Spike sorting for massive neurophysiological datasets : sliding window working set strategy for the estimation of convolutional models in high dimension

Laurent Dragoni

▶ **To cite this version:**

Laurent Dragoni. Spike sorting for massive neurophysiological datasets : sliding window working set strategy for the estimation of convolutional models in high dimension. Optimization and Control [math.OC]. Université Côte d'Azur, 2022. English. NNT : 2022COAZ4016 . tel-03774851

HAL Id: tel-03774851

https://theses.hal.science/tel-03774851v1

Submitted on 12 Sep 2022

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

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

UNIVERSITÉ CÔTE D'AZUR

ÉCOLE DOCTORALE
SCIENCES FONDAMENTALES ET APPLIQUÉES

THÈSE DE DOCTORAT

Tri de potentiels d'action sur des données neurophysiologiques massives

Stratégie d'ensemble actif par fenêtre glissante pour l'estimation de modèles convolutionnels en grande dimension

Laurent DRAGONI

Laboratoire Jean-Alexandre Dieudonné (LJAD)

Présentée en vue de l'obtention du grade de docteur en mathématiques **d'**Université Côte d'Azur

Dirigée par : Karim Lounici, Rémi Flamary, Patricia Reynaud-Bouret
Soutenue le : 25 avril 2022

Devant le jury, composé de :
Thomas Moreau, CR, Inria Saclay
Christophe Pouzat, CR, Université de Strasbourg
Nelly Pustelnik, CR, ENS Lyon
Alain Rakotomamonjy, Pr, Université de Rouen
Joseph Salmon, Pr, Université de Montpellier (Président du jury)

Université Côte d'Azur
École Doctorale de Sciences Fondamentales et Appliquées

Thèse présentée en vue de l'obtention du grade de
Docteur en mathématiques d'Université Côte d'Azur
par Laurent Dragoni

Tri de potentiels d'action sur des données neurophysiologiques massives

Stratégie d'ensemble actif par fenêtre glissante pour l'estimation de modèles convolutionnels en grande dimension

Thèse soutenue le 25 avril 2022
devant le jury composé de

Président du jury :
Joseph Salmon, Professeur des universités, Université de Montpellier

Rapporteurs :
Christophe Pouzat, Chargé de recherche, Université de Strasbourg
Nelly Pustelnik, Chargée de recherche, École Normale Supérieure de Lyon

Examinateurs :
Thomas Moreau, Chargé de recherche, INRIA de Saclay
Alain Rakotomamonjy, Professeur des universités, Université de Rouen

Directeurs de thèse :
Karim Lounici, Professeur des universités, École Polytechnique
Rémi Flamary, Maître de conférences, École Polytechnique
Patricia Reynaud-Bouret, Directeur de recherche, Université Côte d'Azur

Université Côte d'Azur
École Doctorale de Sciences Fondamentales et Appliquées

Thesis presented for the obtention of the title of
Doctor in mathematics of Université Côte d'Azur
by Laurent Dragoni

Spike sorting for massive neurophysiological datasets

Sliding window working set strategy for the estimation of convolutional models in high dimension

Thesis defended on April 25, 2022
before the jury composed of

President of the jury:
Joseph Salmon, University Professor, Université de Montpellier

Rapporteurs:
Christophe Pouzat, Research Associate, Université de Strasbourg
Nelly Pustelnik, Research Associate, École Normale Supérieure de Lyon

Examiners:
Thomas Moreau, Research Associate, INRIA de Saclay
Alain Rakotomamonjy, University Professor, Université de Rouen

Thesis directors:
Karim Lounici, University Professor, École Polytechnique
Rémi Flamary, University Lecturer, École Polytechnique
Patricia Reynaud-Bouret, Research Director, Université Côte d'Azur

Tri de potentiels d'action sur des données neurophysiologiques massives

Résumé: Au sein du système nerveux, des cellules appelées neurones sont spécialisées dans la communication de l'information. À travers l'émission et la propagation de courants électriques nommés potentiels d'action, les neurones peuvent transmettre l'information dans le corps. Étant donné le rôle prééminent des neurones, afin de mieux comprendre le fonctionnement du système nerveux, une vaste gamme de méthodes ont été proposées pour l'étude de ces cellules. Dans cette thèse, nous nous intéressons à l'analyse de signaux ayant été enregistrés par des électrodes, et plus spécifiquement, des tétrodes et des multi-electrode arrays (MEA). Ces appareils mesurant en général l'activité d'un ensemble de neurones, les signaux enregistrés forment souvent un mélange de l'activité de plusieurs neurones. Afin de gagner plus d'information sur ce type de données, un pré-traitement crucial appelé tri de potentiels d'action est requis pour séparer l'activité de chaque neurone. Actuellement, la procédure générale de tri de potentiels d'action repose sur une procédure en trois étapes : seuillage, extraction de caractéristiques et partitionnement de données. Malheureusement cette méthodologie requiert un grand nombre d'opérations manuelles. De plus, elle devient encore plus difficile à mettre en oeuvre sur de grands volumes de données, en particulier pour des enregistrements de MEA qui ont tendance à présenter davantage de synchronisations de neurones. Dans cette thèse, nous présentons une stratégie de tri de potentiels d'action permettant l'analyse de grands volumes de données et qui requiert peu d'opérations manuelles. Cette stratégie utilise un modèle convolutionnel dont le but est de représenter les signaux mesurés en convolutions temporelles entre deux facteurs : les activations de neurones et les formes de potentiels d'action. L'estimation de ces deux facteurs est généralement traitée par optimisation alternée. Étant la tâche la plus difficile, nous nous concentrons ici sur l'estimation des activations, en supposant que les formes de potentiels d'action sont connues. Le célèbre estimateur Lasso présente d'intéressantes propriétés mathématiques pour la résolution d'un tel problème. Néanmoins son calcul demeure difficile sur des problèmes de grande taille. Nous proposons un algorithme basé sur la stratégie d'ensemble actif afin de calculer efficacement le Lasso. Cet algorithme exploite la structure particulière du problème, déduite de propriétés biologiques, en utilisant des fenêtres glissantes temporelles, lui permettant d'être appliqué en grande dimension. De plus, nous adaptons des résultats théoriques sur le Lasso pour montrer que, sous des hypothèses raisonnables, notre estimateur retrouve le support du vrai vecteur d'activation avec grande probabilité. Nous proposons également des modèles pour la distribution spatiale et des temps d'activations des neurones qui nous permettent de quantifier la taille du problème et de déduire la complexité temporelle théorique de notre algorithme. En particulier, nous obtenons une complexité quasi-linéaire par rapport à la taille du signal enregistré. Finalement nous présentons des

expériences numériques illustrant à la fois les résultats théoriques et les performances de notre approche.

Mots clés : Tri de potentiels d'action, neuroscience, apprentissage automatique, optimisation, sparsité, Lasso.

Spike sorting for massive neurophysiological datasets

Abstract: In the nervous system, cells called neurons are specialized in the communication of information. Through the generation and propagation of electrical currents named action potentials, neurons are able to transmit information in the body. Given the importance of the neurons, in order to better understand the functioning of the nervous system, a wide range of methods have been proposed for studying those cells. In this thesis, we focus on the analysis of signals which have been recorded by electrodes, and more specifically, tetrodes and multi-electrode arrays (MEA). Since those devices usually record the activity of a set of neurons, the recorded signals are often a mixture of the activity of several neurons. In order to gain more knowledge from this type of data, a crucial pre-processing step called spike sorting is required to separate the activity of each neuron. Nowadays, the general procedure for spike sorting consists in a three steps procedure: thresholding, feature extraction and clustering. Unfortunately this methodology requires a large number of manual operations. Moreover, it becomes even more difficult when treating massive volumes of data, especially MEA recordings which also tend to feature more neuronal synchronizations. In this thesis, we present a spike sorting strategy allowing the analysis of large volumes of data and which requires few manual operations. This strategy makes use of a convolutional model which aims at breaking down the recorded signals as temporal convolutions between two factors: neuron activations and action potential shapes. The estimation of these two factors is usually treated through alternative optimization. Being the most difficult task, we only focus here on the estimation of the activations, assuming that the action potential shapes are known. Estimating the activations is traditionally referred to convolutional sparse coding. The well-known Lasso estimator features interesting mathematical properties for the resolution of such problem. However its computation remains challenging on high dimensional problems. We propose an algorithm based of the working set strategy in order to compute efficiently the Lasso. This algorithm takes advantage of the particular structure of the problem, derived from biological properties, by using temporal sliding windows, allowing it to scale in high dimension. Furthermore, we adapt theoretical results about the Lasso to show that, under reasonable assumptions, our estimator recovers the support of the true activation vector with high probability. We also propose models for both the spatial distribution and activation times of the neurons which allow us to quantify the size of our problem and deduce the theoretical complexity of our algorithm. In particular, we obtain a quasi-linear complexity with respect to the size of the recorded signal. Finally we present numerical results illustrating both the theoretical results and the performances of our approach.

Keywords: Spike sorting, neuroscience, machine learning, optimization, sparsity, Lasso.

Remerciements

Je remercie en premier lieu mes encadrants, Karim Lounici, Rémi Flamary et Patricia Reynaud-Bouret, pour leur bienveillance, leur soutien et leur infinie patience. Cette thèse n'aurait sans doute pas pu aboutir si je n'avais pas ainsi bénéficié de leur profonde culture scientifique. Merci également aux membres de mon comité de thèse, Thomas Laloë et Charles Bouveyron, pour leurs conseils et leur disponibilité.

Je remercie ensuite Nelly Pustelnik et Christophe Pouzat d'avoir accepté d'être les rapporteurs de mon manuscrit. Je remercie également Joseph Salmon, Alain Rakotomamonjy et Thomas Moreau d'avoir bien voulu composer la suite du jury.

Je remercie également mes collègues d'enseignement, Eliot, Maud, Gaëtan et Nahla, pour la préparation et l'organisation des travaux dirigés.

Merci aussi aux doctorants avec qui j'ai partagé le bureau 801, Armand, Angel, M'hammed et Benjamin, pour leur gentillesse et leur discrétion.

Je remercie enfin Isabelle Delorme et Anita Ibrahim pour leur appui administratif, Jean-Marc Lacroix et Roland Ruelle pour leurs nombreuses assistances informatiques, et plus généralement l'ensemble du laboratoire Jean-Alexandre Dieudonné.

À Mickaël

Contents

Notations **14**

1 Introduction **15**

Introduction **15**
1.1 Introduction to the spike sorting problem 15
1.2 Contributions 18
1.3 Organization of the manuscript 20

2 Production and transmission of the information in the nervous system **22**
2.1 Organization of the nervous system 22
2.1.1 Cells of the nervous system 22
2.1.2 Morphology of the neuron 22
2.1.3 Parts of the nervous system 23
2.2 Membrane potential and action potentials 25
2.2.1 The membrane potential 25
2.2.2 Generation of an action potential 27
2.2.3 Properties of the action potentials 31
2.3 Synaptic potentials and integration 32
2.3.1 Types of synaptic transmission: electrical and chemical 32
2.3.2 Excitatory and inhibitory postsynaptic potentials . . 33
2.3.3 Spatial and temporal summation mechanisms 35

3 Spike sorting **37**
3.1 Recording neural activity 37
3.2 The spike sorting problem 39
3.3 Traditional spike sorting 40
3.4 Spike sorting difficulties 42
3.5 Sparse convolutional linear model 43

4 Optimization methods **48**
4.1 The Lasso problem 48
4.1.1 Choice of the Lasso estimator 48
4.1.2 Vectorization of the convolutional model 50
4.1.3 Optimality conditions of the Lasso 51
4.2 Generic working set algorithm for the Lasso 52

4.2.1 Principle of the algorithm . . . 52
4.2.2 Efficient implementation on convolutional models . . 53
4.3 Proximal optimization methods . . . 54
4.3.1 Proximity operator . . . 54
4.3.2 Proximal algorithm . . . 55
4.4 State of the art Lasso solvers . . . 58
4.4.1 FISTA for the Lasso . . . 58
4.4.2 Coordinate Descent for the Lasso . . . 60
4.4.3 Least Angle Regression (LARS) . . . 61

5 Sliding window working set algorithm 63
5.1 Biologically based assumptions . . . 63
5.2 Overlaps . . . 64
5.2.1 Spatial overlaps . . . 65
5.2.2 Temporal overlaps . . . 66
5.3 Sliding window working set . . . 67
5.3.1 Principle of the algorithm . . . 67
5.3.2 Algorithm solution *w.r.t.* the original Lasso . . . 69
5.3.3 Numerical complexity and efficient implementation . 70

6 Mathematical results 73
6.1 Control of the spatial and temporal overlaps . . . 73
6.1.1 Spatial overlaps . . . 73
6.1.2 Temporal overlaps . . . 76
6.2 Control of the noise . . . 77
6.3 Theoretical properties of the Lasso estimator . . . 78
6.4 Complexity of the sliding window working set algorithm . . 85
6.5 Attenuation model . . . 87

7 Numerical experiments 90
7.1 Computational complexity . . . 90
7.2 Influence of the noise and the regularization parameter . . . 93
7.3 Comparison with distance-based spike sorting methods . . . 94

Conclusion 96

Bibliography 105

List of Figures

2.1 (a) Simplified representation of the structure of a neuron. The dendritic tree is colored in blue, the axon in red. These two structures extend from the cell body as branches. The presynaptic terminals are represented as red triangles at the edges of the branches of the axon. The brown cylinders covering the axon represent the myelin sheats which accelerate the transmission of information (Figure 1-9 from [Luo, 2015]). (b) Illustration of the specialization of neurons. Their specific morphologies permit to achieve specific functions. Despite this variety, most neurons share the same organization: axon and dendrites organized around the soma of the cell. Figure 45.3 from [Hillis et al., 2009]. 24

2.2 Representation of the two main parts of the nervous system. The CNS (brain and spinal cord) is in yellow, the PSN (cranial and spinal nerves) is in purple. Figure 1.12 from [Purves et al., 2018]. 25

2.3 Illustration of the electrochemical gradient. The width of the green arrows represents the magnitude of the electrochemical gradients. Left: in the case where there is no membrane potential, only the concentration difference decides the movement of the ions (from high concentration to low concentration areas). Conversely, in the case where there is a potential difference between the membrane sides, both the electrical and chemical gradients influence the ion movements. Middle: when these two factors operate in the same direction. Right: when these two factors operate in opposing directions. Figure 2-9 from [Luo, 2015]. 27

2.4 Typical shape of an action potential. Starting from the resting potential, which is slightly above the K^+ equilibrium potential, the membrane potential quickly rises around the Na^+ equilibrium potential, then drops down to the K^+ equilibrium potential. Finally the membrane potential slowly returns to its resting value. The span of the whole process takes only some milliseconds. Figure 2-24 from [Luo, 2015]. 30

2.5 Representation of a chemical synapese and the series of events taking place for the transmission of signals between two neurons. Figure 5.4 C from [Purves et al., 2018]. 34

2.6 (A) Spatial summation of two excitatory inputs. Arriving at the cell body at almost the same time, two EPSPs sum and produce a depolarization larger than the depolarization of each EPSP alone. (B) Temporal summation of two excitatory inputs. Arriving from the same synapse during a short span of time, two EPSPs sum at the cell body and also produce a larger depolarization than the depolarization of each EPSP alone. Figure 3-43 from [Luo, 2015]. 36

3.1 Illustration of the different issues to deal with after collecting an extracellular recording: which neuron is responsible of any observed spike? How many neurons generated such signal? The data were recorded from zebra finch lateral magnocellular nucleus of the anterior neostriatum, with a glass-coated platinum–iridium electrode. Figure 2 from [Lewicki, 1998]. . 39

3.2 Illustration of the three main steps in the traditional spike sorting procedure. First, voltage thresholding is performed in order to detect each spike peak instant. Using a small window around these peak instants, the spike waveforms can be extracted. Then feature extraction is achieved. As an illustration, we can see here the result of the projection of the spikes in the first two principal components. Finally, clustering of the spikes is performed, for instance with the K-means algorithm. Points in color coincide with extracted waveforms in the first step. Remark that some of them do not belong to any cluster since they result from the superposition of individual spikes. Figure 1 from [Ekanadham et al., 2014]. 42

3.3 Illustration of the convolutional model in a simple context of $E = 2$ electrodes and $N = 2$ neurons. The shapes of the action potentials are represented in the bottom left corner. The activations vectors are represented in the top right corner, and the result of the temporal convolution between the shapes and the activations is represented in the bottom right corner. For each activation time (depicted as a vertical trait), the corresponding shape appears at the same instant on the convolution. Remark the unusual shapes appearing around the center of the graph. This phenomenon may occur when two neurons activate at almost the same time (synchronization). This causes a superposition of the action potentials that creates waveforms which are not in the model. 47

4.1 Graphical representation of the soft-thresholding operator. . 59

5.1 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. 68
5.2 Illustration of the different steps in the proposed algorithm. Left: observed signal $\boldsymbol{S}$ with model reconstruction and sparse model $\boldsymbol{a}_i$. The current window is represented as a light blue background. True activations are illustrated with transparency. Right: optimality conditions (named KKT) violation at the current step. Temporal instants and neurons violating the optimality conditions are over the black dashed line. 72

7.1 First portion (0.1s = 1000 points) of the recorded signal by one of the electrodes. Notice the five different action potential shapes corresponding to the five neurons in this simulation. . 92
7.2 Comparison of the execution times for four different algorithms, when the size of the signals T growths. We represented the median execution times over 40 simulations as the dotted lines. The bands represent the execution times between the bottom and top percentiles. 93
7.3 Influence of λ and the signal-to-noise ratio on the performances of the Lasso estimator. Results are averaged over 5 draws. 94
7.4 Comparison of Lasso and clustering performances (F-measure). Results are averaged over 50 draws. 95

List of Tables

2.1 The properties of the major species of ions across the membrane at rest of the giant axon of the squid. K^+ are more concentrated inside the cell, while Na^+ and Cl^- are more concentrated outside. The membrane at rest is much more permeable to K^+ ions than to Na^+ ions. Table 9-1 from [Kandel et al., 2000]. 28

3.1 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. The number of electrodes and neurons may greatly variate, depending on the context: a single recording tetrode only comprises four electrodes, while a MEA can contain several thousands of electrodes. The number of recorded neurons ensues from this recording context. 46

7.1 Orders of magnitude for the parameters introduced in Assumption 5.1.3. 91

List of Algorithms

1 Generic working set algorithm for the Lasso 54
2 Fast iterative soft-thresholding algorithm (FISTA) 57
3 FISTA for the Lasso 60
4 Sliding window working set 70

Notations

In the following, bold upper-case letters (*e.g.* $\boldsymbol{M}$) refer to matrices and bold lower-case letters (*e.g.* $\boldsymbol{x}$) refer to vectors.

Notation	Quantity	Definition
$\ell \in \mathbb{N}$	size of the shapes	1.1
$E \in \mathbb{N}$	number of electrodes	1.1
$N \in \mathbb{N}$	number of neurons	1.1
$T \in \mathbb{N}$	number of time steps of each signal	1.1
$\boldsymbol{Y} \in \mathbb{R}^{E \times T}$	recorded signals (matrix)	1.1
$\boldsymbol{W}_n \in \mathbb{R}^{E \times \ell}$	shapes of the action potentials of neuron n	1.1
$\boldsymbol{a}_n^\star \in \mathbb{R}^T$	temporal activations of neuron n	1.1
$\boldsymbol{\Xi} \in \mathbb{R}^{E \times T}$	random noise (matrix)	1.1
$\boldsymbol{w}_{n,e} \in \mathbb{R}^\ell$	spike shape of neuron n and electrode e	3.5
$\lambda \in \mathbb{R}$	regularization parameter of the Lasso	4.1.1
$\boldsymbol{y} \in \mathbb{R}^{ET}$	recorded signals (vector)	4.1.2
$\boldsymbol{\xi} \in \mathbb{R}^{ET}$	random noise (vector)	4.1.2
$\boldsymbol{a}^\star \in \mathbb{R}^{NT}$	temporal activations (vector)	4.1.2
$\boldsymbol{H} \in \mathbb{R}^{ET \times NT}$	design matrix of the problem vectorized	4.1.2
$\boldsymbol{h}_{t,n} \in \mathbb{R}^{ET}$	$\boldsymbol{H}$ column for activation of neuron n at time t	4.1.2
$\boldsymbol{w}^{\to t} \in \mathbb{R}^T$	push of vector $\boldsymbol{w}$ at position t	4.1.2
$\partial g(\boldsymbol{x}) \subset \mathbb{R}^d$	subdifferential of function g in $\boldsymbol{x} \in \mathbb{R}^d$	4.1.3
$J \subset \{1, \dots, NT\}$	working set	4.2.1
prox_g	proximity operator of function g	4.3.1
$S^* \subset \{1, \dots, NT\}$	support of $\boldsymbol{a}^\star$	5.1
$\boldsymbol{G} \in \mathbb{R}^{NT \times NT}$	Gram matrix $\boldsymbol{H}^\top \boldsymbol{H}$	5.1
$\omega = [\![\omega_1, \omega_2]\!]$	temporal window of times $1 \leq \omega_1 < \omega_2 \leq T$	5.2.2
$\delta \in \mathbb{R}$	distance between electrodes	6.1.1
$r_0 \in \mathbb{R}$	range of detection of a neuron by an electrode	6.1.1
$\gamma_c \in \mathbb{R}$	critical parameter of spatial percolation	6.1.1
$p \in \mathbb{R}$	global activation rate of temporal activations	6.1.2

Chapter 1

Introduction

1.1 Introduction to the spike sorting problem

The nervous system is the part of an animal which handles the communication of the information between various segments of its body. Scientists refer to the field of study of the nervous system as **neuroscience**. Encompassing a wide range of disciplines, such as physiology, molecular biology or computer science, neuroscience notably aims at discovering the rules and mechanisms of complex processes such as consciousness and learning. The nervous system contains various families of cells which are specialized in the transmission of the information in the body. Among those cells, some are capable of generating and transporting electrical potentials, which is the way animals transmits information inside their body. These cells, called **neurons**, are an important focus of the studies of neuroscientists. Because neurons play such an important role in the nervous system, especially in the brain, a great variety of approaches have been developped for their study.

As such, neurons can be studied individually, at the microscopic level, in order to better understand them from molecular and cellular perspectives. Such studies notably revealed that the morphology of the neurons allows them to integrate, directly or indirectly, incoming electrical currents from other cells, and in turns generate and transmit an electrical current to other cells. Those inward and outward currents are referred as **action potentials**. They generally take the form of a sudden rise of the membrane electrical potential of the neuron, quickly followed by an equally sharp drop of this potential. This gives action potentials a characteristic shape when recorded, commonly named **spikes**. This ability to modify their membrane electrical potential, through subtle movements of ions accross their membrane, is the reason why neurons occupy such a central position in the study of the nervous sytem ([Kandel et al., 2000]).

But because each neuron is the fundamental component of a larger network, neurons can also be studied at a more macroscopic level. In order to gain knowledge about valuable elements of the neural code, various techniques for macroscopic recordings of the activity of the brain have been

developped. In this thesis, we focus on the analysis of signals which have been recorded by **electrodes**. Let us introduce two important types of recording devices belonging to this electrode paradigm. Tetrodes, which are a bundle of four small electrodes made of metal, are commonly used in order to record the extracellular field potentials. Another important type of recording devices, which emerged in the 1970s but gained important popularity in the recent years, is the multi-electrode array, or MEA ([Thomas Jr et al., 1972]). This apparatus, usually containaing hundreds or thousands electrodes, is especially interesting for the study of neural networks ([Whitson et al., 2006]).

Although these various devices allow convenient recordings of population of neurons, the signals they gather are usually a mixture of the activity of multiple neurons ([Pouzat et al., 2004]). In order to extract the precise activity of each neuron during the recording, an important pre-processing step, called **spike sorting**, is necessary. Since the shape of an action potential generated by a neuron essentially depends on the morphology of this neuron, the shape tends to remain the same along time ([Hill et al., 2011]). Therefore, viewing each shape as the signature of the activity of a particular neuron, the goal of spike sorting is to extract the shapes of the action potentials, to associate them to their respective neuron and to detect at which times each particular neuron has generated an action potential (i.e. to determine the *spike train* of each neuron).

Considering the importance of this pre-processing step, various strategies have been developped in the last decades in order to solve the spike sorting problem. In the fullness of time, these strategies have converged to a general methodology which we will call *traditional spike sorting*. It can be roughly described as a three steps procedure: thresholding (retrieval of the spike shapes from the signal), feature extraction (extraction from these shapes relevant features) and clustering (discrimination of the shapes based on their features). Then a template matching procedure associates each detected spike to the neuron which has the closest shape ([Lewicki, 1998, Pouzat and Detorakis, 2014]).

Although quite popular in the field of neuroscience, this traditional spike sorting methodology presents some important practical difficulties. A large number of manual operations are indeed necessary in order to achieve the aforementioned three steps. As a consequence, it has been established that the results obtained strongly depend of the person performing the task ([Wood et al., 2004, Harris et al., 2000]). This task becomes even more challenging when the recordings present some neuronal **synchronizations**, that is when two or more neurons tend to generate their action potentials at almost the same time. In this context, the shapes of the action potentials are often mixed, which makes the clustering task more difficult. Since synchronizations between neurons are believed to be an important property of the neural code ([Lambert et al., 2018, Albert et al., 2016, Eytan and Marom, 2006]), one of the major ambition of our work is to propose a methodology which estimates more precisely the spike trains and is

more robust in presence of such synchronizations. The limitations of the traditional approach usually worsens with data obtained from MEA. Because these devices can record large populations of neurons, they tend to generate large amounts of data, which are harder to sort manually. Moreover, for an increasing number of recording sites, it has been noted that the synchronization problem exacerbates ([Einevoll et al., 2012]).

We present in this thesis a spike sorting procedure which would permit the analysis of large volumes of data while being less demanding in terms of manual operations than the traditional approach. This procedure builds on a convolutional model, which role is to link the recorded signals with the activity of the neurons. More precisely, writing respectively ℓ, E, N and T the size of the shapes, number of electrodes, number of neurons and number of time steps of each signal, this model aims at representing the recorded signals $\boldsymbol{Y} \in \mathbb{R}^{E \times T}$ as a sum over all the neurons of the temporal convolution between the shapes $\boldsymbol{W}_n \in \mathbb{R}^{E \times \ell}$ of the action potentials and the temporal activations $\boldsymbol{a}_n^\star \in \mathbb{R}^T$ of neuron n. At each instant of activation of neuron n, the corresponding coordinate in $\boldsymbol{a}_n^\star$ equals 1, elsewhere it is 0. Mathematically, the model writes as:

$$\boldsymbol{Y} = \sum_{n=1}^{N} \boldsymbol{W}_n * \boldsymbol{a}_n^\star + \boldsymbol{\Xi}, \tag{1.1}$$

where $\boldsymbol{\Xi}$ is a random noise matrix in $\mathbb{R}^{E \times T}$ and $*$ is the convolution operator along time. Originally proposed in [Taylor et al., 1979], this type of model has been applied for instance to neuroscience ([Roberts, 1979]), speech signals ([Smaragdis, 2007]), and more recently to spike sorting ([Ekanadham et al., 2011]). An important property of this model is based on the linearity of the convolution operator, which enables the handling of synchronizations as additive superpositions of the shapes, which is not possible with traditional spike sorting relying on clustering [Lewicki, 1998].

The terms in model (1.1) are usually estimated by alternative optimization. However, the main computational bottleneck lies in the estimation of the activations $\boldsymbol{a}_n^\star$, since the numbers of variables in these terms are proportional to T, while those in the shapes $\boldsymbol{W}_n$ remain bounded. For this reason, in this thesis we focus on the estimation of the activations $\boldsymbol{a}_n^\star$, assuming that the shapes of the action potentials $\boldsymbol{W}_n$ are known. In this context of convolutional dictionary learning, this particular task is often refered to convolutional sparse coding: since the neurons tend to generate few action potentials with respect to T, the number of 1 in $\boldsymbol{a}_n^\star$ is very small compared to the number of 0. In more mathematical terms, each activation vector $\boldsymbol{a}_n^\star$ is expected to be a sparse vector. This setting, quite common in the framework of convolutional sparse coding, motivates the application of an estimation procedure which promotes the sparsity of its solutions. The favoured procedure here is the well-known Least Absolute Shrinkage and Selection Operator (Lasso), introduced by [Tibshirani, 1996]. This estimator is particularly relevant as it tends to produce a solution containing a lot of coordinates equal to zero. Moreover, its computational feasibility and its statistical precision

have brought it an important popularity in a large range of applications. In particular, it has already been applied in the context of spike sorting, for example by [Ekanadham et al., 2011] (under the equivalent term *basis pursuit estimator*). From a theoretical perspective, the Lasso is also interesting thanks to theoretical evidences showing that it is able to recover the support of the true activations (i.e. the activation times and the active neurons) under specific conditions ([Bickel et al., 2009, Bunea, 2008, Lounici, 2008]).

Despite these interesting assets, computing the Lasso estimator can be challenging for high dimensional problems. Indeed, its computational complexity growths cubically with the number of variables in the optimization problem. Remark that in our setting, this number is proportional to the length of the signal. In the general context of sparse coding problems, [Lee et al., 2007] proposed the Feature Sign Search (FSS) algorithm, which essentially consists in a working set strategy paired with the resolution of a Quadratic Programming problem. The working set strategy, taking advantage of the optimality condition satisfied by any Lasso solution, consists in the iterative activation of the coordinates in the solution vector until convergence. Doing so, the strategy aims at solving multiple subproblems of small size in order to accelerate the computations. They notably showed that this strategy grants better results than the LARS on practical applications. [Grosse et al., 2012] 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 [Moreau et al., 2018], 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.

1.2 Contributions

We propose to solve the sparse coding problem related to model (1.1) with an approach based on the working set strategy. This approach refines the working set strategy by using temporal sliding windows, allowing it to scale in high dimension. Although close to the methods proposed by [Grosse et al., 2012] and [Moreau et al., 2018], in our approach the temporal windows are not fixed a priori, rather they analyze the signal by adapting to its structure. This allows to treat the whole signal in a single pass and quickly enough to permit an online analysis.

Then we explain how we can take advantage of some biological properties of the problem in order to allow this analysis of the signal with sliding windows. We notably present the notion of overlaps between activations, which essentially represent the temporal portions of the signal which can be considered independent in the context of our optimization problem. This

notion of temporal overlaps ensures the validity of the sliding window approach, and also grants the size of the subproblems that need to be solved. Along the temporal overlaps notion, we also present the notion of spatial overlaps: taking into account the attenuation phenomenon that an action potential experiences as the distance with its source neuron increases, we take advantage of the fact that our problem can be further split into smaller spatial subproblems.

Furthermore, we also describe how the biological characteritics related to the spike sorting problem translate into useful mathematical properties. By adapting standard theoretical results about the Lasso, we show that these properties ensure the quality of the Lasso estimation. Indeed, we can prove that with high probability, our estimator recovers the support of the true activation vector.

In addition, we propose simple yet plausible models for both the spatial distribution and activation times of the neurons. These models allow us to quantify the sizes of the aforementioned spatial and temporal overlaps. Since the sliding window working set algorithm performances strongly depend on these quantities, we can deduce the theoretical time complexity of the algorithm. In particular, we note that it grows quasi-linearly with respect to the size of the recorded signal. Other works have tackled similar optimization problems such as [Jas et al., 2017] and [La Tour et al., 2018], but to the best of our knowledge, the sliding window working set approach that we present is the first to fully take advantage of the structure of the problem and to attain a quasi-linear complexity. Interestingly, since the algorithm only needs to perform linear operations such as convolutions, it can be adapted to GPU architectures, allowing a very efficient scaling.

Finally we present numerical results illustrating both the theoretical results and the performances of our approach. Note that since this algorithm is designed to solve an estimation problem associated to a convolutional model, it can potentially be used to other domains, as long as the quantities of interest verifies similar structural and sparsity properties as in the spike sorting problem (for instance the recognition of musical notes).

These various contributions have been the subject of three publications:

Technical report

Dragoni L., Flamary R., Lounici K., and Reynaud-Bouret P. (2019). Large scale lasso with windowed active set for convolutional spike sorting. arXiv preprint `https://arxiv.org/abs/1906.12077`.

This article submitted to IEEE CAMSAP 2019 introduces a preliminary version of the algorithm and its performances with simulations.

Conference proceedings

Dragoni L., Lounici K., Flamary R., and Reynaud-Bouret P., Algorithme d'ensembles actifs par fenêtre glissante pour l'estimation parcimonieuse de

modèle convolutionel, Actes des 52èmes Journées de Statistique de la Société Française de Statistique, Nice, Juin 2021, pages 308-313.

This article submitted and presented to JDS 2021 adds the size of the temporal overlaps and the complexity of the algorithm.

Preprint

Dragoni L., Flamary R., Lounici K., and Reynaud-Bouret P. (2021). Sliding window strategy for convolutional spike sorting with lasso: Algorithm, theoretical guarantees and complexity. arXiv preprint `https://arxiv.org/abs/2110.15813`.

This article submitted to the peer-reviewed mathematics journal Acta Applicandae Mathematicae presents in details our whole approach, and in particular the spatial overlaps as well as the final algorithm.

1.3 Organization of the manuscript

We present here the general organization of the manuscript. Apart from this first introductory chapter, this report is divided in six main chapters as follows:

Chapter 2 Production and transmission of the information in the nervous system: we describe in the second chapter how the nervous system is organized. Focusing on the role of the neurons, we explain why those specialized cells are able to receive electrical inputs, and in turn, generate action potentials and transmit them to other neurons. We see in particular the importance of the morphology of these cells in order to perform such complex tasks.

Chapter 3 Spike sorting: we present in the third chapter various techniques for recording the activity of one or multiple neurons. Then aiming our attention at the devices using electrodes, we motivate the importance of the spike sorting pre-processing step in order to gain relevant insights from recordings of those type of devices. We explain the general methodology which is commonly used nowadays, as well as some of its limitations. Finally, we give our formulation of the spike sorting problem using a convolutional model between the shapes of the action potentials and the activation vectors of the neurons.

Chapter 4 Optimization methods: in the fourth chapter, we begin by introducing the Lasso estimator in order to solve the sparse coding problem associated with our convolutional model. We explain why this estimator is particularly relevant for this task, as opposed to other estimators such as the least square or the ridge estimators. Then we present the generic formulation of the working set strategy in order to compute the Lasso more

efficiently in high dimension. Lastly, we describe various state of the art algorithm for the computation of the Lasso, focusing in particular on proximal methods (FISTA). We justify in particular why we choose FISTA as the inner solver of our working set algorithm.

Chapter 5 Sliding window working set algorithm: we begin the fifth chapter by formulating various mathematical assumptions about our problem. These assumptions, inspired from biological properties related to neurons and action potentials, will be of crucial importance for the analysis of our methodology. Then the notions of temporal and spatial overlaps are defined. Thanks to these ideas of overlaps, we see that we can split our original problem into independent smaller subproblems. Taking advantage of this, we finally present the sliding window working set algorithm.

Chapter 6 Mathematical results: the sixth chapter is devoted to the mathematical analysis of the sliding window working set algorithm. We establish first the size of the temporal and spatial overlaps under reasonable probabilistic assumptions on both he activation times and the neuron positions. Then we focus on the statistical properties of the Lasso estimator under the mathematical assumptions that we introduced in the previous chapter. We prove an important theorem which ensures that, with high probability, the Lasso can estimate the support of the true activation vector. Lastly, we explain how the temporal complexity of the algorithm can be deduced from the sizes of the overlaps. In particular, we see that this complexity is quasi-linear with respect to the length of the signal, which represents an important improvemeent with respect to the generic working set algorithm.

Chapter 7 Numerical experiments: we give in the seventh chapter the results of some numerical experiments illustrating the efficiency of our approach. First we show that the sliding window working set algorithm achieves as expected quasi-linearity in terms of computation time, with respect to length of the signal. Then we investigate the robustness of the method for various signal-to-noise ratios. We see in particular the importance of the calibration of the tuning parameter of the Lasso. Finally we show that the Lasso is less prone to errors than the clustering method when the number of synchronizations increases, an attractive consequence of the convolutional model.

Chapter 2

Production and transmission of the information in the nervous system

2.1 Organization of the nervous system

2.1.1 Cells of the nervous system

We begin this chapter by introducing fundamental neurobiology elements about the organization of the nervous system. In an animal, the nervous system is the component which organizes the transmission of the information, from and to the different parts of its body, essentially thanks to the activity of specialized cells named neurons ([Byrne et al., 2014, Chapter 1]). The nervous system essentially consists of two types of cells:

- neurons, also called nerve cells, which are electrically excitable cells,
- glial cells, which support the activity of neurons.

The neuron doctrine asserts that the neurons are the fundamental building blocks and signaling units of the nervous system. By establishing connections and interacting with other nerve cells, each neuron is an element of a network which has one or multiple behavioral functions.

Although neurons and glial cells derive from the same precursor cells, they do not share the same morphological and functional features. In particular, their membranes show notable differences: the membrane of a neuron is electrically excitable, while the membrane of a glial cell is not. Therefore, electrical signaling is directly performed by neurons, while glial cells surround neurons and support their activity ([Kandel et al., 2000, Chapter 3]).

2.1.2 Morphology of the neuron

We now present the morphological properties of neurons, which allow to understand how such cells can receive, produce and transmit signals. Even

though there is a great variability in neurons anatomy, almost all neurons may be consistently described as the assembly of different functional regions (see Figure 2.1).

For the vertebrates, the cell body, called **soma**, includes the nucleus and the endoplasmic reticulum. This part of the cell is notably responsible for the synthesis of the proteins. The soma is also the place where the incoming signals from other neurons are processed. Typically, the diameter of the soma from a cortical neuron varies from about 10 to 50µm ([Dayan and Abbott, 2001, Chapter 1]).

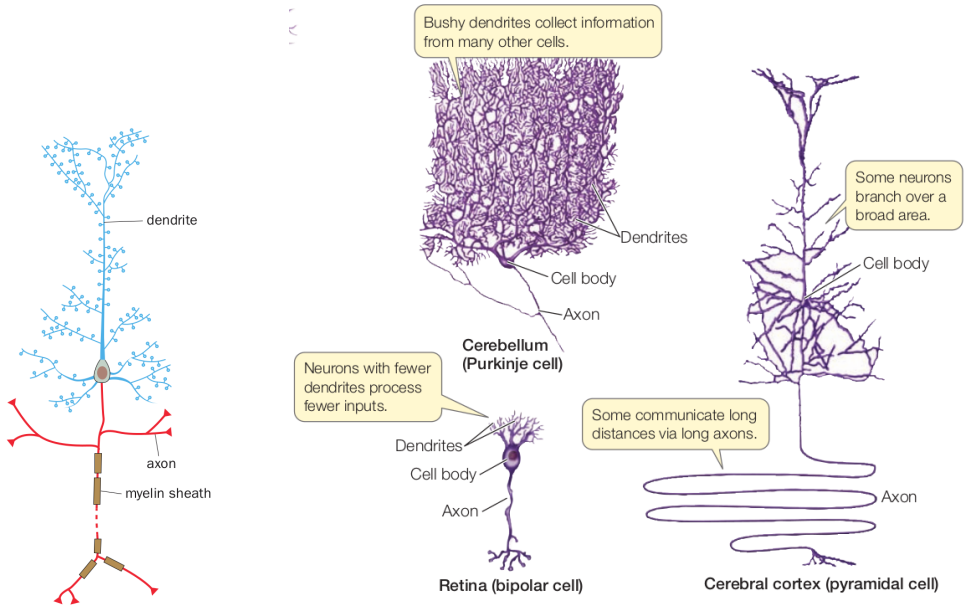
Then, usually two types of extensions grow from the cell body. First, the short leafy extensions called **dendrites**. The complex branching organization of the dendritic tree allows a neuron to receive signals from other neurons. Second, the long tubular extension called the **axon**. After extending along some distance from the soma, the axon branches, which permits the transmission of signals to target neurons. There is a great variability in the axons anatomy. Although most of them are quite thin compared to the soma, their diameter may range from 0.2µm to 20µm. Their length also exhibits a large variability, some of them limited in the brain, but others traversing large parts of the entire body, with a length sometimes exceeding 1m ([Kandel et al., 2000, Chapter 3]). An isolating sheath made of myelin, which is a lipid susbtance, surrounds large axons in order to increase the speed of transmission of signals.

At its end, the axon splits in thin branches which can reach other neurons (note that some of these branches may also reach other types of cells, such as muscle cells). These structures, called **presynaptic terminals** are specialized in the transfer of signals between nerve cells. More generally, we call **synapse** the region where the presynaptic cell (the cell generating the signal) and the postsynaptic cell (the cell receiving the signal) form a connection. The vast majority of the presynaptic terminals form a connection with the dendrites, less frequently the soma, of the postsynaptic neurons. Estimations indicate that, in the neocortical regions, a given neuron can receive connections from up to 10.000 presynaptic cells, and conversely, can connect to up to 10.000 postsynaptic cells. Since the number of neurons in the human brain is of order 10^{11}, the complexity of the whole network is prodigious (Table 1-1 from [Luo, 2015, Chapter 1]).

2.1.3 Parts of the nervous system

The nervous system of all vertebrates and many invertebrate animals is essentially divided in two parts (see Figure 2.2):

- the **central nervous system** (CNS), which consists, for the vertebrates, of the brain and the spinal cord,
- the **peripheral nervous system** (PNS), which consists of the nerves and ganglia not in the brain or in the spinal cord.



(a) Basic structure of a neuron.

(b) Specialized neurons.

Figure 2.1: (a) Simplified representation of the structure of a neuron. The dendritic tree is colored in blue, the axon in red. These two structures extend from the cell body as branches. The presynaptic terminals are represented as red triangles at the edges of the branches of the axon. The brown cylinders covering the axon represent the myelin sheats which accelerate the transmission of information (Figure 1-9 from [Luo, 2015]). (b) Illustration of the specialization of neurons. Their specific morphologies permit to achieve specific functions. Despite this variety, most neurons share the same organization: axon and dendrites organized around the soma of the cell. Figure 45.3 from [Hillis et al., 2009].

The role of the CNS is to analyze the information coming from the body and, in return, make decision to coordinate its activity. The brain comprises different structures (e.g. the cerebellum, the diencephalon, etc. [Luo, 2015, Chapter 1]), which consists of distinctive groups of neurons. These groups can be distinguished both in terms of development and connectivity properties. Due to the importance of the brain in the processing of the information, the majority of the studies tend to focus on this part of the nervous system. Nowadays, various brain imaging methods allow to study precisely the activity of these different regions of the brain. More specifically, the usage of such techniques in the context of controlled experiments, during which people or animals perform certain tasks, shows that specific behaviors mobilize specialized parts of the brain.

The PNS is formed by nerves (groups of axons wrapped together as cables) which connect the CNS with the rest of the body, including internal organs, and also isolated ganglia (bundles of cell bodies of neurons). We can divide the PNS into two parts, namely the somatic nervous system and the

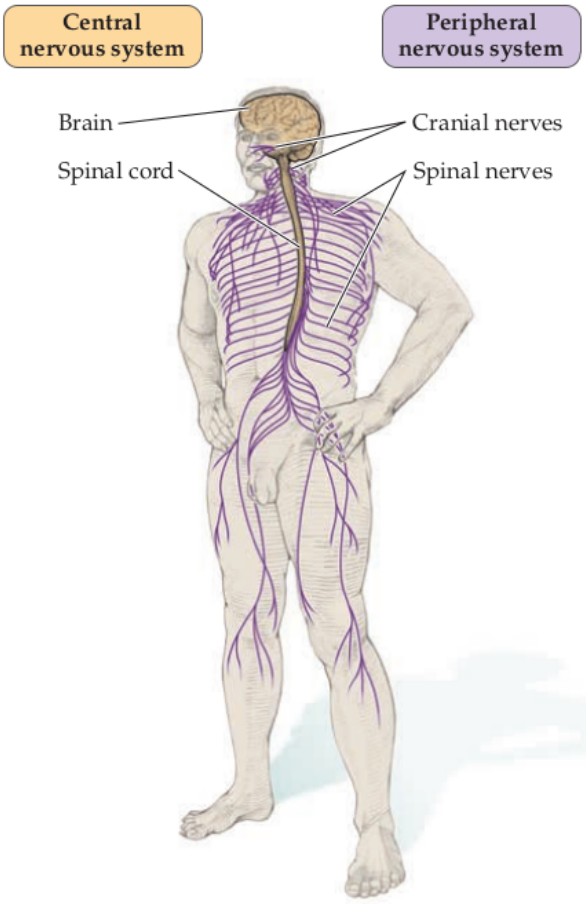


Figure 2.2: Representation of the two main parts of the nervous system. The CNS (brain and spinal cord) is in yellow, the PSN (cranial and spinal nerves) is in purple. Figure 1.12 from [Purves et al., 2018].

autonomic nervous system. The somatic nervous system is formed by motor neurons, which transport signals out of the CNS, to organs such as muscles. This part of the PNS is under voluntary control. Additionally, the somatic nervous system also contains sensory neurons, which transmit sensations such as heat, pressure, or pain. On the other hand, the autonomic nervous system is a self-regulating system which affects the activity of organs not in conscious control, such as the digestive system or the heart ([Luo, 2015, Chapter 1]).

2.2 Membrane potential and action potentials

2.2.1 The membrane potential

In the previous section, we saw that neurons are the cells in charge of the processing and the transmission of the information in the body. Neurons

communicate with each other by analyzing and exchanging signals. In this section we delve deeper into the neurons anatomy, more specifically their membrane, in order to explain the nature of these signals and how neurons generate them.

All the animals feature an uneven distribution of ions between the inside of their cells and their extracellular fluid: inside of the cells, there are more K^+ ions, while in the extracellular fluid, Na^+ ions are more abundant ([Hillis et al., 2009, Chapter 45.2]). Focusing only on the inside or the outside of the cell, the presence of negative ions counterbalance the presence of the positive charges, such that both the interior of the cell and the extracellular fluid are electrically neutral. On the other hand, because of the presence of channels in the membrane of the cell, leak currents imply the existence of a difference between the electrical charges inside and outside. The movements of any given type of ion across a membrane channel depend essentially on two factors. First, the concentration difference of that ion between the interior and the exterior: ions tend to move from regions of high concentration to regions of low concentration (chemical gradient). Second, the voltage difference between the interior and the exterior: positive ions tend to move to negative regions, while negative ions tend to move to positive regions (electrical gradient). The combination of these two forces on any given ion is called its **electrochemical gradient**. It characterizes the direction and the magnitude of the force ruling its movement (see Figure 2.3).

Most channels of the membrane only allow a given type of ions to passively leak through them. Among these discriminating channels, the majority of them specifically allow K^+ ions leakage. Since K^+ ions are more abundant inside of the cell than outside, the chemical gradient tends to move these ions out the cell. On the other hand, each time a K^+ ion leaves the cell, it also leaves behind a negative charge which is no longer balanced. Therefore, the electric gradient tends to pull back the K^+ ion into the cell. When both these conflicting gradients cancel out, the K^+ ions dynamic reaches an equilibrium. The resulting electrical charges difference is called **membrane potential** ([Hillis et al., 2009, Chapter 45.2]). The potential difference may be measured using a microelectrode, which is essentially a very thin glass pipette containing an electrode and of diameter of order 1µm at its tip, that can penetrate the cell. In the absence of exterior stimulation, a constant potential difference of about -50mV is measured in the cell compared to the exterior. Although all cells in the body present a membrane potential, some cells (such as neurons or muscle cells) have an excitable membrane which allows them to modify their membrane potential. Therefore for these excitable cells, the membrane potential measured in the absence of any stimulation is called the **resting potential**. Even though K^+ ions are not the unique ions there, since the most common leak channels are potassium channels, the measured resting potential V_m is close to the potassium equilibrium potential E_{K^+}, which can be determined by the Nernst equation

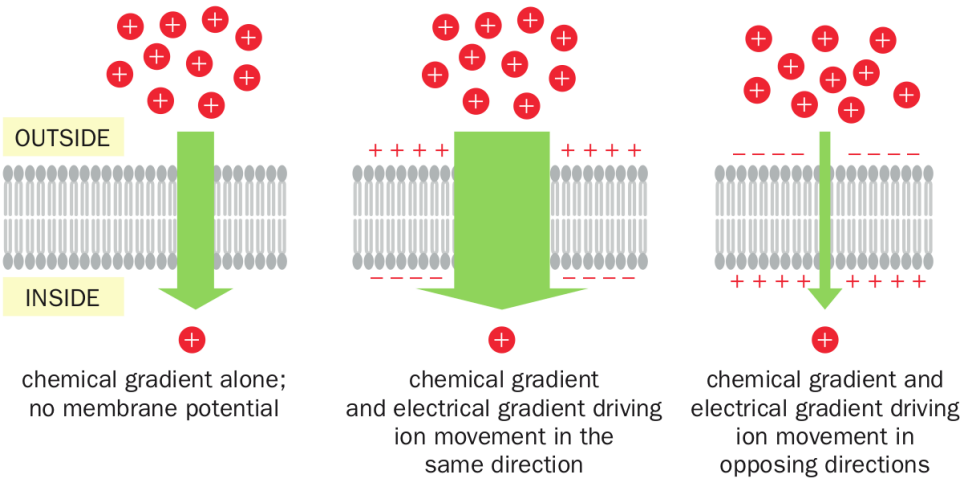


Figure 2.3: Illustration of the electrochemical gradient. The width of the green arrows represents the magnitude of the electrochemical gradients. Left: in the case where there is no membrane potential, only the concentration difference decides the movement of the ions (from high concentration to low concentration areas). Conversely, in the case where there is a potential difference between the membrane sides, both the electrical and chemical gradients influence the ion movements. Middle: when these two factors operate in the same direction. Right: when these two factors operate in opposing directions. Figure 2-9 from [Luo, 2015].

$$E_{K^+} = \frac{RT}{F} \ln \frac{[K^+]_o}{[K^+]_i}, \tag{2.1}$$

where R is the gas constant, T the temperature (in Kelvin), F the Faraday constant, and $[K^+]_o$ and $[K^+]_i$ respectively the concentrations of K^+ ions outside and inside the cell ([Kandel et al., 2000, Chapter 9]).

2.2.2 Generation of an action potential

Regarding most neurons, their membrane potential may undergo fast and noticeable changes, which are called **action potentials**. It consists in a very fast rise followed by a very quick fall of the membrane potential. This distinctive shape explains why the action potentials are also called **spikes** (see Figure 2.4).

The underlying mechanism of these rapid events relies upon the opening and closing of specific ion channels of the neuron membrane. The K^+ dynamic presented above is sufficient to roughly explain the existence of a resting potential, but the notion of action potential requires to study more deeply the mechanisms of the membrane. More specifically, we now need to take into account other abundant ions, which are Na^+ and Cl^- , both present at high concentrations outside of the cell. When V_m needs to be calculated in presence of more than a single permeant ion species, each species con-

Species of ion	Concentration in cytoplasm (mM)	Concentration in extracellular fluid (mM)	Equilibrium potential (mV)	Permeability ratios
K^+	400	20	-75	1.0
Na^+	50	440	+55	0.04
Cl^-	52	560	-60	0.45

Table 2.1: The properties of the major species of ions across the membrane at rest of the giant axon of the squid. K^+ are more concentrated inside the cell, while Na^+ and Cl^- are more concentrated outside. The membrane at rest is much more permeable to K^+ ions than to Na^+ ions. Table 9-1 from [Kandel et al., 2000].

tribution still depends on its concentrations outside and inside the cell, but also depends on its ability to cross the membrane, which is called the permeability (noted P) of this species. Then the membrane potential V_m can be expressed as the balance of the concentrations and permeability of each species of ion with the Goldman equation ([Kandel et al., 2000, Chapter 9])

$$V_m = \frac{RT}{F} \ln \frac{P_{K^+}[K^+]_o + P_{Na^+}[Na^+]_o + P_{Cl^-}[Cl^-]_i}{P_{K^+}[K^+]_i + P_{Na^+}[Na^+]_i + P_{Cl^-}[Cl^-]_o}. \tag{2.2}$$

Note that this applies when the system is at the equilibrium. Clearly when a given species of ions has an overwhelmingly large permeability compared to the others, the Goldman equation (2.2) reduces to the Nerst equation (2.1) associated with this species. An experiment on the giant axon of the squid performed by [Hodgkin and Katz, 1949] estimated the values of the permeabilities when the membrane is at rest (see Table 2.1). These values confirm that the permeability P_{K^+} dominates when the membrane is at rest. Therefore at the equilibrium, the main contributor for the value of the resting potential is the K^+ ion species. The Cl^- and Na^+ species presence tends to slightly raise the resting potential compared to the equilibrium potential of K^+ alone.

We now explain how the membrane is able to leave its electrical equilibrium state, in order to generate action potentials. Thus far, we only presented ion channels that are able to let leak a specific ion species. Another important class of channels consists of **voltage-gated ion channels**. These channels are also able to select a certain species of ion, but contrarily to leak channels which are always open, these voltage-gated channels are able to open or close, depending on the evolution of the voltage across the membrane.

The joint actions of voltage-gated K^+ and Na^+ channels in the membrane of the axon are responsible for the production of action potentials. When the neuron is at rest, these channels are closed. When the membrane **depolarizes** (i.e. its potential gets more positive than at the resting state) due to some external stimulus, some voltage-gated K^+ and Na^+ channels

start to open. This causes K^+ and Na^+ ions to move down their respective electrochemical gradient. Since K^+ ions are more concentrated inside of the cell, K^+ ions tend to leave it through leak channels and the K^+ voltage-gated channels which are opened. Positive charges leaving the cell makes the potential more negative. On the contrary, Na^+ ions are more concentrated outside of the cell, therefore they tend to enter the cell through the voltage-gated Na^+ channels which are opened. This inward flow of positive charges makes the potential of the cell less negative. An important distinction between K^+ and Na^+ voltage-gated channels is the fact that the Na^+ channels tend to open more quickly than K^+ channels. Therefore the inward current of Na^+ ions tends to depolarize even more the membrane, thus opening additional voltage-gated channels ([Hillis et al., 2009, Chapter 45.2]).

If the initial depolarization is small, this depolarizing action of the Na^+ ions is counterbalanced by the hyperpolarizing action of the K^+ ions, since their electrochemical gradient is further increased due to the inflow of positive charges. In this situation, the net ionic current returns to zero. However, if the initial depolarization reaches a certain threshold, roughly 10mV over the resting potential, the more rapid opening of Na^+ channels induces a positive feedback loop in which the additional opening of Na^+ channels causes more depolarization, and the other way around. This state corresponds to the first phase of the generation of an action potential: its rising phase. Under these circumstances, the inward Na^+ current surpasses the outward K^+ current, in other words there is a net inward ionic current. The sudden opening of Na^+ channels, that is the increasing of the Na^+ membrane permeability, leads the membrane potential to quickly move towards the Na^+ equilibrium potential, around +40mV. This rising phase of the action potential stops swiftly after 1 to 2 milliseconds, when the Na^+ channels close. At this point, only the K^+ channels are left open. This state corresponds to the second phase of the action potential: its falling phase. At this stage, the K^+ ions leave the cell, causing a **repolarization** of the membrane. Because the membrane permeability to K^+ ions is higher during this phase than at the resting state, and since there was an accumulation of positive charges inside the cell during the rising phase, the electrochemical gradient of K^+ ions is now higher than at rest. Consequently, the membrane potential returns to a negative value, which is usually more negative than the resting potential. In this case, the membrane is said to be in a **hyperpolarized** state. Then the voltage-gated K^+ channels close, which eventually brings the potential back to its resting state ([Kandel et al., 2000, Chapter 10], [Squire et al., 2012, Chapter 5]).

This mechanism of action potential generation essentially takes place at the **axon hillock** of the neuron, that is where the axon emerges from the cell body. This is due to the fact that this region is particularly rich in voltage-gated K^+ and Na^+ channels. After the emission of an action potential, this one propagates rapidly down the axon, in direction to the presynaptic terminals.

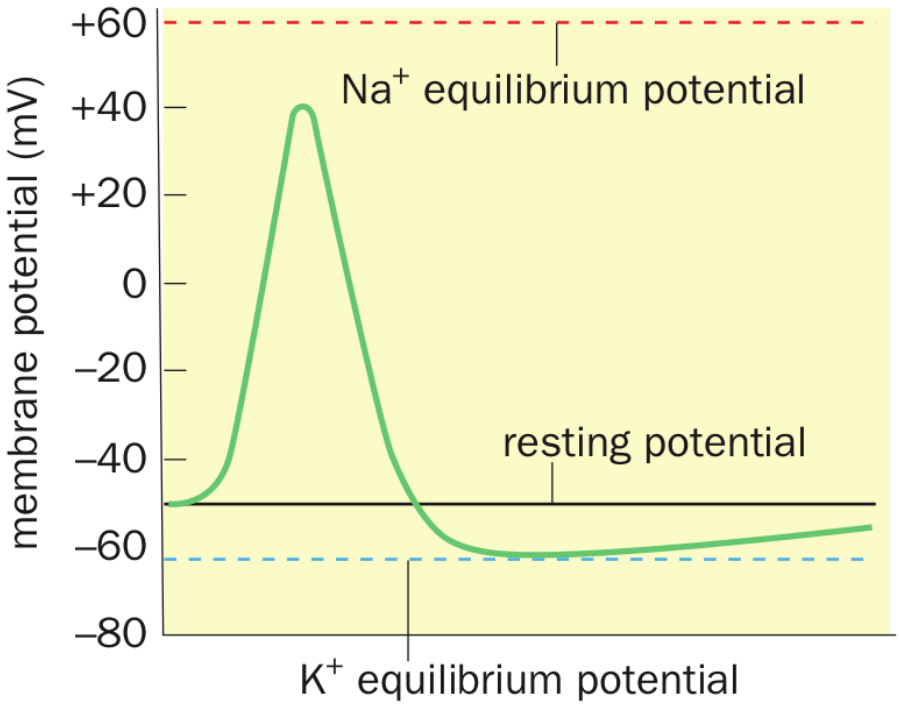


Figure 2.4: Typical shape of an action potential. Starting from the resting potential, which is slightly above the K^+ equilibrium potential, the membrane potential quickly rises around the Na^+ equilibrium potential, then drops down to the K^+ equilibrium potential. Finally the membrane potential slowly returns to its resting value. The span of the whole process takes only some milliseconds. Figure 2-24 from [Luo, 2015].

Even though the membrane potential undergoes a large variation of its potential during the generation of a single action potential, the respective concentrations of K^+ and Na^+ ions inside and outside the cell are marginally modified. Therefore the concentration gradients of K^+ and Na^+ ions remain almost unchanged, and the cell is able to repeat the whole process in order to generate another action potential. But reiterating this a substantial number of times may eventually alter the initial concentrations and finally prevent the cell to fire new action potentials. In order to maintain suitable concentrations of K^+ and Na^+ ions inside and outside the cell, **sodium-potassium pumps** present in the membrane actively brings K^+ ions back into the cell and Na^+ ions back out of it. This transfer of the ions against their respective electrochemical gradients requires a supply of energy, in the form of adenosine triphosphate (ATP). For each molecule of ATP hydrolyzed, the pump is able to bring back two K^+ ions into the cell, and push out three Na^+ ions ([Kandel et al., 2000, Chapter 9]).

2.2.3 Properties of the action potentials

The mechanism described above implies several properties of the action potentials. First, action potentials are said to be **all-or-none**: as we saw earlier, if the stimulus depolarizing the membrane is small so that the potential does not exceed the threshold, no action potential is fired. On the contrary, if the depolarization makes the potential surpass the threshold, an action potential is fired. Then its waveform essentially depends on the timings of the voltage-gated channels openings, and on the concentrations of K^+ and Na^+ inside and outside the cell. For any given neuron, these two factors stay approximately constant so, for any stimulus causing the potential to exceed the threshold, the waveform of the action potential is essentially the same ([Luo, 2015, Chapter 2]).

Another important property of the actions potentials relies on the opening and closing dynamics of the voltage-gated Na^+ channels. These channels actually have two gates: an activation gate and an inactivation gate. At rest, the activation gate is closed and the inactivation gate is open. If the membrane depolarizes to a value higher than the threshold, then the activation gate quickly opens, while the inactivation gate closes more slowly. Therefore, the channel remains momentarily open. Then the inactivation gate stays closed for several milliseconds, after which they open. During this period of inertia of the inactivation gate, the associated channel is forced to remain closed. Consequently, after the generation of an action potential, the Na^+ channels which opened become inactivated. This period of inactivation is called the **refractory period**. If a new depolarizing stimulus is received during the refractory period, the neuron is not able to fire a new action potential, because too few Na^+ channels are available to open. As a result, the action potential firing rate of a neuron is limited by its refractory period ([Luo, 2015, Chapter 2]).

These properties (activation threshold, all-or-none, refractory period)

are quite peculiar with respect to biological phenomenons, which more commonly react in a continuous manner to exterior stimuli. Since action potentials are all-or-none, varying their amplitude in order to encode information is not possible. Instead, the nervous system encodes information by adjusting the frequency rate at which neurons generate action potentials. For instance, in response to a higher intensity of light, the central nervous system receives a higher number of action potentials. The firing rate being limited by the refractory period, the information transmitting ability of the axon is also limited.

Note that, although most neurons fire action potentials in order to transmit information, some of them do not. Rather, they communicate by releasing graded electrical signals, which then induce the release of graded neurotransmitters ([Luo, 2015, Chapter 2]).

2.3 Synaptic potentials and integration

We now know under which circumstances a neuron is able to fire an action potential: if the membrane region around its axon undergoes a sufficient depolarization, and if the neuron is ready to fire (ie is not in its refractory period), then it generates an action potential. In this section, we explain where does this initial triggering depolarization originate from. More specifically, we describe how the neuron receives signals from other neurons and how it integrates these inputs. In section 2.1, we saw that the regions where the neurons exchange signals are their synapses. For the majority of synapses, their presynaptic side are on the axon and their postsynaptic side are on a dendrite, or less commonly on the soma ([Kandel et al., 2000, Chapter 13]). We need to analyze the synaptic transmission process in order to understand fully how neurons communicate with one another.

Although the number of synaptic connections may vary greatly for any given neuron, on average, a neuron shares thousands of connections with other neurons. Despite this multiplicity and the existence of very specific synaptic connections in the nervous system, all neurons utilize one of these two fundamental forms of synapses: **chemical** or **electrical**.

2.3.1 Types of synaptic transmission: electrical and chemical

The most common synapses are chemical synapses. In these synapses, the transmission of the signal follows a complex series of events (see Figure 2.5). In chemicals synapses, the presynaptic terminal and the postsynaptic membrane are separated by a small gap called the **synaptic cleft**, which is essentially made of extracellular fluid. Let us now describe how a signal coming from one end is transmitted to the other. First, when an action potential arrives at the terminal of the presynaptic neuron, the local membrane potential changes. Then voltage-gated Ca^{2+} channels in the membrane open, and due to the important concentration gradient of Ca^{2+} ions,

a quick inflow of Ca^{2+} penetrates the presynaptic terminal. The supply of these ions cause some **synaptic vesicles**, which essentially contain **neurotransmitters**, to coalesce with the membrane of the presynaptic terminal. This fusion makes the neurotransmitters move through the synaptic cleft. This mechanism is known as exocytosis ([Kandel et al., 2000, Chapter 11]).

On the other side, **receptors** in the postsynaptic membrane allow the binding of the neurotransmitters that have been released. This induces a modification of permeability of the postsynaptic membrane, which finally produces a postsynaptic potential. Ultimately the neurotransmitters are removed from the synaptic cleft, through diffusion and the action of neighboring glial cells. Note that, between the generation of an action potential in the presynaptic cell and the postsynaptic potential, there is a delay of about 1ms. This delay of transmission is essentially caused by the whole neurotransmitters transfer process: release from the vesicles, diffusion in the synaptic cleft, binding with the receptors. Chemical synapses can handle complex interactions between neurons: they can for instance amplify signals, in the sense that even small presynaptic terminals are able to modify the reaction of larger postsynaptic cells. They can also cause modifications in the postsynaptic cells which last from milliseconds to hours. Also note that the transmissions through chemical synapses are usually unidirectional, since the postsynaptic neurons do not cause any effect on the presynaptic neurons ([Blaustein et al., 2011, Chapter 12,13]).

The electrical synapses differ from chemical synapses, in the sense that they couple two neurons electrically. For these synapses, a tiny space of a few nanometers separate the presynaptic and the postsynaptic cell. A specialized structure named **gap junction** assumes the role of electric current pathway between two neurons. Therefore, thanks to this direct ionic pathway, a depolarization coming from the presynaptic terminal directly modifies the potential of the postsynaptic terminal. Electrical synapses are essentially used to send very quick and stereotyped depolarizing signals. For example, for tasks which require rapid transmissions such as predator escape, electrical synapses are particularly relevant. In fact, their first identification dates back to the 1950s, between the giant axon and motor neurons in the crayfish escape circuit. Contrarily to chemical synapses, electrical synapses allow transmissions in either direction, blurring the distinction between presynaptic and postsynaptic neurons. Despite their advantages, electrical synapses are less common than chemical synapses, mainly because of their lack of flexibility, which is required for complex processes such as learning ([Hillis et al., 2009, Chapter 45.3]).

2.3.2 Excitatory and inhibitory postsynaptic potentials

In response to the signal transmitted by the presynaptic neuron, the postsynaptic neuron can react in two ways. First, the potential generated in the postsynaptic membrane can excite (ie depolarize) the postsynaptic cell.

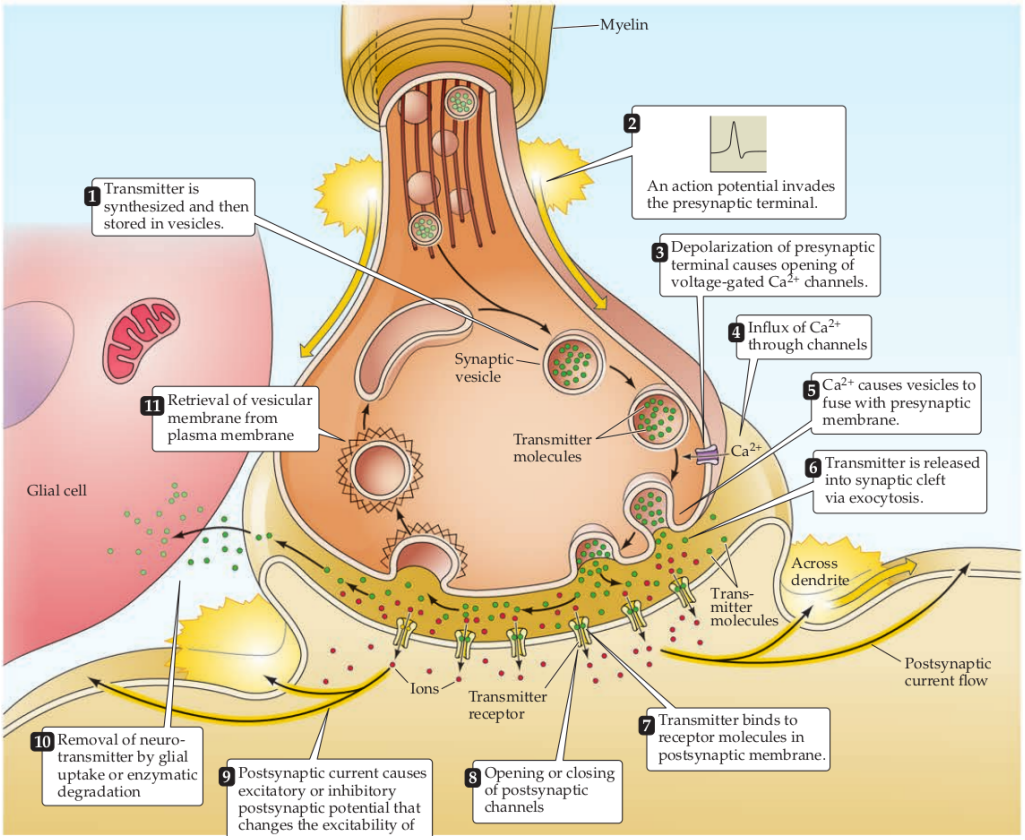


Figure 2.5: Representation of a chemical synapese and the series of events taking place for the transmission of signals between two neurons. Figure 5.4 C from [Purves et al., 2018].

This type of reponse is called an **excitatory postsynaptic potential** (EPSP). An EPSP contributes to bring the membrane potential of the postsynaptic neuron closer to its firing threshold, therefore it increases the likelihood for a postsynaptic action potential to happen. On the contrary, the response can inhibit (ie hyperpolarize) the postsynaptic cell. This second type of reaction is named an **inhibitory postsynaptic potential** (IPSP). An IPSP decreases the likelihood for the postsynaptic neuron to fire an action potential ([Purves et al., 2018, Chapter 5]). Note that inhibitory synapses only involve chemical synapses, since electrical synapses are always excitatory. This is another important limitation for the electrical synapses.

Whether the response of the postsynaptic cell is excitatory or inhibitory essentially depends on the class of ion channels in the postsynaptic membrane. Although not directly responsible for the type of response, the majority of neurotransmitters induce an unique response. For instance, the majority of receptors which receive glutamate tend to produce an excitatory response. On the other hand, gamma-aminobutyric acid (GABA) tends to induce inhibition. Glutamate and GABA constitute the two main neurotransmitters used in the vertebrate CNS, but more than five hundred different neurotransmitters have been found so far ([Luo, 2015, Chapter 3]).

2.3.3 Spatial and temporal summation mechanisms

Usually, a single EPSP generated at an excitatory synapse is quite below the firing threshold of its neuron, and therefore, is not sufficient to produce an action potential. We need to explain the mechanism ruling the firing pattern of a neuron in order to understand how postsynaptic potentials can trigger an action potential. The dendritic tree of a neuron in a mammalian CNS receives, on average, several thousands of excitatory synaptic inputs. At any given time, the neuron sums the postsynaptic potentials produced at its synapses, both in space and time. This summation process (see Figure 2.6) takes place, for most neurons, at the base of the axon, in the axon hillock ([Kandel et al., 2000, Chapter 13]).

First let us explain the spatial summation mechanism by considering the very simplified setting where a neuron would receive EPSPs from only two excitatory synapses. If these EPSPs arrive at the axon hillock at almost the same time, the two EPSPs sum in the membrane of the axon hillock, producing a larger EPSP, which may exceed the firing threshold of the neuron. By this mechanism, the summation of subthreshold EPSPs is able to cause the production of action potentials.

On the other hand, temporal summation consists in the summation of EPSPs generated in a rapid sequence at the same synapse. The characteristic of the neuron membrane which allows this summation is its ability to temporarily store electrical charges. The charge of the second EPSP is added to the charge of the first EPSP which is temporarily stored. This temporal summation is only possible if the EPSPs arrive in a short time

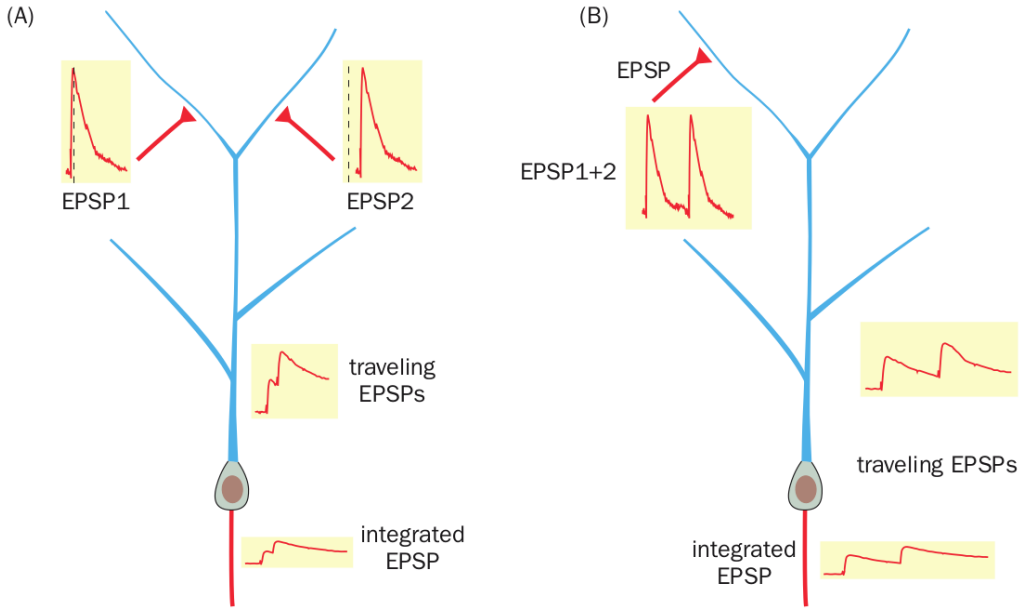


Figure 2.6: (A) Spatial summation of two excitatory inputs. Arriving at the cell body at almost the same time, two EPSPs sum and produce a depolarization larger than the depolarization of each EPSP alone. (B) Temporal summation of two excitatory inputs. Arriving from the same synapse during a short span of time, two EPSPs sum at the cell body and also produce a larger depolarization than the depolarization of each EPSP alone. Figure 3-43 from [Luo, 2015].

frame, which essentially depends on the passive properties of the membrane ([Luo, 2015, Chapter 3]).

Since these summations mechanisms also apply to IPSPs, the decision to fire an action potential actually results from the balance between excitation and inhibition, at any given time. In the situation where the sum of EPSPs and IPSPs gives a depolarization large enough so that the membrane potential exceeds the firing threshold, then the postsynaptic neuron generates an action potential. On the other hand, when inhibition dominates, the postsynaptic neuron stays quiet.

In conclusion, we can see that at any given moment a neuron acts like a complex integrative device. Its anatomical properties allow the integration and conversion of a large number of analog input signals into a single binary signal (release of a spike or not). On a larger scale, by organizing neurons as complex neuronal circuits, and taking advantage of the rich topological possibilities (number of connections, excitation, inhibition), the nervous system is able to produce a wide range of tasks, from sensory perception to behavioral control.

Chapter 3

Spike sorting

We have seen in chapter 2 that the nervous system essentially processes information through the activity of specialized cells, the neurons. In this chapter, we describe an important tool in neuroscience, which aim to bridge the gap between recordings from the brain and interpretable results in terms of neuronal activity: spike sorting.

3.1 Recording neural activity

In the late 18th century, Luigi Galvani discovered that electrical stimulations were able to provoke physical reactions to the legs of a dead frog. This remarkable event is probably the first time a reliationship was found between electrical signals and nervous transmission. However, modern devices allowing to measure the nervous activity only appeared in 1921, with the invention of **microelectrodes** by Ida Hyde ([Lewicki, 1998]). A microelectrode is a very thin glass pipette shaped electrode. In order to study the characteristics of the membrane channels, the first investigations were carried out by recording the membrane potentials using microelectrode on the whole cell. Then in 1949, an important achievement due to [Ling and Gerard, 1949], who tailored the microelectrodes in order to measure the interior of a single cell, allowed the precise recordings of action potentials and resting membrane potentials. These microelectrodes of a diameter smaller than 0.5µm are able to pierce the membrane of the cell and enter its cytoplasm. When the insertion is correctly achieved, the membrane binds over the microelectrode, so the cytosplasm of the cell and the extracellular fluid stay separated ([Nicholls et al., 2001, Chapter 4]).

Although these recording devices provided significant improvements of our knowledge at the scale of a single neuron, such as the electrical mechanisms which trigger action potentials, their contribution to our understanding of the functioning of the brain at the level of a neuronal network was insufficient. The main reason essentially relies on the technical challenge that recording the activity of a substancial number of neurons represents ([Taketani and Baudry, 2010]). Being able to bring together our understanding of the behavior of a single neuron and a whole network has been

an important effort in the last decades. Indeed, complex brain functions such as memory acquisition or speech recognition rely on large populations of neurons acting in concert, both in time and space. Pathophysiological conditions, for instance the Alzheimer's disease, can also be seen as the inappropriate activity of many neurons ([Fejtl et al., 2006]).

The introduction of alternative technologies allowed the monitoring of large numbers of neurons. For instance, the **tetrode** is another popular tool for recording the neural activity. It consists of a bundle of four small electrodes made of metal, named channels. Contrarily to the intracellular microelectrodes described earlier, tetrodes are used to record the extracellular field potentials ([McNaughton et al., 1983]). Because the four channels of a tetrode are usually not equidistant to a given neuron, the action potentials recorded generally do not share the same amplitude. This important geometrical feature makes tetrodes more effective at separing spikes than single recording units ([Harris et al., 2000]), an interesting property for the task named spike sorting and described in the next section. Another technique, calcium imaging, makes use of the variations in the intracellular concentrations of calcium ions to keep track of the electrical activity of a whole population of neurons ([Purves et al., 2018, Chapter 1]). On the other hand, many researchers pursued the extension of the microelectrode approach. The first **multi-electrode array (MEA)** was built in the 1970s ([Thomas Jr et al., 1972]), yet the contribution of these devices to our knowledge of higher brain functions remained modest until more recent progresses, which were twofold: first, technological advances allowed the conception of devices able to record and stimulate large populations of neurons. On the other hand, new computational methods emerged to study the massive amounts of data that these new tools generate. In order to perform a direct examination of the temporal dynamics of a population of neuron, these new devices must satisfy two major constraints: a large (up to 100 or more) number of electrodes must be put, without major harm, in a relatively small volume of tissue. They must also provide an adequate identification of action potentials generated by individual neurons ([Harris et al., 2000]).

Let us briefly review the advantages and disadvantages of the MEA, with respect to traditional microelectrodes. Since the electrodes of a MEA cannot be moved individually, the size of the array must match the physical configuration of the experiment. The simplicity and quickness of the installation of a MEA comes at the price of a lower placement precision. Another consequence is that the electrodes of a MEA do not penetrate the inside of the cells, therefore the amplitude of their recordings tend to be smaller than with intracellular recording microelectrodes. From a signal processing point of view, this reduction of the signal-to-noise ratio makes the analysis of the data more difficult. Nevertheless, the capacity of recording in parallel data from thousands of neurons, and changing rapidly both recording and stimulation eletrodes of the array, makes the MEA a formidable tool for the study of neural networks ([Whitson et al., 2006]).

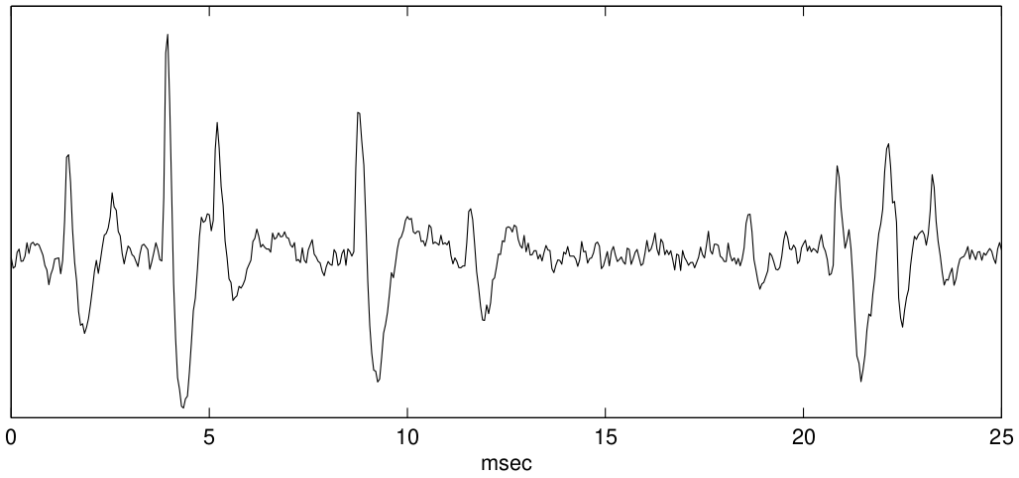


Figure 3.1: Illustration of the different issues to deal with after collecting an extracellular recording: which neuron is responsible of any observed spike? How many neurons generated such signal? The data were recorded from zebra finch lateral magnocellular nucleus of the anterior neostriatum, with a glass-coated platinum–iridium electrode. Figure 2 from [Lewicki, 1998].

3.2 The spike sorting problem

In the rest of this chapter, we focus on monitoring methods based on extracellular recordings with microelectrodes, such as tetrodes and MEA. In modern neuroscience, these methods are particularly appreciated for their simplicity and their reasonable cost. Moreover, the high sampling rates of their recordings, of order 10kHz, allow a precise investigation of the temporal activity of large numbers of neurons. Regrettably, unlike intracellular microelectrodes, these methods often do not record the activity of a single neuron on each of their channel. Therefore, extracellular recordings tend to generate raw data which feature the combination of the activities of multiple neurons ([Pouzat et al., 2004]).

In order to interpret the results of such recordings, several questions must be adressed: how many neurons are involved in the observed data, and, assuming that the activity of the neurons can be properly separated from the recording noise, which neuron is responsible for any given spike. These different problems can be envisioned in the recording example presented in 3.1. Also observe that some spikes may overlap. How do we classify these overlapping action potentials? Aiming to answer these different interrogations is commonly refered as the **spike sorting** problem. For decades, it has represented an important obstacle between the data acquisition and the interpretation of the results for the field of neuroscience ([Lewicki, 1998]).

An important assumption related to the observed action potentials is based upon the idea that any given neuron generates spikes of consistent shape during the experiment. From a biophysical perspective, the waveform of any action potential essentially depends on the morphology of the cell

which generated it, as well as its position and the filtering characteristics of the milieu ([Hill et al., 2011]). Although reasonable from a biological point of view, we see in section 3.4 that, in practice, this assumption of shape consistency can be disrupted by various factors. Using these consistent shapes of the action potentials as the signature of the electrical activity of each neuron, the spike sorting procedure should identify and associate each observed spike to a specific neuron. As a result, the temporal activity of every neuron can be summarized in a **spike train**, which in turn can be analyzed by the experimenter.

One of the main motivation of our work is to provide a more precise estimation of the spike trains, and more particularly, a better detection of neuronal **synchronizations**. Synchronizations between neurons are presumably a sensible part of the neural code ([Lambert et al., 2018, Albert et al., 2016, Eytan and Marom, 2006]). Recording these synchronizations demands to monitor the activity of a large number of neurons using a precise time resolution. To perform this task, MEA recordings represent the right approach ([Pouzat and Detorakis, 2014]). We describe in the next section how spike sorting is traditionally performed. In particular, we see that it involves a large portion of manual operations. These tasks get even more laborious as the number of synchronizations growths. As such, in the next chapters we will aim to provide a spike sorting methodology which is scalable to the large amounts of data that a MEA can generate. Since the synchronization problem worsens as the number of recording sites increases ([Einevoll et al., 2012]), our approach must also treat synchronizations in a more efficient manner. Nowadays, implantable prostheses comprising hundreds of electrodes can generate massive amounts of data. Ideally, our approach should also be able to sort in real-time such high volumes in order to provide steady neural commands ([Wood et al., 2004]).

3.3 Traditional spike sorting

In the previous section, we have seen that spike sorting is an essential task in order to convert raw extracellular recordings into interpretable neuronal activity. In this section, we describe the traditional spike sorting methodology. Although this problem has been addressed for a long time and a large variety of approaches and refinements exists nowadays, automated solutions somewhat converged to a general procedure to which we will call *traditional* spike sorting, summarized in Figure 3.2. In the literature, the term *clustering* can also be found frequently. It essentially consists of three consecutive steps: spike detection, feature extraction and clustering of spikes ([Lewicki, 1998]).

The goal of the first step, **spike detection**, is to identify on every recording channels the instants when neurons may have fired action potentials. Because action potentials can be observed by a sudden increase of the local potential, under favorable recording circumstances they can be discriminated from the noise level. Therefore, after applying a high-pass filter

in order to remove the low frequency portion of the signal, spike detection is generally treated by voltage **thresholding** detection: each time the voltage exceeds some settled threshold, this instant is registered as a spike. The main benefit of this approach is its simplicity, as it requires little software and hardware efforts. The major downside is the inherent trade-off between false positives, that is when the background noise cross the threshold, and false negatives, that is when some true spikes are below the threshold and are therefore missed. Clearly the higher the signal-to-noise ratio is, the easier this trade-off problem becomes.

For any detected spike in the first step, its waveform can be derived from the signal. Then the purpose of the second step, **feature extraction**, consists in summarizing these waveforms with a set of relevant features. In this way, the dimensional complexity of the data is reduced to its most explanatory parts. First, in order to reduce the variability of the spikes waveforms, they are temporally aligned on a shared characteristic, such as their voltage peak instant. Then, for any recorded spike, a feature vector is computed. One of the first strategy was to quantify some characteristics of the shape, like its peak-to-peak amplitude or its width and height. This approach, although cost-effective in terms of required computations, generally grants lacking spike separation. More advanced solutions, such as wavelets or principal component analysis, take advantage of modern computational ressources and provide sets of features of higher value.

Once the spikes features have been extracted, the last step aims at classifying spikes, using a **clustering** strategy in the feature space. In an ideal situation, each cluster should contain every spike of a single neuron. Then the computation of the average waveform of every spike in each cluster gives a representative spike waveform for each neuron, called the *template* of this neuron. In order to classify spikes which are not yet classified, for instance if the clustering model has been determined on a portion of the whole signal, these templates can be used as a template matching strategy ([Einevoll et al., 2012]). A lot of practitioners achieve the clustering task by hand, using scatter plots of the feature space and delimiting clusters manually, defining regions with straight lines and polygons. Since this procedure might be influenced by subjective considerations and is quite cumbersome, automatic clustering algorithms such as K-means have been suggested in order to lessen these issues ([Harris et al., 2000]). By definition, any spike within the boundaries of a cluster is classified as part of this cluster. On the other hand, if a spike is not in any cluster, it is removed. Therefore, as in the threshold detection step, there is an intrinsic trade-off between false negatives (missed spikes) and false positives, and this trade-off applies to both manual and automatic clustering approaches ([Lewicki, 1998]). Also note that, for many reasons, the number of clusters is not necessarily equal to the number of observed neurons. For instance, two neurons that regularly synchronize so that their respective spikes overlap in the recorded signal might induce a third cluster to appear which is in fact not related to a third neuron.

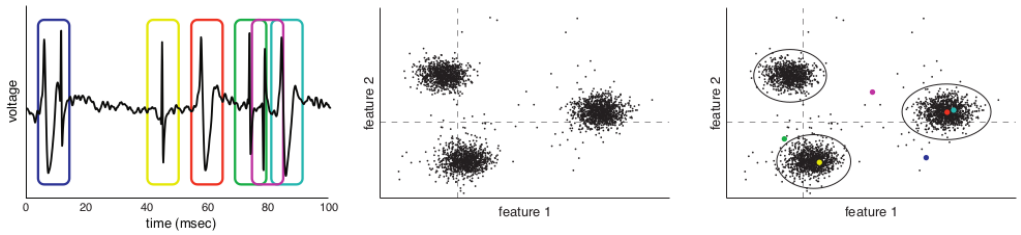


Figure 3.2: Illustration of the three main steps in the traditional spike sorting procedure. First, voltage thresholding is performed in order to detect each spike peak instant. Using a small window around these peak instants, the spike waveforms can be extracted. Then feature extraction is achieved. As an illustration, we can see here the result of the projection of the spikes in the first two principal components. Finally, clustering of the spikes is performed, for instance with the K-means algorithm. Points in color coincide with extracted waveforms in the first step. Remark that some of them do not belong to any cluster since they result from the superposition of individual spikes. Figure 1 from [Ekanadham et al., 2014].

3.4 Spike sorting difficulties

We now discuss more in details the difficulties related to the traditional spike sorting methodology described in the previous section. We have seen that this approach relies greatly on manual operations and subjective decisions. The thresholding step depends notably on the trade-off between false positives and false negatives that the experimenter desires, especially in presence of important noise. One can also be biased by large spikes, neglecting neurons that generate smaller action potentials. The features extraction step is also affected by subjective decisions, for instance the number of features selected after applying a principal component analysis. [Harris et al., 2000] showed that the clustering step could greatly benefit from a semi-automatic classification approach. Indeed, defining manually the clusters in a feature space of high dimension is prone to errors from the human practitioner. Semi-automatic strategies provided results closer to the theoretical optimum. In short, the traditional spike sorting methodology, because of its structure, is subject to error accumulation, time consuming human decisions and performance variability ([Wood et al., 2004]). Moreover, since the whole procedure is often calibrated to fit a certain environment (recording device, nervous tissue or brain area), a specific spike sorting pipeline might work in one setting but fail in others ([Einevoll et al., 2012]).

Another significant difficulty in the traditional approach is related to overlaps between spikes. When two neighboring neurons both fire action potentials in a small time interval, the recorded waveform is the merging of the two spikes. Depending on the delay variability due to the firing patterns of the two neurons, multiple shapes may therefore appear, which all differ from the respective spikes of these neurons. A wide variety of situations may arise from this phenomenon. The two spikes may cancel out, thus not been

detected by the threshold step. On the other hand, a superposition which crosses the threshold may be discarded (false negative) by the classification step, because its waveform is too dissimilar to the waveforms of the clusters. Since such superposition may be the consequence of a functional synchronization between neurons, removing it from the data might cause a loss of valuable information, especially if the goal of the practitioner is to explore a local circuit behavior ([Lewicki, 1998]). With the appearance of large MEA capable of recording large populations of neurons, this issue becomes even more problematic. In order to address this problem, various strategies have been introduced. Most of them are either based on brute-force exploration of every combinations of waveforms (computationally expensive), or greedy iterative subtractions of waveforms until the residual reaches the noise level ([Ekanadham et al., 2014]).

An important assumption of the traditional spike sorting procedure is the stability of the spike waveforms during the experiment. Indeed, it is fundamental for the characterization of the neurons and the discrimination of the spikes into clusters. Unfortunately, this assumption may be violated by certain factors. The first one is the biological phenomenon called *bursting*. A neuron is said to be bursting when it repeatedly generates discrete bursts of action potentials. During these bursting events, the waveform of the action potential often varies. Typically, each successive spike is attenuated, and may even ultimately shrink to the noise level. Some techniques can mitigate this issue, such as the cluster aggregation idea of [Fee et al., 1996]. Another factor which may affect the stability of the clusters is the electrode drift effect ([Bar-Hillel et al., 2006]). After the insertion of the electrode into the neural tissue, it may slowly move to another position because of the reaction of the local environment. As a result, this phenomenon may also slowly modify the waveforms of the recorded spikes. Naturally the drift might become more important as the experiment duration increases. Methods applied to the bursting problem can also be satisfyingly applied to the electrode drift effect.

The last difficulty that we want to briefly mention is related to the comparison of scientific studies based on spike sorting. Indeed, as we have seen, there is no automatic spike sorting reference procedure. This problem is aggravated by the fact that it is difficult to evaluate the performance of a given spike sorting procedure, due to the absence of ground truth data. [Hill et al., 2011] proposed some metrics in order to assess the quality of spike sorting results, without knowing the ground truth. Other approaches aim to compare the results with ground truth data, obtained for instance by simultaneous intra and extracellular recordings, or using numerical simulations mimicking the electrical dynamic of a neural network.

3.5 Sparse convolutional linear model

We present here the mathematical model on which our spike sorting approach is based. As in traditional spike sorting approaches, we aim to

extract from the measured signal the different shapes of the action potentials and the activation times of the neurons. However, in our approach, the measured signal is viewed as the temporal convolution between the shapes and the activation times. In this context, the shapes can be seen as shift-invariant atoms, which can appear at any time in the signal. The activation times of the neurons decide at which instants these atoms appear. Let us first introduce some notations.

We assume that the number of recorded neurons during the experiment is known and we write it N. This population of neurons is recorded by E electrodes. Each of these electrodes records a signal of size T, which is therefore the number of time steps. Let $\boldsymbol{Y}$ the matrix of the observations in $\mathbb{R}^{E\times T}$, which contains the E recorded signals of size T. Then the model, which use in neuroscience dates back to [Roberts, 1979], writes as:

$$\boldsymbol{Y} = \sum_{n=1}^{N} \boldsymbol{W}_n * \boldsymbol{a}_n^\star + \boldsymbol{\Xi}, \tag{3.1}$$

where $\boldsymbol{\Xi}$ is a random noise matrix in $\mathbb{R}^{E\times T}$ and $*$ is the convolution operator along time [1] defined for any two vectors $\boldsymbol{x} \in \mathbb{R}^\ell$ and $\boldsymbol{y} \in \mathbb{R}^T$ as:

$$(\boldsymbol{x} * \boldsymbol{y})_j = \sum_{i=1}^{\min(\ell,j)} x_i \; y_{j-i+1}, \tag{3.2}$$

for any j in $\{1,\dots,T\}$. For any neuron n, the matrix $\boldsymbol{W}_n$ in (3.1) can be described as $\boldsymbol{W}_n = [\boldsymbol{w}_{n,1},\dots,\boldsymbol{w}_{n,E}]^\top \in \mathbb{R}^{E\times\ell}$, where $\boldsymbol{w}_{n,e}$ represents the shape of the action potential of neuron n recorded by electrode e. As we saw in chapter 2, the duration of an action potential is of order some milliseconds. Therefore these shapes $\boldsymbol{w}_{n,e}$ are described by ℓ points, which can be assumed very small with respect to T. Note that this model assumes that the action potentials shapes do not vary during the whole experiment. As we explained in section 3.4, although these shapes essentially depend on the physiological properties of the neurons, which generally remain unchanged throughout the experiment, in practice some factors may induce alteration of the shapes. For instance, the tissues and the recording electrodes may slightly move. Additionally, bursting neurons often see the amplitudes of their action potentials to progressively decrease ([Lewicki, 1998]). Finally for any neuron n, the vector $\boldsymbol{a}_n^\star$ in $\mathbb{R}^T$ is called its activation vector. The non zero entries of $\boldsymbol{a}_n^\star$ correspond to the activation times of this neuron. Remark that, although a reasonable assumption would be that the vectors $\boldsymbol{a}_n^\star$ are binary, the resulting optimization problem would become NP-complete. Moreover, taking their values in $\mathbb{R}$, these activation vectors are able to take into account the change of amplitude of the action potentials, which can happen as stated in section 3.4.

[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`

An illustration of this convolutional model (3.1) is given in figure 3.3 for $E = 2$ electrodes and $N = 2$ neurons. One of the main benefit of this model relies of the linear property of the convolution operator. As we mentioned in section 3.4, if two neighboring cells fire their action potentials at almost the same time, their respective shapes will overlap in the recorded signal. To a first order approximation, we can see this superposition as the summation of the two shapes. Therefore the convolutional model appears better adapted for handling this problem than the traditional spike sorting approaches that we described earlier.

In view of model (3.1), we can see our spike sorting approach as the estimation of the parameters $\boldsymbol{W}_n$ and $\boldsymbol{a}_n^\star$, which is an inverse problem. This estimation task is often stated in the signal processing literature as convolutional dictionary learning. It essentially relies on an optimization problem that seek to determine the *atoms* (here the shapes of the action potentials) and the activations of these atoms (here the activation times of the neurons). This stategy has been successfully applied to a wide range of domains, such as biomedical data ([Pachitariu et al., 2013, Adler et al., 2015, Jas et al., 2017]), audio data ([Mailhé et al., 2008, Grosse et al., 2012]), computer vision ([Kavukcuoglu et al., 2010, Zeiler et al., 2010]), etc.

The convolutional dictionary learning problem usually takes the following generic form:

$$\min_{\boldsymbol{W},\boldsymbol{A}} \left\| \boldsymbol{Y} - \sum_{n=1}^{N} \boldsymbol{W}_n * \boldsymbol{a}_n \right\|^2 + \lambda_A \Omega(\boldsymbol{A}) + \lambda_W \Omega(\boldsymbol{W}). \tag{3.3}$$

The terms $\boldsymbol{A}$ and $\boldsymbol{W}$ are tensors of appropriate sizes which respectively gather the activation vectors $\boldsymbol{a}_n$ and the shapes matrices $\boldsymbol{W}_n$. $\Omega(\boldsymbol{A})$ and $\Omega(\boldsymbol{W})$ are penalization terms which promote the sparsity of $\boldsymbol{A}$ and $\boldsymbol{W}$, proportionately to the regularization parameters $\lambda_A \geq 0$ and $\lambda_W \geq 0$.

Let us explain why it is reasonable to look for sparse solutions in $\boldsymbol{A}$ and $\boldsymbol{W}$. We give in Table 3.1 the typical orders of magnitude of the different parameters of the model. Remark that the temporal quantities notably depends on the sampling recording rate used during the experiment, which is here equal to 30kHz. Therefore a recording session of one hour generates signals of size $T = 10^8$. In this setting, for a typical neuron spiking rate of 30Hz, one can expect that the number of non zero coordinates in each activation vector $\boldsymbol{a}_n^\star$ to be about 10^5, which is very small with respect to their size of 10^8. Similarly, we can expect the matrices $\boldsymbol{W}_n$ to be sparse, because the activity of each neuron is only recorded by its neighboring electrodes, which should be very few with respect to the total number of electrodes E, especially for a large recording device such as a MEA.

Let us assume that the penalization terms in (3.3) are convex. Then the whole objective function is convex in both $\boldsymbol{W}$ and $\boldsymbol{A}$, but not jointly convex. This type of problem can be solved using an alterative minimization strategy, consisting in alternatively minimizing a variable while keeping the other one fixed. The step which focuses on the estimation of the activations

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 3.1: 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. The number of electrodes and neurons may greatly variate, depending on the context: a single recording tetrode only comprises four electrodes, while a MEA can contain several thousands of electrodes. The number of recorded neurons ensues from this recording context.

$\boldsymbol{A}$, called convolutional sparse coding, is the hardest part of the optimization problem, especially for large data sets. Indeed, due to the orders of magnitude presented in table 3.1, we can see that the number of variables in the activation vectors growths linearly with T. On the other hand, the parameter ℓ controling the size of the shapes in $\boldsymbol{W}$ is fixed and small. For this reason, we decide to focus on the estimation of the activations while the shapes are assumed to be known and stay fixed. Consequently the initial optimization problem reduces to:

$$\min_{\boldsymbol{A}} \left\| \boldsymbol{Y} - \sum_{n=1}^{N} \boldsymbol{W}_n * \boldsymbol{a}_n \right\|^2 + \lambda_A \Omega(\boldsymbol{A}). \tag{3.4}$$

The action potential shapes can be estimated by using the results of a traditional spike sorting approach. For instance, one can use the centroids of the clusters determined by the K-means clustering (see section 3.3). In particular, we also get the number of neurons N that was assumed to be known. Our approach therefore lies between two strategies, the first one learning the dictionary from the data ([Mairal et al., 2010]), the second one fixing the dictionary, for instance with wavelets ([Mallat, 1999]). Building our own dictionary offers a better signal reconstruction than generic bases (*e.g.* wavelets, Fourier), which are restricted to certain frequency bands.

Our estimation of the activations allows to take care of the problem of synchronizations more properly. Besides, spikes which were not sorted by the first algorithm may now be correctly treated. Note also that the biological signal that we are modeling here, is really small so that there is no problem in considering an additive superposition of the shapes since it can never reach the saturation of the device. In the next chapter, we specify the problem (3.4), by defining the matricial norm $\|.\|$ and the penalization term $\Omega(\boldsymbol{A})$ used. Then we describe some state of the art algorithms for the resolution of such problem.

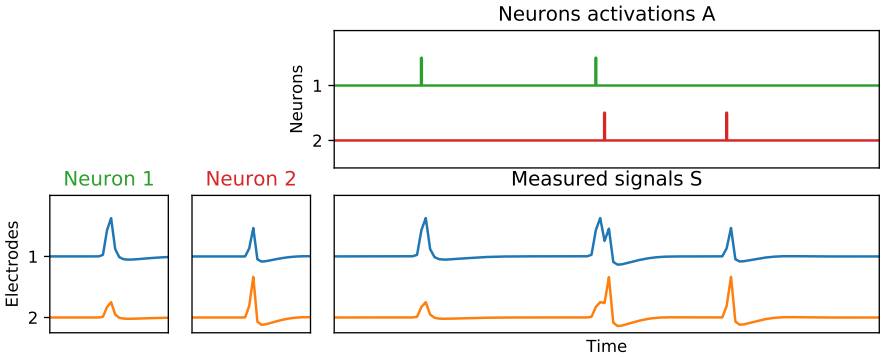


Figure 3.3: Illustration of the convolutional model in a simple context of $E = 2$ electrodes and $N = 2$ neurons. The shapes of the action potentials are represented in the bottom left corner. The activations vectors are represented in the top right corner, and the result of the temporal convolution between the shapes and the activations is represented in the bottom right corner. For each activation time (depicted as a vertical trait), the corresponding shape appears at the same instant on the convolution. Remark the unusual shapes appearing around the center of the graph. This phenomenon may occur when two neurons activate at almost the same time (synchronization). This causes a superposition of the action potentials that creates waveforms which are not in the model.

Chapter 4

Optimization methods

In this chapter, we state precisely the optimization problem that we aim to solve in order to determine the activations of the neurons. We begin by specifying problem 3.4 as a Lasso problem, then we motivate the choice of the Lasso estimator. The rest of the chapter is devoted to the presentation of some useful mathematical and algorithmic results which will be of great importance for solving the Lasso problem in high dimension, consequently for the implementation of our spike sorting algorithm.

4.1 The Lasso problem

4.1.1 Choice of the Lasso estimator

Keeping the notations introduced in section 3.5, we now state precisely our optimization problem for the estimation of the activations. In problem (3.4), we set the matricial norm $\|.\|$ as the Frobenius norm, written $\|.\|_2$, and defined as $\|\boldsymbol{S}\|_2 = \left(\sum_{i,j} S_{ij}^2\right)^{\frac{1}{2}}$. Then for the penalization term $\Omega(\boldsymbol{A})$ on the activations, we choose the sum of the ℓ_1 norm of the activation vectors, defined as $\|\boldsymbol{u}\|_1 = \sum_i |u_i|$. Using a renormalizing factor which will simplify ulterior computations, we get the following optimization problem:

$$\min_{\boldsymbol{A}} \quad \left\| \boldsymbol{Y} - \sum_{n=1}^{N} \boldsymbol{W}_n * \boldsymbol{a}_n \right\|_2^2 + 2\lambda \sum_{n=1}^{N} \|\boldsymbol{a}_n\|_1. \tag{4.1}$$

This problem corresponds to a convolutional formulation of the well-known *Lasso* (Least Absolute Shrinkage and Selection Operator) problem, proposed by [Tibshirani, 1996]. Remark that the only tuning parameter of this method is the regularization parameter $\lambda > 0$, which promotes the sparsity of the activation vectors. In practice, the value of λ can be calibrated according to the signal-to-noise ratio. In the following, we briefly describe the main motivations for the developement of the Lasso, and why it is relevant to the problem of convolutional sparse coding. The Lasso problem is more generally formulated in the context of the classical linear regression model, which we can write as follows:

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}, \tag{4.2}$$

with $\boldsymbol{Y}$ in $\mathbb{R}^n$ the endogenous variable, $\boldsymbol{X}$ in $\mathbb{R}^{n\times d}$ the exogenous variables, and $\boldsymbol{\epsilon}$ in $\mathbb{R}^n$ the error term. Then a classical estimator for the parameter $\boldsymbol{\beta}$ is the least-square estimator, defined as:

$$\hat{\boldsymbol{\beta}}_{LS} = \underset{\boldsymbol{\beta}\in\mathbb{R}^d}{\operatorname{argmin}} \|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 . \tag{4.3}$$

Let us assume that $d \leq n$ and that the design matrix $\boldsymbol{X}$ has full rank d. Then the least-square estimator is unique and equals $\hat{\boldsymbol{\beta}}_{LS} = (\boldsymbol{X}^\top\boldsymbol{X})^{-1}\boldsymbol{X}^\top\boldsymbol{Y}$. The original motivation of [Tibshirani, 1996] for introducing the Lasso problem:

$$\hat{\boldsymbol{\beta}}_{lasso} = \underset{\boldsymbol{\beta}\in\mathbb{R}^d}{\operatorname{argmin}} \|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_1 , \quad \lambda > 0, \tag{4.4}$$

was related to several limitations of the least-square estimator $\hat{\boldsymbol{\beta}}_{LS}$. First, the least-square estimator often exhibits low bias but high variance. In other words, its prediction accuracy can be unsatisfying. The second and main motivation was linked to the problem of variable selection: a large number of predictors in $\hat{\boldsymbol{\beta}}_{LS}$ can make the interpretation of the results difficult. The variable selection stategy aims at selecting a smaller subset of predictors, which have the largest influences on the response $\boldsymbol{Y}$. At that time, two different stategies were used to perform this task: subset selection and ridge regression.

The goal of subset selection is to find the subset of variables of a given size which minimizes the squared residual. Unfortunately, this problem is NP-complete ([Welch, 1982]). Other approaches such as forward selection and backward elimination, which respectively adds and removes variables in a sequential manner, can be computationally fast, but their greedy essence may produce unstable and suboptimal results ([Breiman, 1996]). By adding $\lambda \|\boldsymbol{\beta}\|_2^2$ as a penalty term in (4.3), the ridge regression tends to shrink the coefficients of the estimator, but does not set any of them to zero. Therefore the resulting model is still not straightforward to interpret.

Another severe limitation of the least-square estimator appears when working on a model with more unknown parameters than observations ($d > n$). In this case, $\hat{\boldsymbol{\beta}}_{LS}$ is not unique and tends to highly overfit the data. The ideas of sparsity naturally emerged in order to tackle these different mathematical and computational problems, enabling statistical inference for high-dimensional complex data. For geometrical reasons, and contrarily to the ridge, the ℓ_1-norm tend to produce coordinates estimations exactly equal to zero, thus allowing the Lasso estimator to properly perform variable selection ([Bühlmann and Van De Geer, 2011, Chapter 2]). More generally, the Lasso gained a large popularity in a wide range of applications, thanks to its statistical precision and its computational feasibility. As such, it frequently appears in the convolutional dictionary learning litterature as an estimator of choice (see the references in section 3.5). Also, it

has already been proposed in the context of spike sorting, for instance by [Ekanadham et al., 2011].

4.1.2 Vectorization of the convolutional model

Using the linearity of the convolution operator, we can reformulate the convolutional Lasso (4.1) in its original linear form (4.4), that is to say for a linear regression problem. Let us introduce some notations in order to perform this vectorization step. We define the vectorization as the concatenation of the temporal signals, *i.e.* the lines of the multivariate signals, with:

$$\begin{aligned}\boldsymbol{y} &= (Y_{1,1}, Y_{1,2}, \dots, Y_{E,T-1}, Y_{E,T})^\top,\\ \boldsymbol{\xi} &= (\Xi_{1,1}, \Xi_{1,2}, \dots, \Xi_{E,T-1}, \Xi_{E,T})^\top,\\ \boldsymbol{a}^\star &= (A^\star_{1,1}, A^\star_{1,2}, \dots, A^\star_{N,T-1}, A^\star_{N,T})^\top = (\boldsymbol{a}_1^{\star\top}, \dots, \boldsymbol{a}_N^{\star\top})^\top,\end{aligned}$$

thus $\boldsymbol{y}$, $\boldsymbol{\xi}$ and $\boldsymbol{a}^\star$ are respectively the vectorization of the recorded signals in $\boldsymbol{Y}$, the noise signals in $\boldsymbol{\Xi}$ and the activations in $\boldsymbol{A}^\star$. Note that in the following we will sometimes index vectors $\boldsymbol{a}$ with double indices $a_{n,t} = A_{n,t}$, for readability reason. The linear convolutional model in (3.1) can be expressed in a vectorized format as:

$$\boldsymbol{y} = \boldsymbol{H}\boldsymbol{a}^\star + \boldsymbol{\xi}, \tag{4.5}$$

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

For a given neuron n and a given electrode e, the corresponding block of size $T \times T$ which appears in the columns $(\boldsymbol{h}_{n,1}, \dots, \boldsymbol{h}_{n,T})$ of this neuron has the following form:

$$\begin{pmatrix} w_{n,e,1} & 0 & \dots & \dots & \dots & 0 \\ \vdots & w_{n,e,1} & \ddots & & & \vdots \\ w_{n,e,\ell} & \vdots & \ddots & \ddots & & \vdots \\ 0 & w_{n,e,\ell} & & w_{n,e,1} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots & \ddots & 0 \\ 0 & \dots & 0 & w_{n,e,\ell} & \dots & w_{n,e,1} \end{pmatrix} \in \mathbb{R}^{T \times T}. \tag{4.6}$$

This particular form of matrix is called a Toeplitz matrix: it notably allows to write the temporal convolution between the shapes and the activations as a matrix-vector product. Since the structure of this block is the generic

form of the neuron-electrodes blocks which appear in $\boldsymbol{H}$, the design matrix of the vectorized model (4.5) is a very structured block Toeplitz matrix. Let us recall that the length of the shapes ℓ is expected to be much smaller than T, thus $\boldsymbol{H}$ is also a very sparse matrix.

Although this form of Lasso problem can be formulated on a wide range of situations, the dimensionality growth that we observe after the vectorization step makes the computations of a solution extremely expensive. In the following, we will explain how we can use structural properties of our problem in order to solve it efficiently, despite its large dimension.

4.1.3 Optimality conditions of the Lasso

We conclude this section by giving an important characterization of the Lasso solutions. This property plays a central role in our strategy for an efficient resolution of our Lasso problem:

$$\underset{\boldsymbol{a}\in\mathbb{R}^{NT}}{\operatorname{argmin}} \|\boldsymbol{y}-\boldsymbol{H}\boldsymbol{a}\|_2^2 + 2\lambda \|\boldsymbol{a}\|_1 , \tag{4.7}$$

with $\lambda > 0$. First, let us introduce the notion of subdifferential of a continous convex function.

Definition 4.1.1. *Let g a continuous convex function from $\mathbb{R}^d$ to $\mathbb{R}$. For any $\boldsymbol{x}$ in $\mathbb{R}^d$, the subdifferential of g in $\boldsymbol{x}$ is defined as:*

$$\partial g(\boldsymbol{x}) = \left\{ \boldsymbol{u} \in \mathbb{R}^d \ : \ \forall \boldsymbol{v} \in \mathbb{R}^d \ \boldsymbol{u}^{\boldsymbol{T}}(\boldsymbol{v}-\boldsymbol{x}) + g(\boldsymbol{x}) \leq g(\boldsymbol{v}) \right\}. \tag{4.8}$$

The vectors $\boldsymbol{u}$ in $\partial g(\boldsymbol{x})$ are called subgradients, in the sense that these vectors provide a lower bound of function g in the form of tangent hyperplanes at point $\boldsymbol{x}$. Note that in the case where g is also differentiable on $\boldsymbol{x}$ in $\mathbb{R}^d$, its subdifferential reduces to $\partial g(\boldsymbol{x}) = \{\nabla g(\boldsymbol{x})\}$.

By definition of the subgradient, any $\boldsymbol{x}$ in $\mathbb{R}^d$ minimizes the function g if and only if $\mathbf{0} \in \partial g(\boldsymbol{x})$. For any $\boldsymbol{a}$ in $\mathbb{R}^{NT}$, the first term in (4.7) is convex, differentiable, and its gradient is $-2\boldsymbol{H}^\top(\boldsymbol{y}-\boldsymbol{H}\boldsymbol{a})$. Therefore for any $\boldsymbol{a}$ in $\mathbb{R}^{NT}$, since the subdifferential of the sum is the sum of the subdifferentials (Moreau-Rockafellar theorem), we have that the subgradient of the cost function in $\boldsymbol{a}$ is $\{-2\boldsymbol{H}^\top(\boldsymbol{y}-\boldsymbol{H}\boldsymbol{a})\} + 2\lambda\partial \|.\|_1 (\boldsymbol{a})$. Consequently, any $\hat{\boldsymbol{a}}$ in $\mathbb{R}^{NT}$ is a solution of (4.7) if and only if:

$$\begin{aligned} &\mathbf{0} \in \{-2\boldsymbol{H}^\top(\boldsymbol{y}-\boldsymbol{H}\hat{\boldsymbol{a}})\} + 2\lambda\partial \|.\|_1 (\hat{\boldsymbol{a}}) \\ &\Leftrightarrow \{\boldsymbol{H}^\top(\boldsymbol{y}-\boldsymbol{H}\hat{\boldsymbol{a}})\} \in \lambda\partial \|.\|_1 (\hat{\boldsymbol{a}}). \end{aligned}$$

We can show that for any $\boldsymbol{x}$ in $\mathbb{R}^d$, the subgradient of the ℓ_1-norm takes the following form:

$$\partial \|.\|_1 (\boldsymbol{x}) = A_1 \times \cdots \times A_d, \quad \text{with } A_j = \begin{cases} \{\operatorname{sign}(x_j)\} & \text{, if } x_j \neq 0, \\ [-1,1] & \text{, if } x_j = 0. \end{cases} \tag{4.9}$$

This grants us the following optimality conditions for the Lasso. Any $\hat{\boldsymbol{a}}$ in $\mathbb{R}^{NT}$ is a Lasso solution, that is a solution of problem (4.7), if and only if:

$$\begin{cases} \boldsymbol{h}_{n,t}^\top(\boldsymbol{y} - \boldsymbol{H}\hat{\boldsymbol{a}}) = \lambda \operatorname{sign}(\hat{a}_{n,t}) & \text{, if } \hat{a}_{n,t} \neq 0, \\ |\boldsymbol{h}_{n,t}^\top(\boldsymbol{y} - \boldsymbol{H}\hat{\boldsymbol{a}})| \leq \lambda & \text{, if } \hat{a}_{n,t} = 0. \end{cases} \tag{4.10}$$

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

$$\text{if} \quad |\boldsymbol{h}_{n,t}^\top(\boldsymbol{y} - \boldsymbol{H}\hat{\boldsymbol{a}})| < \lambda, \quad \text{then} \quad \hat{a}_{n,t} = 0. \tag{4.11}$$

Condition (4.11) is simple to test and really useful in a context of high sparsity where the majority of coordinates in $\hat{\boldsymbol{a}}$ 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 $\hat{\boldsymbol{a}}$. This strategy, called working set, is presented in more details in the next section 4.2.

4.2 Generic working set algorithm for the Lasso

Although efficient algorithms (presented in the remaining parts of this chapter) have been developed for solving the Lasso, the large size of our problem makes the computation of a solution prohibitive. In this section, we present the stategy named working set, also known as active set, which makes notably use of the sparsity of the problem in order to compute a solution more efficiently. This strategy has been used for several problems such as the Lasso ([Lee et al., 2007]), the group-Lasso ([Roth and Fischer, 2008]), as well as in different contexts such as kernel learning ([Bach, 2008]), variable selection ([Obozinski et al., 2010]), or detection of objects in images ([Boisbunon et al., 2014]).

4.2.1 Principle of the algorithm

The main idea of the working set algorithm is, starting from the null vector, to sequentially activate the coordinates of the solution, using the optimality condition (4.11) as a criterion and solve the associated subproblems. Since it uses an inner solver for the computation of the solutions, the working set strategy is in fact a meta-algorithm.

Let us describe it in the case of the Lasso problem (4.7). We note J the set of the active coordinates in $\{1, \ldots, NT\}$ of the solution. 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 optimality condition (4.11) (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 using the columns of $\boldsymbol{H}$ in J, noted

$\boldsymbol{H}_J$ (line 6). Note that we can use the previous $\hat{\boldsymbol{a}}_J$, augmented by the new coordinate j_0, as a warm start for the resolution of the new subproblem.

Thanks to the sparsity of $\boldsymbol{a}^*$, we expect to solve several Lasso problems of size $\leq |J|$, which remains small with respect to NT. Although other strategies for the evolution of J exist, for instance adding multiple coordinates at once, or removing some coordinates ([Bach et al., 2011]), adding the coordinate which violates the most the optimality condition, in other words which is the most correlated with the current residual, provides a simple and efficient approach. In particular, this ensures that the working set algorithm ends in finite time, since in the worst possible case, every coordinates would be activated.

4.2.2 Efficient implementation on convolutional models

One simple approach to solve problem (4.1) would be to pre-compute the matrix $\boldsymbol{H}$ and use it for the optimality condition at line 3 and for the Lasso solver at line 6 of Algorithm 1, which are the most expensive steps of the algorithm. But this approach does not scale properly in memory since the memory complexity of storing $\boldsymbol{H}$ is $O(ENT^2)$, with T typically very large. The computational complexity of line 3 for a dense matrix $\boldsymbol{H}$ is also $O(ENT^2)$ which does not scale well with the problem dimensionality.

But in practice $\boldsymbol{H}$ is very sparse, completely defined through $\boldsymbol{W}$, 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 optimality condition and to compute the gradient in the inner Lasso solver at line 6. Also note that in addition to the efficient implementation for the convolution operator, the matrix $\boldsymbol{H}_J$ 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 $O(E|J|\ell \times |J|)$.

The efficient implementation discussed above allows to solve larger problems but several computational bottleneck persists: at each iteration of the working set, one needs to perform an $O(ENT\ell)$ 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. Also note that using the Fast Fourier Transform, as suggested in [Wohlberg, 2015] for convolutional sparse coding, only reduces the computation cost of the optimality conditions to $O(ENT\log_2(T))$ which is still problematic. We propose in chapter 5 the idea of sliding window working set, which takes advantage of the temporal structure of the problem to solve it more efficiently.

Algorithm 1 Generic working set algorithm for the Lasso

Require: $\boldsymbol{y}, \boldsymbol{H}, \lambda > 0, \epsilon > 0$
1: $J \leftarrow \emptyset$, $\hat{\boldsymbol{a}} \leftarrow \boldsymbol{0}$
2: **repeat**
3: $\quad \boldsymbol{g} \leftarrow \boldsymbol{H}^\top(\boldsymbol{y} - \boldsymbol{H}\hat{\boldsymbol{a}})$
4: $\quad j_0 \leftarrow \operatorname{argmax}_{l \in J^c} |g_l|$
5: $\quad J \leftarrow J \cup \{j_0\}$
6: $\quad \hat{\boldsymbol{a}}_J \leftarrow$ Solve Lasso (4.7) for sub-problem $(\boldsymbol{H}_J, \boldsymbol{y})$
7: **until** $g_{j_0} < \lambda + \epsilon$ **return** $\hat{\boldsymbol{a}}$, J

4.3 Proximal optimization methods

We have seen in section 4.2 that the working set algorithm repeatedly makes use of a Lasso solver in order to compute efficiently a global solution. In this section, we present some mathematical results related to proximal optimization. These ideas are essential for the Lasso solver, named FISTA, that we chose to use in practice. We notably state interesting results in terms of convergence of this algorithm. We will see in the next section that some of these ideas also apply to other Lasso solvers. This presentation of the proximal optimization methods essentially follows those in [Combettes and Pesquet, 2011].

4.3.1 Proximity operator

Let us begin by introducing the idea of proximity operator of a convex function. This mathematical tool introduced by [Moreau, 1962] naturally extends the notion of projection operator on a convex set. It plays a major role in the field of convex optimization problems, and consequently, in the study of inverse problems, for instance for the signal processing community.

The optimization problem that we consider in this section takes the following general form:

$$\operatorname*{argmin}_{\boldsymbol{x} \in \mathbb{R}^d} F(\boldsymbol{x}), \tag{4.12}$$

with $F(\boldsymbol{x}) = f(\boldsymbol{x}) + g(\boldsymbol{x})$. We also make the following assumptions:

- $f : \mathbb{R}^d \to \mathbb{R}$ is a continuously differentiable convex function, with L_f-Lipschitz continous gradient, ie for any $\boldsymbol{x}$ and $\boldsymbol{y}$ in $\mathbb{R}^d$

$$\|\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y})\|_2 \leq L_f \|\boldsymbol{x} - \boldsymbol{y}\|_2, \tag{4.13}$$

  where L_f is the Lipschitz constant of ∇f, the gradient of f.

- $g : \mathbb{R}^d \to \mathbb{R}$ is a continuous convex function, non necessarily smooth.

- There exists a solution to problem (4.12). Because F is a continuous function on $\mathbb{R}^d$, a sufficient condition for the existence of such solution would be for instance F coercive, ie $\lim_{\|\boldsymbol{x}\|_2 \to +\infty} F(\boldsymbol{x}) = +\infty$.

The methods and results that we give here can be stated for more general spaces than $\mathbb{R}^d$, and for larger classes of functions than those described in the previous assumptions. We choose to avoid technical difficulties to better focus on the main ideas.

The presence of a non differentiable function g in problem (4.12) may prevent the application of standard smooth optimization techniques. This typically happens for problems involving a sparsity penalty term, such as the Lasso. We now introduce the notion of proximity operator of a convex function that is of great importance in such scenarios.

Definition 4.3.1. *Let g a continuous convex function from $\mathbb{R}^d$ to $\mathbb{R}$. For any $\boldsymbol{x}$ in $\mathbb{R}^d$, the following minimization problem:*

$$\underset{\boldsymbol{y}\in\mathbb{R}^d}{\operatorname{argmin}}\, g(\boldsymbol{y}) + \frac{1}{2}\left\|\boldsymbol{x}-\boldsymbol{y}\right\|_2^2, \tag{4.14}$$

admits an unique solution on $\mathbb{R}^d$. We note this solution $prox_g(\boldsymbol{x})$. Thus we can define the proximity operator of g as:

$$\begin{array}{rccc} prox_g: & \mathbb{R}^d & \longrightarrow & \mathbb{R}^d \\ & \boldsymbol{x} & \longmapsto & prox_g(\boldsymbol{x}). \end{array} \tag{4.15}$$

Let C be a convex subset of $\mathbb{R}^d$. If we set $g = 1_C$, with $1_C(\boldsymbol{x}) = 0$ for any $\boldsymbol{x}$ in C, and $1_C(\boldsymbol{x}) = +\infty$ for any $\boldsymbol{x}$ in $\mathbb{R}^d \setminus C$, then problem (4.14) corresponds to the problem of projection on the convex set C of $\mathbb{R}^d$. Therefore the proximity operator defined above can be seen as a generalization of the projection operator on a convex set.

In order to characterize this solution $\operatorname{prox}_g(\boldsymbol{x})$, we use once more the results related to the subdifferential of a function, introduced in subsection 4.1.3. For any $\boldsymbol{x}$ in $\mathbb{R}^d$, the second term in (4.14) is convex and differentiable, therefore the subgradient at any $\boldsymbol{y}$ in $\mathbb{R}^d$ of the whole quantity to minimize is $\{\boldsymbol{y}-\boldsymbol{x}\}+\partial g(\boldsymbol{y})$. This implies for the proximity operator of g the following characterization:

$$\forall(\boldsymbol{x},\boldsymbol{p}) \in \mathbb{R}^d \times \mathbb{R}^d, \quad \boldsymbol{p} = \operatorname{prox}_g(\boldsymbol{x}) \Leftrightarrow \boldsymbol{x}-\boldsymbol{p} \in \partial g(\boldsymbol{p}). \tag{4.16}$$

4.3.2 Proximal algorithm

Having introduced the necessary results related to proximity operators, we can approach the resolution of the general optimization problem (4.12) as follows: first, we give a characterization of any solution of (4.12) in terms of proximity operators. We then deduce a general algorithm for the practical computation of such solution. This algorithm belongs to the group of methods called proximal methods. This name derives from the proximity operator of the non smooth functions involved in the optimization problem.

It can be shown that a large class of well known algorithms such as the projected gradient or the alternating-direction method of multipliers are

particular cases of proximal algorithms ([Combettes and Pesquet, 2011]). Therefore this proximal formalism grants a general structure for the study of a large number of convex optimization algorithms. We now give the main result from which we later deduce the FISTA algorithm.

Proposition 4.3.2. *Let f and g functions satisfying the assumptions in section 4.3.1. Then $\boldsymbol{x}$ in $\mathbb{R}^d$ is a solution of problem* (4.12) *if and only if for any positive number γ, we have:*

$$\boldsymbol{x} = prox_{\gamma g}\left(\boldsymbol{x} - \gamma \nabla f(\boldsymbol{x})\right). \tag{4.17}$$

Proof. $F = f + g$ is a continuous convex function, therefore for any $\boldsymbol{x}$ in $\mathbb{R}^d$ we have

$$\begin{aligned}
\boldsymbol{x} = \underset{\boldsymbol{y}\in\mathbb{R}^d}{\operatorname{argmin}} F(\boldsymbol{y}) &\Leftrightarrow \mathbf{0} \in \partial F(\boldsymbol{x}) \\
&\Leftrightarrow \mathbf{0} \in \partial f(\boldsymbol{x}) + \partial g(\boldsymbol{x}) \\
&\Leftrightarrow \mathbf{0} \in \{\nabla f(\boldsymbol{x})\} + \partial g(\boldsymbol{x}) \\
&\Leftrightarrow -\nabla f(\boldsymbol{x}) \in \partial g(\boldsymbol{x}) \\
&\Leftrightarrow (\boldsymbol{x} - \gamma \nabla f(\boldsymbol{x})) - \boldsymbol{x} \in \gamma \partial g(\boldsymbol{x}) \\
&\Leftrightarrow \boldsymbol{x} = \operatorname{prox}_{\gamma g}\left(\boldsymbol{x} - \gamma \nabla f(\boldsymbol{x})\right).
\end{aligned}$$

Aside from trivial operations on subgradients, we used the fact that the subgradient of a differentiable function only contains its gradient on the third line, and the proximity operator characterization (4.16) on the last line. □

Then proposition 4.3.2 allows us to derive an algorithm for the computation of a minimizer of function F, in the form of a fixed point iterative scheme:

$$\boldsymbol{x^{(k+1)}} = \operatorname{prox}_{\gamma_k g}\left(\boldsymbol{x^{(k)}} - \gamma_k \nabla f(\boldsymbol{x^{(k)}})\right), \tag{4.18}$$

with the step size γ_k in an appropriate bounded interval. Inspired by the numerical analysis terminology used for discretization schemes, the algorithm (4.18) is called a forward–backward splitting algorithm. Indeed it relies on a forward (explicit) gradient step on function f, then on a backward (implicit) step on function g.

Starting from an initial point $\boldsymbol{x^{(0)}}$ in $\mathbb{R}^d$ and iterating (4.18) with a suitable step size γ_k provides a proximal method called iterative soft-thresholding algorithm (ISTA). The main advantage of ISTA is its simplicity. Strong convergence of the iterates to a solution can also be shown. Nevertheless, ISTA can be a slow algorithm. Although [Bredies and Lorenz, 2008] showed that ISTA converges linearly in almost every case, they also demonstrated that under some assumptions, the convergence can be arbitrarily slow.

Let us set $\boldsymbol{x^*}$ a minimizer of function F. [Beck and Teboulle, 2009] also showed that ISTA produces a slow convergence of the iterates $F(\boldsymbol{x^{(k)}})$

to $F(\boldsymbol{x}^*)$, but not worse than $O(C/k)$, where C is a constant depending on L_f and $\left\|\boldsymbol{x}^* - \boldsymbol{x}^{(0)}\right\|_2^2$. In other words, in order to reach a prescribed precision $\epsilon > 0$, ISTA would need at most C/ϵ steps. Several researchers attempted to design an accelerated version of ISTA, for instance [Bioucas-Dias and Figueiredo, 2007] with a two-step ISTA named TWIST, as well as [Nesterov, 2013] with an algorithm which gather the past iterates. In our spike sorting approach, we chose to use an improvement of ISTA proposed by [Beck and Teboulle, 2009], named FISTA. Smartly choosing a point as a linear combination of the previous iterates, they managed to obtain a better rate of convergence of $F(\boldsymbol{x}^{(k)})$ to $F(\boldsymbol{x}^*)$. We give a detailed description of FISTA in Algorithm 2.

Algorithm 2 Fast iterative soft-thresholding algorithm (FISTA)

Require: $\boldsymbol{x}^{(0)} \in \mathbb{R}^d$.
1: $\boldsymbol{z}^{(0)} \leftarrow \boldsymbol{x}^{(0)}$, $t_0 \leftarrow 1$
2: **for** $k \in \{1, \dots, K\}$ **do**
3: $\quad \boldsymbol{y}^{(k)} \leftarrow \boldsymbol{z}^{(k)} - L_f^{-1} \nabla f(\boldsymbol{z}^{(k)})$
4: $\quad \boldsymbol{x}^{(k+1)} \leftarrow \mathrm{prox}_{L_f^{-1} g}\left(\boldsymbol{y}^{(k)}\right)$
5: $\quad t_{k+1} \leftarrow \frac{1+\sqrt{1+4t_k^2}}{2}$
6: $\quad \boldsymbol{z}^{(k+1)} \leftarrow \boldsymbol{x}^{(k+1)} + \frac{t_k - 1}{t_{k+1}}(\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)})$
7: **end for**

Note that we set the step size $\gamma_k = L_f^{-1}$ in Algorithm 2. Unfortunately, the Lipschitz constant L_f of ∇f is not always accessible. In such situations, the sequence γ_k can be specified with a backtracking strategy, without changing the later results. Then we give the main result from [Beck and Teboulle, 2009], which grants the convergence rate of the $F(\boldsymbol{x}^{(k)})$ generated by FISTA.

Theorem 4.3.3. *Let $(\boldsymbol{x}^{(k)})_{k\in\mathbb{N}}$ the sequence generated by FISTA and let $\boldsymbol{x}^*$ a minimizer of function F. Then we have for all $k > 0$*

$$F(\boldsymbol{x}^{(k)}) - F(\boldsymbol{x}^*) \leq \frac{2L_f \left\|\boldsymbol{x}^{(0)} - \boldsymbol{x}^*\right\|_2^2}{(k+1)^2}. \tag{4.19}$$

Theorem 4.3.3 demonstrates the convergence rate enhancement, now of order at worst $1/k^2$. Also note that the additional operations, that is the computations of t_k and $\boldsymbol{z}^{(k)}$, increase marginally the overall computational complexity. Indeed, the essential computational cost of FISTA is due to the gradient and proximal steps.

Although the strong convergence of FISTA has not been established yet, [Chambolle and Dossal, 2015] demonstrated more recently that a slight modification of FISTA produces a sequence $(\boldsymbol{x}^{(k)})_{k\in\mathbb{N}}$ which weakly converges to a minimizer $\boldsymbol{x}^*$, in other terms, for any $\boldsymbol{v} \in \mathbb{R}^d$, the quantity $< \boldsymbol{x}^{(k)}, \boldsymbol{v} >$ converges to $< \boldsymbol{x}^*, \boldsymbol{v} >$ for $k \to +\infty$.

4.4 State of the art Lasso solvers

In this section, we present three important state of the art algorithms for the resolution of the Lasso. First, we apply FISTA (section 4.3) to the Lasso, in particular we see how the proximity operator can be computed for this problem. Then we present the Least Angle Regression (LARS) and the coordinate descent algorithms, and explain why we chose to use FISTA as the inner solver of the working set.

4.4.1 FISTA for the Lasso

In this section, we explain how proximal methods can apply for the practical computation of the Lasso. First, let us recall the general form of the Lasso, as we introduced it in 4.1.1.

$$\hat{\boldsymbol{\beta}}_{lasso} = \underset{\boldsymbol{\beta}\in\mathbb{R}^d}{\operatorname{argmin}} \|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_1 , \quad \lambda > 0. \tag{4.20}$$

First, let us remark that problem (4.20) is a particular instance of the more general problem that we introduced in (4.12). Indeed, we have $F(\boldsymbol{\beta}) = f(\boldsymbol{\beta}) + g(\boldsymbol{\beta})$, with $f(\boldsymbol{\beta}) = \|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2$ and $g(\boldsymbol{\beta}) = \lambda \|\boldsymbol{\beta}\|_1$. These two functions satisfy the assumptions in 4.3.1. Therefore it is reasonable to try to apply FISTA for the computation of a solution $\hat{\boldsymbol{\beta}}_{lasso}$. Recall that the backward step in Algorithm 2 is based upon the computation of the proximity operator of $L_f^{-1}g$, and is the main difficulty of this algorithm. For the Lasso, we can give a simple expression of this proximity operator.

This problem takes the following form. For any $\boldsymbol{\beta^0}$ in $\mathbb{R}^d$, solve:

$$\underset{\boldsymbol{\beta}\in\mathbb{R}^d}{\operatorname{argmin}} L_f^{-1}\lambda \|\boldsymbol{\beta}\|_1 + \frac{1}{2}\left\|\boldsymbol{\beta^0} - \boldsymbol{\beta}\right\|_2^2. \tag{4.21}$$

Let $\alpha = L_f^{-1}\lambda > 0$. An interesting property of this problem relies on the fact that it can be separated into independent scalar problems, by definition of the ℓ_1 and ℓ_2 norms. Thus, for any i in $\{1, \dots, d\}$, we consider the following problem:

$$\underset{\beta_i\in\mathbb{R}}{\operatorname{argmin}} \alpha|\beta_i| + \frac{1}{2}(\beta_i^0 - \beta_i)^2. \tag{4.22}$$

Therefore, using the proximity operator characterization (4.16) on this scalar optimization problem grants:

$$\forall(\beta_i^0, p_i) \in \mathbb{R}\times\mathbb{R}, \quad p_i = \operatorname{prox}_{\alpha|.|}\left(\beta_i^0\right) \Leftrightarrow \beta_i^0 - p_i \in \partial(\alpha|.|)(p_i). \tag{4.23}$$

The subdifferential of this function can be easily described as follows:

$$\partial(\alpha|.|)(p_i) = \begin{cases} +\alpha & \text{, if } p_i > 0, \\ -\alpha & \text{, if } p_i < 0, \\ [-\alpha, +\alpha] & \text{, if } p_i = 0. \end{cases} \tag{4.24}$$

Then using the characterization of p_i (4.23), we get:

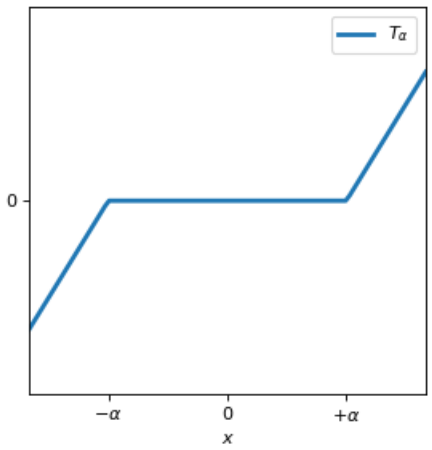


Figure 4.1: Graphical representation of the soft-thresholding operator.

$$p_i = \begin{cases} \beta_i^0 - \alpha & \text{, if } \beta_i^0 > \alpha, \\ \beta_i^0 + \alpha & \text{, if } \beta_i^0 < -\alpha, \\ 0 & \text{, if } \beta_i^0 \in [-\alpha, +\alpha]. \end{cases} \tag{4.25}$$

These different cases can be more conveniently summarized by considering the following function (see Figure 4.1):

$$T_\alpha(x) = (|x| - \alpha)_+ \text{sign}(x), \tag{4.26}$$

for any x in $\mathbb{R}$. This function, called the soft-thresholding operator according to the terminology from [Donoho, 1995], is an important element for sparse optimization methods. Notice that the terms *shrinkage thresholding* from ISTA are based on this property. Here it provides a closed form for the proximity operator of FISTA applied to the Lasso. Indeed we have that for any $\boldsymbol{\beta^0}$ in $\mathbb{R}^d$, the proximity operator $\boldsymbol{p} = \text{prox}_{L_f^{-1}\lambda\|\cdot\|_1}(\boldsymbol{\beta^0})$ verifies:

$$p_i = T_{L_f^{-1}\lambda}(\beta_i^0), \quad \forall i \in \{1, \dots, d\}. \tag{4.27}$$

We close this chapter by giving FISTA for the Lasso in Algorithm 3. For simplicity, we chose to present it with the constant step L_f^{-1}. Since the Lipschitz constant $L_f = 2\lambda_{max}(\boldsymbol{X}^\top \boldsymbol{X})$, with $\lambda_{max}(\boldsymbol{X}^\top \boldsymbol{X})$ being the biggest eigenvalue of $\boldsymbol{X}^\top \boldsymbol{X}$, this choice may be not suitable for large problems. In such cases where L_f is hard to compute, a backtracking strategy for the FISTA steps can be implemented. Also note that the fifth line has to be understood component by component of $\boldsymbol{x^{(k+1)}}$ and $\boldsymbol{y^{(k)}}$, since T_α has been defined as a scalar function.

[Chalasani et al., 2013] applied FISTA on convolutional problems, in a generic context. Despite their implementation of efficient operations, for instance the application of the convolution operator in order to avoid expensive matrix products (discussed in 4.2.2), they still needed to compute at

each iteration the convolution on the whole data. Therefore, in such generic settings, FISTA does not provide a scalable method for large signals.

Algorithm 3 FISTA for the Lasso

Require: $\boldsymbol{x^{(0)}} \in \mathbb{R}^d$.
1: $\boldsymbol{\beta^{(0)}} \leftarrow \boldsymbol{x^{(0)}}$
2: $t_0 \leftarrow 1$
3: **for** $k \in \{1, \dots, K\}$ **do**
4: $\quad \boldsymbol{y^{(k)}} \leftarrow \boldsymbol{\beta^{(k)}} - 2L_f^{-1}\boldsymbol{X}^\top(\boldsymbol{X\beta^{(k)}} - \boldsymbol{Y})$
5: $\quad \boldsymbol{x^{(k+1)}} \leftarrow T_{L_f^{-1}\lambda}(\boldsymbol{y^{(k)}})$
6: $\quad t_{k+1} \leftarrow \frac{1+\sqrt{1+4t_k^2}}{2}$
7: $\quad \boldsymbol{\beta^{(k+1)}} \leftarrow \boldsymbol{x^{(k+1)}} + \frac{t_k - 1}{t_{k+1}}(\boldsymbol{x^{(k+1)}} - \boldsymbol{x^{(k)}})$
8: **end for**

4.4.2 Coordinate Descent for the Lasso

We now briefly present an alternative strategy for solving the Lasso problem (4.20), the coordinate descent (CD) algorithm. The main idea of the CD is to, at any step, select a single coordinate of the vector to estimate, then minimize the cost function of the problem with respect to this coordinate, keeping the other coordinates fixed. Therefore, solving such smaller optimization problems until convergence allows to compute a solution. This idea is quite general and can therefore be applied to other problems than the Lasso. Nevertheless, we will see that the main step of the CD for the Lasso takes a very particular form, intimately related to the ideas developed for proximal methods (subsection 4.4.1).

At any step, we select a coordinate j in $\{1, \dots, d\}$ and we want to minimize the cost function of (4.20) with respect to the variable β_j. Then using the ideas of subgradient and convex optimizations given in 4.3.1, simple computations show that the minimizer $\beta_j^\star$ of (4.20) with respect to β_j is given by:

$$\beta_j^\star = \frac{1}{z_j} T_\lambda(w_j), \tag{4.28}$$

with:

$$w_j = \sum_{i=1}^{n} X_{i,j}\left(Y_i - \sum_{k \neq j} \beta_k X_{i,k}\right) \quad \text{and} \quad z_j = \|\boldsymbol{X_j}\|_2^2, \tag{4.29}$$

where $\boldsymbol{X_j}$ is the j-the column of $\boldsymbol{X}$. Remark that, as in proximal methods, the soft-thresholding operator plays an important role for the computation of a Lasso solution. Moreover, we get a closed form for the solution, which is notable since the Lasso does not have a closed form in the multivariate context. In practice, the choice of the coordinate j at any step can be handled in various ways, for instance cycle through all the coordinates,

choose a random one or greedily select a coordinate based on a certain criterion. The latter grants better improvements, but at the price of a higher computational cost. However, it seems to provide the best results in experimental settings ([Nutini et al., 2015]).

We present in the next chapter our efficient implementation of the working set for the estimation of the activations. This algorithm aims to apply the working set strategy on small temporal windows. We chose FISTA over CD as its inner Lasso solver for two reasons. First, although this algorithm works on smaller parts of the signal, the size of the associated design matrix remains quite large, due to the temporal structure of the problem. Moreover, on average the working set tends to add activations which are not too correlated. In such setting, experimental results from [Bach et al., 2011] show that FISTA is a more interesting approach than CD in terms of computation times.

An analogous approach which relies on a similar idea of splitting the signal using small temporal windows can be found in [Moreau et al., 2018], named Distributed Convolution Coordinate Descent (DICOD). By considering a temporal partition of the whole signal, this 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. Under mild assumptions, they could prove that the computed solution is indeed a Lasso solution. Although they also 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 properties of the problem, we prove in the following chapters 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.

4.4.3 Least Angle Regression (LARS)

We end this chapter with the presentation of another popular Lasso solver, the Least Angle Regression (LARS) introduced by [Efron et al., 2004]. Originally proposed as a variable selection algorithm for the linear regression problem, because of the deep reliationships that the Lasso shares with this problem, LARS has been successfully adapted to the Lasso.

We essentially follow the presentation of LARS given in [Hastie et al., 2008, Chapter 3]. As in the forward stepwise regression strategy, LARS iteratively adds variables to an *active set*. The coefficients of these active variables move towards their least-square value, until their correlations with the current residual reach the same value of another variable. Then this new variable is added to the active set, and the whole process is repeated until all predictors have entered the active set.

Assume that the predictors are standardized so that they have zero

mean and unit variance. At the beginning of step k, let A_k the active set of variables and $\boldsymbol{\beta}_{A_k}$ the associated coefficients of the active variables. Since a new variable has just entered A_k, its value in $\boldsymbol{\beta}_{A_k}$ is zero, and the $k-1$ others are non zero. Then set $\boldsymbol{r}^{(k)} = \boldsymbol{Y} - \boldsymbol{X}_{A_k}\boldsymbol{\beta}_{A_k}$ the current residual. The direction at step k is defined as:

$$\boldsymbol{\delta}^{(k)} = (\boldsymbol{X}_{A_k}^\top \boldsymbol{X}_{A_k})^{-1} \boldsymbol{X}_{A_k}^\top \boldsymbol{r}^{(k)}. \tag{4.30}$$

Then moving the coefficients of the active variables as:

$$\boldsymbol{\beta}_{A_k} + \alpha \boldsymbol{\delta}^{(k)}, \tag{4.31}$$

with increasing α, we can show that the correlations of these variables with the residual remain tied and decrease. The value of α is increased until another variable has the same correlation with the current residual, then this variable is added to the active for step $k+1$. This process continues until all the variables entered the model, thus ending on the global least-square.

Remark that due to (4.31), the coefficients move in a piecewise linear manner in LARS. Using the Lagrangian of the Lasso problem and its Karush-Kuhn-Tucker optimality conditions, one can show that the regularization parameter λ is equal (in absolute value) to the correlation of any active variable with the current residual. Then assuming that the active set does not change for $\lambda_0 \geq \lambda \geq \lambda_1$, it can be deduced that the Lasso solution path is also linear, when λ moves from λ_0 to λ_1. From this important observation, we can conclude that LARS is also able to compute the full Lasso path, which is piecewise-linear by construction.

LARS is highly efficient, especially in small dimension settings, and does not seem too sensitive to highly correlated features ([Bach et al., 2011]). Nevertheless, since LARS provides the full Lasso path instead of a single estimate for a given λ, it is not directly comparable to solvers such as FISTA or the coordinate descent. Moreover, an important limitation of LARS relies on its inability to handle warm starts. Therefore in the context of our study, which is the design of an efficient working set algorithm, LARS should not be viewed as a relevant candidate for the inner Lasso solver of the working set. Using a working set strategy, [Lee et al., 2007] showed that they obtained better results than with LARS. In the same manner, we will see in the next chapter that, by making use of the structure of the problem, especially its temporal characteristic, our efficient working set algorithm paired with FISTA provides better results than LARS alone.

Chapter 5

Sliding window working set algorithm

In this chapter, 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 section 5.2. The algorithm is then introduced and illustrated in section 5.3 and discussed more in detail in subsection 5.3.2.

5.1 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 $\boldsymbol{a}^*$ 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 $S^* = Supp(\boldsymbol{a}^\star)$ of the true model parameter $\boldsymbol{a}^\star$.

Assumption 5.1.1. *(Absolute refractory period) For any given neuron, all indices in the support S^* which correspond to this neuron 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. This refractory period duration is several times longer than the action potential duration ([Kandel et al., 2000]). 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 $\boldsymbol{G} = \boldsymbol{H}^\top \boldsymbol{H}$. From the definition of the columns of $\boldsymbol{H}$ (see below Equation 4.5), we can recover the following Lemma.

Lemma 5.1.2. *For all t and t' in $\{1, ..., T\}$ and for all n and n' in $\{1, ..., N\}$, we have:*

$$G_{(n,t),(n',t')} = \boldsymbol{h}_{n,t}^\top \boldsymbol{h}_{n',t'} = \sum_{e=1}^{E} (\boldsymbol{w}_{n,e}^{\to t})^\top \boldsymbol{w}_{n',e}^{\to t'} \tag{5.1}$$

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

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

Assumption 5.1.3.
2.a *(Neurons are recognizable) There exists $\epsilon > 0$ such that for all $n \neq n'$ and $|t - t'| \leq \ell$*

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

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

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

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 produce different impacts on different electrodes ([Biffi et al., 2013]). 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 [Pouzat, 2016]). 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 [Pouzat, 2016].

5.2 Overlaps

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 4.1 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.

5.2.1 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 generate simultaneous spikes.

The problem is in fact a bit more complex than that because there might be transitive effect. Indeed in Figure 5.1, 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 [Biffi et al., 2013]. 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 200µm [Biffi et al., 2013] to about 30µm [Muthmann et al., 2015]. Finally the electrical signal that is generated 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 [Muthmann et al., 2015], which corresponds to a range of about 200µm). 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 6.1.1, we will leverage percolation results to upper bound the size of the spatial overlaps with large probability. In order to qualify the previous argument, we should mention that some areas of the brain may present a much higher density of recordable neurons, for instance the hippocampal area CA1 of the rat ([Henze et al., 2000]).

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:

$$\forall e, \quad \boldsymbol{w}_{n,e} = 0 \text{ or } \boldsymbol{w}_{n',e} = 0. \tag{5.2}$$

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 5.1.left where three independent spatial overlaps are recovered. Note that (5.2) implies that between two neurons n and n' in two independent clusters we have $\boldsymbol{h}_{n,t}^\top \boldsymbol{h}_{n',t'} = 0, \quad \forall t, t'$, 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).

5.2.2 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 introduce the following notations: we define a temporal window $\omega = [\![\omega_1, \omega_2]\!]$, where $\omega_1 \leq \omega_2$ and $\omega_1, \omega_2 \in \{1, \dots, T\}$, which contains all samples whose temporal indices $\omega_1 \leq t \leq \omega_2$. This temporal window will be used in the following to index vectors with $\boldsymbol{a}_\omega$, that contains the temporals samples $\omega_1 \leq t \leq \omega_2$ for all neurons n, and matrix $\boldsymbol{H}_\omega$ where are selected only the columns $\boldsymbol{h}_{n,t}$ where $\omega_1 \leq t \leq \omega_2$, for all neurons n. Let us first review the biological phenomenon which explains how we can temporally split the problem into smaller subproblems.

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 [Tuleau-Malot et al., 2014, Reynaud-Bouret et al., 2014] and the references 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 [Tuleau-Malot et al., 2014], 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 $\boldsymbol{h}_{n,t}^\top \boldsymbol{h}_{n',t'} = 0, \quad \forall n, n'$. This is interesting because it means that for any windows ω and ω' such that $\omega_2 < \omega'_1 + \ell$ we have $\boldsymbol{H}_\omega^\top \boldsymbol{H}_{\omega'} = \mathbf{0}$ 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 5.1.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. This will not only permit to analyze signals in real time, but also to reestimate the shapes in order to take into account experimental perturbations, such as the electrode drift phenomenon [Ekanadham et al., 2014].

Mathematically, the size of the temporal overlaps themselves is also controlled with large probability (see Section 5.2) and this will impact the overall complexity of the algorithm.

5.3 Sliding window working set

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 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.

5.3.1 Principle of the algorithm

The main algorithm is detailed in Algorithm 4 where line 3 denotes an update of the large vector $\hat{\boldsymbol{a}}$ on the current window. The idea is to solve the Lasso problem on small windows ω starting at 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 $\hat{\boldsymbol{a}}_\omega$, on the window ω with the working set algorithm.

2. Updating the window ω depending on the support of $\hat{\boldsymbol{a}}_\omega$:

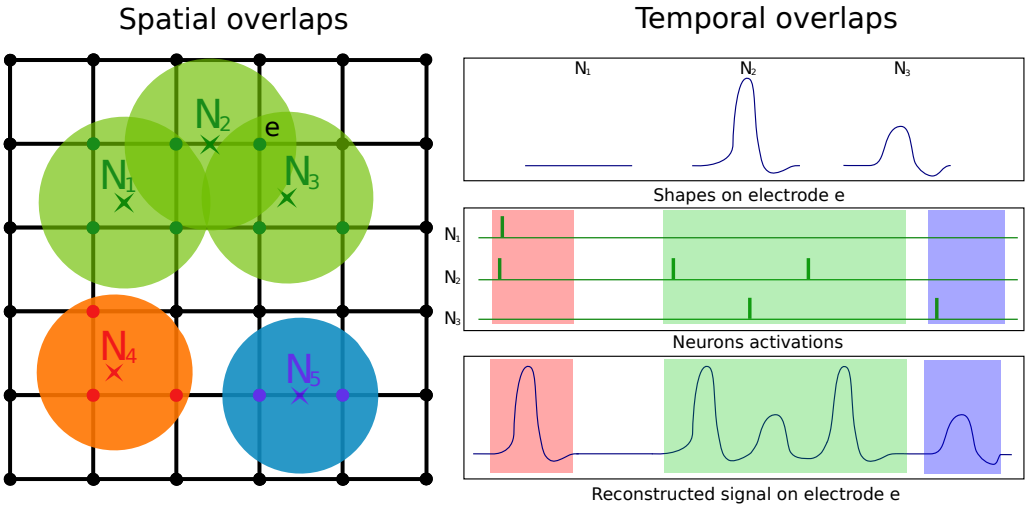


Figure 5.1: Illustration of the spatial and temporal overlaps. At the left hand side, we present an example of 3 spatial overlaps in the case of 5 neurons, on a regular grid of 36 electrodes. The position N_j of neuron j is represented by a check, and the reach of its spikes by a disc of radius r. On 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.

(a) If the support $Supp(\hat{\boldsymbol{a}}_\omega) \in [\![\omega_1+\ell, \omega_2-2\ell]\!]$ then the current problem is independent from the rest of the signal and the window is updated as $\omega = [\![\omega_2 + 1 - \ell, \omega_2 + 3\ell]\!]$.

(b) If the support $Supp(\hat{\boldsymbol{a}}_\omega) \cap [\![\omega_1, \omega_1 + \ell - 1]\!] \neq \emptyset$ has components in the first ℓ samples of the window, then we merge the current window with the previous (because the optimality conditions on the last ℓ samples of 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 $Supp(\hat{\boldsymbol{a}}_\omega) \subset [\![\omega_1 + \ell, \omega_2 - 2\ell]\!]$, it means that the reconstructed signal after convolution 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). The right border was chosen as 2ℓ instead of ℓ to promote a better exploration ahead of time and well separated temporal clusters minimizing the occurrence of the more expensive case 2.b discussed below. If there are activations at the beginning of the window however (first ℓ samples), it means that the residual and the optimality 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 window with the previous one. Else, if there are activations in the last ℓ samples of the window, we extend it in order to 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 5.2. It shows both configurations: when the window is extended and when the window is shifted to the right.

5.3.2 Algorithm solution *w.r.t.* the original Lasso

We now address the following question: is the proposed algorithm actually solving the global optimization problem (4.1)? We provide to this end the following theorem.

Theorem 5.3.1. *The solution $\hat{\boldsymbol{a}}$ computed by the sliding window working set is a solution of the initial Lasso problem* (4.7).

Proof. By construction of the algorithm, line 4 of the algorithm will return a list of windows Ω such that $\forall \omega \in \Omega$ the support $Supp(\hat{\boldsymbol{a}}_\omega) \in [\![\omega_1+\ell, \omega_2-2\ell]\!]$ 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 $|t' - t| > 3\ell$. This means that as discussed in subsection 5.2.2, the Lasso

Algorithm 4 Sliding window working set

Require: $\boldsymbol{y}, \boldsymbol{H}, \lambda > 0$

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

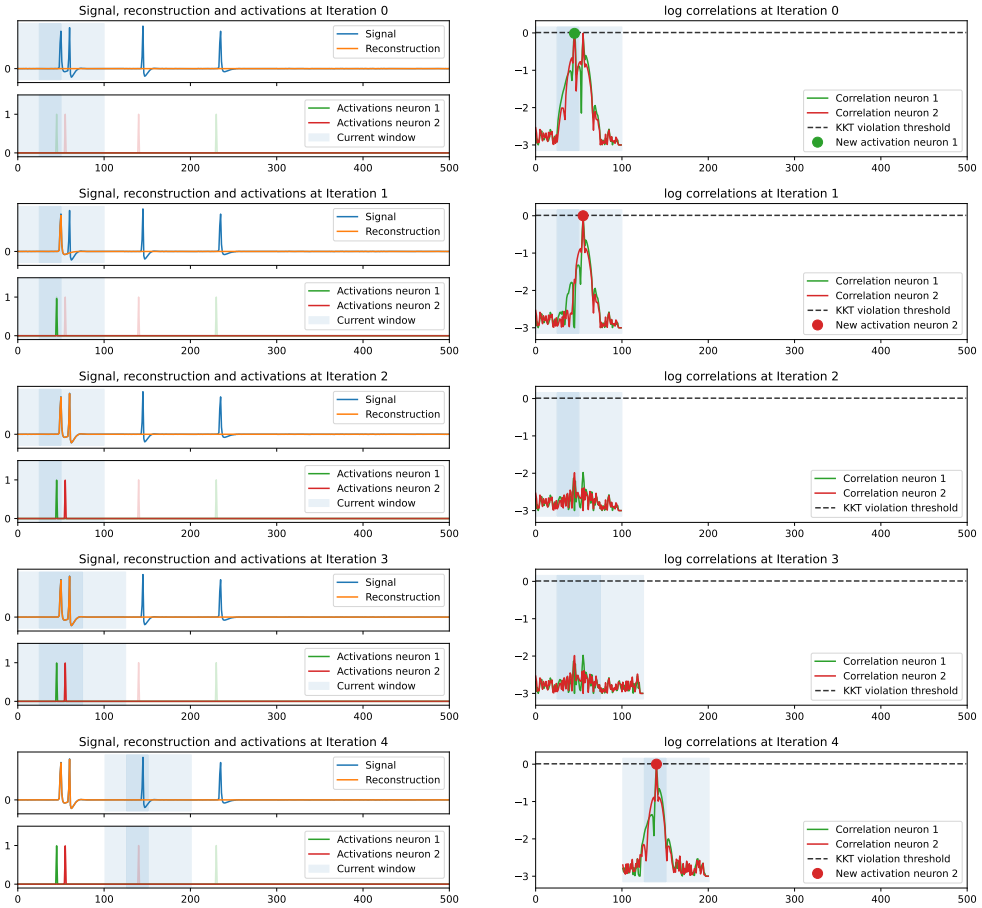
problems can be split as two independent problems on the support and all the other components that are not active satisfy the optimality conditions, implying that they will be 0. The solution $\hat{\boldsymbol{a}}$ is obtained by successive juxtaposition of the solutions of independent problems on disjoint windows that are estimated on line 3. □

5.3.3 Numerical complexity and efficient implementation

We now discuss why the proposed algorithm is more efficient than the convolutional working set discussed in section 4.2. 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 optimality conditions. In the sliding window working set, the optimality conditions 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 chapter 6. 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.

Also note that the sliding window working set proposed in Algorithm 4 can be slightly improved: in the case where a new activation is close to the beginning or the end of the window, instead of waiting for the convergence on the current window, the algorithm could immediately merge this window

with the previous one (that is if the new activation is in $[\![\omega_1, \omega_1 + \ell - 1]\!]$), or extend this window (that is if the new activation is larger than $\omega_2 - 2\ell$). Since this modification only marginally improves its performances, we keep in the following the sliding window working set as formulated in Algorithm 4.



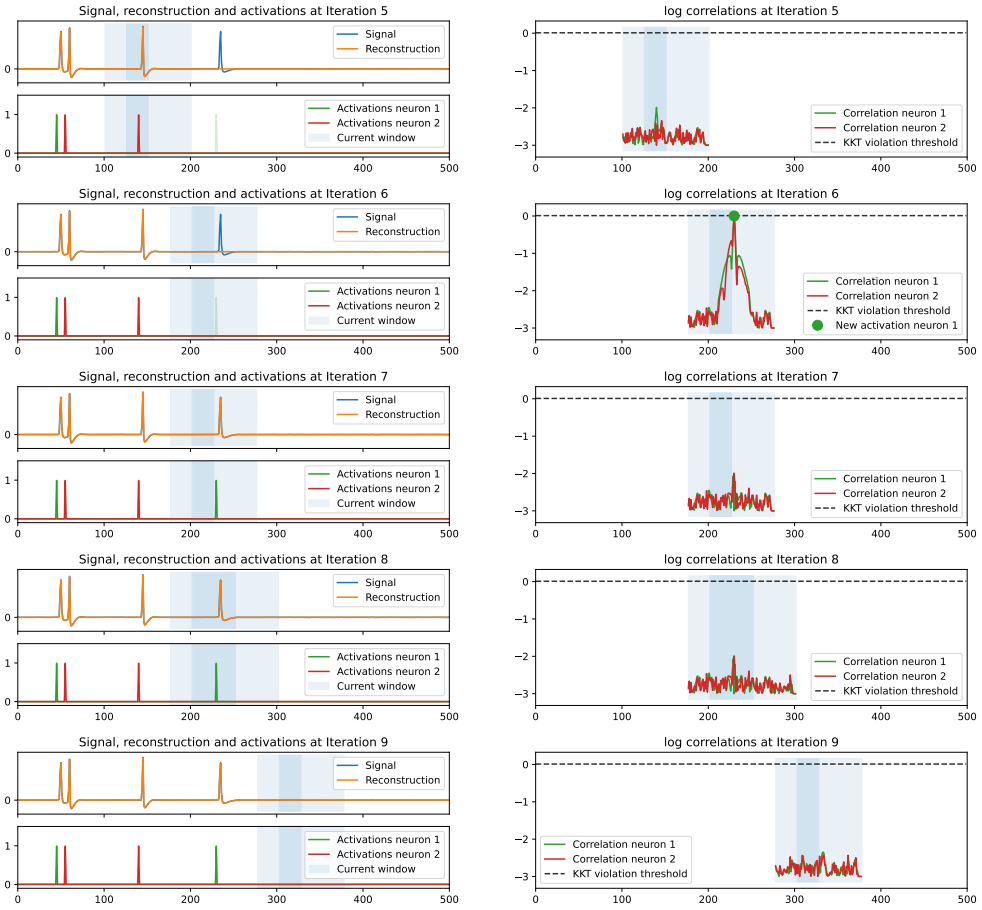


Figure 5.2: Illustration of the different steps in the proposed algorithm. Left: observed signal $\boldsymbol{S}$ with model reconstruction and sparse model $\boldsymbol{a}_i$. The current window is represented as a light blue background. True activations are illustrated with transparency. Right: optimality conditions (named KKT) violation at the current step. Temporal instants and neurons violating the optimality conditions are over the black dashed line.

Chapter 6

Mathematical results

In this chapter, we present the mathematical results for the analysis of the sliding window working set algorithm. We begin by giving bounds on the sizes of the spatial and temporal overlaps. Then we present the main statistical theorem which informs us that the support of the computed Lasso solution is close to the support of the true activation vector. Finally, we derive from this theorem the theoretical temporal complexity of the sliding window working set algorithm.

6.1 Control of the spatial and temporal overlaps

6.1.1 Spatial overlaps

In the MEA case, the E electrodes are placed on a square lattice with fixed distance δ between adjacent 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. Note that this approximation is reputed valid for neuron cultures ([Millet et al., 2011]), but not for real tissues.

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 $2r_0$.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 [Duminil-Copin et al., 2018] and references therein). Thanks to this, we can prove the following proposition.

Proposition 6.1.1. *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 cardi-*

nality N_c, satisfies

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

with $\kappa > 0$, which only depends on γ, δ, τ and r_0. Similarly the number of active electrodes for a given spatial overlap c can be bounded as $E_c \leq \bar{\kappa}[\log(E/\alpha)]^2$, with $\bar{\kappa} > 0$, 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.

Proof. In the sequel we use the term percolation term "cluster" to refer to spatial overlap. It has been shown in [Duminil-Copin et al., 2018] 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 $\exp(-c(\gamma, 2r_0)r)$, with $c(\gamma, 2r_0) > 0$, 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 [Reynaud-Bouret, 2003]), 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 $(\sqrt{E}+2)\delta$. 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

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

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 which belong 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, and (iii) the number of neurons per ball for the first Q balls is controlled, we obtain the desired result. □

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 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.

Nevertheless, in the continuum percolation literature, we can find various estimations of the percolation constant γ_c. Taking into account an exact result from [Meester and Roy, 1996, Theorem 3.10, p85], the bound $\gamma < 0.174/(2r_0)^2$ would guarantee the hypothesis $\gamma < \gamma_c$ of Proposition 6.1.1.

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 varies logarithmically with E or not.

In the rest of this thesis,

(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 whole 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.

6.1.2 Temporal overlaps

Here we assume that the activations $\boldsymbol{a}^*$ are the realisation of a given random process. More specifically, and as explained in Section 5.2.2, we do not need to model each neuron individually and we do not need to model the amplitude of $\boldsymbol{a}^*$. 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 that 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, N the number of neurons (of the spatial overlap possibly) (that is the A_i's are i.i.d. Bernoulli with parameter p) and Δ the binning size of the process. Note that in this setting, 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 6.1.2. *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.

Proof. 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 6.1.1, 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 $\Omega_{\alpha,A,s} = \Omega_{\alpha/2,s} \cap \Omega_{\alpha/2,A}$. □

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 records a large 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.

6.2 Control of the noise

Lemma 6.2.1. *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*

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

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

$$\Omega_{\alpha,\xi} = \bigcap_{n,t} \left\{ \left| h_{n,t}^\top \xi \right| \leq z_\alpha \right\}. \tag{6.1}$$

Then we have

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

Proof. For a fixed (n,t), since the noise is Gaussian, the random variable

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

is also a Gaussian variable with variance bounded by $\overline{c}\sigma^2$. Thus the event

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

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

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 $\{1,...,T\}$,

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

with probability larger than $1-\alpha$.

□

This lemma is very classical and helps 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.

6.3 Theoretical properties of the Lasso estimator

Theorem 6.3.1. *Fix $\alpha \in (0,1/2)$. Let Assumptions 5.1.1 and 5.1.3 be satisfied and let us assume that the noises are i.i.d. Gaussian. With the notation of Propositions 6.1.1, 6.1.2 and Lemma 6.2.1, there exists an event Ω_α of probability larger than $1-\alpha$ such that, on Ω_α, for any temporal window ω and any solution $\hat{\boldsymbol{a}}_\omega$ of the Lasso problem $(\boldsymbol{H}_\omega, \boldsymbol{y})$ 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*

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

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

$$\lambda > \frac{\underline{c} + 2\rho\overline{c}}{\underline{c} - 2\rho\overline{c} - 4\epsilon\mathcal{N}}\left(z_\alpha + 2(\rho\overline{c} + \epsilon\mathcal{N})\|\boldsymbol{a}^*\|_{\infty,\partial\omega}\right) \quad \textit{and} \quad \underline{c} > 2\rho\overline{c} + 4\epsilon\mathcal{N} \tag{6.2}$$

with the convention that

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

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

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

2. *Moreover, if*

$$\inf_{(n,t)\in S^*\cap\omega} |\boldsymbol{a}^*_{n,t}| > \frac{z_\alpha + \lambda + 2\|\boldsymbol{a}^*\|_{\infty,\partial\omega}\,(\rho\overline{c} + \epsilon\mathcal{N})}{\underline{c} - 2\epsilon\,\mathcal{N}}, \tag{6.3}$$

then

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

Proof. Note that if Algorithm 4 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 $\boldsymbol{a}_J$ as the vector obtained by setting to 0 all the coordinates from $\boldsymbol{a}$ with their index not in J. We define similarly $\boldsymbol{H}_J$ as the matrix obtained by replacing all the columns from $\boldsymbol{H}$ with their index not in J by the zero vector. Therefore in the sequel vectors $\boldsymbol{a}$ and matrix $\boldsymbol{H}$ have always the same dimensions.

We work on the event Ω_α, which as stated in the remarks below Lemma 6.2.1 is of probability greater 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 6.1.1. 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:

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

where we have used a slight abuse of language: "$Supp(\boldsymbol{a}) \subset \omega$" means that the temporal indices t of $\boldsymbol{a} = (\boldsymbol{a}_{n,t})_{n,t}$ 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 $\boldsymbol{a}^*$:

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

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

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

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

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

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

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

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

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

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

In view of Lemma 5.1.2, we have for all $(n,t) \in S^* \cap \omega$ that

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

In addition, Assumption 5.1.3 guarantees that $|G_{(n,t),(n',t')}| \leq \epsilon$ 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 $(n,t) \in S^* \cap \omega$,

$$|\boldsymbol{h}_{n,t}^\top \boldsymbol{H}_{S^*\cap\omega^c}\,\boldsymbol{a}^*| \leq 2(\rho\bar{c} + \epsilon\mathcal{N})\|\boldsymbol{a}^*\|_{\infty,\partial\omega}. \tag{6.4}$$

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

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

Given the block-band structure of the Gram matrix G (see Lemma 5.1.2), the first sum in the above display contains exactly 1 nonzero term corresponding to $t' = t$:

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

Regarding the second sum, we also have in view of Lemma 5.1.2:

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

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

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

Assumption 5.1.3 guarantees that $|\boldsymbol{G}_{(n,t),(n',t')}| \leq \epsilon$ 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 $(n,t) \in S^* \cap \omega$,

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

and consequently, since we are on Ω_α,

$$(c - 2\epsilon\mathcal{N})\|\Delta\|_{\infty,\omega} \leq z_\alpha + \lambda + 2(\rho\overline{c} + \epsilon\mathcal{N})\|\boldsymbol{a}^*\|_{\infty,\partial\omega}.$$

Combining this result with Lemma 6.2.1, we get

$$\|\boldsymbol{a}^* - \hat{\boldsymbol{a}}_{S^*\cap\omega}\|_{\infty,\omega} \leq \frac{z_\alpha + \lambda + 2(\rho\overline{c} + \epsilon\mathcal{N})\|a^*\|_{\infty,\partial\omega}}{\underline{c} - 2\epsilon\mathcal{N}}. \tag{6.5}$$

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

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

In view of (6.4) and (6.5), we have on the event $\Omega_{\alpha,A,s} \cap \Omega_{\alpha,\xi}$, for all $(n,t) \in \omega \setminus S^*$,

$$\begin{aligned}&|\boldsymbol{h}_{n,t}^\top(\boldsymbol{y} - \mathbf{H}_\omega \hat{\boldsymbol{a}}_{S^*\cap\omega})| \\ &\leq z_\alpha + 2(\rho\overline{c} + \epsilon\mathcal{N})\left(\|\boldsymbol{a}^*\|_{\infty,\partial\omega} + \frac{z_\alpha + \lambda + 2(\rho\overline{c} + \epsilon\mathcal{N})\|\boldsymbol{a}^*\|_{\infty,\partial\omega}}{\underline{c} - 2\epsilon\mathcal{N}}\right).\end{aligned} \tag{6.6}$$

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

$$\lambda > z_\alpha + 2(\rho\overline{c} + \epsilon\mathcal{N})\left(\|\boldsymbol{a}^*\|_{\infty,\partial\omega} + \frac{z_\alpha + \lambda + 2(\rho\overline{c} + \epsilon\mathcal{N})\|\boldsymbol{a}^*\|_{\infty,\partial\omega}}{\underline{c} - 2\epsilon\mathcal{N}}\right), \tag{6.7}$$

or equivalently

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

This means that $\hat{\boldsymbol{a}}_{S^*\cap\omega}$ satisfies the strict optimality conditions in (6.8) below on the temporal window ω. Thus we proved that $\hat{\boldsymbol{a}}_{S^*\cap\omega}$ is a solution

of the Lasso minimization problem on the temporal window ω, on the event Ω_α.

By Lemma 6.3.2 applied to $\tilde{\boldsymbol{a}}_\omega = \hat{\boldsymbol{a}}_{S^*\cap\omega}$, we get the first inclusion.

We assume in addition (6.3). Then, in view of (6.5), we have on the event on Ω_α

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

□

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

Lemma 6.3.2. *Consider* $Crit(\boldsymbol{a}) = \|\boldsymbol{y} - \mathbf{H}_\omega \boldsymbol{a}\|_2^2 + 2\lambda\|\boldsymbol{a}\|_1$. *Let* $\tilde{\boldsymbol{a}}_\omega$ *be a minimizer of* $Crit(\boldsymbol{a})$, *hence satisfying the optimality conditions:*

$$\begin{cases} \boldsymbol{h}_{n,t}^\top(\boldsymbol{y} - \mathbf{H}_\omega\tilde{\boldsymbol{a}}_\omega) = \lambda\ sign((\tilde{a}_\omega)_{n,t}) & ,\ if\ (\tilde{\boldsymbol{a}}_\omega)_{n,t} \neq 0, \\ |\boldsymbol{h}_{n,t}^\top(\boldsymbol{y} - \mathbf{H}_\omega\tilde{\boldsymbol{a}}_\omega)| \leq \lambda & ,\ if\ (\tilde{\boldsymbol{a}}_\omega)_{n,t} = 0. \end{cases} \tag{6.8}$$

Let

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

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

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

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

Proof. In view of (6.8), we have for any $(n,t) \in \omega$,

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

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

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

Therefore, we have

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

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

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

Thus,

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

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

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

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

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

We obtain a contradiction. This means that

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

□

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 6.1.2, it is roughly pT.

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

$$\underline{c}>2\rho+4\epsilon\mathcal{N}.$$

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 $\mathcal{N}$ 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 [Tuleau-Malot et al., 2014], it usually involves a few neurons, except during epileptic crisis.

Next, $\boldsymbol{a}^*$ is usually assumed to be a binary 0/1 vector in the classical problem. Therefore (6.2) means that we need

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

On the other hand, Condition (6.3) becomes

$$\lambda < \underline{c} - 2\rho\overline{c} - 4\epsilon\mathcal{N} - \sqrt{2\sigma^2\overline{c}\log(2NT/\alpha)}.$$

So there is room to find such a λ if typically

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

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 4 are built to guarantee that

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

which would imply that

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

So in practice,

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

should be sufficient.

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

Corollary 6.3.3. *Fix $\alpha \in (0, 1/2)$. Let Assumptions 5.1.1 and 5.1.3 be satisfied and let us assume that the noise variables are i.i.d. Gaussian. With the notations of Propositions 6.1.1, 6.1.2 and Lemma 6.2.1, on the same event Ω_α of probability larger than $1-\alpha$, if λ is chosen so that* (6.2) *and* (6.3) *are satisfied, then all the windows $\omega \in \Omega$ constructed by Algorithm 4 have a length controlled by*

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

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

Proof. With the choices provided in Algorithm 4, 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 $\hat{\boldsymbol{a}}_\omega$ is found in the last 2ℓ coordinates. Thanks to Theorem 6.3.1, this means that this is also the first time that $\boldsymbol{a}^*$ 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 6.1.2). 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. □

6.4 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 6.4.1 ([Loth, 2011], Section 2.4)**.** *Consider the Lasso problem* (4.5) *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*

$$C_{ws}(n, p, k) = O(n^2pk + nk^3 + k^4). \tag{6.9}$$

Therefore applying Theorem 6.4.1 to the resolution of the global Lasso problem with the working set strategy gives the following complexity.

Proposition 6.4.2. *Under the hypothesis of Theorem 6.3.1, the algorithm 1 solves a problem of TE observations with TN features and a number of true activations $O(TN)$, an application of Theorem 6.4.1 recovers an approximate complexity of*

$$C_{ws}(TE, TN, TN) = O(T^4(E^2N^2 + EN^3 + N^4)). \tag{6.10}$$

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 remain 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 6.4.3. *Under the hypothesis of Theorem 6.3.1 and Corollary 6.3.3, there exists an event Ω_α of probability larger than $1-\alpha$ such that, on Ω_α, the sliding window working set algorithm 4 has the following approximate complexity*

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

$$O(T\log^4(T/\alpha)(E^2N^2 + EN^3 + N^4)). \tag{6.11}$$

- *In the subcritical MEA case*

$$O(ET\log^4(ET/\alpha)\log(E/\alpha)^8). \tag{6.12}$$

Proof. In this proof we assume that the problem respects the hypothesis of Theorem 6.3.1 and Corollary 6.3.3, so there exists such 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. Let us define $\tilde{T} = \log(T/\alpha)$ for the bounds on the window sizes. In this case we need to solve $O(T)$ temporal independent problems whose window size is bounded by $O(\tilde{T})$ (as proven in Corollary 6.3.3). Those independent problems, using notations from Theorem 6.4.1, have dimensionalities of $O(E\tilde{T})$ observations, $O(N\tilde{T})$ features and $O(N\tilde{T})$ selected features. This means that the complexity is

$$\begin{aligned}\mathcal{C}(Alg.4, Tetrode) &= O(TC_{ws}(E\tilde{T}, N\tilde{T}, N\tilde{T}) \\ &= O(T((E\tilde{T})^2(N\tilde{T})^2 + (E\tilde{T})(N\tilde{T})^3 + (N\tilde{T}^4))) \\ &= O(T\log(T/\alpha)^4(E^2N^2 + EN^3 + N^4))\end{aligned}$$

which proves the result in equation (6.11).

In the subcritical MEA case, the problem can be solved with $O(ET)$ independent problems (using both spatial and temporal overlaps). But those problems are of much smaller size. In order to simplify the notations, let us now define $\tilde{T} = \log(ET/\alpha)$ and $\tilde{N} = \tilde{E} = \log(T/\alpha)^2$ for the bounds on the window sizes, N_c and E_c. Indeed Corollary 6.3.3 tells us that the size of the temporal window is bounded in this case by $O(\tilde{T})$ and Proposition 6.1.1 tells us that the size of the spatial overlaps N_c and E_c are bounded in this case by $O(\tilde{N})$ and $O(\tilde{E})$. This means that the problems we need to solve have maximum dimensionality of $O(\tilde{E}\tilde{T})$ observations, $O(\tilde{N}\tilde{T})$ features and again $O(\tilde{N}\tilde{T})$ selected features. This means that the complexity of solving the whole problem is

$$\begin{aligned}\mathcal{C}(Alg.4, MEA) &= O(ETC_{ws}(\tilde{E}\tilde{T}, \tilde{N}\tilde{T}, \tilde{N}\tilde{T})) \\ &= O(ET\tilde{T}^4(\tilde{E}^2\tilde{N}^2 + \tilde{E}\tilde{N}^3 + \tilde{N}^4)) \\ &= O(ET\log(ET/\alpha)^4\log(E/\alpha)^8).\end{aligned}$$

This proves result in equation (6.12) concludes the proof of the theorem. □

The above result reveals that our sliding window working set can avoid the high computational quartic costs from Proposition 6.4.2 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 $O(T\log(T)^4)$ 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.

6.5 Attenuation model

In section 4.1.2, we presented the general form of the columns $\boldsymbol{h}_{n,t}$ of the design matrix $\boldsymbol{H} = (\boldsymbol{h}_{1,1}, \boldsymbol{h}_{1,2}, \dots, \boldsymbol{h}_{N,T-1}, \boldsymbol{h}_{N,T})$, where n stands for a neuron in $\{1, \dots, N\}$ and t stands for a time in $\{1, \dots, T\}$. Each column can be deduced from the spike shapes $\boldsymbol{w}_{n,e}$ as follows:

$$\boldsymbol{h}_{n,t} = ((\boldsymbol{w}_{n,1}^{\to t})^\top, \dots, (\boldsymbol{w}_{n,E}^{\to t})^\top)^\top. \tag{6.13}$$

In order to study more deeply the properties of the design matrix $\boldsymbol{H}$, we introduce a model for its columns. The main goal of this model is to take into account the attenuation phenomenon which affects the voltage amplitude of the actions potentials as they diffuse in the extracellular environment. Therefore we want the recorded shapes $\boldsymbol{w}_{n,e}$ to satisfy such property. This encourages us to consider the following assumption:

Assumption 6.5.1 (attenuation model for the columns of $\boldsymbol{H}$)**.** *For any neuron n, we assume that there exists a shape $\boldsymbol{w}_{n,\cdot} \in \mathbb{R}^\ell$ and E real numbers $(\alpha_{n,1}, \dots, \alpha_{n,E})$ in $[0,1]$ (called attenuation weights) such that the shape recorded by any electrode e writes as:*

$$\boldsymbol{w}_{n,e} = \alpha_{n,e}\boldsymbol{w}_{n,\cdot}.$$

Using Assumption 6.5.1, we can derive any column $\boldsymbol{h}_{n,t}$ of a given neuron n from the shape $\boldsymbol{w}_{n,\cdot}$. This shape can be viewed as an ideal recording of the action potential generated by neuron n. In view of the recording techniques presented in section 3.1, we can interpret this shape as the recording we would get from an intracellular recording of neuron n.

Therefore for any neuron n and any time t, we can write:

$$\boldsymbol{h}_{n,t} = \begin{pmatrix} \alpha_{n,1}\boldsymbol{w}_{n,\cdot}^{\to t} \\ \vdots \\ \alpha_{n,E}\boldsymbol{w}_{n,\cdot}^{\to t} \end{pmatrix} = \boldsymbol{\alpha}_{n,\cdot} \otimes \boldsymbol{w}_{n,\cdot}^{\to t}, \tag{6.14}$$

where $\boldsymbol{\alpha}_{n,\cdot} = (\alpha_{n,1}, \ldots, \alpha_{n,E})^\top$ and $\otimes$ stands for the tensor product between two vectors.

In practice, the value of any given attenuation weight $\alpha_{n,e}$ essentially depends on the distance r between the source of the signal, roughly speaking the center of the soma of the neuron n, and the electrode e recording it. Naturally the greater the distance r, the larger the attenuation is. In other words, the amplitude of the recorded voltage is expected to decrease as the signal propagates in the extracellular environment.

Interestingly, the form which takes this decreasing depends itself on the shape of the neuron involved. For instance, the potential measured from a spherical shaped neuron, which can be modeled as a point source, decreases as $1/r$. On the other hand, by the dipolar approximation, the potential measured from a pyramidal shaped neuron is expected to decrease approximately as $1/r^2$ ([Plonsey et al., 2007, Chapter 2]). Moreover, experimental works from [Henze et al., 2000] inform us that beyond a certain distance r_{lim} (roughly 200µm), the attenuation becomes large enough so that the action potentials can no longer be distinguished from the background noise. Therefore, taking $\alpha_{n,e} = 0$ when the distance r between neuron n and electrode e is larger than r_{lim} is realistic and remains coherent with our assumption of spatial clustering between neurons formulated in chapter 5.

Using this attenuation model for the columns of the design matrix $\boldsymbol{H}$, we can see that:

Proposition 6.5.2. *For all t and t' in $\{1, ..., T\}$ and for all n and n' in $\{1, ..., N\}$, we have:*

$$G_{(n,t),(n',t')} =< \boldsymbol{\alpha}_{n,\cdot}, \boldsymbol{\alpha}_{n',\cdot} >_{\mathbb{R}^E} < \boldsymbol{w}_{n,\cdot}^{\to t}, \boldsymbol{w}_{n',\cdot}^{\to t'} >_{\mathbb{R}^T}, \tag{6.15}$$

where $< \boldsymbol{x}, \boldsymbol{y} >_{\mathbb{R}^p}$ stands for the usual scalar product $\boldsymbol{x}^\top \boldsymbol{y}$, between $\boldsymbol{x}$ and $\boldsymbol{y}$ in $\mathbb{R}^p$.

Proof.

$$G_{(n,t),(n',t')} =< \boldsymbol{h}_{n,t}, \boldsymbol{h}_{n',t'} >_{\mathbb{R}^{ET}} \tag{6.16}$$

$$= \sum_{e=1}^{E} \alpha_{n,e}\alpha_{n',e} < \boldsymbol{w}_{n,\cdot}^{\to t}, \boldsymbol{w}_{n',\cdot}^{\to t'} >_{\mathbb{R}^T} \quad \text{by (6.14)} \tag{6.17}$$

$$=< \boldsymbol{\alpha}_{n,\cdot}, \boldsymbol{\alpha}_{n',\cdot} >_{\mathbb{R}^E} < \boldsymbol{w}_{n,\cdot}^{\to t}, \boldsymbol{w}_{n',\cdot}^{\to t'} >_{\mathbb{R}^T} . \tag{6.18}$$

□

In this chapter, we have seen that the assumption of weak correlations between the columns of $\boldsymbol{H}$ ((2.a)) is of great importance in order for the estimation procedure to be able to distinguish the activity of each neuron. In practice, neighboring neurons tend to share the same biophysiological properties, and therefore tend to generate action potentials with similar shapes. Consequently we see from this attenuation model that the geometrical configuration of the neurons and the electrodes is essential for the study of the identifiability of our problem.

Taking $w_{max} = \sup_{n \in \{1,\dots,N\}} \|\boldsymbol{w}_{n,.}\|_2 < +\infty$, we see from proposition 6.5.2 that, for two neurons n and n' in the same spatial overlap, that:

$$| < \boldsymbol{\alpha}_{n,.}, \boldsymbol{\alpha}_{n',.} >_{\mathbb{R}^E} | \leq \frac{\epsilon}{w_{max}^2} \Rightarrow |G_{(n,t),(n',t')}| \leq \epsilon \quad \forall t, t'. \tag{6.19}$$

Therefore if the correlations between the attenuation weights are small enough, we would still be able to guarantee the validity of our hypothesis of weak correlations, even in the worst case where two neurons have the same shapes, as in (6.19). In this regard, an interesting prospect would be to examine the behavior of the terms $< \boldsymbol{\alpha}_{n,.}, \boldsymbol{\alpha}_{n',.} >_{\mathbb{R}^E}$ for any neurons n and n' in the same spatial overlap. Using a Poisson distribution for the position of the neurons (as in subsection 6.1.1), and considering the appropriate form of attenuation, which, as stated previously depends greatly on the type of cell recorded, would allow the analysis of the correlations between the attenuation weights, and as a consequence grant an interesting contribution for the study of the practicality of our method.

Chapter 7

Numerical experiments

In this chapter we show with numerical simulations the performance of the sliding window working set presented in section 5.3. First we compare its computation time with other approaches. As all these approaches require to solve efficiently Lasso problems, we used the parallel implementation of the accelerated proximal gradient FISTA [Beck and Teboulle, 2009] from [Mairal et al., 2014]. 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 grows. 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.

7.1 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 [Mairal et al., 2014] using the accelerated proximal gradient FISTA to solve the global problem (4.7) with a pre-computed matrix $\boldsymbol{H}$. Since the size of the design matrix $\boldsymbol{H}$ 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 optimality conditions 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 $\boldsymbol{H}$ is not scalable with the

Quantity	Mean	Stdev
ϵ	173	175
ρ	0.828	0.013
$\underline{c}$	24.5	7.7
$\overline{c}$	918	1052

Table 7.1: Orders of magnitude for the parameters introduced in Assumption 5.1.3.

signal length T. In this method we adapt the standard working set algorithm Algorithm 1 to take into account the structure of $\boldsymbol{H}$. The computational bottleneck comes from the optimality conditions (line 3 of Algorithm 1). But in practice those conditions can be computed efficiently using a convolution by the shapes $\boldsymbol{W}$ instead of expensive matrix products. The residual $\boldsymbol{y} - \boldsymbol{H}\hat{\boldsymbol{a}}$ and the correlation $\boldsymbol{H}^T(\boldsymbol{y} - \boldsymbol{H}\hat{\boldsymbol{a}})$ (a convolution with reversed shapes) can be computed with complexity $O(ENT\ell)$, hence linear in T. The use of a working set also means that the storage of $\boldsymbol{H}$ is not necessary anymore, since we solve the Lasso on the much smaller $\boldsymbol{H}_J$. We use the FISTA solver from [Mairal et al., 2014] to solve the sub-problems at each iteration. Finally note that $\boldsymbol{H}_J$ will be very sparse due to the convolutional model and the sub-problem can be solved on a matrix $\tilde{\boldsymbol{H}}_J$ of $O(E|J|\ell)$ lines instead of $O(ET)$.

- **Sliding window working set**: This is the method proposed in section 5.3 and described in Algorithm 4. It focuses only on a small temporal window and slides the window when the problem is locally solved. Furthermore, since the research of the new activation is only carried out on the current window ω and not on the full signal, the computation cost of the optimality conditions is greatly reduced from $O(ENT\ell)$ to $O(EN|\omega|\ell)$.

We have simulated our dataset realistically by using the classical model from [Hodgkin and Huxley, 1952] for the description of the shape of the action potentials, and implemented by [Pouzat, 2016]. 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 also set the sampling rate at 10kHz and the single neuron firing rate at 20Hz. The recorded signals are computed as the convolution of the neuron activations and their respective action potential shapes, without noise (figure 7.1). We present in Table 7.1, numerical values which correspond to the parameters introduced in Assumption 5.1.3. These values depend a lot on the recordings in practice.

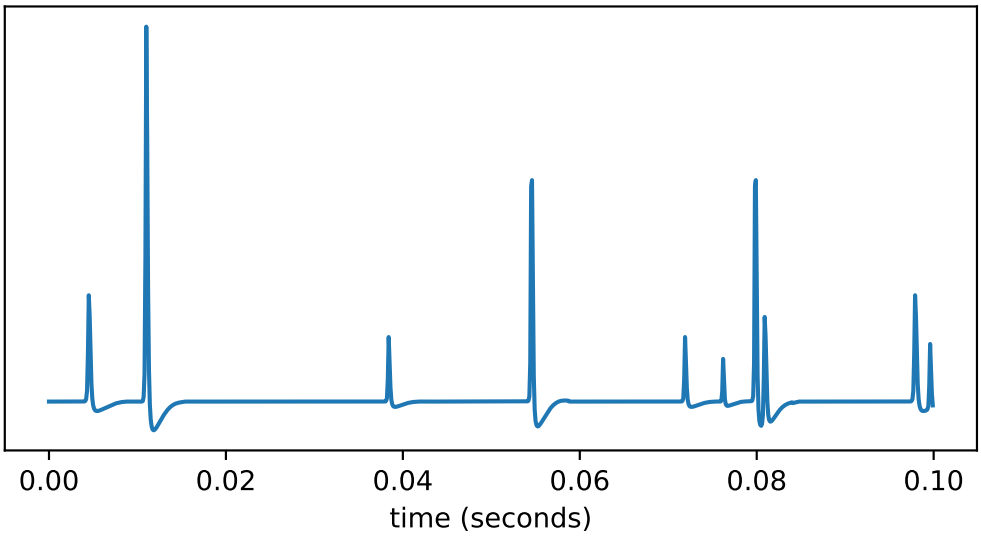
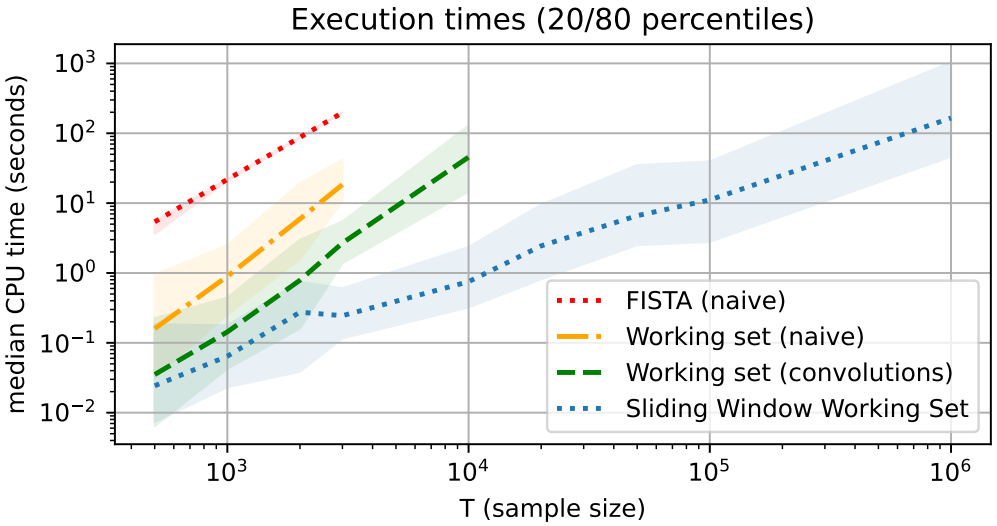


Figure 7.1: First portion (0.1s = 1000 points) of the recorded signal by one of the electrodes. Notice the five different action potential shapes corresponding to the five neurons in this simulation.

We present in figure 7.2 the computation times of the different methods and their 20/80 and 5/95 percentiles for different values of T (each simulation is performed 40 times). It is clear from the figure 7.2 that the proposed algorithm is the most efficient and is actually the only one that can solve problems with $T = 10^6$ temporal samples. The slopes 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 corresponds to the $O(T \log T)$ obtained in the theoretical results.

Also note that we applied the algorithms to the same data, in order to verify that they all compute the same activation vectors, and therefore estimate the same supports.



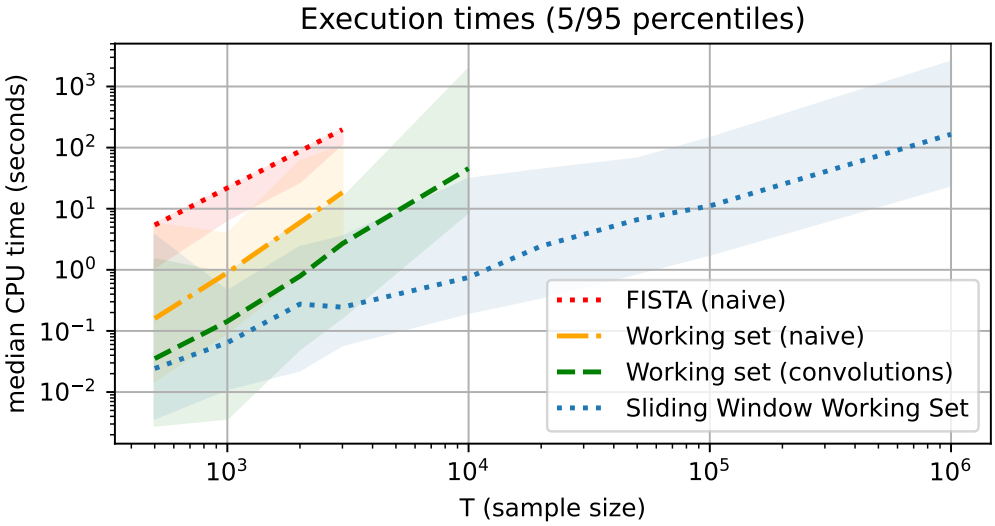


Figure 7.2: Comparison of the execution times for four different algorithms, when the size of the signals T growths. We represented the median execution times over 40 simulations as the dotted lines. The bands represent the execution times between the bottom and top percentiles.

7.2 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 [Bethus et al., 2012] and that have been already spike sorted by classical algorithms, we simulate small signals of size $T = 500$ for different noise levels, with $N = 2$ neurons firing at $50Hz$ and recorded by $E = 4$ electrodes. As in the previous section, we set the sampling rate at 10kHz.

In order to measure the performance of the algorithm to recover the true support, we consider first 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 estimated activation times. We introduce a softer measure of performance: $CP(\boldsymbol{x}, \boldsymbol{y}) = 1 - \|K * (\boldsymbol{x} - \boldsymbol{y})\|_1 / (\|\boldsymbol{x}\|_1 + \|\boldsymbol{y}\|_1)$, where K is an uniform kernel 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 error tolerance of 1ms.

We provide on figure 7.3 the performance in F1 score (left) and the

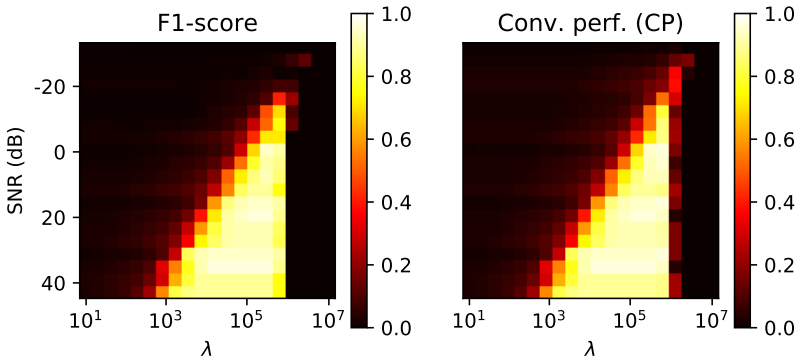


Figure 7.3: Influence of λ and the signal-to-noise ratio on the performances of the Lasso estimator. Results are averaged over 5 draws.

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 [Lewicki, 1998]. Therefore these experiments show that our method is robust enough to even treat low SNR recordings, provided that the regularization parameter λ is well chosen.

7.3 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 $\boldsymbol{W}_n$ are the centroids of the method ([Ekanadham et al., 2014]). After the activation times have been estimated, classical approaches select the neuron corresponding to the activation as the one closest to the centroid. Here in the distance-based approach, we actually do not perform the clustering, but directly use the true action potential shapes as what would be the centroids of the K-means. We now compare the Lasso and distance-based spike sorting in the presence of synchronization between neurons (simultaneous spikes).

Using similar data as in section 7.2, we compare in Figure 7.4 the performances of the methods when the number of synchronizations increases. To this end, we use the true activation times (left). Therefore the only unknown information is which neuron generated each activation. We also use the support recovered with our Lasso estimator (right). 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 imply that it can handle well simultaneous activations.

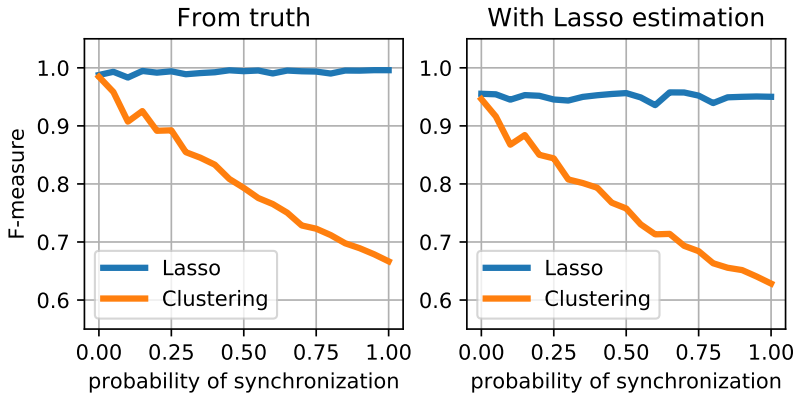


Figure 7.4: Comparison of Lasso and clustering performances (F-measure). Results are averaged over 50 draws.

Conclusion

In this thesis, our goal was to propose a new spike sorting methodology in order to improve the performances of this pre-processing step, especially when in presence of synchronizations and for large volumes of data. Using the Lasso for the convolutional sparse coding estimation, we have seen that this procedure offers nice theoretical guarantees in terms of estimation quality. Although there exist various state-of-the-art algorithms for computing such estimator, the orders to magnitude of our problem dictates the need of a faster algorithm. Taking advantage of the convolutional structure of the problem, the sliding window working set algorithm can analyze such large signals in a fast and efficient way, as well as satisfying interesting mathematical properties. In particular, we saw that under reasonable mathematical assumptions, the Lasso estimator estimates correctly the support of the true activation vectors, with high probability. Moreover, taking into account the sizes of the temporal and spatial overlaps, we could establish the theoretical temporal complexity of the sliding window working set algorithm for the computation of the Lasso. We showed in particular that this algorithm scales quasi-linearly with the length of the experiment. This favorable behavior has also been observed on numerical experiments. As such, it deals with some limitations of the current spike sorting methodology. We believe that the approach we developed has the potential to help achieving fast and precise spike sorting, in particular for signals recorded by large multi-electrode arrays.

In future works, the natural extension of our approach would focus on the simultaneous estimation of the spike shapes and the activations. It should especially look for an update of the shapes along time, in an online manner. This extension would therefore permit to take into account progressing variations of the shapes during the experiment, for instance during bursting episodes or because of the electrode drift phenomenon. As such, our approach would gain a wider range of applications, provided that its extension also verifies nice statistical properties, in particular the statistical identifiability of the problem.

Another extension could revolve around the substitution of the ℓ_1-norm in the Lasso problem by a non-convex penalization term. These types of approaches are known to often provide faster computations. Moreover, recent works from [Rakotomamonjy et al., 2021] demonstrated theoretical convergence guarantees for non-convex sparse regularizers.

From a more practical point of view, our approach would need to be

experimented on real data, in various settings: tetrode recordings, MEA recordings, different areas of the brain and different types of neurons recorded, in order to examine its robustness, and in particular the domain of validity of our mathematical assumptions. Moreover, since the sliding window working set uses linear operations such as convolutions, it could be adapted to GPU architectures in order to provide even faster computation times. Finally, we believe that it would be interesting to consider a joint collaboration with practitioners for who the online aspect of our estimation procedure is critical in their studies. Devices working with real time data such as brain-machine interfaces could indeed take advantage of the rapidity of the sliding window working set.

Bibliography

[Adler et al., 2015] Adler, A., Elad, M., Hel-Or, Y., and Rivlin, E. (2015). Sparse coding with anomaly detection. *Journal of Signal Processing Systems*, 79(2):179–188.

[Albert et al., 2016] Albert, M., Bouret, Y., Fromont, M., and Reynaud-Bouret, P. (2016). Surrogate data methods based on a shuffling of the trials for synchrony detection : the centering issue. *Neural Computation*, 28(11):2352–2392.

[Bach, 2008] Bach, F. (2008). Exploring large feature spaces with hierarchical multiple kernel learning. *arXiv preprint arXiv:0809.1493*.

[Bach et al., 2011] Bach, F., Jenatton, R., Mairal, J., Obozinski, G., et al. (2011). Convex optimization with sparsity-inducing norms. *Optimization for Machine Learning*, 5:19–53.

[Bar-Hillel et al., 2006] Bar-Hillel, A., Spiro, A., and Stark, E. (2006). Spike sorting: Bayesian clustering of non-stationary data. *Journal of neuroscience methods*, 157(2):303–316.

[Beck and Teboulle, 2009] Beck, A. and Teboulle, M. (2009). A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202.

[Bethus et al., 2012] Bethus, I., Poucet, B., and Sargolini, F. (2012). Neural correlates of goal-directed spatial navigation in the rat dorsal striatum. In *Forum of European Neuroscience*, Barcelona, Spain.

[Bickel et al., 2009] Bickel, P., Ritov, Y., and Tsybakov, A. (2009). Simultaneous analysis of lasso and dantzig selector. *Annals of Statistics*, 37(4):1705–1732.

[Biffi et al., 2013] Biffi, E., Regalia, G., Menegon, A., Ferrigno, G., and Pedrocchi, A. (2013). The influence of neuronal density and maturation on network activity of hippocampal cell cultures: a methodological study. *Plos one*, 8(12):e83899.

[Bioucas-Dias and Figueiredo, 2007] Bioucas-Dias, J. M. and Figueiredo, M. A. (2007). A new twist: Two-step iterative shrinkage/thresholding algorithms for image restoration. *IEEE Transactions on Image processing*, 16(12):2992–3004.

[Blaustein et al., 2011] Blaustein, M. P., Kao, J. P., and Matteson, D. R. (2011). *Cellular Physiology and Neurophysiology E-Book: Mosby Physiology Monograph Series*. Elsevier Health Sciences.

[Boisbunon et al., 2014] Boisbunon, A., Flamary, R., Rakotomamonjy, A., Giros, A., and Zerubia, J. (2014). Large scale sparse optimization for object detection in high resolution images. In *MLSP-24th IEEE Workshop on Machine Learning for Signal Processing*.

[Bredies and Lorenz, 2008] Bredies, K. and Lorenz, D. A. (2008). Linear convergence of iterative soft-thresholding. *Journal of Fourier Analysis and Applications*, 14(5-6):813–837.

[Breiman, 1996] Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2):123–140.

[Bühlmann and Van De Geer, 2011] Bühlmann, P. and Van De Geer, S. (2011). *Statistics for high-dimensional data: methods, theory and applications*. Springer Science & Business Media.

[Bunea, 2008] Bunea, F. (2008). Honest variable selection in linear and logistic regression models via l1 and l1 + l2 penalization. *Electron. J. Statist.*, 2:1153–1194.

[Byrne et al., 2014] Byrne, J. H., Heidelberger, R., and Waxham, M. N. (2014). *From molecules to networks: an introduction to cellular and molecular neuroscience*. Academic Press.

[Chalasani et al., 2013] Chalasani, R., Principe, J. C., and Ramakrishnan, N. (2013). A fast proximal method for convolutional sparse coding. In *The 2013 International Joint Conference on Neural Networks (IJCNN)*, pages 1–5. IEEE.

[Chambolle and Dossal, 2015] Chambolle, A. and Dossal, C. (2015). On the convergence of the iterates of the "fast iterative shrinkage/thresholding algorithm". *Journal of Optimization theory and Applications*, 166(3):968–982.

[Combettes and Pesquet, 2011] Combettes, P. L. and Pesquet, J.-C. (2011). Proximal splitting methods in signal processing. In *Fixed-point algorithms for inverse problems in science and engineering*, pages 185–212. Springer.

[Dayan and Abbott, 2001] Dayan, P. and Abbott, L. F. (2001). *Theoretical neuroscience: computational and mathematical modeling of neural systems*. MIT press.

[Donoho, 1995] Donoho, D. L. (1995). De-noising by soft-thresholding. *IEEE transactions on information theory*, 41(3):613–627.

[Duminil-Copin et al., 2018] Duminil-Copin, H., Raoufi, A., and Tassion, V. (2018). Subcritical phase of d-dimensional poisson-boolean percolation and its vacant set.

[Efron et al., 2004] Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R. (2004). Least angle regression. *The Annals of statistics*, 32(2):407–499.

[Einevoll et al., 2012] Einevoll, G. T., Franke, F., Hagen, E., Pouzat, C., and Harris, K. D. (2012). Towards reliable spike-train recordings from thousands of neurons with multielectrodes. *Current opinion in neurobiology*, 22(1):11–17.

[Ekanadham et al., 2011] Ekanadham, C., Tranchina, D., and Simoncelli, E. P. (2011). Recovery of sparse translation-invariant signals with continuous basis pursuit. *IEEE transactions on signal processing*, 59(10):4735–4744.

[Ekanadham et al., 2014] Ekanadham, C., Tranchina, D., and Simoncelli, E. P. (2014). A unified framework and method for automatic neural spike identification. *Journal of neuroscience methods*, 222:47–55.

[Eytan and Marom, 2006] Eytan, D. and Marom, S. (2006). Dynamics and effective topology underlying synchronization in networks of cortical neurons. *Journal of Neuroscience*, 26(33):8465–8476.

[Fee et al., 1996] Fee, M. S., Mitra, P. P., and Kleinfeld, D. (1996). Automatic sorting of multiple unit neuronal signals in the presence of anisotropic and non-gaussian variability. *Journal of neuroscience methods*, 69(2):175–188.

[Fejtl et al., 2006] Fejtl, M., Stett, A., Nisch, W., Boven, K.-H., and Möller, A. (2006). On micro-electrode array revival: its development, sophistication of recording, and stimulation. In *Advances in network electrophysiology*, pages 24–37. Springer.

[Grosse et al., 2012] Grosse, R., Raina, R., Kwong, H., and Ng, A. Y. (2012). Shift-invariance sparse coding for audio classification. *arXiv preprint arXiv:1206.5241*.

[Harris et al., 2000] Harris, K. D., Henze, D. A., Csicsvari, J., Hirase, H., and Buzsaki, G. (2000). Accuracy of tetrode spike separation as determined by simultaneous intracellular and extracellular measurements. *Journal of neurophysiology*, 84(1):401–414.

[Hastie et al., 2008] Hastie, T., Tibshirani, R., and Friedman, J. (2008). *The elements of statistical learning Second Edition*. Springer series in statistics New York.

[Henze et al., 2000] Henze, D. A., Borhegyi, Z., Csicsvari, J., Mamiya, A., Harris, K. D., and Buzsaki, G. (2000). Intracellular features predicted

by extracellular recordings in the hippocampus in vivo. *Journal of neurophysiology*, 84(1):390–400.

[Hill et al., 2011] Hill, D. N., Mehta, S. B., and Kleinfeld, D. (2011). Quality metrics to accompany spike sorting of extracellular signals. *Journal of Neuroscience*, 31(24):8699–8705.

[Hillis et al., 2009] Hillis, D., Sadava, D., Berenbaum, M., and Heller, C. (2009). *Life The Science of Biology 9th edition*. WH Freeman and Company and Sinauer Associates, Inc.

[Hodgkin and Huxley, 1952] Hodgkin, A. L. and Huxley, A. F. (1952). A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of Physiology*, 117(4):500–544.

[Hodgkin and Katz, 1949] Hodgkin, A. L. and Katz, B. (1949). The effect of sodium ions on the electrical activity of the giant axon of the squid. *The Journal of physiology*, 108(1):37–77.

[Jas et al., 2017] Jas, M., La Tour, T. D., Şimşekli, U., and Gramfort, A. (2017). Learning the morphology of brain signals using alpha-stable convolutional sparse coding. *arXiv preprint arXiv:1705.08006*.

[Kandel et al., 2000] Kandel, E. R., Schwartz, J. H., Jessell, T. M., Siegelbaum, S., Hudspeth, A. J., and Mack, S. (2000). *Principles of neural science*, volume 4. McGraw-hill New York.

[Kavukcuoglu et al., 2010] Kavukcuoglu, K., Sermanet, P., Boureau, Y.-L., Gregor, K., Mathieu, M., Cun, Y., et al. (2010). Learning convolutional feature hierarchies for visual recognition. *Advances in neural information processing systems*, 23:1090–1098.

[La Tour et al., 2018] La Tour, T. D., Moreau, T., Jas, M., and Gramfort, A. (2018). Multivariate convolutional sparse coding for electromagnetic brain signals. *arXiv preprint arXiv:1805.09654*.

[Lambert et al., 2018] Lambert, R., Tuleau-Malot, C., Bessaih, T., Rivoirard, V., Bouret, Y., Leresche, N., and Reynaud-Bouret, P. (2018). Reconstructing the functional connectivity of multiple spike trains using hawkes models. *Journal of Neuroscience Methods*, 297:9–21.

[Lee et al., 2007] Lee, H., Battle, A., Raina, R., and Ng, A. Y. (2007). Efficient sparse coding algorithms. In *Advances in neural information processing systems*, pages 801–808.

[Lewicki, 1998] Lewicki, M. S. (1998). A review of methods for spike sorting: the detection and classification of neural action potentials. *Network: Computation in Neural Systems*, 9(4):R53–R78.

[Ling and Gerard, 1949] Ling, G. and Gerard, R. (1949). The normal membrane potential of frog sartorius fibers. *Journal of cellular and comparative physiology*, 34(3):383–396.

[Loth, 2011] Loth, M. (2011). *Active Set Algorithms for the LASSO.(Algorithmes d'Ensemble Actif pour le LASSO).* PhD thesis, Lille University of Science and Technology, France.

[Lounici, 2008] Lounici, K. (2008). Sup-norm convergence rate and sign concentration property of lasso and dantzig estimators. *Electron. J. Statist.*, 2:90–102.

[Luo, 2015] Luo, L. (2015). *Principles of neurobiology*. Garland Science.

[Mailhé et al., 2008] Mailhé, B., Lesage, S., Gribonval, R., Bimbot, F., and Vandergheynst, P. (2008). Shift-invariant dictionary learning for sparse representations: extending k-svd. In *2008 16th European Signal Processing Conference*, pages 1–5. IEEE.

[Mairal et al., 2010] Mairal, J., Bach, F., Ponce, J., and Sapiro, G. (2010). Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research*, 11(1).

[Mairal et al., 2014] Mairal, J., Bach, F., Ponce, J., Sapiro, G., Jenatton, R., and Obozinski, G. (2014). Spams: A sparse modeling software. *URL http://spams-devel. gforge. inria. fr/downloads. html*.

[Mallat, 1999] Mallat, S. (1999). *A wavelet tour of signal processing*. Elsevier.

[McNaughton et al., 1983] McNaughton, B. L., O'Keefe, J., and Barnes, C. A. (1983). The stereotrode: a new technique for simultaneous isolation of several single units in the central nervous system from multiple unit records. *Journal of neuroscience methods*, 8(4):391–397.

[Meester and Roy, 1996] Meester, R. and Roy, R. (1996). *Continuum percolation*, volume 119. Cambridge University Press.

[Millet et al., 2011] Millet, L. J., Collens, M. B., Perry, G. L., and Bashir, R. (2011). Pattern analysis and spatial distribution of neurons in culture. *Integrative Biology*, 3(12):1167–1178.

[Moreau, 1962] Moreau, J. J. (1962). Fonctions convexes duales et points proximaux dans un espace hilbertien. *Comptes rendus hebdomadaires des séances de l'Académie des sciences*, 255:2897–2899.

[Moreau et al., 2018] Moreau, T., Oudre, L., and Vayatis, N. (2018). Dicod: Distributed convolutional coordinate descent for convolutional sparse coding. In *International Conference on Machine Learning*, pages 3626–3634. PMLR.

[Muthmann et al., 2015] Muthmann, J.-O., Amin, H., Sernagor, E., Maccione, A., Panas, D., Berdondini, L., Bhalla, U. S., and Hennig, M. H. (2015). Spike detection for large neural populations using high density multielectrode arrays. *Frontiers in neuroinformatics*, 9:28.

[Nesterov, 2013] Nesterov, Y. (2013). Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1):125–161.

[Nicholls et al., 2001] Nicholls, J. G., Martin, A. R., Wallace, B. G., and Fuchs, P. A. (2001). *From neuron to brain*, volume 271. Sinauer Associates Sunderland, MA.

[Nutini et al., 2015] Nutini, J., Schmidt, M., Laradji, I., Friedlander, M., and Koepke, H. (2015). Coordinate descent converges faster with the gauss-southwell rule than random selection. In *International Conference on Machine Learning*, pages 1632–1641. PMLR.

[Obozinski et al., 2010] Obozinski, G., Taskar, B., and Jordan, M. I. (2010). Joint covariate selection and joint subspace selection for multiple classification problems. *Statistics and Computing*, 20(2):231–252.

[Pachitariu et al., 2013] Pachitariu, M., Packer, A. M., Pettit, N., Dalgleish, H., Hausser, M., and Sahani, M. (2013). Extracting regions of interest from biological images with convolutional sparse block coding. *Advances in neural information processing systems*, 26:1745–1753.

[Plonsey et al., 2007] Plonsey, R., Barr, R. C., and Bioelectricity, A. (2007). *Quantitative Approach*. Springer.

[Pouzat, 2016] Pouzat, C. (2016). Origin of the (high frequency) extracellular signal. `http://christophe-pouzat.github.io/LASCON2016/OriginOfTheHighFrequencyExtraCellularSignal.html`.

[Pouzat et al., 2004] Pouzat, C., Delescluse, M., Viot, P., and Diebolt, J. (2004). Improved spike-sorting by modeling firing statistics and burst-dependent spike amplitude attenuation: a markov chain monte carlo approach. *Journal of neurophysiology*, 91(6):2910–2928.

[Pouzat and Detorakis, 2014] Pouzat, C. and Detorakis, G. (2014). Spysort: Neuronal spike sorting with python. *CoRR*, abs/1412.6383.

[Purves et al., 2018] Purves, D., Augustine, G., Fitzpatrick, D., Hall, W., LaMantia, A., Mooney, R., and White, L. (2018). *Neuroscience*. Sinauer.

[Rakotomamonjy et al., 2021] Rakotomamonjy, A., Flamary, R., Gasso, G., and Salmon, J. (2021). Provably convergent working set algorithm for non-convex regularized regression.

[Reynaud-Bouret, 2003] Reynaud-Bouret, P. (2003). Adaptive estimation of the intensity of inhomogeneous poisson process via concentration inequalities. *Probab. Theory related Fields*, 126(1):103–153.

[Reynaud-Bouret et al., 2014] Reynaud-Bouret, P., Rivoirard, V., Grammont, F., and Tuleau-Malot, C. (2014). Goodness-of-fit tests and nonparametric adaptive estimation for spike train analysis. *The Journal of Mathematical Neuroscience*, 4(1):1–41.

[Roberts, 1979] Roberts, W. M. (1979). Optimal recognition of neuronal waveforms. *Biological cybernetics*, 35(2):73–80.

[Roth and Fischer, 2008] Roth, V. and Fischer, B. (2008). The group-lasso for generalized linear models: uniqueness of solutions and efficient algorithms. In *Proceedings of the 25th international conference on Machine learning*, pages 848–855.

[Smaragdis, 2007] Smaragdis, P. (2007). Convolutive speech bases and their application to supervised speech separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(1):1–12.

[Squire et al., 2012] Squire, L., Berg, D., Bloom, F. E., Du Lac, S., Ghosh, A., and Spitzer, N. C. (2012). *Fundamental neuroscience*. Academic Press.

[Taketani and Baudry, 2010] Taketani, M. and Baudry, M. (2010). *Advances in network electrophysiology*. Springer.

[Taylor et al., 1979] Taylor, H. L., Banks, S. C., and McCoy, J. F. (1979). Deconvolution with the l1 norm. *Geophysics*, 44(1):39–52.

[Thomas Jr et al., 1972] Thomas Jr, C., Springer, P., Loeb, G., Berwald-Netter, Y., and Okun, L. (1972). A miniature microelectrode array to monitor the bioelectric activity of cultured cells. *Experimental cell research*, 74(1):61–66.

[Tibshirani, 1996] Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288.

[Tuleau-Malot et al., 2014] Tuleau-Malot, C., Rouis, A., Grammont, F., and Reynaud-Bouret, P. (2014). Multiple tests based on a gaussian approximation of the unitary events method with delayed coincidence count. *Neural computation*, 26(7):1408–1454.

[Welch, 1982] Welch, W. J. (1982). Algorithmic complexity: three np-hard problems in computational statistics. *Journal of Statistical Computation and Simulation*, 15(1):17–25.

[Whitson et al., 2006] Whitson, J., Kubota, D., Shimono, K., Jia, Y., and Taketani, M. (2006). Multi-electrode arrays: Enhancing traditional methods and enabling network physiology. In *Advances in Network Electrophysiology*, pages 38–68. Springer.

[Wohlberg, 2015] Wohlberg, B. (2015). Efficient algorithms for convolutional sparse representations. *IEEE Transactions on Image Processing*, 25(1):301–315.

[Wood et al., 2004] Wood, F., Black, M. J., Vargas-Irwin, C., Fellows, M., and Donoghue, J. P. (2004). On the variability of manual spike sorting. *IEEE Transactions on Biomedical Engineering*, 51(6):912–918.

[Zeiler et al., 2010] Zeiler, M. D., Krishnan, D., Taylor, G. W., and Fergus, R. (2010). Deconvolutional networks. In *2010 IEEE Computer Society Conference on computer vision and pattern recognition*, pages 2528–2535. IEEE.