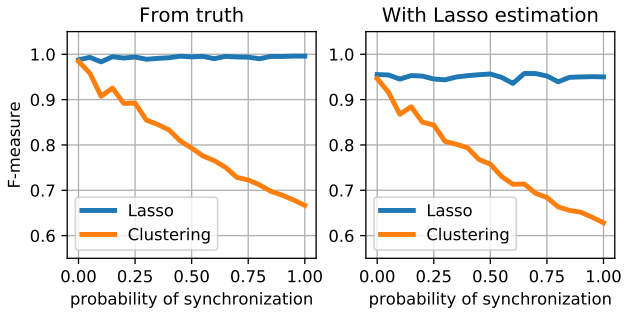

Fig. 6 Comparison of Lasso and clustering performances (F-measure). Results are averaged over 50 draws.

5.3 Comparison with distance-based spike sorting methods

We now compare the performance of the Lasso estimator with distance based methods that rely usually on K-means clustering for spike sorting. In a clustering setting,the spike shapes $\mathbf{W}_n$ are the centroids of the method ([12]). After the activation times have been estimated classical approach select the neuron corresponding to the activation as the one closest to the centroid. We now compare the Lasso and distance-based spike sorting in the presence of synchronization between neurons (simultaneous spike).

Using similar data as in subsection 5.2, we compare in Figure 6 the performance of the methods when the number of synchronizations increases. To this end we use the true time activation (left) where only which neurons are active is unknown and the support recovered with our Lasso estimator The synchronizations have a minor impact on the performances of the lasso estimator, illustrating the robustness of the method due to the fact that the Lasso estimator is additive which mens that it can handle simultaneous activation.

6 Proofs

6.1 Proof of Proposition 1

In the sequel we use the term percolation term "cluster" to refer to spatial overlap. It has been shown in [9] that there exists a critical value $\gamma_c > 0$ which only depends on δ and r_0, such that if $\gamma < \gamma_c$, the probability for a typical cluster to reach a radius r (or a diameter $2r$) is less than $\exp(-c(\gamma, r_0)r)$, with $c(\gamma, r_0) > 0$, depending on γ and r_0.

But the number of neurons, N, that can be sensed by the MEA is the number of neurons which are in a square of area $(\sqrt{E}+2)^2\delta^2$. This is therefore a Poisson variable with mean $(\sqrt{E}+2)^2\delta^2\gamma$.

So by using basic concentration inequalities for Poisson variables (see for instance [29]), we obtain that, for all positive x

$$\mathbb{P}(N > (\sqrt{E}+2)^2\delta^2\gamma + (\sqrt{E}+2)\delta\sqrt{2\gamma x} + x/3) \leq e^{-x}.$$

Let us take $e^{-x} = \alpha/3$ and let us enumerate the points (neurons) of the Poisson process in the whole plan with the first being the ones in the square of size $\sqrt{E}+2$. We say that a cluster is attached to a neuron if the neuron belongs to this cluster.

We use a union bound to control the size of each cluster attached to each neuron n such that $n \leq Q$, with Q the largest integer such that

$$Q \leq (\sqrt{E}+2)^2\delta^2\gamma + (\sqrt{E}+2)\delta\sqrt{2\gamma\log(3/\alpha)} + \log(3/\alpha)/3.$$

So we get that the probability to have one of these clusters of diameter larger than r is smaller than

$$\left[(\sqrt{E}+2)^2\delta^2\gamma + (\sqrt{E}+2)\delta\sqrt{2\gamma\log(3/\alpha)} + \log(3/\alpha)/3\right] e^{-c(\gamma,r_0)r}.$$

We take r such that this bound is less than $\alpha/2$, that is

$$r = \kappa' \log(E/\alpha),$$

with

$$\kappa' = \frac{1}{c(\gamma, r_0)}\left(1 + \frac{\log(2C)}{\log(\tau)}\right),$$

and

$$C = 4\delta^2\gamma + 2\delta\sqrt{\frac{2\gamma}{\tau}} + \frac{1}{3\tau}.$$

which depend only on γ, δ, τ and r_0.

Finally, we can also control the number of neurons that belongs to each of the Q balls that are used to encompass the Q clusters of the first Q points.

With similar arguments as before on the control of Poisson variables and union bound, we can upper bound by $\alpha/3$, the probability that there is one of the Q balls with more than $\kappa'' \log(E/\alpha)^2$ neurons in it.

Therefore if we define $\Omega_{\alpha,s}$ as the event where (i) the total number of neurons is controlled (ii) the range of the Q first clusters is controlled (iii) the number of neurons per ball for the first Q balls is controlled, we obtain the desired result.

6.2 Proof of Proposition 2

If T_i is the ith index where $A_{T_i} = 1$, then for all i, $\tau_i = T_i - T_{i-1}$ are independent Geometric variable on $\{1,2,...\}$ with parameter p, with the convention $T_0 = 0$.

We define $X_0 = 1$ and

$$X_1 = \min\{j > 1, \tau_j > \eta\} \quad \text{and} \quad X_i = \min\{j > X_{i-1}, \tau_j > \eta\}.$$

Similarly, for $i \geq 1$, $\delta_i = X_i - X_{i-1}$ are independent geometric variables of parameter $(1-p)^\eta$.

Therefore the ith overlap, which happens between $T_{X_{i-1}}$ and T_{X_i} has a length $D_i = T_{X_i} - T_{X_{i-1}} + 1$.

So for all integer k,

$$\mathbb{P}(D_i > k\eta + 1) \leq \mathbb{P}(\delta_i > k) \leq (1-(1-p)^\eta)^k \leq (1-0.5^\eta)^k.$$

We have at most R overlaps with R the largest integer such that $R \leq T/\eta$.

By a union bound we can control all the R first overlaps and the probability to have at least one overlap larger than $k\eta + 1$ is controlled by

$$T/\eta(1-0.5^\eta)^k.$$

Forcing this last term to be α gives the value of k and concludes the proof in the non subcritical case. The complementary event is $\Omega_{\alpha,A}$.

In the subcritical case, using the notation of the proof of Proposition 1, we need to control it for all the first Q clusters, which lead us to

$$QT_{max}/\eta(1-(1-p)^\eta)^k,$$

hence the other choice of k. The complementary event is $\Omega_{\alpha,A}$. We then use here $\Omega_{\alpha,A,s} = \Omega_{\alpha/2,s} \cap \Omega_{\alpha/2,A}$

6.3 Proof of Lemma 2

For a fix (n,t), since the noise is Gaussian, the random variable

$$\mathbf{h}_{n,t}^\top \xi = \sum_{e=1}^{E} (\xi_t^e, ..., \xi_{t+\ell}^e) \mathbf{w}_{n,e}^{\to t}$$

is also a Gaussian variable with variance bounded by $C\sigma^2$. Thus the event

$$\left| \mathbf{h}_{n,t}^\top \xi \right| \geq z$$

is of probability less than $2e^{-z^2/(2\sigma^2 \bar{c})}$.

This argument is valid for any (n,t). Therefore, by an union bound argument, we get, for a fixed $\alpha \in (0,1)$, with the choice $z = z_\alpha$, for all n in $1,...,N$ and all t in $1,...,T$,

$$\left| \mathbf{h}_{n,t}^\top \xi \right| \leq z_\alpha,$$

with probability larger than $1-\alpha$. □

6.4 Proof of Theorem 2

Note that if Algorithm 3.2 needs to work with sparse matrices for computational reasons, mathematically speaking, we can as well work with the corresponding inflated matrices and this will not change the value of the solution, but just the space in which it is represented. Therefore, for the sake of convenience when we investigate the statistical properties of the method, we define $\mathbf{a}_J$ as the vector obtained by setting to 0 all the coordinates from $\mathbf{a}$ with their index not in J. We define similarly $\mathbf{H}_J$ as the matrix obtained by replacing all the columns from $\mathbf{H}$ with their index not in J by the zero vector. Therefore in the sequel vectors $\mathbf{a}$ and matrix $\mathbf{H}$ have always the same dimensions.

We work on the event Ω_α, which as stated in the remarks below Lemma 2 is of probability less than $1-\alpha$. We fix a spatial overlap if we are in the subcritical MEA case or we work with the whole set of sensors in the other cases. In every cases, the number of neurons in the restricted problem is bounded by $\mathscr{N}$ thanks to Proposition 1. We also fix a given window ω.

We now solve the Lasso problem on the temporal window ω on the possibly restricted set of neurons:

$$\hat{\mathbf{a}}_\omega = \arg \min_{\mathbf{a}/Supp(\mathbf{a})\subset\omega} \|\mathbf{y}-\mathbf{H}_\omega\,\mathbf{a}\|_2^2 + 2\lambda\,\|\mathbf{a}\|_1,$$

where we have used a slight abuse of language: "$Supp(\mathbf{a})\subset\omega$" means that the temporal indices t of $\mathbf{a}=(\mathbf{a}_{n,t})_{n,t}$ have to be in the temporal window ω. Let us define the solution of the Lasso optimization problem on $S^*\cap\omega$ where we recall that S^* is the true support of $\mathbf{a}^*$:

$$\hat{\mathbf{a}}_{S^*\cap\omega} = \arg \min_{\mathbf{a}/Supp(\mathbf{a})\subset S^*\cap\omega} \|\mathbf{y}-\mathbf{H}_{S^*\cap\omega}\,\mathbf{a}\|_2^2 + 2\lambda\,\|\mathbf{a}\|_1,$$

where we have also made a slight abuse of language: $S^*\cap\omega=\{(n,t)\in S^*/t\in\omega\}$. Note that $\hat{\mathbf{a}}_{S^*\cap\omega}$ and $\hat{\mathbf{a}}_\omega$ are both of dimension NT, with (temporal) support inside ω.

Our goal is to prove that $\hat{\mathbf{a}}_\omega$ is null outside of S^*. To this end, we first prove that the vector $\hat{\mathbf{a}}_{S^*\cap\omega}$ satisfies the KKT conditions on the temporal window ω.

Noting that $\mathbf{H}_\omega\,\hat{\mathbf{a}}_{S^*\cap\omega}=\mathbf{H}_{S^*\cap\omega}\,\hat{\mathbf{a}}_{S^*\cap\omega}$, we see that $\hat{\mathbf{a}}_{S^*\cap\omega}$ already satisfies the KKT condition for any $(n,t)\in S^*\cap\omega$. We only have to check the KKT conditions for $(n,t)\in\omega\setminus S^*$ (with the same kind of language abuse as before).

To this end, we first need to prove a bound on $\|\mathbf{a}^*-\hat{\mathbf{a}}_{S^*\cap\omega}\|_{\infty,\omega}$ where $\|\mathbf{a}\|_{\infty,J}:=\max_{n,t\in J}|a_{n,t}|$.

By definition, the Lasso solution $\hat{\mathbf{a}}_{S^*\cap\omega}$ satisfies the following necessary condition for all $(n,t)\in S^*\cap\omega$:

$$|\mathbf{h}_{n,t}^\top(\mathbf{y}-\mathbf{H}_{S^*\cap\omega}\,\hat{\mathbf{a}}_{S^*\cap\omega})|\le\lambda.$$

We deduce that, for all $(n,t)\in S^*\cap\omega$,

$$|\mathbf{h}_{n,t}^\top\mathbf{H}_{S^*\cap\omega}(\mathbf{a}^*-\hat{\mathbf{a}}_{S^*\cap\omega})|\le\lambda+|\mathbf{h}_{n,t}^\top\xi|+|\mathbf{h}_{n,t}^\top\mathbf{H}_{S^*\cap\omega^c}\,\mathbf{a}^*|,$$

with ω^c being the complementary in $[\![1,T]\!]$ of the temporal window ω.

In view of Lemma 1, we have for all $(n,t)\in S^*\cap\omega$ that

$$\mathbf{h}_{n,t}^\top\mathbf{H}_{S^*\cap\omega^c}\,\mathbf{a}^*=\mathbf{h}_{n,t}^\top\mathbf{H}_{S^*\cap\partial\omega}\,\mathbf{a}^*.$$

In addition, Assumption 2 guarantees that $|G_{(n,t),(n',t')}| \leq \varepsilon$ for all $n \neq n'$ and $|G_{(n,t),(n,t')}| < \rho$ for any $t \neq t'$. Also, given the refractory period, there can be at most only 1 activation on any interval of length l. Thus we get, for all $(n,t) \in S^* \cap \omega$,

$$|\mathbf{h}_{n,t}^\top \mathbf{H}_{S^* \cap \omega^c} \mathbf{a}^*| \leq 2(\rho + \varepsilon \mathscr{N}) \|\mathbf{a}^*\|_{\infty, \partial\omega}. \tag{16}$$

Next, we have for all $(n,t) \in S^* \cap \omega$

$$\begin{aligned}\mathbf{h}_{n,t}^\top \mathbf{H}_{S^* \cap \omega} (\mathbf{a}^* - \hat{\mathbf{a}}_{S^* \cap \omega}) = &\sum_{t' : (n,t') \in S^* \cap \omega} \mathbf{G}_{(n,t),(n,t')} (\mathbf{a}^*_{n,t'} - (\hat{\mathbf{a}}_{S^* \cap \omega})_{n,t'}) \\ &+ \sum_{n' \neq n} \sum_{(n',t') \in S^* \cap \omega} \mathbf{G}_{(n,t),(n',t')} (\mathbf{a}^*_{n',t'} - (\hat{\mathbf{a}}_{S^* \cap \omega})_{n',t'}).\end{aligned}$$

Given the block-band structure of the Gram matrix G (see Lemma 1), the first sum in the above display contains exactly 1 nonzero term corresponding to $t' = t$:

$$\sum_{t' : (n,t') \in S^* \cap \omega} \mathbf{G}_{(n,t),(n,t')} (\mathbf{a}^*_{n,t'} - (\hat{\mathbf{a}}_{S^* \cap \omega})_{n,t'}) := \mathbf{G}_{(n,t),(n,t)} (\mathbf{a}^*_{n,t} - (\hat{\mathbf{a}}_{S^* \cap \omega})_{n,t}).$$

Regarding the second sum, we also have in view of Lemma 1:

$$\sum_{n' \neq n} \sum_{(n',t') \in S^* \cap \omega} \mathbf{G}_{(n,t),(n',t')} (\mathbf{a}^*_{n',t'} - (\hat{\mathbf{a}}_{S^* \cap \omega})_{n',t'}) = \sum_{n' \neq n} \sum_{(n',t') \in S^* \cap \omega, |t'-t| \leq l} \mathbf{G}_{(n,t),(n',t')} (\mathbf{a}^*_{n',t'} - (\hat{\mathbf{a}}_{S^* \cap \omega})_{n',t'})$$

Set $\Delta_{n,t} := \mathbf{a}^\star_{n,t} - (\hat{\mathbf{a}}_{S^* \cap \omega})_{n,t}$. Combining the last four displays, we get for all $(n,t) \in S^* \cap \omega$,

$$\mathbf{G}_{(n,t),(n,t)} |\Delta_{n,t}| \leq \sum_{n' \neq n} \sum_{(n',t') \in S^* \cap \omega, |t'-t| \leq l} |\mathbf{G}_{(n,t),(n',t')}| \, |\Delta_{n',t'}| + |\mathbf{h}_{n,t}^\top \xi| + \lambda + 2(\rho + \varepsilon \mathscr{N}) \|\mathbf{a}^*\|_{\infty, \partial\omega}.$$

Assumption 2 guarantees that $|\mathbf{G}_{(n,t),(n',t')}| \leq \varepsilon$ for all $n \neq n'$ and $G_{(n,t),(n,t)} > \underline{c}$. Also for a given n', because of the refractory period, there is at most 2 activations for this particular neuron at distance ℓ of t. Thus we get, for all $(n,t) \in S^* \cap \omega$,

$$\underline{c} \, |\Delta_{n,t}| \leq 2\varepsilon \mathscr{N} \, \|\Delta\|_{\infty,\omega} + \max_{(n,t) \in S^* \cap \omega} |\mathbf{h}_{n,t}^\top \xi| + \lambda + 2(\rho + \varepsilon \mathscr{N}) \|\mathbf{a}^* \mathbf{a}^*\|_{\infty, \partial\omega},$$

and consequently, since we are on Ω_α,

$$(c - 2\varepsilon \mathscr{N}) \|\Delta\|_{\infty,\omega} \leq z_\alpha + \lambda + 2(\rho + \varepsilon \mathscr{N}) \|\mathbf{a}^*\|_{\infty, \partial\omega}.$$

Combining this result with Lemma 2, we get

$$\|\mathbf{a}^* - \hat{\mathbf{a}}_{S^* \cap \omega}\|_{\infty,\omega} \leq \frac{z_\alpha + \lambda + 2(\rho + \varepsilon \mathscr{N}) \|a^*\|_{\infty, \partial\omega}}{\underline{c} - 2\varepsilon \mathscr{N}}. \tag{17}$$

We now check the KKT conditions for $\hat{\mathbf{a}}_{S^* \cap \omega}$ on $\omega \setminus S^*$. For any $(n,t) \in \omega \setminus S^*$, we have

$$\begin{aligned}\mathbf{h}_{n,t}^\top (\mathbf{y} - \mathbf{H}_\omega \hat{\mathbf{a}}_{S^* \cap \omega}) &= \mathbf{h}_{n,t}^\top (\mathbf{H}\mathbf{a}^* + \xi - \mathbf{H}_\omega \hat{\mathbf{a}}_{S^* \cap \omega}) \\ &= \mathbf{h}_{n,t}^\top \mathbf{H}_\omega (\mathbf{a}^* - \hat{\mathbf{a}}_{S^* \cap \omega}) + \mathbf{h}_{n,t}^\top \mathbf{H}_{S^* \cap \partial\omega} \mathbf{a}^* + \mathbf{h}_{n,t}^\top \xi.\end{aligned}$$

In view of (16) and (17), we have on the event $\Omega_{\alpha,A,s} \cap \Omega_{\alpha,\xi}$, for all $(n,t) \in \omega \setminus S^*$,

$$|\mathbf{h}_{n,t}^\top (\mathbf{y} - \mathbf{H}_\omega \hat{\mathbf{a}}_{S^* \cap \omega})| \leq z_\alpha + 2(\rho + \varepsilon \mathscr{N}) \left(\|\mathbf{a}^*\|_{\infty, \partial\omega} + \frac{z_\alpha + \lambda + 2(\rho + \varepsilon \mathscr{N}) \|\mathbf{a}^*\|_{\infty, \partial\omega}}{\underline{c} - 2\varepsilon \mathscr{N}} \right). \tag{18}$$

We need the following condition to satisfy the strict KKT conditions:

$$\lambda > z_\alpha + 2(\rho + \varepsilon_{\mathscr{N}}) \left(\|\mathbf{a}^*\|_{\infty,\partial\omega} + \frac{z_\alpha + \lambda + 2(\rho+\varepsilon_{\mathscr{N}})\|\mathbf{a}^*\|_{\infty,\partial\omega}}{\underline{c} - 2\varepsilon_{\mathscr{N}}} \right), \tag{19}$$

or equivalently

$$\lambda > \frac{1 + \frac{2(\rho+2\varepsilon_{\mathscr{N}})}{\underline{c}-2\varepsilon_{\mathscr{N}}}}{1 - \frac{2(\rho+2\varepsilon_{\mathscr{N}})}{\underline{c}-2\varepsilon_{\mathscr{N}}}} \left(z_\alpha + 2(\rho+\varepsilon_{\mathscr{N}})\|\mathbf{a}^*\|_{\infty,\partial\omega} \right).$$

This means that $\hat{\mathbf{a}}_{S^*\cap\omega}$ satisfies the strict KKT conditions in (20) below on the temporal window ω. Thus we proved, that $\hat{\mathbf{a}}_{S^*\cap\omega}$ is a solution of the Lasso minimization problem on the temporal window ω, on the event Ω_α.

The following property is an immediate consequence of the convexity of the Lasso objective function.

Lemma 3 *Consider* $Crit(\mathbf{a}) = \|\mathbf{y} - \mathbf{H}_\omega \mathbf{a}\|_2^2 + 2\lambda\|\mathbf{a}\|_1$. *Let* $\tilde{\mathbf{a}}_\omega$ *be a minimizer of* $Crit(\mathbf{a})$, *hence satisfying the KKT conditions:*

$$\begin{cases} \mathbf{h}_{n,t}^\top(\mathbf{y} - \mathbf{H}_\omega\tilde{\mathbf{a}}_\omega) = \lambda\ sign((\tilde{a}_\omega)_{n,t}) & ,\ if\ (\tilde{\mathbf{a}}_\omega)_{n,t} \neq 0, \\ |\mathbf{h}_{n,t}^\top(\mathbf{y} - \mathbf{H}_\omega\tilde{\mathbf{a}}_\omega)| \leq \lambda & ,if\ (\tilde{\mathbf{a}}_\omega)_{n,t} = 0. \end{cases} \tag{20}$$

Let

$$\tilde{S} = \{(n,t)/|\mathbf{h}_{n,t}^\top(\mathbf{y} - \mathbf{H}_\omega\tilde{\mathbf{a}}_\omega)| = \lambda\}.$$

Then for any other minimizer $\hat{\mathbf{a}}_\omega$ *of* $Crit(\mathbf{a})$, *we have*

$$Supp(\hat{\mathbf{a}}_\omega) \subset \tilde{S}.$$

Note that $\tilde{S}$ might be larger than the true support of $\tilde{\mathbf{a}}_\omega$ because there might be coordinates (n,t) such that $(\tilde{\mathbf{a}}_\omega)_{n,t} = 0$ and for which $|\mathbf{h}_{n,t}^\top(\mathbf{y} - \mathbf{H}_\omega\tilde{\mathbf{a}}_\omega)| = \lambda$.

Proof (Lemma 3)

In view of (20), we have for any $(n,t) \in \omega$,

$$\mathbf{h}_{n,t}^\top(\mathbf{y} - \mathbf{H}_\omega\tilde{\mathbf{a}}_\omega) = \lambda\, s_{n,t},$$

where $(s_{n,t})_{(n,t)\in\omega}$ is such that,

$$\begin{cases} |s_{n,t}| \leq 1 & ,\text{ in all cases} \\ s_{n,t} = \text{sign}((\tilde{a}_\omega)_{n,t}) & ,\text{ if } (\tilde{\mathbf{a}}_\omega)_{n,t} \neq 0, \\ |s_{n,t}| < 1 & ,\text{if } (n,t) \in \tilde{S}^c. \end{cases}$$

Therefore, we have

$$\begin{aligned} Crit(\tilde{\mathbf{a}}_\omega + \mathbf{a}) - Crit(\tilde{\mathbf{a}}_\omega) &= \|\mathbf{y} - \mathbf{H}_\omega(\tilde{\mathbf{a}}_\omega + \mathbf{a})\|_2^2 - \|\mathbf{y} - \mathbf{H}_\omega\tilde{\mathbf{a}}_\omega\|_2^2 + 2\lambda(\|\tilde{\mathbf{a}}_\omega + \mathbf{a}\|_1 - \|\tilde{\mathbf{a}}_\omega\|_1) \\ &= \|\mathbf{H}_\omega\mathbf{a}\|_2^2 - 2 < \mathbf{y} - \mathbf{H}_\omega\tilde{\mathbf{a}}_\omega, \mathbf{H}_\omega\mathbf{a} > + 2\lambda(\|\tilde{\mathbf{a}}_\omega + \mathbf{a}\|_1 - \|\tilde{\mathbf{a}}_\omega\|_1) \\ &= \|\mathbf{H}_\omega\mathbf{a}\|_2^2 + 2\lambda\Big(\sum_{(n,t)\in\omega} |(\tilde{\mathbf{a}}_\omega)_{n,t} + \mathbf{a}_{n,t}| - |(\tilde{\mathbf{a}}_\omega)_{n,t}| - \mathbf{a}_{n,t}s_{n,t}\Big). \end{aligned}$$

Set $\mathbf{a} = \hat{\mathbf{a}}_\omega - \tilde{\mathbf{a}}_\omega$. By convexity of the l_1-norm, we have for all $(n,t) \in \omega$,

$$|(\tilde{\mathbf{a}}_\omega)_{n,t} + \mathbf{a}_{n,t}| - |(\tilde{\mathbf{a}}_\omega)_{n,t}| - s_{n,t}\mathbf{a}_{n,t} \geq 0.$$

Thus,

$$\sum_{(n,t)} (|(\tilde{\mathbf{a}}_\omega)_{n,t} + \mathbf{a}_{n,t}| - |(\tilde{\mathbf{a}}_\omega)_{n,t}| - s_{n,t}\mathbf{a}_{n,t}) \geq 0$$

Assume that $Supp(\hat{\mathbf{a}}_\omega)$ is not included in $\tilde{S}$. Then there exists $(n_0,t_0) \in Supp(\hat{\mathbf{a}}_\omega) \cap \tilde{S}^c$ such that $\mathbf{a}_{n_0,t_0} = (\hat{\mathbf{a}}_\omega)_{n_0,t_0} \neq 0$ and $|s_{n_0,t_0}| < 1$. Consequently, we get

$$|\mathbf{a}_{n_0,t_0}| - s_{n_0,t_0}\mathbf{a}_{n_0,t_0} > 0.$$

Since both $\hat{\mathbf{a}}_\omega$ and $\tilde{\mathbf{a}}_\omega$ are Lasso solution, we have

$$0 = Crit(\hat{\mathbf{a}}_\omega) - Crit(\tilde{\mathbf{a}}_\omega)) \geq 2\lambda \sum_{(n,t)} |(\tilde{\mathbf{a}}_\omega)_{n,t} + \mathbf{a}_{n,t}| - |(\tilde{\mathbf{a}}_\omega)_{n,t}| - \mathbf{a}_{n,t}s_{n,t}) \geq 2\lambda(|\mathbf{a}_{n_0,t_0}| - s_{n_0,t_0}\mathbf{a}_{n_0,t_0}) > 0.$$

We obtain a contradiction. This means that

$$Supp(\hat{\mathbf{a}}_\omega) \subset \tilde{S}.$$

Proof of Theorem 2 (continued) By Lemma 3 applied to $\tilde{\mathbf{a}}_\omega = \hat{\mathbf{a}}_{S^* \cap \omega}$, we get the first inclusion.

We assume in addition (11). Then, in view of (17), we have on the event on Ω_α

$$Supp(\hat{\mathbf{a}}_{S^* \cap \omega}) = S^*.$$

6.5 Proof of Corollary 1

With the choices provided in Algorithm 3.2, we start with the window $\omega = [\![1, 4\ell]\!]$. So ω is included in the first temporal overlap of size $\eta = 4\ell$. Next the algorithm will compute and expand this window ω to the first time that no activation of $\hat{\mathbf{a}}_\omega$ is found in the last 2ℓ coordinates. Thanks to Theorem 2, this means that this is also the first time that $\mathbf{a}^*$ has no activation in a segment of length 2ℓ. This is not necessarily the first hole of size 2ℓ, because the algorithm only looks at $k\ell$ for some integer k, but definitely, the algorithm will stop and start a new window at the first "hole" of size 4ℓ.

In this sense, the first window will not be expanded not after the first hole of size η. Therefore its length is controlled by the control of the temporal overlap (see Proposition 2). The next step of the algorithm (Steps 7,8,9) cannot happen on the same event, indeed it would mean that the lasso estimator finds something at the beginning of the new window, whereas the estimator on the previous window (and therefore the truth) have no activation there. This is not possible since on every window, the Lasso estimator has the same support as the truth.

Therefore we start a new window without merging with the one before and expand it again. The same arguments as before will apply recursively to prove our statement.

This conclude the proof of the theorem.

6.6 Proof of complexity in Theorem 4

In this proof we assume that the problem respects the hypothesis of Theorem 2 and Corollary 1, there exists an event Ω_α of probability larger than $1-\alpha$. The following of the proof suppose that we are on Ω_α.

We first investigate the non-subcritical MEA case or in the tetrode case. In this case we need to solve $O(T)$ temporal independent problems whose window size is bounded by $O(\log(T/\alpha))$ (as proven in Corollary 1). Those independent problems, using notations from Theorem 3, have dimensionalities of $O(E\log(T/\alpha))$ observations, $O(N\log(T/\alpha))$ features and $O(N\log(T/\alpha))$ selected features. This means that the complexity is

$$\begin{aligned}\mathscr{C}(Alg.3.2, Tetrode) &= O(TC_{ws}(E\log(T/\alpha), N\log(T/\alpha), N\log(T/\alpha))) \\ &= O(T(E\log(T/\alpha)^2(N\log(T/\alpha))^2 + 2(N\log(T/\alpha)^4))) \\ &= O(T\log(T/\alpha)^4(E^2N^2+N^4))\end{aligned}$$

which proves the result in equation (14).

In the subcritical MEA case, the problem can be solved with $O(NE)$ independent problems (using both spatial and temporal overlaps). But those problems are of much smaller size. Indeed Corollary 1 tells us that the size of the temporal window is bounded in this case by $O(\log(ET/\alpha))$ and Proposition 1 tells us that the size of the spatial overlaps N_c and E_c are bounded in this case by $O(\log(E/\alpha)^2)$. This means that the problems we need to solve have maximum dimensionality of $O(\log(E/\alpha)^2\log(ET/\alpha))$ observations, $O(\log(E/\alpha)^2\log(ET/\alpha))$ features and again $O(\log(E/\alpha)^2\log(ET/\alpha))$ selected features. This means that the complexity of solving the whole problem is

$$\begin{aligned}\mathscr{C}(Alg.3.2, MEA) &= O(ETC_{ws}(\log(E/\alpha)^2\log(ET/\alpha), \log(E/\alpha)^2\log(ET/\alpha), \log(E/\alpha)^2\log(ET/\alpha))) \\ &= O(ET\log(ET/\alpha)^4\log(E/\alpha)^8)\end{aligned}$$

This proves result in equation (15) concludes the proof of the theorem.

7 Conclusion

In this paper we propose a novel sliding window working set algorithm that can solve exactly the large scale Lasso in spike sorting in an efficient way by exploiting the convolutional structure of the problem. Under some realistic assumptions, we prove that the complexity of the proposed algorithm is quasi-linear with high probability with respect to the temporal dimensionality of the signal. Under some conditions on the neurons firing rate, the complexity is also quasi-linear with the number of electrodes which is a very important aspect for MEA that can have a large number of electrodes. We perform numerical experiments on a realistic signals and recover the theoretical computational complexity.

We believe that this result opens the door for large scale spike sorting on the recently available MEA sensor but also in other potential application which rely on a sparse estimation with a convolutional model. Future work will investigate the simultaneous estimation of the spike shape and activation and the online update of those shapes along time.

References

1. Albert, M., Bouret, Y., Fromont, M., Reynaud-Bouret, P.: Surrogate data methods based on a shuffling of the trials for synchrony detection : the centering issue. Neural Computation **28**(11), 2352–2392 (2016)
2. Beck, A., Teboulle, M.: A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM journal on imaging sciences **2**(1), 183–202 (2009)
3. Bethus, I., Poucet, B., Sargolini, F.: Neural correlates of goal-directed spatial navigation in the rat dorsal striatum. In: Forum of European Neuroscience. Barcelona, Spain (2012)
4. Bickel, P., Ritov, Y., Tsybakov, A.: Simultaneous analysis of lasso and dantzig selector. Annals of Statistics **37**(4), 1705–1732 (2009)
5. Biffi, E., Regalia, G., Menegon, A., Ferrigno, G., Pedrocchi, A.: The influence of neuronal density and maturation on network activity of hippocampal cell cultures: a methodological study. Plos one **8**(12), e83899 (2013)
6. Boisbunon, A., Flamary, R., Rakotomamonjy, A., Giros, A., Zerubia, J.: Large Scale Sparse Optimization for Object Detection in High Resolution Images. In: MLSP - 24th IEEE Workshop on Machine Learning for Signal Processing. Reims, France (2014). URL `https://hal.inria.fr/hal-01066235`
7. Bunea, F.: Honest variable selection in linear and logistic regression models via l1 and l1 + l2 penalization. Electron. J. Statist. **2**, 1153–1194 (2008). DOI 10.1214/08-EJS287. URL `https://doi.org/10.1214/08-EJS287`
8. Dayan, P., Abbott, L.F.: Theoretical neuroscience: computational and mathematical modeling of neural systems (2001)
9. Duminil-Copin, H., Raoufi, A., Tassion, V.: Subcritical phase of d-dimensional poisson-boolean percolation and its vacant set (2018)
10. Einevoll, G.T., Franke, F., Hagen, E., Pouzat, C., Harris, K.D.: Towards reliable spike-train recordings from thousands of neurons with multielectrodes. Current opinion in neurobiology **22**(1), 11–17 (2012)
11. Ekanadham, C., Tranchina, D., Simoncelli, E.P.: Recovery of sparse translation-invariant signals with continuous basis pursuit. IEEE transactions on signal processing **59**(10), 4735–4744 (2011)
12. Ekanadham, C., Tranchina, D., Simoncelli, E.P.: A unified framework and method for automatic neural spike identification. Journal of neuroscience methods **222**, 47–55 (2014)
13. Eytan, D., Marom, S.: Dynamics and effective topology underlying synchronization in networks of cortical neurons. Journal of Neuroscience **26**(33), 8465–8476 (2006). DOI 10.1523/JNEUROSCI.1627-06.2006. URL `http://www.jneurosci.org/content/26/33/8465`
14. Grosse, R., Raina, R., Kwong, H., Ng, A.Y.: Shift-invariance sparse coding for audio classification. arXiv preprint arXiv:1206.5241 (2012)
15. Harris, K.D., Henze, D.A., Csicsvari, J., Hirase, H., Buzsaki, G.: Accuracy of tetrode spike separation as determined by simultaneous intracellular and extracellular measurements. Journal of neurophysiology **84**(1), 401–414 (2000)
16. Hodgkin, A.L., Huxley, A.F.: A quantitative description of membrane current and its application to conduction and excitation in nerve. The Journal of Physiology **117**(4), 500–544 (1952). DOI 10.1113/jphysiol.1952.sp004764. URL `https://physoc.onlinelibrary.wiley.com/doi/abs/10.1113/jphysiol.1952.sp004764`
17. Jas, M., Dupré La Tour, T., Şimşekli, U., Gramfort, A.: Learning the Morphology of Brain Signals Using Alpha-Stable Convolutional Sparse Coding. In: Advances in neural information processing systems. Long Beach, United States (2017). URL `https://hal.archives-ouvertes.fr/hal-01590988`
18. La Tour, T.D., Moreau, T., Jas, M., Gramfort, A.: Multivariate convolutional sparse coding for electromagnetic brain signals. In: Advances in Neural Information Processing Systems, pp. 3292–3302 (2018)
19. Lambert, R., Tuleau-Malot, C., Bessaih, T., Rivoirard, V., Bouret, Y., Leresche, N., Reynaud-Bouret, P.: Reconstructing the functional connectivity of multiple spike trains using hawkes models. Journal of Neuroscience Methods **297**, 9–21 (2018)
20. Lee, H., Battle, A., Raina, R., Ng, A.Y.: Efficient sparse coding algorithms. In: Advances in neural information processing systems, pp. 801–808 (2007)
21. Lewicki, M.S.: A review of methods for spike sorting: the detection and classification of neural action potentials. Network: Computation in Neural Systems **9**(4), R53–R78 (1998)
22. Loth, M.: Active set algorithms for the lasso.(algorithmes d'ensemble actif pour le lasso). Ph.D. thesis, Lille University of Science and Technology, France (2011)
23. Lounici, K.: Sup-norm convergence rate and sign concentration property of lasso and dantzig estimators. Electron. J. Statist. **2**, 90–102 (2008). DOI 10.1214/08-EJS177. URL `https://doi.org/10.1214/08-EJS177`
24. Mairal, J., Bach, F., Ponce, J., Sapiro, G., Jenatton, R., Obozinski, G.: Spams: A sparse modeling software. URL http://spams-devel. gforge. inria. fr/downloads. html (2014)

25. Moreau, T., Oudre, L., Vayatis, N.: Dicod: Distributed convolutional coordinate descent for convolutional sparse coding. In: International Conference on Machine Learning, pp. 3626–3634. PMLR (2018)
26. Muthmann, J.O., Amin, H., Sernagor, E., Maccione, A., Panas, D., Berdondini, L., Bhalla, U.S., Hennig, M.H.: Spike detection for large neural populations using high density multielectrode arrays. Frontiers in neuroinformatics **9**, 28 (2015)
27. Pouzat, C.: Origin of the (high frequency) extra-cellular signal. `http://christophe-pouzat.github.io/LASCON2016/OriginOfTheHighFrequencyExtraCellularSignal.html` (2016)
28. Pouzat, C., Detorakis, G.: Spysort: Neuronal spike sorting with python. CoRR **abs/1412.6383** (2014). URL `http://arxiv.org/abs/1412.6383`
29. Reynaud-Bouret, P.: Adaptive estimation of the intensity of inhomogeneous poisson process via concentration inequalities. Probab. Theory related Fields **126**(1), 103–153 (2003)
30. Reynaud-Bouret, P., Rivoirard, V., Grammont, F., Tuleau-Malot, C.: Goodness-of-fit tests and nonparametric adaptive estimation for spike train analysis. The Journal of Mathematical Neuroscience **4**(1), 1–41 (2014)
31. Smaragdis, P.: Convolutive speech bases and their application to supervised speech separation. IEEE Transactions on Audio, Speech, and Language Processing **15**(1), 1–12 (2007). DOI 10.1109/TASL.2006.876726
32. Szafranski, M., Grandvalet, Y., Morizet-Mahoudeaux, P.: Hierarchical penalization. In: Advances in neural information processing systems, pp. 1457–1464 (2008)
33. Taylor, H.L., Banks, S.C., McCoy, J.F.: Deconvolution with the l1 norm. Geophysics **44**(1), 39–52 (1979)
34. Tibshirani, R.: Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society: Series B (Methodological) **58**(1), 267–288 (1996)
35. Tuleau-Malot, C., Rouis, A., Grammont, F., Reynaud-Bouret, P.: Multiple tests based on a gaussian approximation of the unitary events method with delayed coincidence count. Neural computation **26**(7), 1408–1454 (2014)
36. Tuleau-Malot, C., Rouis, A., Grammont, F., Reynaud-Bouret, P.: Multiple tests based on a gaussian approximation of the unitary events method with delayed coincidence count. Neural computation **26**(7), 1408–1454 (2014)
37. Wood, F., Black, M.J., Vargas-Irwin, C., Fellows, M., Donoghue, J.P.: On the variability of manual spike sorting. IEEE Transactions on Biomedical Engineering **51**(6), 912–918 (2004)