# Modelling Spikes with Mixtures of Factor Analysers

Dilan Görür, Carl Edward Rasmussen, Andreas S. Tolias, Fabian Sinz, and Nikos K. Logothetis

Max Planck Institute for Biological Cybernetics
72076 Tübingen, Germany
{first.last}@tuebingen.mpg.de

**Abstract.** Identifying the action potentials of individual neurons from extracellular recordings, known as spike sorting, is a challenging problem. We consider the spike sorting problem using a generative model, *mixtures of factor analysers*, which concurrently performs clustering and feature extraction. The most important advantage of this method is that it quantifies the certainty with which the spikes are classified. This can be used as a means for evaluating the quality of clustering and therefore spike isolation. Using this method, nearly simultaneously occurring spikes can also be modelled which is a hard task for many of the spike sorting methods. Furthermore, modelling the data with a generative model allows us to generate simulated data.

## 1 Introduction

Recording the spiking activity from well isolated single neurons is important for studying the physiological functions of the brain. Although intracellular electrodes provide good quality signals from a single neuron, recording with an intracellular electrode in awake behaving animals is extremely difficult. Extracellular electrodes introduced into the brain isolating a single neuron have been successfully used for years. Recently, there has been excitement about recording simultaneously from multiple neurons in order to study their interactions. Electrodes placed in the extracellular medium can record the activity of multiple nearby neurons but this leads us to the question of distinguishing between the activity of individual neurons known as spike sorting. Under the assumption that the extracellular space is electrically homogeneous, four-tip electrodes (tetrodes) provide the minimal number necessary to identify the spatial position of a source based on the relative spike amplitudes on different electrodes. Recording with multi-tip electrodes improves the identification of individual neurons compared to standard single-tip electrodes ([3]).

Spike sorting is usually done in three steps, namely spike detection, feature extraction and clustering. Determination of the occurrence of spikes, which is usually achieved by high-pass filtering followed by thresholding is known as the spike detection step. In the feature extraction stage, a feature vector for each spike is calculated and clustering is done on this low dimensional feature space.

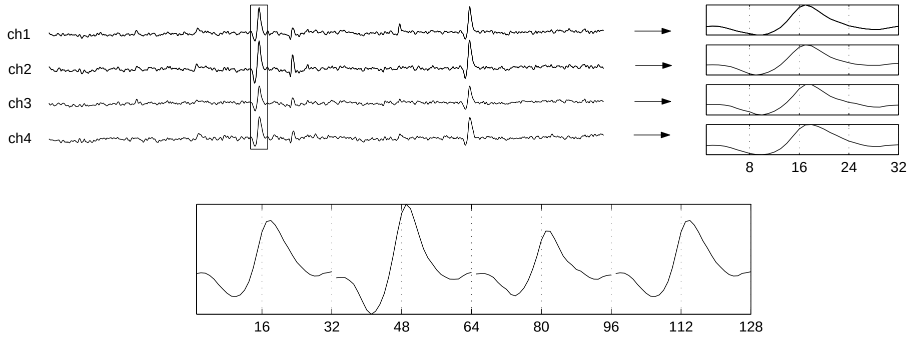

**Fig. 1.** Data recording and representation: Extracellular waveforms are recorded with a tetrode. Every time the signal exceeded the threshold in one of the channels, a window of 1ms around this event was extracted from the recordings of each channel and joined to form the data vectors

In most laboratories the clustering is done manually, usually using only the peak-to-peak amplitude data as features in order to make visualisation possible. There are also automatic spike sorting techniques that have been proposed which use different methods for feature extraction and clustering (see [5] for a review). A common problem of the spike sorting techniques is that the true labels for the recorded data cannot be known without the verification of intracellular recordings, which makes it very hard to evaluate the results obtained by clustering techniques.

Here, we present an automated way of spike sorting based on mixtures of factor analysers (MFA) using data collected with tetrodes. MFA is a statistical method that concurrently performs clustering and feature extraction. It models the spike waveforms and therefore overcomes the major cause of separation error, overlapping clusters in the low dimensional feature space. MFA can also model the nearly simultaneously occurring spikes which is a hard task for many spike sorting methods. It assigns responsibility degrees to each cluster for each spike and the entropy of these responsibilities can be used as a means for clustering evaluation. In addition, modelling the data with a generative model allows us to generate simulated data. The next section describes the data used in this study. In section 3, the clustering method is explained and in section 4 we demonstrate the method with real data collected with tetrodes.

## 2 Data Collection

We used data recorded with tetrodes from awake behaving macaque monkeys. The data were collected using a multi-channel data acquisition system (Cheetah Inc.). The signal was band-pass filtered between 600-6000$Hz$ and digitised at 32$kHz$. The neuronal spikes were acquired via an interrupt-driven, spike voltage threshold-triggered data acquisition system. That is, when a signal above threshold was detected in one of the channels, the occurrence of a spike was assumed and therefore the signal in all channels was stored within a window length of 1 ms around the triggering event with a corresponding time stamp. The 32-dimensional signals of all four channels were joined to form the 128-dimensional data vectors for the MFA model (see figure 1).

## 3 Methods

In unsupervised learning, dimensionality reduction and clustering are usually two steps that come sequentially. In dimensionality reduction, the features that are highly correlated are compressed, whereas in clustering, the data with similar features are grouped. MFA combines a well known dimensionality reduction technique, factor analysis, with a widely used clustering technique, mixture of Gaussians.

### 3.1 Factor Analysis

Factor analysis (FA) is a latent variable model in which a $p$-dimensional real-valued data vector $\mathbf{y}$ is modelled using a $k$-dimensional vector of factors where $k << p$. Dimensionality reduction is achieved by finding a low dimensional projection of the high dimensional data that captures most of the correlation structure of the data. The generative model is given by:

$$\mathbf{y} = \mathbf{\Lambda x} + \mu + \varepsilon \tag{1}$$

where $\mathbf{\Lambda}$ is called the *factor loading matrix.* The factors $\mathbf{x}$ and the noise are assumed to be normally distributed, $\mathbf{x} \sim \mathcal{N}(0, \mathbf{I})$ and $\varepsilon \sim \mathcal{N}(0, \mathbf{\Psi})$, where $\mathbf{\Psi}$ is a diagonal matrix. Therefore, $\mathbf{y}$ is also normally distributed with mean $\mu$ and covariance $\mathbf{\Lambda\Lambda}^T + \mathbf{\Psi}$. If $\mathbf{\Psi}$ is constrained to be $\delta\mathbf{I}$, then in the limit $\delta \to 0$, the FA becomes a PCA. In FA, the scaling of the coordinates is not important, but the axis rotation in which the original data arrived is important since noise is independent along the axes the input data are represented [7].

### 3.2 Mixtures of Factor Analysers

By using a mixture of factor analysers, dimensionality reduction and clustering can be achieved simultaneously. If we consider a mixture of $M$ factor analysers, the distribution of the data becomes:

$$P(\mathbf{y}) = \sum_{i=1}^{M} \pi_i \mathcal{N}(\mu_i, \mathbf{\Lambda}_i \mathbf{\Lambda}_i{}^T + \mathbf{\Psi}) \tag{2}$$

where $\pi_i$ denote the mixing proportions.

Given a set of observations $\mathbf{Y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_N\}$, we can describe the joint distribution of the data and the hidden factors using binary indicator variables $z_i, i = 1, \ldots, M$ as

$$P(\mathbf{Y}, \mathbf{x}, z) = \prod_{j=1}^{N} \sum_{i=1}^{M} P(z_i) P(\mathbf{x}|z_i) P(\mathbf{y}_j|\mathbf{x}, z_i) \tag{3}$$

where $z_i = 1$ indicates that the example was generated by the $i^{th}$ mixture. The unknown parameters $\mathbf{\Lambda}_i, \mu_i, \pi_i$ and $\mathbf{\Psi}$ that best model the covariance structure of the data can be found by maximum likelihood estimation using the EM algorithm, described below.

In the following, $\mathbf{Y}$ is the observation matrix and $\mathbf{H}$ are the unobserved (missing) quantities, in our case the latent variables of the factor analysis model ($\mathbf{x}$) and the identity of the mixture component which generated the observation ($z_i$). For *any* distribution $Q$ over the latent variables, the log likelihood $\big(\mathcal{L} \equiv \ln P(\mathbf{Y}|\theta)\big)$ can be lower bounded using Jensen's inequality:

$$\mathcal{L} = \ln \int d\mathbf{H} P(\mathbf{H}, \mathbf{Y}|\theta) \geq \int d\mathbf{H} Q(\mathbf{H}) \ln \frac{P(\mathbf{H}, \mathbf{Y}|\theta)}{Q(\mathbf{H})} \equiv \mathcal{F}(Q, \theta), \tag{4}$$

defining the $\mathcal{F}(Q, \theta)$ functional. Alternately optimising $\mathcal{F}$ with respect to the distribution of the hidden variables $Q(\mathbf{H})$, and the parameters $\theta$ is guaranteed not to decrease $\mathcal{L}$.

In the E step, we specify the distribution of the hidden variables that maximises $\mathcal{F}$[1] and calculate the expectation of the log likelihood with respect to this distribution. In the M step, we maximise the expected log likelihood, $\log P(\mathbf{H}, \mathbf{Y}|\theta)$, with respect to the parameters, keeping the $Q$ distribution constant.

There are two sets of conditionally independent hidden variables, thus the $Q$ distribution is of the form $Q(\mathbf{x}, z_i) = Q(\mathbf{x}|z_i)Q(z_i)$. Therefore, in the E step we compute the distributions of the hidden factors given the indicator variables $Q(\mathbf{x}|z_i)$, the distribution of the indicator variables $Q(z_i)$ and the expected log likelihood with respect to the $Q$ distribution (for details see [2]).

In the M step, the update rule for the parameters is obtained simply by setting the derivative of the expected log likelihood with respect to the parameters to zero and solving for the parameters[2]:

$$\tilde{\mathbf{\Lambda}}_i = [\mathbf{\Lambda}_i \, \mu_i] = (\sum_j \mathbf{y}_j \langle z_i \mathbf{x} | \mathbf{y}_j \rangle^T)(\sum_j \tilde{\mathbf{\Lambda}}_i \langle z_i \mathbf{x}\mathbf{x}^T | \mathbf{y}_j \rangle)^{-1} \tag{5}$$

$$\mathbf{\Psi} = \frac{1}{N} \operatorname{diag}(\sum_{ij} \langle z_i | \mathbf{y}_j \rangle \mathbf{y}_j \mathbf{y}_j^T - \sum_{ij} \mathbf{\Lambda}_i \langle z_i \mathbf{x} | \mathbf{y}_j \rangle \mathbf{y}_j{}^T) \tag{6}$$

$$\pi_i = \frac{1}{N} \sum_j \langle z_i | \mathbf{y}_j \rangle \tag{7}$$

### 3.3 Split and Merge EM

The EM algorithm is a hill climbing approach, thus local maxima is a serious problem of EM. When there are many components in one part of the space, and too few in another, it might not be possible to move a component from the overpopulated region to the underpopulated region without passing through

[1] This is equivalent to minimising the KL-divergence between $Q(\mathbf{H})$ and $P(\mathbf{H}, \mathbf{Y}|\theta)$.

[2] $\langle . \rangle$ denotes expectation wrt. $Q$

positions that has lower likelihood. To overcome this problem we used split and merge EM (SMEM) algorithm of Ueda et al. [8].

In the SMEM algorithm, when the model converges using the EM steps described above, two components are merged into one, and one component is split into two. Then, only the parameters of these three components are updated until convergence (referred to as the *partial EM procedure*). If this helps to increase the likelihood, the new parameters are accepted, otherwise another set of candidates are selected for splitting and merging and the partial EM procedure is repeated. The ordering of the split and merge candidates is important for the speed of the SMEM algorithm. We have used the criteria given in [8] to sort the candidates which suggests that the components that share the responsibilities[3] of many examples are good candidates for merging:

$$\mathcal{J}_{\text{merge}}(i,j;\theta) = \frac{\mathbf{R}_i(\theta)^T \mathbf{R}_j(\theta)}{||\mathbf{R}_i(\theta)||\,||\mathbf{R}_j(\theta)||} \tag{8}$$

where $\mathbf{R}_i(\theta)$ is the $N$-dimensional vector consisting of the responsibilities of the $i^{th}$ model for each of the examples. The split criterion uses the *local KL divergence* between the local density $f_k(\mathbf{y})$ around the $k^{th}$ model and the density of the $k^{th}$ model specified by the current parameters $\theta$:

$$\mathcal{J}_{\text{split}}(k;\theta) = \int f_k(\mathbf{y};\theta) \log \frac{f_k(\mathbf{y};\theta)}{P(\mathbf{y}|z_k,\theta_k)} d\mathbf{y} \tag{9}$$

where $z_k$ denotes the indicator variables of the $k^{th}$ model. The local density is defined as:

$$f_k(\mathbf{y};\theta) = \frac{\sum_{n=1}^{N} \delta(\mathbf{y}-\mathbf{y}_n) P(z_k|\mathbf{y}_n;\theta)}{\sum_{n=1}^{N} P(z_k|\mathbf{y}_n;\theta)} \tag{10}$$

### 3.4 Model Selection

An important issue in using MFA is choosing the number of factors ($k$) and the number of mixtures ($M$) to be used. The likelihood of the fit of the model to data is used to assess the goodness of fit in the maximum likelihood models. Factor analysis is a constrained Gaussian model, therefore, the best likelihood one can achieve using a FA is that of the full Gaussian model and the likelihood gets closer to this limit as the number of factors is increased. Thus, a $k$ value can be chosen depending on the closeness of the likelihood of different models to the full Gaussian model. Cross-validation can be used for determining the number of mixtures, in which several values of $M$ are fit to the data and the log likelihood on a validation set is used to select the the final value. Alternatively, a Bayesian analysis in which these parameters are determined automatically may also be used [1, 6].

[3] $r_{ij} = \langle z_i | \mathbf{y}_j \rangle$

## 4 Results

We first trained FA models with varying number of factors to decide how much dimensionality reduction we could have while keeping a good representation of the data. The results obtained by modelling the raw data was not very promising since only the likelihood of the models with large number of factors were close to the limiting likelihood value mentioned in section 3.4. As mentioned in section 3.1, the FA is sensitive to the orientation of the input axis. Therefore we tried to find a representation of the data that would allow good dimensionality reduction. Using the Fourier transform coefficients of the data helped to have high likelihood with lower number of factors (see figure 2 for a comparison). As seen in the figure, both raw data and Fourier transformed data likelihood converges to the full Gaussian model likelihood. The Fourier transformed data gets closer to this value with fewer factors (32 in this case). Therefore, we used the Fourier transformed data to train the MFA models that had 32 factors per mixture. It should be emphasised that we have used all coefficients of the Fourier transform obtained by a linear transform of the data, thus kept all information about the waveforms.

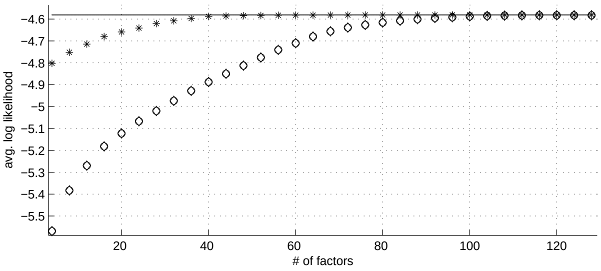

**Fig. 2.** Log likelihood for the FA models with only single mixture components trained with raw data (o) and with Fourier transformed data (*). The solid line shows the full Gaussian model likelihood, which is the limiting value. Note that using MFA, this limit can be exceeded, therefore this approach gives only an approximation of the necessary number of factors

After determining the number of components used in each mixture, the next step was to find an optimal number of mixtures $M$ in terms of the data likelihood and neuron isolation. We used a cross-validation scheme of MFA models trained on the whole waveform to determine $M$. We evaluated the resulting model by looking at the value of the entropies, observing the waveforms assigned to different clusters and comparing with manual clustering. In manual clustering the amplitude of one channel versus another is plotted in 2-D graphs in every possible combination of the channels, and the human operator manually draws boundaries around regions of high spike density. Therefore, we plotted the same kind of amplitude plots for comparison.

We trained the MFA models using SMEM algorithm to avoid local maxima. The cross-validation scheme determined $M = 9$ to be the optimal number of clusters. Unfortunately the average entropy of the responsibilities over all test samples was not as small as expected. In addition, we observed that the spikes assigned to some of the clusters seemed to be generated from different neurons, because their waveforms showed large deviations. As a consequence of this, those clusters were composed of multiple disjoint regions in the the amplitude plots and therefore did not match with the manual clustering. Thus we concluded that the clustering was not successful in terms of finding meaningful clusters. This inability to isolate the spikes with different characteristics is probably due to the fact that the model gets caught in local maxima where it cannot escape even with the help of the SMEM. To overcome this problem, we trained a model that had many more mixtures than the assumed number of neurons and merged some of the clusters after training, using the merge criteria of the SMEM algorithm.

Training a model with 30 mixtures and merging these clusters after training resulted in clusters that were similar to those of manual clustering. Specifically for the data set used in this study, the MFA model found all clusters that were found in manual clustering with similar boundaries, except for one cluster which it clustered as a part of noise. On the other hand, the model found some tiny clusters that had double spike waveforms, which correspond to nearly synchronously firing neurons. Also, the model identified another big cluster as a neuron, which was assigned to be noise in manual clustering due to its low amplitude. Figure 3 shows the examples that are assigned to different clusters. The trained model was also used to generate simulated data. As can be seen in figure 3, the simulated signals are very realistic in the sense that they resemble the recorded waveforms while showing some deviations.

Furthermore, the entropy of the responsibilities of both the training and test examples is close to zero, meaning that the model is almost always sure about to which cluster it should assign an example.

We have also trained a model on the 4 dimensional feature space used in manual clustering. SMEM algorithm helped to escape from the local maxima in this lower dimensional space, therefore training with more clusters was not necessary in this case. The clusters found by this low dimensional model were similar to the manually found clusters, but the double spike waveforms were not detected and the low amplitude cluster found by the higher dimensional model was assigned to the noise cluster, as expected.

## 5 Conclusion

We have demonstrated a successful approach to spike sorting of tetrode recordings using MFA. This method allows to model the whole spike waveforms and therefore can discriminate between neurons with similar amplitude characteristics across channels and also detect nearly simultaneously occurring spikes. The entropies of the responsibilities gives a measure of quality of clustering. The trained model can also be used to synthesise realistic data sets with labels that can be used to compare different spike sorting methods. A drawback of this method is that it is not fully unsupervised. The spike waveforms assigned to the resulting clusters should be assessed by an expert.

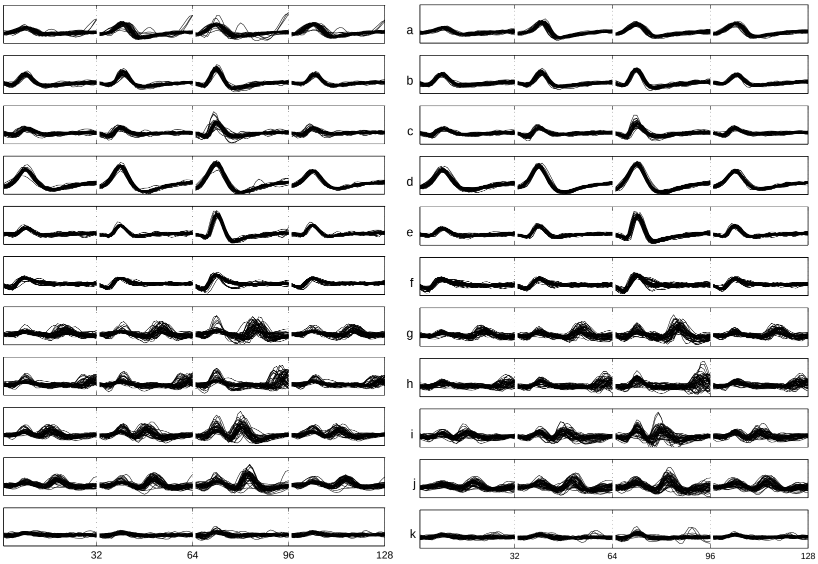

**Fig. 3.** Recorded samples assigned to different clusters by the model (left) and samples generated by the model (right). Waveforms produced by (a-f) different neurons, (g-j) nearly simultaneously firing neurons, (k) noise

**Acknowledgements** CER was supported by the German Research Council (DFG) through grant RA 1030/1.

## References

1. Ghahramani Z., Beal M.J.: Variational inference for Bayesian mixtures of factor analysers. In Adv. Neur. Inf. Proc. Sys. 12, MIT Press, 2000
2. Ghahramani Z., Hinton G.E.: The EM algorithm for mixtures of factor analysers. Technical Report CRG-TR-96-1, Dept. of Comp. Sci., Univ. of Toronto, 1996
3. Gray CM, Maldonado PE, Wilson M, McNaughton B.: Tetrodes markedly improve the reliability and yield of multiple single-unit isolation from multi-unit recordings in cat striate cortex. J Neurosci Methods. 1995 Dec;63(1-2):43-54
4. Harris K.D., Henze D.A., Csicsvari J., Hirase H., Buzsáki G.: Accuracy of tetrode spike separation as determined by simultaneous intracellular and extracellular measurements
5. Lewicki M.S.: A review of methods for spike sorting: the detection and classification of neural action potentials. Network 9 R53-R78, 1998
6. Rasmussen C.E.: The infinite Gaussian mixture model, In Adv. Neur. Inf. Proc. Sys. 12, MIT Press, 2000
7. Roweis S.T., Ghahramani Z.: A unifying review of linear Gaussian models. Neural Computation 11(2):305-345, 1999
8. Ueda N., Nakano R., Ghahramani Z., Hinton G.E.: SMEM algorithm for mixture models. In Adv. Neur. Inf. Proc. Sys. 11, MIT Press, 1999