BAYESIAN CLUSTERING OF NON-STATIONARY DATA

A thesis submitted in partial fulfillment of the requirements for

the degree of

Master of Science

by

Adam Spiro

Supervised by

Prof. Daphna Weinshall

May 2005

School of Computer Science and Engineering

The Hebrew University of Jerusalem, Israel

Acknowledgements

I wish to thank Prof. Daphna Weinshall for her guidance, help and continuing support that enabled me to fulfill this research and motivated me all through the process.

I owe special thanks to Aharon Bar-Hillel who without I could not carry out this research. My gratitude is for his patience and cooperation throughout this period, and his participation in the writing, especially of Section 4.

I would also like to thank Eran Stark for valuable discussions and ideas and for providing us with the neuronal data.

Thanks to Itay Asher and Prof. Moshe Abeles for their hard work in manual data clustering and for their insightful comments.

Abstract

Non-stationary data clustering is a hard, ill-posed problem, which is nevertheless unavoidable in several scientific fields. A representative example is the problem of Spike Sorting, which involves clustering spike trains recorded from the brain by a micro-electrode, according to source neuron. It is a complicated problem which requires a lot of human labor, partly due to the non-stationary nature of the data. We propose an automatic clustering process using a Bayesian framework, with the relevant clusters modeled as a non-stationary mixture of Gaussians. At a first search stage, data is divided into short time frames and candidate descriptions of the data as a mixture of Gaussians are computed for each frame separately. At a second stage transition probabilities between candidate mixtures are computed, and a globally optimal clustering is found as the MAP solution of the resulting probabilistic model. Transition probabilities are computed using local stationarity assumptions and are based on a Gaussian version of the Jensen-Shannon divergence. The method was tested on 46 recordings including movement, merges and splits of clusters, and exhibited clustering behavior similar to a human expert. In another experiment the agreement between automated clustering and 3 human experts was shown to be comparable to the agreement among the experts. If needed, the method can adapt to the preferences of a specific human expert with minimal human supervision. In addition to the Spike Sorting application the algorithm was also tested on the problem of tracking storm cells of hurricane in satellite images. In this domain we successfully identify and track the main storm cell in several hard sequences and capture the essential clusters and their dynamic aspects.

תקציר

תיוג לצבירים של נתונים שאינם סטציונרים (כלומר, נתונים שזזים עם הזמן) הינה בעיה קשה, שלעתים אף אינה מוגדרת היטב. עם זאת, לא אחת נתקלים בבעיה מעין זו בתחומים מדעיים שונים.
דוגמא מייצגת לכך הינה הבעיה של "מיון אותות עצביים" (Spike Sorting) שמקורם במוח. בבעיה זו נדרש מיון של אותות עצביים רבים, אשר הוקלטו על ידי מיקרו-אלקטרודה, על פי תא העצב אשר יצר את האות. הבעיה הינה מסובכת ודורשת מלאכה ידנית של שעות רבות, בין השאר בגלל האופי הלא סטציונרי שלה.
במאמר, מוצגת שיטת מיון אוטומטית לצבירים, אשר נעשית במסגרת בייסיאנית, כאשר הנתונים מיוצגים באמצעות גאוסיאנים שיכולים להשתנות עם הזמן. בשלב הראשון של האלגוריתם, הנתונים מחולקים לחלונות זמן קטנים ולכל חלון זמן בנפרד, מחושבים מועמדים לתיאור הנתונים כתערובת של גאוסיאנים. בשלב השני, מחושבות הסתברויות מעבר בין המועמדים השונים בחלונות זמן עוקבים, ובעזרתן מחשבים פיתרון כולל על ידי מציאת מקסימום אפוסטריורי של המודל ההסתברותי. הסתברויות המעבר בין התערובות מחושבות בעזרת הנחות של סטציונריות מקומית ומבוססות על גירסא גאוסיאנית של מדד ג'נסן-שאנון (Jensen-Shannon Divergence).
השיטה נוסתה על נתונים מ-46 אלקטרודות שונות, שכללו תזוזות, איחודים ופיצולים של צבירים, ונמצא כי קיים דמיון ניכר בין המיון האוטומטי למיון הידני של אדם מומחה. בניסוי נוסף, מוינו אותם נתונים על ידי שלושה מומחים שונים, ונמצא כי לא ניתן להבדיל בין המיון האוטומטי למיונים הידניים.
במידת הצורך, ניתן לעשות שימוש אוטומטי-למחצה בשיטה, על ידי מיון ידני של מספר מועט של חלונות זמן והשלמה אוטומטית של שאר החלונות. בכך, יכולה השיטה להתאים עצמה להעדפות או סגנונות מיון אישיות.
בנוסף ליישום זה, נוסתה השיטה גם על הבעיה של מיון ומעקב אחר עננים בסופות הוריקן, אשר צולמו מלוויין. במקרה זה, האלגוריתם הצליח לזהות ולעקוב אחרי הצביר הראשי של עין הסופה ולקלוט את המאפיינים הדינמיים של שאר הצבירים.

Contents

1 Introduction

Clustering can be seen as one of the most important unsupervised learning problems. It is essentially an ill-posed problem, yet unavoidable in many domains, and is used as a main tool in data mining and analysis of unlabeled data. The clusters found can be useful for domain understanding and hypothesis generation, and the assignments of individual items into clusters can provide valuable predictions. See [6, 12] for background on traditional clustering methods and concepts.

In the clustering literature problems of non-stationary data are rarely addressed. However, this problem arises in several distinct scientific contexts, and has prompted extensive task-specific research. One prominent example is the problem of clustering extra-cellular spike recordings [20, 7]. In this case neuronal spikes from several neurons are recorded using a single electrode, and clustering is required to identify the source of each spike. Small drifts of the electrode cause changes in the recorded spike shapes, creating the problem of non-stationarity. Since existing algorithms are not satisfactory, clustering is typically done manually by human experts and requires many days of human labor. Another example is the problem of identifying and tracking storm cells in satellite imagery [13, 15], where the image is dynamic, noisy, and often ambiguous. This problem is far from being solved as well. Similar problems appear in other scenarios which involve tracking of multiple objects in cluttered scenes [16] or online non-stationary clustering [10].

The examples above have several difficulties in common. Clusters in these examples can move or perform splits and merges. The number of clusters is unknown a priori and often changes over time. These features highlight the fuzziness and ambiguity of the partition, leading to a problem that is even more ill-posed than traditional clustering. Despite this ill-posedness, clustering of non-stationary data is still useful. Humans achieve fairly high consistency, and useful predictions can be made. An important factor in the success of human clustering is the ability to use information from past and future time frames to disambiguate data partition at a specific frame.

Basic similarities exist between non-stationary clustering and multi-hypothesis tracking [2]. However, the added 'clustering' element of the first makes it harder. Our goal is not only to track an object, but also to assign each data point to a cluster. In addition, we usually do not have a simple motion model for the clusters, as is commonplace in multi-hypothesis tracking. In the examples above, cluster movement is mostly unpredictable and hard to model, especially when it comes to splits and merges. Furthermore, when clustering is done as an exploratory tool, it is preferable not to enforce an a priori motion model of any kind.

In this paper we suggest a new fully automated technique to solve the clustering problem for non-stationary sources in a Bayesian framework. We

divide the data into short time frames in which stationarity is a reasonable assumption. We then look for good mixture of Gaussians descriptions of the data in each time frame independently. Transition probabilities between local mixture solutions are introduced, and a globally optimal clustering solution is computed by finding the Maximum-A-Posteriori (MAP) solution of the resulting probabilistic model. The global optimization allows the algorithm to successfully disambiguate problematic time frames.

The main obstacle in this approach is the need for a probabilistic account of transitions between mixture models. In Section 4 we suggest basing transition probabilities on the differences between 'types' of samples in the two consecutive time frames. To this end, we extend concepts from the 'method of types' [4] to mixtures models with components from the exponential family. We derive a large sample approximation of the transition probability and pose its computation as a relatively small, discrete optimization problem for which we suggest an algorithmic solution. Although we use Gaussian mixtures in our experiments, all our theoretical work applies to general mixture models with components in the exponential family.

Our main effort and the original reason for which this research began concerns the Spike Sorting problem. We thus bring in Section 2 an overview of that problem and current methods for handling it.

In Section 3 we describe the general framework of the algorithm. Throughout this section we will concentrate on the Spike Sorting application and deal with domain specific issues and problems, especially when it comes to pre- and post-processing.

Section 4 describes the mathematical derivation of the transition probabilities between mixtures of consecutive time frames.

In Section 5 we bring the experimental results. Since our initial goal was to design an automatic Spike Sorting tool and mimic human experts, we conducted extensive experiments for comparing our clustering results to the human experts. In Section 5.1 we show that the partitions suggested by the algorithm have a high agreement with manual partitions conducted by human experts. This agreement was comparable to the average agreement between experts, so in this sense our method passes a local 'Turing test'. In Section 5.2 we present experiments regarding Storm Tracking which is an additional non-stationary phenomenon. In this domain we show that our method can successfully identify and track the main storm cell in several hard sequences. In both cases, the algorithm captures the essential clusters and their dynamic aspects.

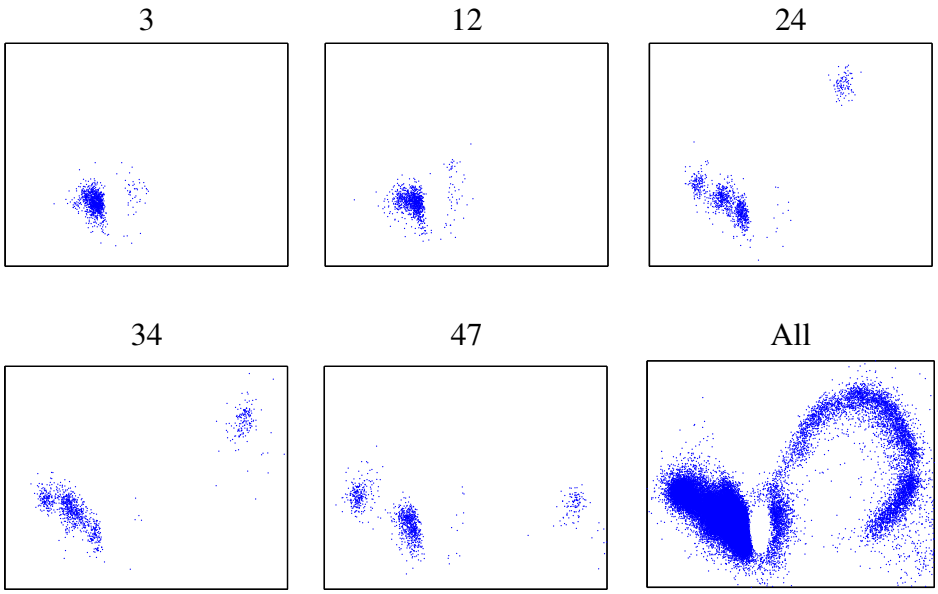


Figure 1: The figures present several time frames from a single electrode recording. 5 time frames from a sequence of 68 are presented with their frame numbers, and the bottom right figure shows all the data in a single frame. Spikes are extracted and projected on the first two eigenvectors. In this recording one cluster moves constantly and another splits into distinguished clusters, which turn partially indistinguishable again toward the end. While single time frames may have several plausible clustering interpretations, the sequence as a whole is usually less ambiguous. Clearly the 'right' clustering cannot be seen when all the data is grouped in a single frame.

2 The Spike Sorting Problem

In extracellular recording of brain activity the micro electrode normally picks up the activity of multiple neurons. Spike Sorting is the task of finding a clustering of the spike data such that each cluster contains the spikes generated by a different neuron. Currently, this task is mostly done manually. It is a tedious mission, requiring many hours of human work for a single recording session. Many algorithms were proposed in order to help automate this process (see [20] for a review), and some were implemented to provide a helping tool for manual sorting ([19],[11]). However, the ability of suggested algorithms to replace the human worker has been quite limited.

One of the main obstacles for a successful application is the non-stationary nature of the data [20]. The primary source of this non-stationarity is slight movements of the recording electrode. Slight drifts of the electrode's location, which are almost inevitable, cause changes in the typical shapes of recorded spikes over time. Other sources of non-stationarity are variable background noise and changes in the characterization of the spike generated by a neuron. The increasing usage of multiple electrode recordings turns non-stationarity into an acute problem, since electrodes are placed in a single location for long durations.

Manual sorting usually uses the first two Principal Components (PC) coefficients to represent spike data. Such a representation preserves up to 93% of the variance of the recorded spikes [1]. A human typically clusters the spikes by visual inspection of the projected spikes in small time frames, in which non stationary effects are not significant. Problematic scenarios which can appear due to non-stationarity are exemplified in Figure 1 and include: (1) Movements and considerable shape changes of the clusters over time. Such movements cause the cluster to be smeared when observed at a large time window. (2) Two clusters which are initially well-separated may move until they converge and become indistinguishable. A split of a single cluster is possible in the same manner. We would like to distinguish between different clusters whenever possible, and state that a cluster is a 'multi-unit' cluster when separation is not possible anymore.

Most Spike Sorting algorithms do not address the presented difficulties at all, as they assume full stationarity of the data. Fee et al. [8] assumes a single frame containing some noisy data and uses a hierarchical clustering scheme. Initially the data is sorted into an overly large number of clusters by recursive bisection and then progressively aggregated into a minimal set of putative single units. Two criteria are used to unite two clusters: (1) the pair should have many points on the shared borderline of the clusters, and (2) the unified cluster should have an inter-spike interval (ISI - the minimal amount of time interval of which a neuron can fire again) distribution similar to the two constituent clusters, without short ISI's prohibited by the refractory period. Snider et al. [24] point out that the ISI criterion assumes independence of spike times for different neurons, and is thus invalid when neurons are correlated. Instead, they unite clusters if the density of points falling between them changes linearly. Shoham et al. [23] suggest to handle non-stationary clusters by modeling clusters using a t-distribution. This distribution can partially describe a-symmetric, smeared clusters. All these methods do not partition the data into time frames and hence cannot cope with complicated cases that require the information found within these frames. For example, clusters with similar shape, existing at distant time frames are always unified by such methods, while precise tracking of the clusters reveals that they are separated.

Emondi et al. [7] suggested a semi-automated method in which the data is divided into small time frames. Each time frame is clustered manually, and then the correspondence between clusters in consecutive time frames is established automatically. In order to establish the correspondence, match scores between clusters are computed using a heuristic metric, and a greedy procedure is then used to choose matching pairs. No splits or merges of clusters are considered in this framework.

In the next sections we solve the clustering problem in a Bayesian framework. Trying to mimic the human experts, we focus on the two dimensional spike representation, although our main algorithm can be extended to higher dimensions.

We tested the method performance using 46 electrode recordings from the pre-motor area of Macaque monkeys. The automatic clustering was evaluated by computing its agreement with manual clustering done by a human expert. In most cases, high agreement rates where found. In a second experiment we compared the agreement between the automatic and human clustering to the agreement among 3 human experts. The algorithm was comparable to the human experts in this test, passing a local 'Turing test'. Finally we tested the algorithm in a semi-supervised scenario, in which the human expert clusters one out of 5 or 10 time frames, and the algorithm fills in the rest. In this mode the clustering results highly agree with the human expert intentions, while the human labor required is reduced significantly.

In Section 5.1 we bring the full details of the experimental results regarding the Spike Sorting application.

3 A Chain of Gaussian Mixtures

A typical recording contains from tens to hundreds of thousands of spikes recorded in several hours. While spike clusters are not necessarily stationary over such periods, it is reasonable to assume stationarity for shorter periods of several minutes. In short periods it is known [20] that the cluster density can be well approximated by a Gaussian with a general covariance matrix. We therefore approach the problem by a two stage process. First, the data is divided into time frames with fixed length ΔT or fixed spike count N (in the presented experiments we used $N = 1000$). We look for a set of likely mixture of Gaussians solutions for each time frame independently. This search stage is heuristic in nature, and its purpose is to find good local solution candidates which constrain the search space in the second, global stage. In the second stage we choose the best combination of local solutions by finding the MAP solution in a global generative model of the complete data. These two algorithmic stages which are the core of the algorithm are described in Sections 3.1 and 3.2 respectively. In Section 3.3 we describe how repetition of local and global optimization stages in a feedback loop can be used to refine the solution. Another performance improvement is gained by introducing a rule-based post processing as described in Section 3.4. The algorithm's flow is described in Algorithm 1.

3.1 Local Solutions

Denote the observable spike data by $D = \{d\}$, where each spike $d \in R^2$ is described by the vector of its first two PCA coefficients. We break the data into T disjoint groups $\{D_t = \{d_i^t\}_{i=1}^{N_t}\}_{t=1}^{T}$. We assume that at each frame, the data can be well approximated by a mixture of Gaussians, where each Gaussian corresponds to a single neuron. Each Gaussian in the mixture

Algorithm 1 Spike Sorting

1: ***Divide the data into T time frames.***

2: ***Search for Local Solutions***
For each $t \in \{1...T\}$ search for M^t candidate mixtures using EM.

Loop step 3,4,5 for i=1,2

3: ***Remixing of Solutions***
For each $t \in \{1...T\}$ import and adapt selected solutions from neighboring frames [$t-k$,...,$t+k$].

4: ***Calculate Transition Scores***
For each $t \in \{1, T-1\}$ calculate scores between all candidates at time frame t and candidates of frame $t+1$.

5: ***Find MAP Solution***
Calculate the MAP path using the Viterbi algorithm.
if $i < 2$ use the found path to re-initialize local candidates pool.

6: ***Post Processing stage***
Identify special clusters.

may have a different covariance matrix. The number of components in the mixture is not known a priori, but is assumed to be within a certain range. Specifically we used 1 - 6 in our experiments.

In the search stage, we use a standard Expectation-Maximization (EM) algorithm [5] to find a set of M^t mixture of Gaussians candidates for each time frame t. We run the EM algorithm with different values for the 'number of clusters' parameter and different initial conditions. This creates the basic pool of solutions in each time frame. Then, we import to each time frame t the best mixture solutions found in the neighboring time frames $[t-k,..,t+k]$ (we used $k = 2$). These solutions are then adapted by using them as the initial conditions for the EM and running a low number of EM rounds. This mixing of solutions between time frames is repeated several times (2 in our experiments), so that good solutions are propagated forward and backward in time. Finally, the solution list at each time frame is pruned to remove similar solutions. Solutions which do not comply with the assumption of well shaped Gaussians are also removed. This is done by observing the actual labeling of the data induced by a certain Gaussian and checking how "far" the empirical distribution of the labeled points is from the Gaussian distribution. By using a threshold on the $Kullback-Leibler$ (D_{kl}) [3] distance between the two distributions we decide whether the specified solution should be pruned.

In order to handle outliers, which are usually background spikes or non-spike events, each mixture candidate contains an additional 'background

model' Gaussian. This model's parameters are set to $0, K \cdot \Sigma_t$ where Σ_t is the covariance matrix of the data at frame t and $K > 1$ is a constant. The weight is the only parameter of the background model that can change during the EM process.

When working in a semi-supervised mode, human guidance can be naturally incorporated at this stage. The human supervisor can provide local solutions to a subset $S \subset \{1..T\}$ of the T time frames. For each frame $s \in S$, the manual clustering is translated into a mixture of Gaussians and considered as the only candidate solution of time frame s. This constraints the algorithm to choose the manual clustering in the appropriate frames and enables the spreading of these solutions into the neighboring frames through the 'Remixing of Solutions' stage (stage 3 in Algorithm 1).

3.2 A Global Model

After the search stage, each time frame t has a candidate list of M^t models $\{\Theta_i^t\}_{t=1,i=1}^{T,M^t}$. Each mixture model is described by a triplet $\Theta_i^t = \{\alpha_{i,l}^t, \mu_{i,l}^t, \Sigma_{i,l}^t\}_{l=1}^{K_{i,t}}$, denoting the Gaussian mixture's weights, means, and covariances respectively. Given these candidate models we define a discrete random vector $Z = \{Z^t\}_{t=1}^T$ in which each component Z^t has a value range of $\{1, 2, .., M^t\}$. "$Z^t = j$" has the semantics of "at time frame t the data is distributed according to the candidate mixture Θ_j^t". In addition we define for each spike d_i^t a hidden discrete 'label' random variable l_i^t. This label indicates which Gaussian in the local mixture hypothesis is the source of the spike. Denote by $L^t = \{l_i^t\}_{i=1}^{N^t}$ the vector of labels at time frame t, and by L the vector of all the labels.

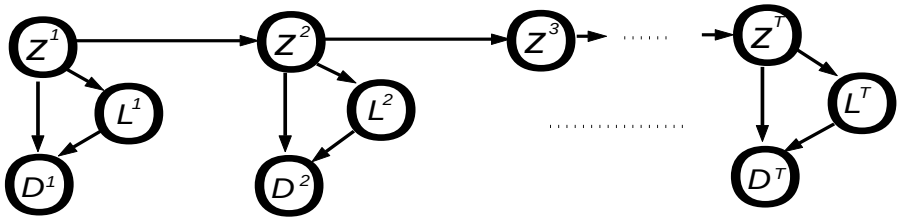


Figure 2: A Bayesian network model of the data generation process. The network has an HMM structure, but unlike HMM it does not have fixed states and transition probabilities over time. The variables and the CPDs are explained in the text.

We describe the probabilistic relations between D, L, and Z using a Bayesian network with the structure described in Figure 2. Using the network structure and assuming i.i.d samples the joint probability decomposes into

$$
\begin{aligned}
P(Z,L,D) &= P(Z)P(L|Z)P(D|Z,L) \\
&= P(Z^1)\prod_{t=2}^{T} P(Z^t|Z^{t-1})\prod_{t=1}^{T} P(L^t|Z^t)P(D^t|L^t,Z^t) \\
&= P(Z^1)\prod_{t=2}^{T} P(Z^t|Z^{t-1})\prod_{t=1}^{T}\prod_{i=1}^{N_t} P(l_i^t|Z^t)P(d_i^t|l_i^t,Z^t)
\end{aligned}
\tag{1}
$$

The complete log-likelihood is therefore given by

$$
\begin{aligned}
&\log P(Z^1) + \sum_{t=2}^{T} \log P(Z^t|Z^{t-1}) \\
&+ \sum_{t=1}^{T}\sum_{i=1}^{N^t} \left[\log P(l_i^t|Z^t) + \log P(d_i^t|l_i^t,Z^t)\right]
\end{aligned}
\tag{2}
$$

and we wish to maximize this log-likelihood over all possible choices of the hidden variables L, Z. Notice that by maximizing the probability of both data and labels we avoid the tendency to prefer mixtures with many Gaussians, which appear when maximizing the probability for the data alone. The conditional probability distributions (CPDs) of the points' labels and the points themselves, given an assignment to Z, are given by

$$
\begin{aligned}
&\log P(l_k^t = j|Z^t = i) = \log \alpha_{i,j}^t \\
&\log P(d_k^t|l_i^t = j, Z^t = i) = -\tfrac{1}{2}[n\log 2\pi + \log|\Sigma_{i,j}^t| \\
&\qquad\qquad + (d_k^t - \mu_{i,j}^t)^t {\Sigma_{i,j}^t}^{-1}(d_k^t - \mu_{i,j}^t)]
\end{aligned}
\tag{3}
$$

The transition probabilities $P(Z^t = i|Z^{t-1} = j)$ are computed based on the assumption that the visible mixtures Θ_i^t , Θ_j^{t-1} arise from a single mixture model with unknown parameters. The transition probability computation process includes the establishment of a correspondence mapping between clusters in the two frames. This idea is formalized and described in Section 4, and is generalized for the case of mixtures of a distribution from the exponential family. For the first frame's prior we use a uniform CPD.

The MAP solution for the model is found using the Viterbi algorithm. Labels can then be unified using the correspondences established between the chosen mixtures in consecutive time frames. When the correspondence is not one-to-one, as in the cases of merges and splits, labels are not unified. Instead, the merged cluster is identified as a multi-unit and receives a new label.

3.3 A Feedback Loop

The initial pool of solution computed at the local search stage is very diverse, but important local solutions may still be missing. This, in turn, damages the quality of the optimal chain found in the global stage. Once we

have found a good global chain candidate using the Viterbi algorithm, we can use it to guide the search for better local candidates. It is hence reasonable to create another, more 'focused' pool of solutions and repeat the global optimization.

In order to create such a focused candidate pool we set the initial pool of each time frame to contain a single solution - the one chosen by the global optimization. Then we repeat the solutions mixing stage several times. In addition, solutions with clusters that appear in the good global chain for only a small fraction of the time frames ($1/10$ in our implementation) are dropped from the candidate list. The new candidate pool at time frame t contains only solutions that appeared in neighboring time frames in the global path, and adaptations of these solutions.

The new 'search space' for the global optimization is now smaller, yet contains a rich collection of variations over the best path found so far. This enables a delicate refinement of the path and improves the final solution. A single repetition of the feedback loop is usually sufficient to obtain the desired result.

3.4 Post Processing

Using only the stages described so far, the algorithm gives reasonable results. However, in order to better mimic the human expert, additional processing is required. It seems that although the basic algorithm catches the main trends of the expert's behavior, the expert supplements this basic logic with many rules-of-thumb, that specifically address exceptional cases. These 'rules' are most useful when basic assumptions of the algorithm fail. To cope with this issue, we added a post processing stage in which several exceptional phenomena are handled. Two main problems, "Large scale" clusters and "Mirror" clusters are exemplified in Figure 3.

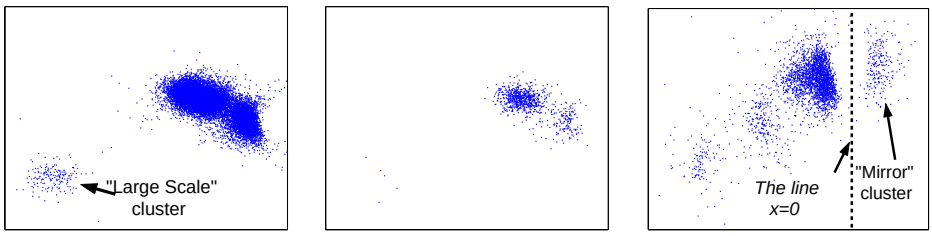


Figure 3: Two special cluster types are identified in the post processing stage. **($\mathbf{a}_1$)** View of the entire data in which the "Large scale" cluster can be observed. **($\mathbf{a}_2$)** A single time frame from the same data. Due to its sparseness, the "Large scale" cluster seen in (a_1) can not be detected in small time windows. **(b)** A "Mirror" cluster which is normally identified as a regular cluster, but has a mirror cluster symmetric to the line x=0.

"Large scale" clusters represent neurons with very low firing rate. Such neurons can not be detected within a small time frame, since such a frame

only contains a few spikes normally labeled as background. In order to detect them we look at all the data in a single time frame, and re-sample the data to reduce the weight of dense areas. We compute several descriptions of the sampled data as a mixture of Gaussians, using a various number of clusters. For each cluster in each description we compute the distance to other clusters in the mixture using the JS divergence (See Section 4). A candidate for a "Large scale" cluster is the cluster for which the distance to the nearest cluster in the mixture is maximal. The candidate is declared a "Large scale" cluster if most of its points (more than 75%) were classified as background in the main algorithm. This process enables us to locate most of the "Large scale" clusters (about 90%) found by humans with almost no spurious detections.

"Mirror" clusters are spurious clusters created due to a problem in the spike alignment. As described in Section 5.1.1, spikes are aligned by looking for the maximal fit of the waveform with 2 library PCs. The first, more influential PC is almost symmetric, including a large positive peak followed by a large negative one. Sometimes the best alignment is achieved when the positive peak of the first PC correlates with the negative peak of the waveform. In such cases the projection on the first PC is negated. This effect creates two clusters, symmetrical to the Y axis $x = 0$ which actually represent only one neuron. The "Mirror" cluster is usually much smaller and it should be either identified as background or labeled with the same label as the mirrored cluster. "Mirror" clusters are usually identified as regular clusters. Therefore, after having clustered the entire data, we check whether there is any cluster that has a mirror cluster symmetrical to the $x = 0$ line.

Abrupt large movements of the recording electrode or other extreme conditions produce spike recordings that are extremely non stationary. In such cases even our modest local stationarity assumptions do not hold, and the correspondence established between clusters in consecutive time frames is not reliable. Such data is usually dropped by the human sorter. We identify these cases by setting thresholds on the transition probabilities computed by the algorithm. We use 2 such thresholds. Transitions with probability lower than the first threshold are identified as 'not continuous'. For each data we compute the length of the longest sequence of continuous transitions. If this length is below a second threshold, the data is marked as extremely non stationary. About 10% of the recordings can be classified as extremely non stationary, and indeed the above classification method manages to correctly identify them in the test data. The results reported in Section 5.1.2 do not include the extremely non stationary data.

The post processing also includes detection of very small clusters with large covariance. Imitating human behavior, such clusters are merged with the background.

4 The Method of Types and Transition Probabilities

We suggest to derive transition probabilities $P(Z^t = i|Z^{t-1} = j)$ between two mixtures assuming a joint hidden mixture model underlying the two. The main concepts in this derivation are the 'Type' and 'Type class' of a sample. These are concepts from Information Theory, usually applied to discrete variables [4], that we extend to continuous variables from the exponential family. In Section 4.1 we briefly overview important properties of the exponential family, and extend the notions of 'Type' and 'Type class' based on these properties. These definitions are then extended to mixture models and labeled samples. In Section 4.2 we formalize the notion of a 'hidden mixture model'. Section 4.3 includes a main lemma and Section 4.4 uses the lemma to derive an expression of the transition probability. This latter expression involves a discrete optimization over the possible correspondence relations between the components of the two mixtures. Section 4.5 includes the derivation of that expression in the special case of a mixture of Gaussians, and in Section 4.6 we suggest algorithmic solutions for two interesting cases of this optimization problem.

4.1 Types and the Exponential Family

For a sample of i.i.d discrete variables, the concept of 'type' is defined in [4] as the multinomial distribution in which for every symbol a, $P(a)$ is the relative frequency of a in the sample. One of the important properties of a type is that it completely characterizes the sample and determines its likelihood under any other distribution from the multinomial family. However, such complete characterization exists for all the exponential families in the form of 'sufficient statistics'. A sufficient statistic is a vector of finite length which summarizes the sample completely with regard to likelihood assessment [18]. In this section we extend the 'type' concept by identifying it with the 'sufficient statistic' of a sample. In later sections we show that using such identification we can extend basic results of the method of types to continuous variables from the exponential families.

A distribution family $\mathcal{F} = \{f(x|\theta)\}_\theta$ is from the exponential family [18] if $f(x|\theta)$ can be written as

$$f(x|\theta) = \exp(\theta \cdot T(x) - A(\theta)) \tag{4}$$

where θ is a parameter vector (in the 'natural form' [18]) and $T(x)$ is a vector containing functions of the data. We use $Dom(\mathcal{F})$ to denote the domain of x. For an i.i.d sample $X = \{x_i\}_{i=1}^N \in Dom(\mathcal{F})^N$, we denote the empirical average of a function $g(x)$ by $\langle g(x) \rangle_X = \frac{1}{N} \sum_{i=1}^N g(x_i)$. Below are a few properties of the exponential family:

1. $A(\theta)$ is convex and $dA/d\theta = E_{p(x|\theta)}T(x)$ (see [18] for a proof).

2. Eq. (4) clearly implies that for a sample X, the likelihood is determined by

$$logf(X|\theta) = N[\ \theta \cdot \langle T(x)\rangle_X - A(\theta)\ ] \tag{5}$$

   The likelihood is determined by the single vector of averages. We thus define

   Definition 4.1. *The $\mathcal{F}$-type of a sample X is the average vector $T(X) = \langle T(x)\rangle_X$*

3. Differentiating Eq. (5) w.r.t θ, we can see that the ML parameters for a sample X with $U = T(X)$ are found by solving $dA/d\theta = U$. Since $A(\theta)$ is convex, there is a unique solution, and so we have an invertible function $\Theta(U)$ which gives the ML parameters for any input vector U. The inverse function $U(\Theta)$ gives (according to property 1) the expectation $E_{p(x|\theta)}T(x)$.

4. The expectation of $logf(x|\theta)$ w.r.t any distribution $q(x)$ can be computed as $\theta \cdot E_{q(x)}[T(x)] - A(\theta)$. Using this fact and property 1 we get

$$D_{kl}[f(x|\theta^1)||f(x|\theta^2)] = \frac{dA(\theta^1)}{d\theta^1} \cdot (\theta^1 - \theta^2) + A(\theta^2) - A(\theta^1) \tag{6}$$

Let us now consider labeled samples from mixture models. A mixture model over the family $\mathcal{F}$ is a model for a discrete label l and $x \in Dom(\mathcal{F})$ with the density $p(x,l) = p(l)f(x|\theta_l)$, where l has a multinomial distribution. The parameters of this model are $\Theta = \{\vec{\alpha}, \theta_1, .., \theta_K\}$, where $\vec{\alpha} = (\alpha_1, .., \alpha_K)$ are the weights and $\{\theta_j\}_{j=1}^K$ are the parameters of the K components of the mixture. A labeled sample is a fully observable sample of i.i.d data points with their labels $S = (X, L) = \{x_i, l_i\}_{i=1}^N$. Since the multinomial family is exponential, the density $p(x,l)$ is exponential too, as a product of two exponential densities. We can hence extend the definitions of sample type $T(X)$ and the ML function $\Theta(U)$ to mixtures and labeled samples. For a sample $S = (X, L)$ and mixture density Θ, we get the following expression for $\log p(S|\Theta)$

$$\begin{aligned}\log p(S|\Theta) &= \sum_{j=1}^{K} \sum_{\{i:l_i=j\}} \log \alpha_j + \theta_j \cdot T(x_i) - A(\theta_j) \\ &= N \sum_{j=1}^{K} \langle 1_{l=j}\rangle_s [\log \alpha_j + \theta_j \cdot \langle T(x)\rangle_{s_j} - A(\theta_j)]\end{aligned} \tag{7}$$

where $S_j = \{x_i | l_i = j\}$. The likelihood again depends on a vector of data averages. We define

Definition 4.2. *The sample type of a labeled sample* S *is* $T(S) = (\vec{\alpha}^s, U_1, .., U_K)$ *with* $\alpha_j^s =< 1_{l=j} >_s$, $U_j = T(\{x_i | l_i = j\})$.

Clearly the ML parameters of the sample are a function of $V = T(S)$. More specifically, the ML function is $\Theta(V) = (\vec{\alpha}^s, \Theta(U_1), .., \Theta(U_K))$.

A second central concept in the method of types is the type class, defined as the set of all N point samples with the same type. These are the basic events used in the theory. In our generalization, when x is continuous such events are of measure 0. We hence resort to the following concept

Definition 4.3. *The* (ϵ, N)*-type class of a type* U *is the set of samples*

$$TC_{N,\varepsilon}(U) = \{X \in Dom(\mathcal{F})^N : \ ||T(X) - U||_\infty < \varepsilon\} \tag{8}$$

And the definition extends trivially to (ϵ, N)-types of labeled samples.

The following lemma tells us that the type of large samples from a mixture Θ converges to the expected type $U(\Theta)$.

Lemma 1. *Let* S *be a labeled sample of size* N *from a mixture model* $\Theta = (\vec{\alpha}, \theta_1, .., \theta_K)$. *For* $0 < \epsilon < 1/2 \min_m \alpha_m$ *we have*

$$P(S \in TC_{N,\varepsilon}(U(\Theta)) \mid \Theta) > 1 - O(\frac{1}{N\varepsilon^2}) \tag{9}$$

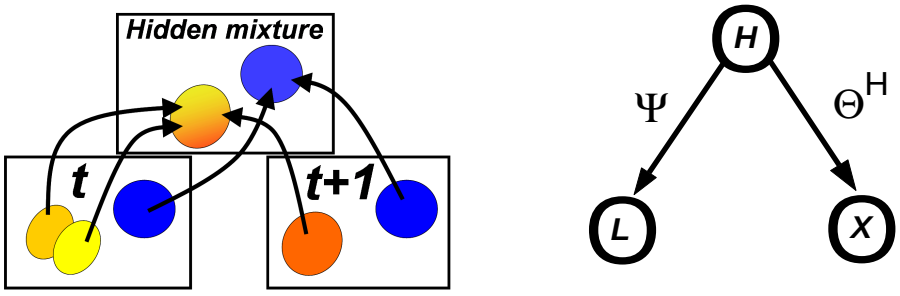
We omit the proof, which is a relatively simple use of the law of large numbers (Chebyshev's ineq.) to the averages of the statistics $T(x)$. As N grows, we are guaranteed to have these averages close to their expectation $U(\Theta)$.

4.2 A Hidden Mixture Model

Assume that in two consecutive time frames we saw two N point labeled samples. The joint sample $S = S^1 \cup S^2$ is a labeled sample with $K^1 + K^2$ possible labels. If we assume the data is approximately stationary in the two frames, then $K^1 \approx K^2$ and the data is actually generated from a hidden mixture with $K \approx K^1$ components (see Figure 4a). The visible labels in this case contain irrelevant 'noise', since points from the same hidden source are labeled with different visible labels in S^1 and in S^2. We capture this notion in the following definition

Definition 4.4. *A hidden mixture model* M^H *is a generative model for labeled data* (x, l) *where* $x \in Dom(\mathcal{F})$ *and* $l \in \{1, .., K\}$. *It is parameterized by a triplet* $M^H = (\Theta^H, R, \Psi)$ *where*

- $\Theta^H = \{\alpha_m^H, \theta_m^H\}_{m=1}^{K^H}$ *is a mixture model with* $K^H \leq K$ *components. The data* x *are generated from this mixture, as well as a hidden label* h *indicating the source of* x *in the hidden mixture.*



(a) (b)

Figure 4: **(a)** A diagram of the correspondence R between components of the Hidden and visible mixtures. The image represents a 'merge' event of the left bottom clusters. **(b)** A Bayesian network representation of the relations between the data and the hidden labels. The visible labels L and the sampled data points X are independent given the hidden labels H. The visible labels are deterministic given the hidden labels, and so we have $p(x,l) = p(h)p(l|h)p(x|h)$.

- *R is a function $R : \{1,..,K\} \rightarrow \{1,..,K^H\}$ which matches each visible label to its hidden source component in Θ^H (see Figure 4a). It thus induces a partition of the set of visible mixture components.*

- *$\Psi = \{\psi_j\}_{j=1}^K$. The visible labels l are sampled multinomially given the hidden labels h. Given a hidden label h, only visible labels $\{j : R(j) = h\}$ are possible. ψ_j is the parameter $P(l = j|h = R(j))$. For $h \neq R(j)$, $P(l = j|h) = 0$.*

Our probabilistic dependence assumptions between a data point, its visible label and its hidden label are captured by the Bayesian network model in Figure 4b. We assume a data point is sampled by choosing a hidden label, then sampling *in an independent manner* the point and the visible label from the relevant hidden component. In our main result (Theorem 3) we assess the probability of obtaining two different labeled samples S^1, S^2 from the same hidden model M^H with unknown parameters. Note that if the hidden model is to be non-trivial, we need $K^H < K^1 + K^2$ where K^1, K^2 are the number of labels in S^1, S^2. This way several different visible components will have to be explained by a single hidden component.

4.3 Main Lemma

In this section we consider type class probabilities under our suggested hidden model. A basic result in the method of types says that the probability $Q^N(T_N(P))$ of a type class $T_N(P)$ under a source Q approaches

$exp(-ND_{kl}(P||Q))$ for large N. We show a similar result for our case. Interestingly, $D_{kl}(P||Q)$ in our result is computed as though P and Q were from the family $\mathcal{F}$, even though we do not make such a demand on the density of P.

Lemma 2. *Let S be a labeled sample of N points from some arbitrary density, with $V = (\vec{\alpha}^s, U_1, .., U_K) = T(S)$. Let M^H be a hidden mixture model parameterized by $(\{\alpha_m^H\}_{m=1}^{K^H}, \{\psi_j\}_{j=1}^{K}, \{\theta_m^H\}_{m=1}^{K^H})$. Then*

$$\begin{aligned} &\lim_{\substack{N\to\infty,\varepsilon\to 0\\ N\varepsilon^2\to\infty}} \tfrac{1}{N}\log P(TC_{N,\varepsilon}(V)|M^H) \\ &= -D_{kl}[\vec{\alpha}^s||\alpha^H\psi] - \sum_{j=1}^{K}\alpha_j^s D_{kl}[f(x|\Theta(U_j))||f(x|\theta^H_{R(j)})] \end{aligned} \tag{10}$$

Where $\alpha^H\psi$ is a probability vector of length K with $\alpha^H_{R(j)}\psi_j$ in coordinate j.

Proof. See Appendix. □

4.4 Transition Probability

Before we prove our main theorem, we need to introduce the notion of the Jensen-Shannon divergence (JS-divergence) restricted to a specific exponential family. In general, the JS-divergence [9] is defined as the difference between the entropy of a mixture and the average entropy of the components. It is a positive quantity that naturally arises as the relevant statistic for the 'two sample problem' [9], in which we wish to decide whether two samples could have been generated by a single source. However, for general continuous densities the JS-divergence is intractable and can only be numerically approximated. Given an exponential distribution $\mathcal{F}$, we now define a specialized $\mathcal{F}$-JS-divergence.

Definition 4.5. *Let $\{p_j(x)\}_{j=1}^K$ be a set of general densities, and $\{\pi_j\}_{j=1}^K$ be a set of mixture coefficients summing to 1. We define the $\mathcal{F}$-type of a density to be $U(p) = E_{p(x)}T(x)$ and the most likely $\mathcal{F}$-parameters of p to be $\Theta(p) = \Theta(U(p))$. The $\mathcal{F}$-Entropy of $p(x)$ is defined as $H^F(p) = H(f(x|\Theta(p)))$. Based on these definitions, we can define the $\mathcal{F}$-Jensen-Shannon divergence to be*

$$\begin{aligned} JS^F_{\pi_1,..,\pi_K}(\{p_j(x)\}_{j=1}^K) &= H^F(\sum_{j=1}^{K}\pi_j p_j(x)) - \sum_{j=1}^{K}\pi_j H^F(p_j(x)) \\ &= \sum_{j=1}^{K}\pi_j D_{kl}[f(x|\Theta(p_j))||f(x|\Theta(\sum_{i=1}^{K}\pi_i p_i))] \end{aligned} \tag{11}$$

We defined the $JS^{\mathcal{F}}$-divergence above as a function of a set of densities, to be consistent with traditional definitions. However, in this definition, $JS^{\mathcal{F}}$ is actually a function of the $\mathcal{F}$-types of the densities alone. We will hence

extend the usage of $JS^{\mathcal{F}}$ to $JS^{\mathcal{F}}_{\pi_1,..,\pi_K}(U_1,..,U_K)$ where U_j are $\mathcal{F}$-types. For a specific $\mathcal{F}$ family the $JS^{\mathcal{F}}$ expression has a closed-form formula, since the entropy and D_{kl} (see Eq. (6)) have such formulas. We now state our main theorem

Theorem 3. *Let* $V^1 = ((\alpha^s_1,..,\alpha^s_{K^1}),U_1,..,U_{K^1})$, $V^2 = ((\alpha^s_{K^1+1},..,\alpha^s_{K^1+K^2}),U_{K^1+1},..,U_{K^1+K^2})$ *be two mixture types with a total of* $K = K^1 + K^2$ *components. Let* S^1, S^2 *be two* N *point samples from a hidden mixture model* M^H *with unknown parameters. Then*

$$\begin{aligned}&\log P(S^1 \in TC_{N,\varepsilon}(V^1)|S^2 \in TC_{N,\varepsilon}(V^2), M^H)\\ &\to \max_R -2N \sum_{m=1}^{K^H} \alpha^H_m JS^{\mathcal{F}}_{\{\psi_j:R(j)=m\}}(\{U_j : R(j)=m\})\end{aligned} \tag{12}$$

Where the limit is of $N \to \infty, \epsilon \to 0, N\epsilon^2 \to \infty$ *and the parameters* $\{\alpha^H_m\}_{m=1}^{K^H}$,$\{\psi_j\}_{j=1}^{K}$ *are given by*

$$\alpha^H_m = \frac{1}{2}\sum_{\{j:R(j)=m\}} \alpha^s_j \quad , \quad \psi_j = \frac{\alpha^s_j}{\alpha^H_{R(j)}} \tag{13}$$

The conditional probability therefore is a weighted sum of $\mathcal{F}$-Jensen-Shannon divergences over the equivalence classes introduced by R.

Proof. We decompose the conditional probability using $P(a|b) = P(a,b)/P(b)$ and estimate the two resulting terms using ML approximations.

$$\begin{aligned}&\log P(S^1 \in TC_{N,\varepsilon}(V^1)|S^2 \in TC_{N,\varepsilon}(V^2), M^H)\\ &\approx \max_{R,\Theta^H,\Psi} \log P(S^1 \in TC_{N,\varepsilon}(V^1), S^2 \in TC_{N,\varepsilon}(V^2)|M^H)\\ &\quad - \max_{R,\Theta^H,\Psi} \log P(S^2 \in TC_{N,\varepsilon}(V^2)|M^H)\end{aligned} \tag{14}$$

The second optimization problem above deals with finding the hidden model from which samples of $TC_{N,\epsilon}(V^2)$ are most likely. Since any reasonable restriction of the hidden mixture model still allows hidden mixtures with $K^H \approx K^2$ components, this problem can be trivially solved by choosing $\Theta^H = \Theta(V^2)$ and trivial R and Ψ (i.e. $R(j) = j$, $\psi_j = 1$ for all $j = 1,..,K^2$). In this case we know from Lemma 1 that $P(S^2 \in TC_{N,\varepsilon}(\Theta(V^2))|R,\Theta^H,\Psi) > 1 - O(\frac{1}{N\epsilon^2})$, and so the second summand is negligible.

The probability in the first optimization can be assessed using Lemma 2. Consider the joint sample $S = S^1 \cup S^2$ and the joint type $V = ((\alpha^s_1,..,\alpha^s_K)/2, U_1,..,U_K)$. Clearly, if $S^i \in TC_{N,\epsilon}(V^i)$ for i=1,2, then $S \in TC_{2N,\epsilon}(V)$. On the other hand $S \in TC_{2N,\epsilon}(V)$ implies $S^i \in TC_{N,2\epsilon}(V^i)$ for i=1,2. Hence in the limit $\epsilon \to 0$ the events are interchangeable, and we

can optimize $logP(S \in TC_{2N,\epsilon}(V)|M^H)$. Using Lemma 2 this expression is written as

$$\max_{R,\Theta^H,\Psi} -2N(D_{kl}[\frac{\vec{\alpha^s}}{2}||\alpha^H\psi] + \sum_{j=1}^{K} \frac{\alpha_j^s}{2} D_{kl}[f(x|\Theta(U_j))||f(x|\theta^H_{R(j)})]) \quad (15)$$

Fixing R, we can find the optimal parameters for Θ^H, Ψ. The parameters α^H, ψ only appear in the first D_{kl} and they should be chosen to minimize it. This is achieved using the values in Eq. (13), and the minimum achieved is 0. As for the hidden components parameters θ^H_m, writing the D_{kl} expressions in (15) using (6) and differentiating we get

$$\frac{dA(\theta^H_m)}{d\theta^H_m} = \sum_{\{j:R(j)=m\}} \psi_j \frac{dA(\Theta(U_j))}{d\Theta(U_j)} \quad (16)$$

According to property 3 in Section 4.1 we have $\theta^H_m = \Theta(\sum_{j:R(j)=m} \psi_j U_j)$. Plugging the optimal expressions found in (15) we have to maximize over R the expression

$$-2N \sum_{m=1}^{K^H} \alpha^H_m \sum_{\{j:R(j)=m\}} \psi_j D_{kl}[f(x|\Theta(U_j))||f(x|\theta^H_m)] \quad (17)$$

which is identical to (12) by Definition 4.5 of the $JS^{\mathcal{F}}$ quantity. □

4.5 The Case of a Gaussian Source

It is very common to model probabilistic sources as Gaussian mixtures (specifically, in our experiments we used such a model to describe spike trains, as well as sequences of satellite imagery). We thus derive the special case for the Gaussian case.

The entropy of a Gaussian is given by $H^G(p(x)) = (1/2)log2\pi e|\Sigma|$, and so the Gaussian-JS divergence is:

$$JS^G_{\pi_1,..,\pi_K}(\{p_j(x)\}) = \frac{1}{2}(\log|\Sigma^*| - \sum_{j=1}^{K} \pi_j \log|\Sigma_j|) \quad (18)$$

Where $\Sigma^* = \sum_{j=1}^{K} \pi_j(\Sigma_j + (\mu_j - \mu^*)(\mu_j - \mu^*)^t)$, $\mu^* = \sum_{j=1}^{K} \pi_j \mu_j$.

4.6 Algorithms

The JS-based score (12) can be trivially optimized if we do not restrict the family of allowed hidden mixture models. Without such restriction, we can choose Θ^H as the concatenation of the empirical distributions Θ^1, Θ^2 and trivial values for R and Ψ ($R(q) = q$, $\psi_q = 1$ for $q = 1, .., K^1 + K^2$).

For this choice we get the maximal value of 0. In general a 'richer' hidden model, which does not merge many Gaussians, gets a higher score. We have to impose restrictions on the hidden model to get a useful score. This is done by posing constraints over the allowed R mappings.

Consider first the case in which a one-to-one correspondence is assumed between clusters in two consecutive frames, and hence the number of Gaussian components K is constant over all time frames. In this case, a mapping R is allowed iff it maps to each hidden source i a single Gaussian from mixture Θ^1 and a single Gaussian from Θ^2. Denoting the Gaussians matched to hidden i by $R_1^{-1}(i), R_2^{-1}(i)$, the transition score (12) takes the form of $-N \cdot \max_R \sum_{i=1}^{K} S(R_1^{-1}(i), R_2^{-1}(i))$. Such an optimization of a pairwise matching score can be seen as a search for a maximal perfect matching in a weighted bipartite graph. The nodes of the graph are the Gaussian components of Θ^1, Θ^2 and the edges' weights are given by the scores $S(a, b)$. The global optimum of this problem can be efficiently found using the Hungarian algorithm [14] in $O(n^3)$, which is feasible in our case.

The one-to-one correspondence assumption is too strong for many data sets, specifically, in the Spike Sorting application, as it ignores the phenomena of either splits or merges of clusters. We wish to allow such phenomena, but nevertheless enforce strong (though not perfect) demands of correspondence between the Gaussians in two consecutive frames. In order to achieve such balance, we place the following constraints on the allowed partitions R:

1. Each cluster of R should contain exactly one Gaussian from Θ^1 or exactly one Gaussian from Θ^2. Hence assignment of different Gaussians from the same mixture to the same hidden source is limited only for cases of a split or a merge.

2. The label entropy of the mixture induced by R should satisfy

$$H(\alpha_1^H, .., \alpha_{K^H}^H) \leq \tfrac{1}{2}(H(\alpha_1^1, .., \alpha_{K^1}^1) + H(\alpha_1^2, .., \alpha_{K^2}^2)) \qquad (19)$$

Intuitively, the second constraint limits the allowed hidden mixtures to ones which are not richer than the average visible mixtures, i.e., do not have much more clusters. Note that the most detailed hidden model (the 'concatenation' model) has a label entropy given by the r.h.s of inequality (19) plus 1 bit. The extra bit can be intuitively understood as the cost of using twice the number of required labels. We look for mixtures which do not pay this extra price in label information.

The optimization for this family of R does not seem to have an efficient global optimization technique, and thus we resort to a greedy procedure. Specifically, we use a bottom up agglomerative procedure to find the partition induced by R. We start from the concatenated mixture model. This is the most detailed partition, as R is a $1-1$ map and each visible Gaussian

is a singleton. At each iteration of the algorithm we merge two clusters of the partition. Only merges that comply with the first constraint are considered. We look for a merge which incurs a minimal loss to the accumulated Jensen-Shannon score (12) and a maximal loss to the mixture label entropy. For two Gaussian clusters $(\alpha_1, \mu_1, \Sigma_1)$, $(\alpha_2, \mu_2, \Sigma_2)$ these two quantities are given by

$$\begin{aligned} \Delta JS &= -N(\alpha_1 + \alpha_2) JS^G_{\pi_1,\pi_2}(G(x|\mu_1, \Sigma_1), G(x|\mu_2, \Sigma_2)) \\ \Delta H &= -N(\alpha_1 + \alpha_2) H(\pi_1, \pi_2) \end{aligned} \tag{20}$$

where π_1, π_2 are $\frac{\alpha_1}{\alpha_1+\alpha_2}, \frac{\alpha_2}{\alpha_1+\alpha_2}$. We choose at each round the merge which minimizes the ratio between these two quantities. The algorithm terminates when the accumulated label entropy reduction is bigger than 1 bit or when no allowed merges remain. In the second case, it may happen that the mapping R found by the algorithm violates the constraint (19). We nevertheless compute the score based on the R found, since this partition obeys the first constraint and usually is not far from satisfying the second.

5 Experimental Results

Our main effort was to achieve an automatic Spike Sorting comparable to human experts. We thus relate to the human expert's clustering as the ground truth and measure our performance based on this assumption. In Section 5.1 we bring a full description of the various experiments.

In Section 5.2 we describe the experiments conducted on the Storm Tracking application, a problem with similar non-stationary properties.

In each problem domain, the algorithm was supplemented with some specific pre- and post-processing based on specific domain knowledge. Clustering examples for both applications can be seen on the author's web site[1].

5.1 Spike Sorting

We describe the experimental design and the data acquisition in 5.1.1. In Section 5.1.2 we assess the algorithms performance in terms of agreement with human sorters. The algorithms agreement with human sorters is shown to be similar to the agreement among humans. In Section 5.1.3 we present experiments with semi supervised clustering, where only $1/10$ of the time frames are manually sorted and the algorithm performs the rest of the clustering.

[1] http://www.cs.huji.ac.il/~adams

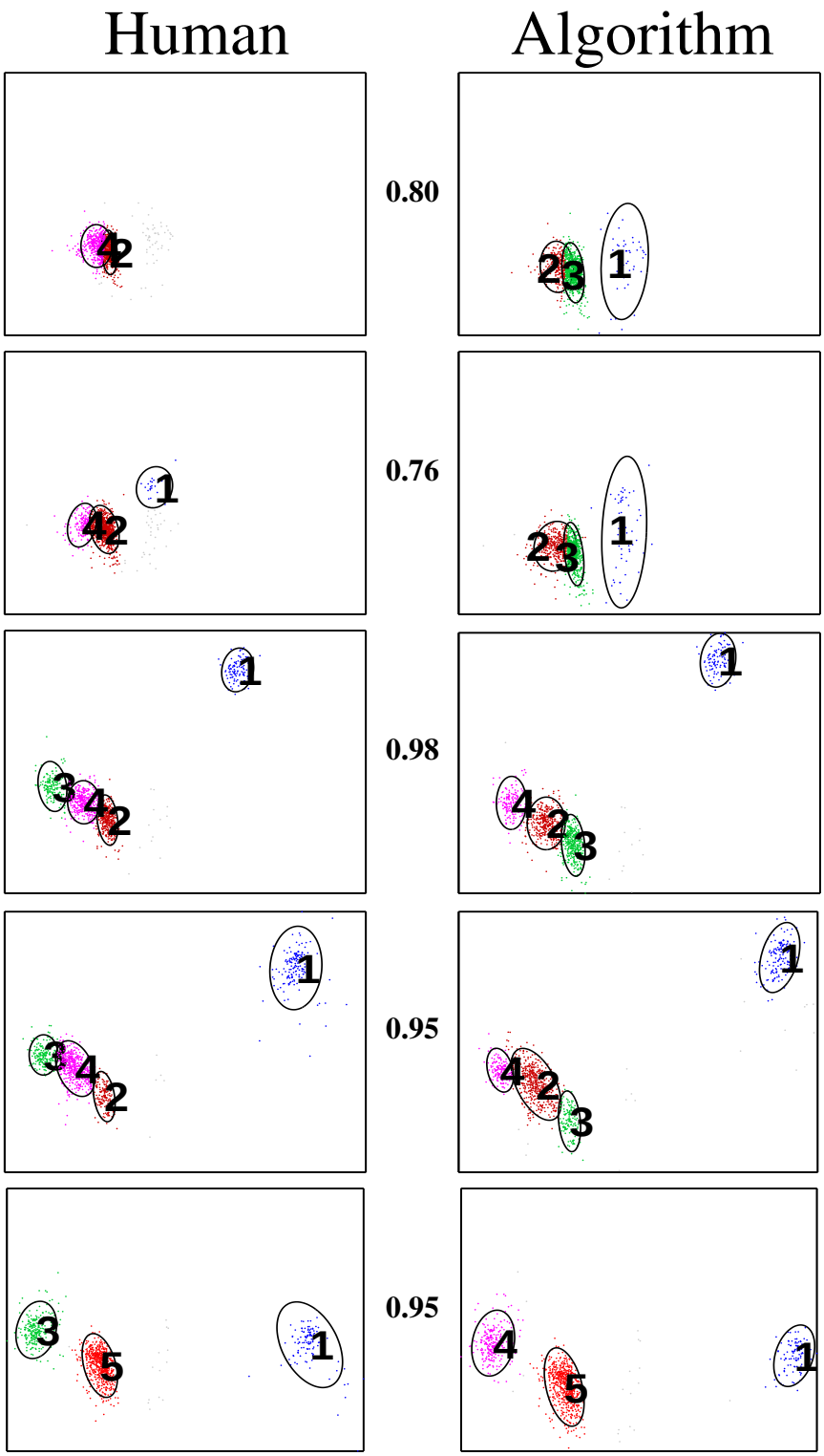


Figure 5: Manual and automatic clustering solutions for the non stationary recording presented in Figure 1. Each frame contains 1000 spikes, plotted here (with random number assignments) according to their first two PCs. The left and right columns show manual and automatic clustering solutions respectively. Notice that during the split process of the bottom left area some ambiguous time frames exist in which 1,2, or 3 cluster descriptions are reasonable. This ambiguity can be resolved using global considerations of past and future time frames. By finding the MAP solution over all time frames, the algorithm manages such considerations. The numbers between the images show the $f_{1/2}$ score of the local match (per frame) between the manual and the automatic clustering solutions (see text).

5.1.1 Experimental design and data acquisition

Neural data were acquired from the dorsal and ventral pre-motor (PMd, PMv) cortices of two Macaque monkeys performing a prehension (reaching and grasping) task. At the beginning of each trial, an object was presented in one of six locations. Following a delay period, a Go signal prompted the monkey to reach for, grasp, and hold the target object. A recording session typically lasted 2 hours during which monkeys completed 600 trials. During each session 16 independently-movable glass-plated tungsten micro-electrodes were inserted through the dura, 8 into each area. Signals from these electrodes were amplified (10K), bandpass filtered (5-6000Hz), sampled (25 kHz), stored on disk (Alpha-Map 5.4, Alpha-Omega Eng.), and subjected to 3-stage preprocessing. (1) Line influences were cleaned by pulse-triggered averaging: the signal following a pulse was averaged over many pulses and subtracted from the original in an adaptive manner. (2) Spikes were detected by a modified second derivative algorithm (7 samples backwards and 11 forward), accentuating spiky features; segments that crossed an adaptive threshold were identified. Within each segment, a potential spike's peak was defined as the time of the maximal derivative. If a sharper spike was not encountered within 1.2ms, 64 samples (10 before peak and 53 after) were registered. (3) Waveforms were re-aligned s.t. each started at the point of maximal fit with 2 library PCs (accounting, on average, for 82% and 11% of the variance, [1]). Aligned waveforms were projected onto the PCA basis to arrive at two coefficients.

5.1.2 Results of the fully automated algorithm

We tuned the algorithm's parameters using a training data set of 42 electrodes containing a total of 1383 time frames. The algorithms was then tested on another set of 46 electrode recordings containing a total of 4544 time frames. Spike trains were manually clustered by a skilled user in the environment of Alpha-Sort 4.0 (Alpha-Omega Eng.). Our performance measure in this section is the agreement rate between automatic and human clustering. We computed the agreement between automatic and human clustering solutions using a combined measure of precision P and recall R scores $f_{\frac{1}{2}} = \frac{2PR}{R+P}$. This criterion establishes mappings between clusters in the two partitions, and roughly measures the percentage of points assigned to a cluster and its chosen correspondent. It ranges in $[0, 1]$ where 1 signifies a perfect agreement. Similar results were obtained when we measured solutions agreement using another criterion - the 'Variation of Information' suggested in [21]. Figure 5 demonstrates the performance of the algorithm using a particularly non-stationary data set.

We measure the algorithm agreement with human clustering using two statistics: (1) label agreement for each time frame independently, and (2) label agreement of entire electrodes (sequences of time frames). The first gives

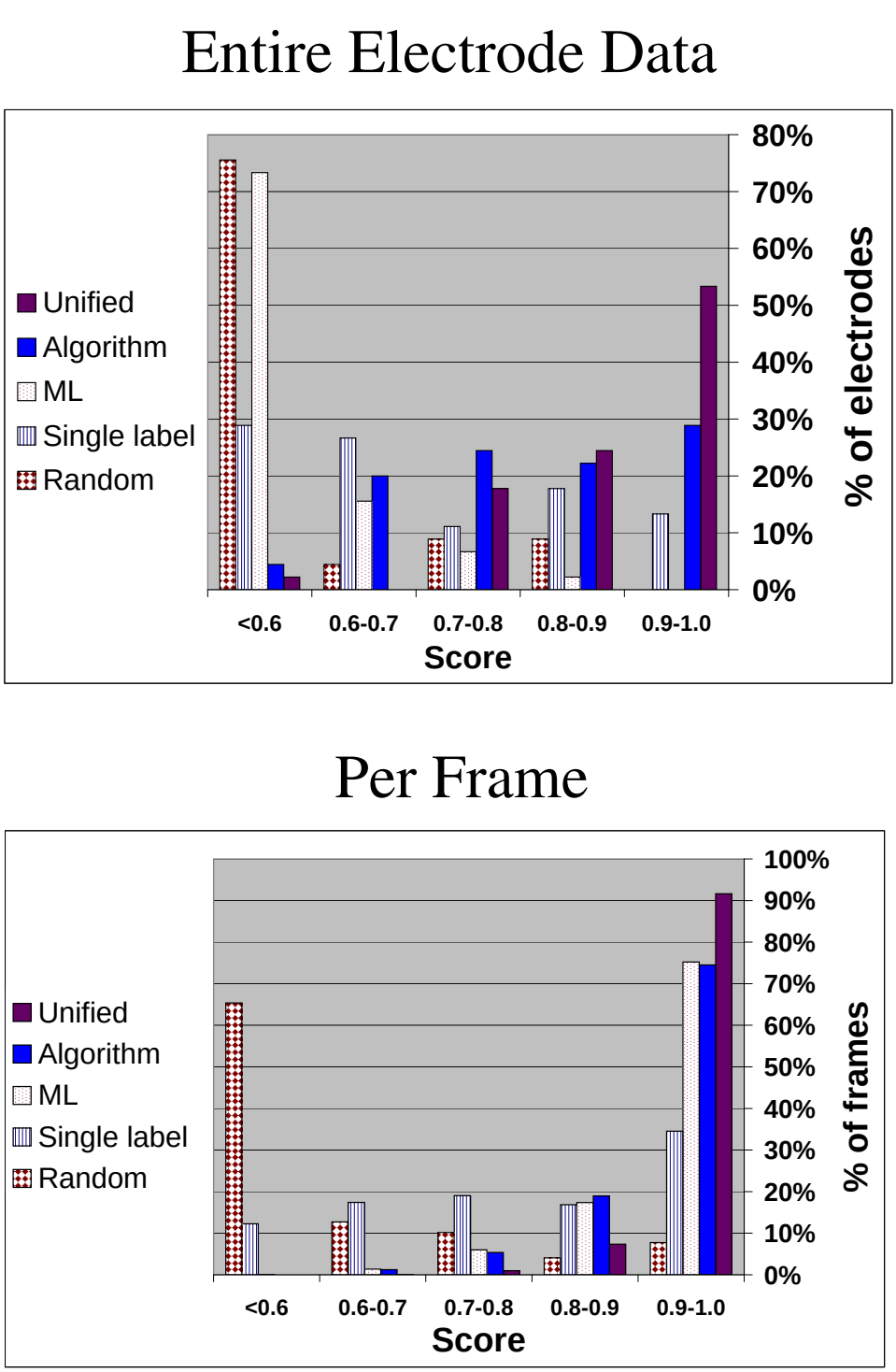


Figure 6: Performance histograms of the automatic algorithm (in blue) and several control conditions. The control conditions are explained in the text. Performance measurements are $f_{1/2}$ agreement scores between suggested clustering solutions and human clustering. Results for complete electrodes are shown on the top and for single time frames on the bottom.

an indication of how well the local solution matches that of the human. However, notice that it is possible to obtain high local frame's agreement together with low agreement scores for the entire electrode. This happens when the clusters correspondence between consecutive frames is not established correctly. A single correspondence failure in the middle of a sequence may completely mix two labels, or split a single label into two. Thus high agreement on the entire electrode requires almost perfect correspondence along all the sequence.

We first compare the algorithm's performance to several control conditions:

- Local maximum likelihood (ML) clustering. This clustering solution is obtained by choosing the mixture description with the highest local likelihood at each time frame. Cluster correspondence between consecutive time frames is established using our standard algorithm for computation of the transition probability. The difference between this approach and ours is that in the ML clustering no global optimization is obtained.

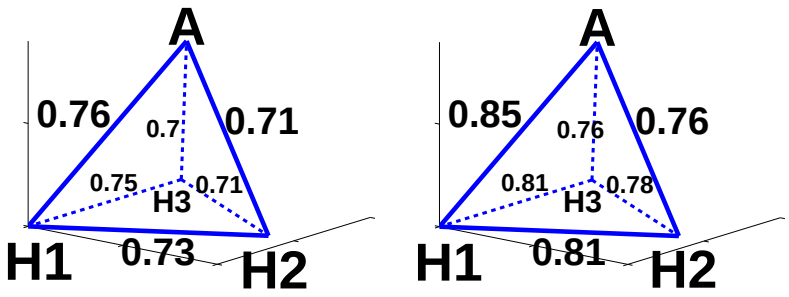
- Unified clusters in the human clustering. An observation of the human and automatic clusterings shows that the automatic solution occasionally unites 2 clusters which were considered separated in the human's solution. This phenomenon usually occurs in cases in which descriptions using one or two clusters are both plausible. We hence compute the highest agreement that can be achieved by unifying two clusters in the human labeling.

- Trivial clustering. All spikes are assigned to the same cluster.

- Random clustering. Labels are randomly assigned to spikes, according to the label distribution in the manual clustering.

Figure 6 summarizes the comparison of our algorithm with these control conditions. Several effects can be seen. The average agreement score of the algorithm was 0.80 for complete electrodes and 0.91 for single frames. These results are much higher than the baseline performance of random and trivial clustering. The ML solution performance for independent frames is comparable to the algorithm performance, but it gets very low scores for entire electrodes. This happens since mixtures are chosen independently for each time frame, and often no natural correspondences between adjacent time frames can be made. This emphasises the importance of the global optimization in our method. Last, we can see that when the human solution is modified using a single cluster unification the agreement score increases dramatically. This means that most of the discrepancy between human and algorithm is usually caused by a single cluster unification decision.

The algorithm gives reasonable, continuously evolving clustering even when it has low agreement with the manual clustering. It often seems that



(a) (b)

Figure 7: The agreement scores between the automatic clustering and clustering solutions of three human sorters. **(a)** Agreement over complete electrodes. **(b)** Agreement over single time frames.

both human and algorithm provide reasonable interpretations of an inherently ambiguous data. This 'inherent ambiguity' component can be quantified by measuring the disagreement between human experts clustering the same data. To this end we obtained manual clustering of 3 human experts (our original expert and two additional skilled sorters) for 6 highly non-stationary electrodes. The average agreement between the 3 humans and the algorithm can be seen in Figure 7. The average agreement of the algorithm with the human sorters (0.723 and 0.79 for electrodes and single frames respectively) is almost identical to the average agreement among humans (0.73 and 0.80). In fact, for electrodes the average agreement of the algorithm with humans is higher than the average agreement of sorter 2 with other humans. The algorithm clearly prefers the clustering solutions of sorter 1 over the other two humans. This is not surprising, as sorter 1 is our original expert, and the algorithm's parameters were tuned according to his preferences.

5.1.3 Results of the semi-supervised version

As mentioned in Section 3.1, human supervision can be easily incorporated into the automatic clustering by letting the human cluster a small fraction of the time frames. We tested this mode on our 46 test electrodes. Two levels of human intervention were checked, one in which the human supplies clustering solution for every fifth time frame, the other for every tenth time frame. Result histograms are given in Figure 8. Human guidance has increased average electrode agreement of the algorithm from 0.80 to 0.84 and 0.837 using $1/5$ and $1/10$ of time frames respectively. Clearly most of the improvement is already gained with $1/10$ of the frames manually clustered.

The main intended use of the semi-supervised clustering mode is to facilitate adaptation of the automatic algorithm to the clustering preferences of a specific user. We checked whether this mode can adapt to the preferences of different users using the 6 data sets for which we have 3 different manual

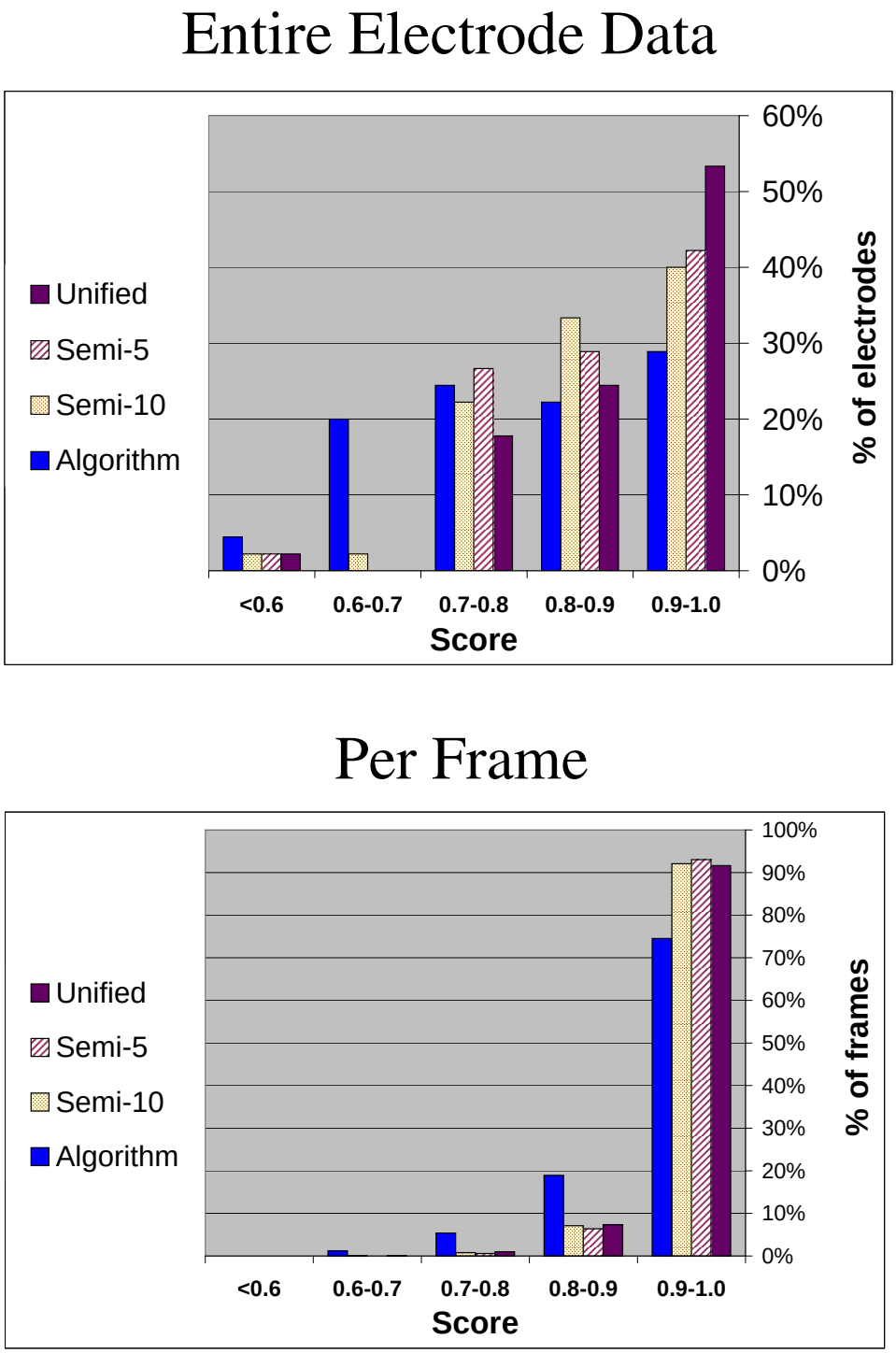


Figure 8: Performance histograms of fully automated and semi supervised clustering. Results are shown for two levels of human supervision, in which manual partition is given for $1/5$ and $1/10$ of the time frames. $f_{1/2}$ scores for complete electrodes are shown on the top and for single time frames on the bottom.

supervisor	H1 match	H2 match	H3 match
H1	**0.86**	0.70	0.71
H2	0.754	**0.785**	0.630
H3	0.778	0.664	**0.780**
None	**0.76**	0.71	0.70

Table 1: Agreement scores between 3 human sorters and the algorithm in a semi supervised mode. Rows in the table present results obtained with different human supervisors. 1/10 of the frames were manually clustered. The three columns present the $f_{1/2}$ agreement of the clustering found with the three human sorters. Results are given for complete electrodes and averaged over 6 data sets. The highest match in each row is shown in bold. The diagonal position of the high matches indicates successful adaptation of the algorithm to the preferences of different sorters.

clusterings. The results with manual clustering of $1/10$ of the data are given in Table 1. Supervision increased the agreement of the algorithm with the supervisor by $0.06 - 0.08$ for the three sorters. It allowed sorter 2 and 3 to override the initial tendency of the algorithm toward sorter 1 (the sorter whose labels were used in tuning the algorithms parameters).

5.2 Hurricane Tracking

We tested the algorithm on hurricane identification and tracking using satellite imagery sequences obtained from the GOES project of NASA[2]. The data includes visible light and infra-red imagery taken from a fixed point satellite at intervals of 15 minutes. Since cold areas tend to be brighter in these images, we obtained a sample from the storm areas by sampling in proportion to image intensity. Because the data is sometimes very ambiguous, and contains many split and merge events, we adopted another policy in label unification across merges and splits. In each such event, the unified cluster kept the label, and thus the 'identity', of the largest component unified (in the Spike Sorting application it received a new label). Adaptations were also made to the local solution pruning process, to remove small or very elongated clusters which do not have a meteorological meaning.

We tested the algorithm on two storm sequences, describing hurricanes 'Alex' and 'Dennis'. In both cases, the algorithm yielded reasonable clustering that captures the main dynamic trends. Specifically, the main hurricane cell was identified and well tracked. Figure 9 shows a few frames from the clustering suggestion of the 'Alex' sequence.

[2]http://rsd.gsfc.nasa.gov/goes/text/hotstuff.html

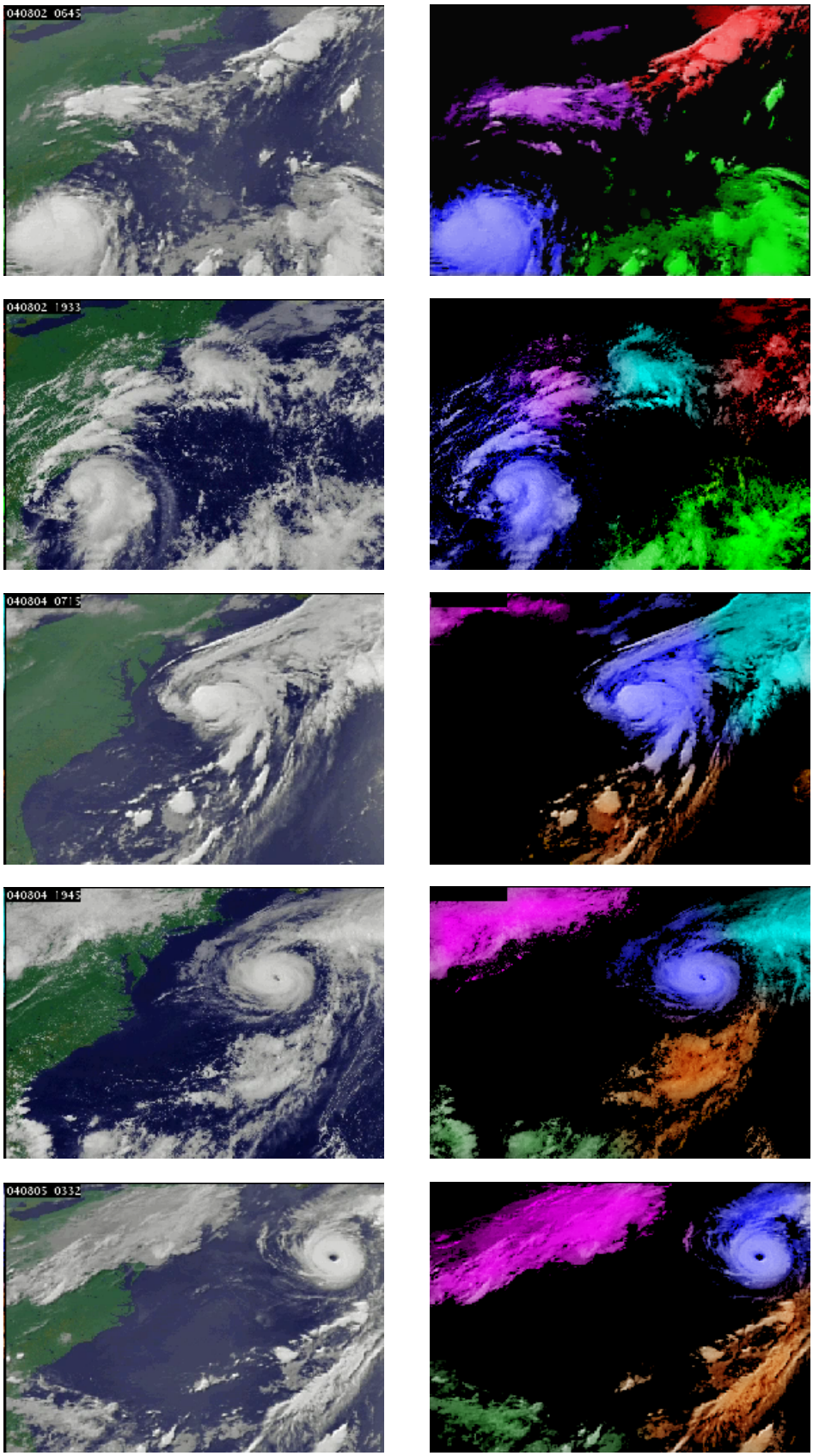


Figure 9: Frames 18, 82, 240, 300, and 344 from a movie of 375-frames. The left column presents the original satellite images. The right column preserves the original intensity of the clouds, but is colored according to the clustering found. Alex's main storm cell is identified and tracked in blue (the figure is best viewed in color).

6 Discussion

We presented an algorithm for clustering non-stationary data that copes well with movement, merges, and splits of clusters. The algorithm can be adapted to specific domains by using various constraints. Two of these domains, Spike Sorting and Storm Tracking, were presented here.

Applying the algorithm in the Spike Sorting domain almost reaches human performance level, as can be judged using measurements of agreement between clustering solutions. There are several main factors which explain the algorithm's success. First is the reasonable statistical model of a chain of Gaussian mixtures, which closely follows the human intuition. The global optimization, which introduces past and future information into the clustering decision at each specific frame, is the second factor. Last, a module of rule based, classic AI system mimics the human treatment of exceptional cases.

Since our main motivation in the development of the method is to save human labor, we have basically tried to mimic the human experts behavior. However, we do not use all the information available for the Spike Sorting problem, and hence it is clearly not optimal. Specifically, the decision to work with 2 dimensional spike representation implies that important information in the original wave form is untapped (approximately 7% of the spike variance [1]). Moreover, when using the low dimensional representation we cannot adequately account for the problem of overlapping spikes [19, 27]. The method we suggest can be naturally augmented to handle spike representations in higher dimensions. Specifically using 3 PC's instead of 2 can account for additional 3% of the spike variance with minor changes to the existing system. We did not follow this path because our main evaluation method is the agreement with human experts, who heavily rely on the visual representations in 2 dimensions. Another source of information we did not attend is the inter spike intervals (ISI). Using refractory period considerations we can extract constraints of the form 'spike 1 cannot be from the same cluster as spike 2'. Such constraints were used by [8] to guide clustering decisions. In our framework, such constraints can be incorporated into the EM algorithm [22] to guide the search for good local solutions. We did not make use of this information since sharp constraints were very rare in our data.

The agreement among human experts clusterings of complicated electrodes was found to be rather low in our experiments (Section 5.1.2). These measurements are consistent with other studies measuring the accuracy of human clustering when 'ground truth' is known. In [11] spikes are recorded simultaneously using intracellular and extracellular electrodes, and intracellular measurements serve as ground truth. When using tetrode recordings human errors were as high as 30%. In [25] similar human error rates are reported for carefully designed synthetic data. In view of these findings,

one may question whether mimicking the human expert should be the goal of automatic Spike Sorting.

Human sorters use high level considerations, and deal very well with exceptional cases. It is unlikely that an automated system can outperform them in these respects. However, there is some evidence indicating that the overall performance of a well designed automated system may be higher than human performance. First, while human clustering is subjective, and may vary with time and person, the automated clustering clearly does not have such internal variance. Second, it is pointed in [11] that a major factor in the human errors is caused by humans inability to efficiently visualize and make decisions regarding high dimensional data. A semi-supervised approach in which part of the clustering is done automatically in high dimension is presented and shown to have error rates lower than fully manual sorting. A third point is that automatic sorting can be done in a Bayesian framework using statistically sound models. This leads to accurate determination of clusters borders, which may be better than the intuitive human choice. In addition, such a framework supports natural confidence measures for the spike labels, as the probability of label confusion can be inferred from the model. In [26] a Gaussian mixture approach to Spike Sorting is used and evaluated in the prediction of hand kinematics. The prediction performance of the algorithm is shown to be better than the predictions obtained using manually sorted data. The algorithm we suggest can be seen as a natural extension of their simple statistical model to long, non-stationary chains.

Based on the considerations above it is reasonable to believe that the extension of the method proposed here in various aspects, including spike representation in higher dimensions and ISI information, can yield better performance than manual human clustering.

7 Appendix

Proof of Lemma 2 from Section 4.3

Lemma 2. *Let S be a labeled sample of N points from arbitrary density, with $V = (\vec{\alpha}^s, U_1, .., U_K) = T(S)$. Let M^H be a hidden mixture model parameterized by $(\{\alpha_m^H\}_{m=1}^{K^H}, \{\psi_j\}_{j=1}^{K}, \{\theta_m^H\}_{m=1}^{K^H})$. Then*

$$\begin{aligned}&\lim_{\substack{N\to\infty,\varepsilon\to 0\\ N\varepsilon^2\to\infty}} \tfrac{1}{N}\log P(TC_{N,\varepsilon}(V)|M^H)\\ &= -D_{kl}[\vec{\alpha}^s||\alpha^H\psi] - \sum_{j=1}^{K}\alpha_j^s D_{kl}[f(x|\Theta(U_j))||f(x|\theta_{R(j)}^H)]\end{aligned} \tag{21}$$

Where $\alpha^H\psi$ is a probability vector of length K with $\alpha_{R(j)}^H\psi_j$ in coordinate j.

Proof.

$$P(TC_{N,\varepsilon}(V)|M^H) = \int\limits_{S'\in TC_{N,\varepsilon}(V)} p(S'|M^H) \tag{22}$$

The proof relies on two steps: First we show that the integrand is roughly constant in the domain, and we compute its value. Then we compute the volume and get the integral value by multiplying it with the constant integrand value.

For a sample $S' = (X', L')$ with type $V' = (\vec{\alpha}^{s'}, U_1^{s'}, .., U_K^{s'})$ the log likelihood can be decomposed in a similar manner to Eq. (7)

$$\log p(S'|M^H) = N\sum_{j=1}^{K}\alpha_j^{s'}[\log\alpha_{R(j)}^H + \log\psi_j + \theta_{R(j)}^H\cdot U_j^{s'} - A(\theta_{R(j)}^H)] \tag{23}$$

For $S' \in TC_{N,\epsilon}(V)$ we can approximate S' statistics with those of S

$$\begin{aligned}&\log p(S'|M^H)\\ &= N[\sum_{j=1}^{K}\alpha_j^s[\log\alpha_{R(j)}^H + \log\psi_j + \theta_{R(j)}^H\cdot U_j - A(\theta_{R(j)}^H)] + O(\varepsilon)]\end{aligned} \tag{24}$$

so the integrand is almost constant w.r.t to S'. For each component j, according to property 3 from Section 4.1, $U_j = dA/d\Theta(U_j)$. However, according to property 1 $dA/d\Theta(U_j) = E_{f(x|\Theta(U_j))}[T(x)]$. We can thus view U_j instead of a sample average from an arbitrary density, as an expectation of $T(x)$ according to $f(x|\Theta(U_j))$. Replacing U_j with this expectation and using property 4 we get

$$\begin{aligned}&\log p(S'|M^H)\\ &= N[\sum_{j=1}^{K}\alpha_j^s[\log\alpha_{R(j)}^H + \log\psi_j + E_{f(x|\Theta(U_j))}\log f(x|\theta_{R(j)}^H)] + O(\varepsilon)]\end{aligned} \tag{25}$$

Moving the 'almost constant' integrand implied by (25) out of the integral (22), we now have to compute the type 'volume' $\int\limits_{TC_{N,\varepsilon}(V)} 1$. We will assess this volume by first considering the mass of $TC_{N,\epsilon}(V)$ under the mixture density which makes it most plausible, i.e., $p(S^{'}|\Theta(V))$. Lemma 1 tells us that this mass is almost 1

$$1 > P(TC_{N,\varepsilon}(V)|\Theta(V)) = \int\limits_{S^{'}\in TC_{N,\varepsilon}(V)} p(S^{'}|\Theta(V)) > 1 - O(\tfrac{1}{N\varepsilon^2}) \tag{26}$$

On the other hand, we can repeat the steps taken in (23) and (24), this time with $p(S^{'}|\Theta(V))$, to see that the integrand in (26) is, again, almost constant

$$\begin{aligned}\log p(S^{'}|\Theta(V)) &= N[\sum_{j=1}^{K} \alpha_j^s[\log \alpha_j^s + \Theta(U_j)\cdot U_j - A(\Theta(U_j))] + O(\varepsilon)] \\ &= -N[H(p(x,l|\Theta(V))) + O(\varepsilon)]\end{aligned} \tag{27}$$

Putting the exponent of (27) into (26) and multiplying the inequality by $\exp(NH(p(x,l|\Theta(V))))$ we can get

$$\begin{aligned}&\exp(N[H(p(x,l|\Theta(V))) + O(\varepsilon)]) < \int\limits_{TC_{N,\varepsilon}(V)} 1 < \\ &(1 - O(\tfrac{1}{N\varepsilon^2}))\exp(N[H(p(x,l|\Theta(V))) + O(\varepsilon)])\end{aligned} \tag{28}$$

We can now compute the integral (22) by multiplying the volume (28) with the integrand (exponent of (25)). Ignoring arbitrary small terms of $O(\epsilon)$ and $O(1/N\epsilon^2)$ we get

$$\begin{aligned}&P(TC_{N,\varepsilon}(V)|M^H) \\ &\approx \exp(N[\sum_{j=1}^{K} \alpha_j^s \log \tfrac{\alpha_{R(j)}^H \psi_j}{\alpha_j^s} + E_{f(x|\Theta(U_j))} \log \tfrac{f(x|\theta_{R(j)}^H)}{f(x|\Theta(U_j))}])\end{aligned} \tag{29}$$

From which (21) follows. □

References

[1] Abeles M., Goldstein M.H., Multispike train analysis, *Proceedings of the IEEE*, vol. 65(5), pp. 762-773, 1977.

[2] Blackman S., Popoli R., Design and Analysis of Modern Tracking Systems, *Artech house*, 1999.

[3] Cover T., Thomas J., Elements of Information Theory, *John Wiley & Sons*, New York 1991.

[4] Csiszar I., The method of types, *IEEE Transactions on Information Theory*, vol. 44(6), pp. 2205–2523, 1998.

[5] Dempster, A. P., Laird N. M., Rubin D. B., Maximum Likelihood from incomplete data via the EM algorithm, *Journal of the Royal Statistical Society*, Series B, vol. 39, pp. 1-38, 1977.

[6] Duda R. O., Hart P. E., Stork David G., Pattern Classification, *John Wiley & Sons*, New York, 2nd edition, 2001.

[7] Emondi A.A, Rebrik S.P, Kurgansky A.V, Miller K.D., Tracking neurons recorded from tetrodes across time, *Journal of Neuroscience Methods*, vol. 135, pp. 95-105, 2004.

[8] Fee M., Mitra P., Kleinfeld D., Automatic sorting of multiple unit neuronal signals in the presence of anisotropic and non-gaussian variability, *Journal of Neuroscience Methods*, vol. 69, pp. 175-188, 1996.

[9] Grosse I., Bernaola J.P., Carpena P., Roman R.R., Oliver J., Eugene S.H., Analysis of symbolic sequences using the Jensen-Shannon Divergence, *Physical Review* E.65, 2002.

[10] Guedalia I.D., London M., Werman M., An on-line agglomerative clustering method for nonstationary data, *Neural Computation*, vol. 11, no. 2, pp. 521-540, 1999.

[11] Harris K.D., Henza D.A., Csicsvari J., Hirase H., Buzsaki G. Accuracy of tetrode spike separation as determined by simultaneous intracellular and extracellular measurements, *Journal of Neurophysiology*, Vol. 84(1), pp. 401-414, 2000. Code available in http://klustakwik.sourceforge.net/

[12] Jain A. K., Dubes. R. C., Algorithms for Data clustering, *Prentice Hall*, 1988.

[13] Johnson J., Mackeen P. , Witt A., Mitchell E., Stumpf G., Eilts M., Thomas K., The storm cell identification and tracking algorithm: An enhanced WSR-88D algorithm, *Weather and Forecasting*, vol. 13, pp. 263-276, 1998.

[14] Kuhn H.W., The Hungarian method for the assignment problem, *Naval research logistics quarterly*, pp. 83-87, 1995.

[15] Lakshmanan, V., Rabin R., DeBrunner V., Identifying and tracking storms in satellite images, *Second Artificial Intelligence Conference*, American Meteorological Society, Long Beach, CA, pp. 90-95, 2000.

[16] Landabaso L., Xu L., Pardas M., Robust Tracking and Object Classification Towards Automated Video Surveillance, *International Conference on Image Analysis and Recognition (ICIAR)*, pp. 463-470, Porto, 2004.

[17] Lehmann E.L., Testing statistical hypotheses, *John Wiley & Sons*, New York, 1959.

[18] Lehmann E.L., Theory of point estimation, *John Wiley & Sons*, New York, 1998.

[19] Lewicki M.S., Bayesian modeling and classification of neural signals, *Neural Computation*, vol. 6, pp. 1005-1030, 1994. Software available at www.cnl.salk.edu/ lewicki/

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

[21] Meila M., Comparing Clusterings by the Variation of Information, *Computational Learning Theory (COLT)*, pp. 173-187, 2003.

[22] Shental N., Bar Hillel A., Hertz T., Weinshall D. Computing Gaussian Mixture Models with EM using Side-Information, *Proceedings of workshop: The Continuum from labeled to unlabeled data in machine learning and data mining*, ICML 2003.

[23] Shoham S., Fellows M.R., Normann R.A., Robust, automatic spike sorting using mixtures of multivariate t-distributions, *Journal of Neuroscience Methods*, vol. 127(2), pp. 111-122, 2003.

[24] Snider R., Bonds A., Classification of non-stationary neural signals, *Journal of Neuroscience Methods*, vol. 84(1-2), pp. 155-166, 1998.

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

[26] Wood, F.D., Fellows, M., Donoghue, J. P., Black, M.J., Automatic spike sorting for neural decoding, *Proceedings of the IEEE Engineering in Medicine and Biology Society*, pp. 4009-4012, 2004.

[27] Zhang P.M., Wu J.Y., Zhou Y., Liang P.J., Yuan J.Q., Spike sorting based on automatic template reconstruction with a partial solution to the overlapping problem, *Journal of Neuroscience Methods*, vol. 135(1-2), pp. 55-65, 2004.