An Approximate EM Approach to Sorting Overlapping Spikes

Sebastian Bitzer

Year Of Submission: 2006
Supervisor: Maneesh Sahani

Disclaimer: This report is submitted as part requirement for the M.Sc. Degree in Intelligent Systems at University College London. It is substantially the result of my own work except where explicitly indicated in the text.

This report may be freely copied and distributed provided the source is explicitly acknowledged.

Created : September 8, 2006

Contents

Abstract

In the past neuroscience has focused on explaining the function of single neurons, but now more and more interest is directed towards the local interactions between neurons. This leads to greater demands on spike sorting algorithms to cope with strongly correlated neurons which predominantly produce overlapping spikes.

While existing spike sorting algorithms work in two stages: the first to identify spike times and the second to cluster the found spike shapes, we propose an algorithm based on the expectation-maximisation (EM) algorithm which iterates between finding spike times given an estimate of spike shapes and estimating spike shapes given the current spike times. Overlapping spikes are automatically resolved in this framework and contribute to the learning process.

Because the computations involved in the original formulation of the algorithm are prohibitive, we further propose three approximations. Two of them have severe problems with local optima. Although the third overcomes these, this is payed for by a deviation from the original problem setting which proposed correction leads to bad convergence behaviour. Additionally we discuss possible improvements of the third approximation.

We define performance in terms of recall and precision. The algorithm is tested and compared to other algorithms on a test data set which has been previously published and very closely resembles real data. On average our algorithm correctly finds a bit less than 2 out of 3 spikes with considerable variation to the better and lower mainly depending on the spike shapes, the number of cells in the data and the noise level. The performance of one of the tested clustering algorithms is often better, although not much. Even a simple matched filter approach often performs as well as our more complicated algorithm. On the other hand, clustering and matched filter exhibit a large drop in performance when only overlapping spikes are considered while the performance of our algorithm stays equal.

In conclusion we find that our algorithm works not as good as wished, but automatically handles overlapping spikes as expected.

1 Introduction

How does the activity of single neurons in the brain lead to achievement of the various tasks that we are confronted with in daily life? To answer this ultimate question neuroscientists are recording the activity of single neurons and relate it to all kinds of behavioural tasks. Most commonly used is the so-called extracellular recording in which an electrode is guided near to single neurons, but not into them. An advantage of that practice is the ability to record more than one neuron with the same electrode at the same time. Actually this has long been seen as a disadvantage, because it makes necessary to establish cell identities through the characteristic shape of their spikes which is hard with the naked eye. This is the task of spike sorting.

When a neuron is active, it generates a spike which can be seen in the extracellular potential recorded by an electrode. The spikes of different neurons usually have a different characteristic shape and this enables us to extract the time at which the neuron was active by finding the characteristic shape in the extracellular recording. The problem is that the characteristic spike shapes are unknown and have to be found from the recording as well. Of course, there is also a lot of noise in the recording some of which is discarded by high-pass filtering with a suitable cutoff near the minimal spike shape frequency.

The spike sorting methods, that we know of, all work in two stages. In the first stage spikes in the recording are detected, aligned according to a certain feature like their peak and extracted from the recording. In the second stage the extracted spikes are clustered. For a review see Lewicki (1998). Once the first stage has been done successfully the clustering methods which are around are reasonably good (Quiroga et al., 2004; Harris et al., 2000; Rutishauser et al., 2006).

A special case that clustering methods can not handle is the case of overlapping spikes in the recording. These result when two cells are active at very close times. Such interactions will not fit in any cluster which is supposed to represent the characteristic shape of a single cell. Sometimes they are simply discarded as outliers, but this clearly has an effect on estimates of correlations between neurons (Bar-Gad et al., 2001) which are studied more and more to reveal how neighbouring neurons interact. Other methods of handling overlapping spikes after clustering have been proposed (Lewicki, 1994; Sahani, 1999; Takahashi et al., 2003), but since they rely on the clustering result they will fail when overlapping spikes are predominant in the data, for example because of strong correlations between recorded neurons.

Here we report about various variations of an algorithm which is designed to naturally handle overlapping spikes. The following sections explain the

algorithms, subsequently we present the performance of them on several test data sets and compare it to that of clustering methods in section 3.

2 Algorithms

The algorithms presented here are based on an iterative expectation-maximisation (EM) scheme which differ from earlier EM approaches in spike sorting in that they explicitly learn spike times as well as spike shapes. we will explain this idea and the underlying model in the next subsections. Three different, but related, approximate approaches to EM are shown. Following on that we will demonstrate how the problem can be reformulated to fit a quadratic program (QP) solver (2.3), describe a way of limiting the problem size to make it feasible for existing QP solvers (2.4) and discuss extensions to the model which have been considered (2.5).

2.1 Probabilistic Model

If two spikes overlap, the resulting waveform can be seen as the sum of two single spike waveforms which can be shifted with respect to each other. The main variables in this scenario are thus the spike times describing the shift and the single spike waveforms which we call spike shapes. Here we introduce the model which describes in a probabilistic framework how these variables supposedly interact to produce the recorded waveform.

Our model is adopted from Sahani (1999). The recorded waveform, v, constitutes its set of observable variables, the spike times, $\boldsymbol{\tau}$, constitute its hidden variables and the spike shapes, w, are parameters of the model. We are trying to capture the joint distribution of observable and hidden variables given all parameters, $\boldsymbol{\theta}$, which include w

$$P(v, \boldsymbol{\tau}|\boldsymbol{\theta}) = P(v|\boldsymbol{\tau}, \boldsymbol{\theta})P(\boldsymbol{\tau}|\boldsymbol{\theta}) \tag{1}$$

The terms on the right hand side are not dependent on all the parameters collected in $\boldsymbol{\theta}$. Where necessary we will make the dependent parameters explicit.

A reasonable first idea for a prior over spike times is a homogeneous Poisson process prior

$$P(\boldsymbol{\tau}|\lambda) = e^{-\lambda T}\lambda^n \tag{2}$$

where T is the length of the time span that we consider and n is the number of spikes in that interval. we focus on the prior in subsection 2.5, so let us

have a closer look at the posterior probability of the recorded waveform given the spike times $P(v|\boldsymbol{\tau}, \boldsymbol{\theta})$.

We assume that the recorded waveform consists of a linear superposition of many spike waveforms, most of which only appear as background noise and some occur as the foreground spikes we are aiming to identify. This assumption of linear additivity of spikes has been reported to hold for a recording from the locust lobula (Wehr et al., 1999) and makes intuitive sense, because currents add linearly, too. Hence, given we know the spike times, the recorded voltage at time t is a random variable with value

$$v(t) = \eta(t) + \sum_{i=1}^{N} \int_{t'} w(t - \tau_i)\delta(t - \tau_i - t') \tag{3}$$

where N is the number of spikes and $\eta(t)$ is a random noise variable determining the noise at time t. Instead of handling each spike, τ_i, separately we introduce a time-dependent indicator variable which is 1 at the time of a spike and 0 everywhere else

$$\rho(t) = \sum_{i=1}^{N} I(t - \tau_i) \qquad I(x) = \begin{cases} 1 & x = 0 \\ 0 & x \neq 0 \end{cases} \tag{4}$$

We also note that eq. (3) corresponds to a convolution, so we can write in short

$$v(t) = (w \otimes \rho)(t) + \eta(t). \tag{5}$$

Up to here we are only modelling a single cell with spike shape w, but for our application only recordings with more than one cell are interesting. Therefore we introduce as many spike shapes in our model as we expect to find in the recording (C)

$$v(t) = \eta(t) + \sum_{c=1}^{C} (w^c \otimes \rho^c)(t). \tag{6}$$

The distribution of v is determined by the distribution of the noise. We are using a Gaussian process with zero mean as an approximation and get

$$P(v|\boldsymbol{\rho}, \mathbf{w}, K) \propto \exp\left[-\frac{1}{2}\int_t \int_{t'} \left(v(t) - \sum_c (w^c \otimes \rho^c)(t)\right) K^{-1}(t, t') \left(v(t') - \sum_c (w^c \otimes \rho^c)(t')\right)\right]. \tag{7}$$

Although this approximation is not true in all cases, it often suits well enough (Sahani, 1999). We are further restricting the model to decorrelated noise and simplify to

$$P(v|\boldsymbol{\rho}, \mathbf{w}, \sigma) \propto \exp\left[-\frac{1}{2\sigma^2}\int_t \left(v(t) - \sum_c (w^c \otimes \rho^c)(t)\right)^2\right]. \tag{8}$$

This simplification introduces a drawback in our model, because it is known that the background noise is locally correlated (see section 3.2.2). However, we accept this loss of accuracy in favour of a gain of simplicity for the moment.

The model is expressed in terms of a continuous variable, but actually we are only handling data which has been sampled and digitised. Thus we write down the discrete version of eq. (8) as

$$P(\mathbf{v}|\boldsymbol{\rho}, \mathbf{W}, \sigma) \propto \exp\left[-\frac{1}{2\sigma^2}\sum_t \left(v_t - \sum_c (\mathbf{w}^c \otimes \boldsymbol{\rho}^c)_t\right)^2\right] \tag{9}$$

and note that v becomes a vector, $\boldsymbol{\rho}$ becomes a matrix[1] and vectors are indexed by subscript. The convolution of two finite vectors is a bit differently defined than the infinite, continuous version above, but we defer the exact explanation of it to section 2.3 where we present the implementation of the algorithm.

In the discrete space the homogeneous Poisson process prior becomes

$$P(\boldsymbol{\rho}|\pi) = \prod_c \prod_t \pi^{\rho_t^c}(1-\pi)^{1-\rho_t^c}. \tag{10}$$

Here π is the probability with which a cell spikes within the time interval defined by a single sample and we assume that each cell spikes with equal probability.

In summary, equations (9) and (10) define our final model. We assume constant spike shapes which linearly superpose, we assume additive, zero-mean, decorrelated Gaussian noise and we work in discrete space. The next subsection shows how spike shapes and times can be learned within this model.

2.2 Learning: Approximate EM

Our model, as given in eqs. (1) and (9) in the last subsection, is a probabilistic latent variable model. The standard way of learning within these

[1] this can actually not be seen, because there is no capital letter ρ in latex

models is the expectation-maximisation (EM) algorithm (see e.g. Bishop, 2006). Given some data from the visible variables it iterates between inferring the posterior distribution for the latent variables while the parameters are fixed (E-step) and learning parameter values given a fixed setting of the posterior distribution (M-step).

In particular, the EM algorithm is a maximum-likelihood approach for parameter learning, but instead of maximising the likelihood directly, which would not be tractable, it is working on the free energy, a lower bound on the log-likelihood, $\mathcal{L}$,

$$\begin{aligned} \mathcal{F}(q, \boldsymbol{\theta}) &= \int_{\boldsymbol{\rho}} q(\boldsymbol{\rho}) \log \frac{P(\mathbf{v}, \boldsymbol{\rho}|\boldsymbol{\theta})}{q(\boldsymbol{\rho})} && (11) \\ &= \mathcal{L}(\boldsymbol{\theta}) - \mathrm{KL}[q(\boldsymbol{\rho}) \| P(\boldsymbol{\rho}|\mathbf{v}, \boldsymbol{\theta})] && (12) \\ &= \langle \log P(\mathbf{v}, \boldsymbol{\rho}|\boldsymbol{\theta}) \rangle_q + \mathrm{H}(q) && (13) \end{aligned}$$

Here KL is the Kullback-Leibler divergence measuring the difference between two distributions, q is an arbitrary distribution defined over $\boldsymbol{\rho}$, H is the entropy and $\langle \cdot \rangle_q$ is the expected value with respect to distribution q.

In the E-step the free energy is maximised with respect to q. From equation (12) we can see that this is the case for $q(\boldsymbol{\rho}) = P(\boldsymbol{\rho}|\mathbf{v}, \boldsymbol{\theta})$, because the KL divergence is 0 there and positive everywhere else. Hence we would have to compute the posterior distribution of the latent variables in the E-step by

$$P(\boldsymbol{\rho}|\mathbf{v}, \boldsymbol{\theta}) = \frac{P(\mathbf{v}, \boldsymbol{\rho}|\boldsymbol{\theta})}{P(\mathbf{v}|\boldsymbol{\theta})} = \frac{P(\mathbf{v}|\boldsymbol{\rho}, \boldsymbol{\theta}) P(\boldsymbol{\rho}|\boldsymbol{\theta})}{\sum_{\boldsymbol{\rho}} P(\mathbf{v}|\boldsymbol{\rho}, \boldsymbol{\theta}) P(\boldsymbol{\rho}|\boldsymbol{\theta})}, \tag{14}$$

but the number of operations needed to compute the sum over the values of $\boldsymbol{\rho}$ increases exponentially with the number of spike times that one considers and the prior, $P(\boldsymbol{\rho}|\boldsymbol{\theta})$, is not a Gaussian distribution. Therefore computing $P(\boldsymbol{\rho}|\mathbf{v})$ directly is not tractable.

In the M-step we maximise the free energy with respect to parameters, $\boldsymbol{\theta}$. Consider eq. (13). Because the entropy of a fixed q is not dependent on the parameters, maximising the free energy with respect to parameters is equivalent to maximising

$$\langle \log P(\mathbf{v}, \boldsymbol{\rho}|\boldsymbol{\theta}) \rangle_q = \langle \log P(\mathbf{v}|\boldsymbol{\rho}, \boldsymbol{\theta}) \rangle_q + \langle \log P(\boldsymbol{\rho}|\boldsymbol{\theta}) \rangle_q \tag{15}$$

with

$$\begin{aligned}\langle \log P(\mathbf{v}|\boldsymbol{\rho},\boldsymbol{\theta})\rangle_q &= \left\langle k_1 - \frac{1}{2\sigma^2}\sum_t \left(v_t - \sum_c (\mathbf{w}^c \otimes \boldsymbol{\rho}^c)_t\right)^2 \right\rangle_q \\ &= k_2 - \frac{1}{2\sigma^2}\left[\sum_t \left\langle \left(\sum_c (\mathbf{w}^c \otimes \boldsymbol{\rho}^c)_t\right)^2 \right\rangle_q - 2v_t \left\langle \sum_c (\mathbf{w}^c \otimes \boldsymbol{\rho}^c)_t \right\rangle_q\right].\end{aligned} \tag{16}$$

A further look on the terms over which we take the expectation with respect to q

$$\begin{aligned}\left\langle \sum_c (\mathbf{w}^c \otimes \boldsymbol{\rho}^c)_t \right\rangle_q &= \left\langle \sum_c \sum_{t'} w_{t'}^c \rho_{t-t'}^c \right\rangle_q \\ &= \sum_c \sum_{t'} w_{t'}^c \left\langle \rho_{t-t'}^c \right\rangle_q\end{aligned} \tag{17}$$

$$\begin{aligned}\left\langle \left(\sum_c (\mathbf{w}^c \otimes \boldsymbol{\rho}^c)_t\right)^2 \right\rangle_q &= \left\langle \left(\sum_{c'} \sum_{t'} w_{t'}^{c'} \rho_{t-t'}^{c'}\right)\left(\sum_{c''} \sum_{t''} w_{t''}^{c''} \rho_{t-t''}^{c''}\right) \right\rangle_q \\ &= \sum_{c',c''} \sum_{t',t''} w_{t'}^{c'} w_{t''}^{c''} \left\langle \rho_{t-t'}^{c'} \rho_{t-t''}^{c''} \right\rangle_q\end{aligned} \tag{18}$$

shows that we need q to calculate the expectation of the spikes, $\langle \rho_t^c \rangle_q$, and the expectation of their interactions, $\langle \rho_t^c \rho_{t'}^{c'} \rangle_q$, but the number of operations needed to compute the expectation will increase exponentially with the number of spike times, too[2]. Hence, the standard EM-algorithm is doubly intractable in this case, because q can not be set to the KL-minimising posterior and even if we had q equal to the posterior, we could not compute the expectations with respect to q. An approximation is needed as is provided by the variational EM algorithm.

2.2.1 Variational EM

If we assume that a spike at a certain time is independent of a spike at any other time given the observed data and the parameters, the posterior

[2]assuming that there is no suitable representation that can be used to skip summation over all settings of spike times

distribution and also q become fully factorised. In particular we have for q

$$q(\boldsymbol{\rho}) = \prod_c \prod_t q_t^c(\rho_t^c) = \prod_c \prod_t (r_t^c)^{\rho_t^c}(1 - r_t^c)^{1-\rho_t^c}. \tag{19}$$

In this case the expectations simply are

$$\begin{array}{rcll} \langle \rho_t^c \rangle_q & = & r_t^c & \\ \langle \rho_t^c \rho_t^c \rangle_q & = & r_t^c & \\ \langle \rho_t^c \rho_{t'}^{c'} \rangle_q & = & r_t^c r_{t'}^{c'} & |(c,t) \neq (c',t'). \end{array} \tag{20}$$

we will call the r_t^c "expected spike times".

Of course, the assumption of independent spikes violates our intuitions about the interactions between spikes. For example, if there is a waveform generated from one spike of one cell at a certain time point in the data, then putting that spike at the corresponding time point in our model will explain away the waveform in the data. As a consequence spikes at nearby times should be anti-correlated (we will look at this point from the perspective of the prior in section 2.5.2). However, despite such shortcomings this assumption is a well-known and efficient approximation which makes learning in our model feasible. It is an example of a variational approximation (Jordan et al., 1999) which in this form in which we use a fully factorised distribution is also called mean field approximation.

The parameters in our model are the variance of the noise, σ^2, the parameters introduced through the prior of the spike times, and the spike shapes, $\mathbf{W}$. We are estimating the first two separately as described in section 2.4. Consequently we only have to learn the spike shapes in the M-step for our model which becomes

$$\begin{aligned} \arg\max_{\mathbf{W}} \mathcal{F}(q, \mathbf{W}) &= \arg\max_{\mathbf{W}} \langle \log P(\mathbf{v}|\boldsymbol{\rho}, \mathbf{W}) \rangle_q \\ &= \arg\min_{\mathbf{W}} \frac{1}{2\sigma^2} \sum_t \left(v_t - \sum_c (\mathbf{w}^c \otimes \mathbf{r}^c)_t \right)^2 . \end{aligned} \tag{21}$$

Because the convolution is only a linear operation, we can see that the M-step simplifies to least squares fitting when we use the mean field approximation. Note that this equation is slightly wrong in that the quadratic, $r_t^c r_t^c$, should be replaced by its linear term, r_t^c, in correspondence with eq. (20). we will address this issue and the implementation in general in section 2.3.

We now see that the M-step becomes easy, but we still have to solve the E-step. Equation (12) tells us to minimise the KL divergence between our approximate distributions and the posterior of spike times in order to maximise the free energy in the E-step. Because we restrict ourselves to a certain

set of distributions, there is no immediate solution to the minimisation of the KL divergence anymore and we have to consider the actual optimisation which is

$$\begin{aligned}\arg\min_q \mathrm{KL}[q(\boldsymbol{\rho})\|P(\boldsymbol{\rho}|\mathbf{v},\boldsymbol{\theta})] &= \arg\min_q \sum_{\boldsymbol{\rho}} q(\boldsymbol{\rho}) \log \frac{q(\boldsymbol{\rho})}{P(\boldsymbol{\rho}|\mathbf{v},\boldsymbol{\theta})} \\ &= \arg\min_q \sum_{\boldsymbol{\rho}} q(\boldsymbol{\rho}) \log q(\boldsymbol{\rho}) - q(\boldsymbol{\rho}) P(\boldsymbol{\rho}|\mathbf{v},\boldsymbol{\theta}) \\ &= \arg\min_q -\mathrm{H}(q) - \langle \log P(\boldsymbol{\rho}|\mathbf{v},\boldsymbol{\theta})\rangle_q\end{aligned}$$

and because $P(\boldsymbol{\rho}|\mathbf{v},\boldsymbol{\theta}) = P(\mathbf{v},\boldsymbol{\rho}|\boldsymbol{\theta})/P(\mathbf{v}|\boldsymbol{\theta})$, because the likelihood is independent of q and using eq. (15) we finally have to minimise with respect to q

$$F = -\langle \log P(\mathbf{v}|\boldsymbol{\rho},\boldsymbol{\theta})\rangle_q - \langle \log P(\boldsymbol{\rho}|\boldsymbol{\theta})\rangle_q - \mathrm{H}(q). \tag{22}$$

So far we stayed general and have not used any of the properties of our special q as given in eq. (19). Let us consider the contributing terms from above in turn.

As seen before, the expected log-probability of the data under our approximation is

$$\langle \log P(\mathbf{v}|\boldsymbol{\rho},\boldsymbol{\theta})\rangle_q = -\frac{1}{2\sigma^2} \sum_t \left(v_t - \sum_c (\mathbf{w}^c \otimes \mathbf{r}^c)_t \right)^2. \tag{23}$$

Because the convolution is linear, we can rewrite this as

$$\begin{aligned}\langle \log P(\mathbf{v}|\boldsymbol{\rho},\boldsymbol{\theta})\rangle_q &= -\frac{1}{\sigma^2} \left(\sum_{c'} \sum_{c''} \frac{1}{2} (\mathbf{r}^{c'})^\top \mathbf{H}^{c',c''} \mathbf{r}^{c''} + \sum_c (\mathbf{f}^c)^\top \mathbf{r}^c \right) \\ &= -\frac{1}{\sigma^2} \sum_c \sum_t \left(f_t^c r_t^c + \frac{1}{2} \sum_{c'} \sum_{t'} r_t^c h_{t,t'}^{c,c'} r_{t'}^{c'} \right)\end{aligned} \tag{24}$$

where $\mathbf{H}^{c',c''}$ and $\mathbf{f}^c$ are the quadratic and linear coefficients which will be derived in section 2.3.

The expected log-prior of the spike times from eq. (10) is

$$\begin{aligned}\langle \log P(\boldsymbol{\rho}|\boldsymbol{\theta})\rangle_q &= \sum_c \sum_t r_t^c \log \pi + (1 - r_t^c) \log[1 - \pi] \\ &= \sum_c \sum_t (\log \pi - \log[1 - \pi]) r_t^c + \log[1 - \pi].\end{aligned} \tag{25}$$

and the entropy of q is

$$\mathrm{H}(q) = -\sum_c \sum_t r_t^c \log r_t^c + (1 - r_t^c) \log[1 - r_t^c]. \tag{26}$$

Then, by substituting eqs. (24), (25) and (26) into eq. (22) we see that minimising the KL divergence with respect to q is equivalent to minimising the following with respect to $\mathbf{R}$

$$\begin{aligned} F = \sum_c \sum_t \frac{1}{\sigma^2} \left(f_t^c r_t^c + \frac{1}{2} \sum_{c'} \sum_{t'} r_t^c h_{t,t'}^{c,c'} r_{t'}^{c'} \right) \\ - \ (\log \pi - \log[1 - \pi]) r_t^c - \log[1 - \pi] \\ + \ r_t^c \log r_t^c + (1 - r_t^c) \log[1 - r_t^c]. \end{aligned} \tag{27}$$

By differentiation of F we derive the fixed point equations. Note that the diagonals of $\mathbf{H}^{c,c'}$ are zero, because the coefficients for the quadratic $r_t^c r_t^c$ have been added to the linear coefficients (see section 2.3) in accordance with eqs. (20).

$$\begin{aligned} \frac{\partial F}{\partial r_\tau^\zeta} &= \frac{1}{\sigma^2} f_\tau^\zeta + \frac{1}{\sigma^2} \sum_{(c,t) \neq (\zeta,\tau)} r_t^c h_{t,\tau}^{c,\zeta} - \log \pi + \log[1 - \pi] + \log r_\tau^\zeta - \log[1 - r_\tau^\zeta] \\ &= \log \left[\frac{r_\tau^\zeta}{1 - r_\tau^\zeta} \right] + \frac{1}{\sigma^2} \sum_{(c,t) \neq (\zeta,\tau)} r_t^c h_{t,\tau}^{c,\zeta} + \frac{1}{\sigma^2} f_\tau^\zeta - \log \pi + \log[1 - \pi] \end{aligned}$$

Setting the derivative to zero we can rearrange and get an equation which solves for r_τ^ζ dependent on all other r_t^c

$$r_\tau^\zeta = g \left(\log \left[\frac{\pi}{1 - \pi} \right] - \frac{1}{\sigma^2} \sum_{(c,t) \neq (\zeta,\tau)} r_t^c h_{t,\tau}^{c,\zeta} - \frac{1}{\sigma^2} f_\tau^\zeta \right) \tag{28}$$

with $g(x) = 1/(1 + e^{-x})$ being the sigmoid function. Each of the expected spike times, r_τ^ζ, is then updated iteratively until the KL divergence does not decrease anymore. It is not guaranteed that this will be a global minimum of $\mathrm{KL}[q(\boldsymbol{\rho}) \| P(\boldsymbol{\rho} | \mathbf{v}, \boldsymbol{\theta})]$ and consequently the result depends on the initial values for the expected spike times.

Finally, we arrived at a complete algorithm. The E-step sets the expected spike times to the converged solutions of eq. (28) and the M-step uses them to update the spike shapes according to eq. (21). The two steps are iterated until the free energy converges.

The original EM algorithm always converges to a maximum of the likelihood, because the E-step sets the free energy equal to the likelihood. By restricting the set of distributions allowable in the E-step the free energy can not be set equal to the likelihood anymore and correspondingly a found maximum in the free energy will not be a maximum in the likelihood.

2.2.2 Assuming a Deterministic Posterior Distribution

The variational EM algorithm as described above is very efficient, but does not give good solutions (see section 3.5). One reason for this is its susceptibility to run into local optima at various stages in the algorithm. We propose further approximations under which we can find a good optimum for expected spike times in the E-step.

In a first step we note from eq. (27) that our problem is quadratic, if the entropy is zero for all choices of q. The entropy of a distribution is zero, if it is deterministic, i.e. the probability for exactly one setting of the variables is equal to one and for all other settings it is zero. In the case of fully factored q this means that all expected spike times, r_t^c, should be either 0 or 1. Therefore instead of minimising eq. (27) for continuous expected spike times we now have

$$\begin{aligned} F &= \sum_c \sum_t \frac{1}{\sigma^2} \left(f_t^c r_t^c + \frac{1}{2} \sum_{c'} \sum_{t'} r_t^c h_{t,t'}^{c,c'} r_{t'}^{c'} \right) \\ &\quad - (\log \pi - \log[1-\pi]) r_t^c - \log[1-\pi] \end{aligned} \tag{29}$$

$$\begin{aligned} F &= -\langle \log P(\mathbf{v}|\boldsymbol{\rho}, \boldsymbol{\theta}) \rangle_q - \langle \log P(\boldsymbol{\rho}|\boldsymbol{\theta}) \rangle_q \\ &= -\langle \log P(\mathbf{v}, \boldsymbol{\rho}|\boldsymbol{\theta}) \rangle_q \\ &= -\langle \log P(\boldsymbol{\rho}|\mathbf{v}, \boldsymbol{\theta}) \rangle_q - \log Z \end{aligned} \tag{30}$$

which has to be minimised for binary expected spike times. At the same time we see that we are actually maximising the expected log-posterior probability of the spike times. With a deterministic q the expected log-posterior probability is simply the log-posterior probability of the spike times, $\boldsymbol{\rho}$, for which $q(\boldsymbol{\rho}) = 1$. In other words, minimising $\mathrm{KL}[q(\boldsymbol{\rho}) \| P(\boldsymbol{\rho}|\mathbf{v}, \boldsymbol{\theta})]$, when q is constrained to be deterministic, is equivalent to finding the most probable spike times given the data. This is exactly what we are finally looking for. Note that we do not have to compute the (intractable) normalising constant for the posterior probabilities, because we are only interested in the position of the maximum which stays the same for any uniform scaling.

Also note that we actually do not have to assume that q is fully factored to get the relationship between q and most probable spike times, but it has to be deterministic. This assumption of determinism seems very drastic, because we take away the uncertainty out of our probabilistic model. However, when spike shapes are sufficiently different and the noise is comparably small, we expect to be very certain about a particular posterior setting of spike times, especially considering the anti-correlation between spikes. Therefore, we expect this assumption to be reasonably faithful. Additionally we tested the hypothesis of (quasi-)deterministic posterior probabilities for very small sets of spike times on an example data set and got confirmed. See section 3.1 for procedures and results.

One could use genetic algorithms for optimisation in this binary space. This would have the advantage that we would not have to make further assumptions. But it is hard to tweak the parameters of genetic algorithms, they are not guaranteed to find the global optimum and they are very slow. Additionally, since the posterior is near zero for almost all sets of spike times, we are facing a very difficult optimisation problem, because the space in which we optimise hardly gives any hints about the position of the optimum and genetic algorithms usually do not work well in these situations.

Instead we make use of the mean field approximation again. If we had r_t^c continuous instead of binary, we could use quadratic programming to find a maximum of eq. (29), but this violates our assumption of determinism. However, if we have a closer look at $\langle \log P(\boldsymbol{\rho}|\mathbf{v}, \boldsymbol{\theta}) \rangle_q$ which has to be maximised

$$\sum_{\boldsymbol{\rho}} q(\boldsymbol{\rho}) \log P(\boldsymbol{\rho}|\mathbf{v}, \boldsymbol{\theta}) = \sum_{\boldsymbol{\rho}} \log P(\boldsymbol{\rho}|\mathbf{v}, \boldsymbol{\theta}) \prod_c \prod_t (r_t^c)^{\rho_t^c} (1 - r_t^c)^{1-\rho_t^c}, \quad (31)$$

we notice that the factor $\prod_c \prod_t (r_t^c)^{\rho_t^c} (1 - r_t^c)^{1-\rho_t^c}$, which has its maximum at 1 and falls off quickly, very strongly pushes towards a deterministic solution. Furthermore, because $\log P$ is concave, the maximum of $\langle \log P(\boldsymbol{\rho}|\mathbf{v}, \boldsymbol{\theta}) \rangle_q$ in continuous space is where the maximum of $P(\boldsymbol{\rho}|\mathbf{v}, \boldsymbol{\theta})$ gets full support, that is where $\mathbf{R}$ is the deterministic setting which is equal to the setting of $\boldsymbol{\rho}$ with the highest posterior probability. Hence, the maximum in continuous space is equal to the maximum in binary space. Consequently we can use quadratic programming in continuous space to solve our binary problem.

Yet, there is one big problem: the quadratic program that we have to solve is non-convex and hence it is not guaranteed that we find the global optimum. Equation (20) tells us that the expectation of a quadratic spike time is the posterior probability of a spike at that time ($\langle \rho_t^c \rho_t^c \rangle_q = r_t^c$). This means that there are no terms $r_t^c r_t^c$ in our quadratic problem and thus the

corresponding hessian has only zeros in the diagonal, but such matrices are non-convex as is shown in appendix A.

So we most likely get stuck in local optima. How are they characterised? Because the posterior probability is zero or almost zero for most settings of spike times and those settings add a very high cost to our objective function (cf. (31)), a local optimum will most likely be a deterministic setting of the expected spike times, r_t^c, which selects the spike times with the largest posterior probability in their neighbourhood. The selected spike times, however, might have a very small posterior themselves.

2.2.3 A Related Convex Problem

To overcome the fallacy of such bad local optima we try to change our problem so we can easily find the global optimum of the new problem without departing from the original problem too much. A change suggesting itself is substituting $\langle \rho_t^c \rho_t^c \rangle_q = r_t^c$ with

$$\langle \rho_t^c \rho_t^c \rangle_q = r_t^c r_t^c. \tag{32}$$

Then, minimising eq. (29) is equivalent to directly minimising

$$\begin{aligned} & \frac{1}{2\sigma^2} \sum_t \left(v_t - \sum_c (\mathbf{w}^c \otimes \mathbf{r}^c)_t \right)^2 - \sum_c \sum_t (\log \pi - \log[1-\pi]) r_t^c \\ = \; & \frac{1}{2\sigma^2} \sum_t \left(v_t - \sum_c (\mathbf{w}^c \otimes \mathbf{r}^c)_t \right)^2 + \sum_c \sum_t \log \left[\frac{1-\pi}{\pi} \right] r_t^c \end{aligned} \tag{33}$$

with respect to r_t^c and we see that the quadratic terms all stem from the left part which can be transformed into a least-squares problem and therefore is convex.

We can interpret the two parts of eq. (33) in an intuitive way. The first summand is responsible for fitting the data with scaled versions of the spike shapes and the second summand adds a cost for large values of the expected spike times which depends on the prior over spike times.

Clearly we want that the values for the expected spike times range between zero and one, $0 \leq r_t^c \leq 1$. Therefore, the analogy to least squares fitting does not fully apply and we have to use a quadratic program solver which allows bounds on the variables to be set. Given our bounds the second summand of eq. (33) pushes the expected spike times towards zero and hence leads to sparse results.

The diagonal in the new problem has been taken from the linear terms in the old problem. We will see in section 2.3 that these entries are all

positive. In combination with linear terms positive values penalise large expected spike times (compare contribution of the prior above), but when the same positive values are combined with quadratic terms they penalise large expected spike times proportionally more than small expected spike times. As a result we find that solutions to our new problem tend to have several non-zero expected spike times most of which are far from one. Such solutions violate our assumption of determinism and do not fit the near-deterministic posterior distributions reported in section 3.1.

We have two ways to correct for this outcome. First, we can seed the optimisation for the original problem with the solution of the new problem. Because the problems are similar, one would hope that the solution of the latter lies in the vicinity of the global optimum of the former problem, although this is not guaranteed at all. The second workaround simply sets all expected spike times above a certain threshold to 1 and all others to 0, thereby producing a deterministic result. If we still interpret the value, r_t^c, coming out of the new problem as the posterior probability that there is a spike of cell c at time t, then a threshold of 0.5 is a good candidate, because this would choose the setting of spike times with the greatest probability.

There is a completely different interpretation of the new problem setting. Indeed it is hard to relate the here presented convex problem to its non-convex brother from the last section, because it is unclear how the new r_t^c relate to that of the mean field approximation. Alternatively, we can look at the minimisation of the error as depicted by eq. (33) as an optimal method to find times at which a spike with a given shape occurs in a scaled version. For the learning in an EM-framework, however, we need estimates of the expected spike times for the M-step. So we can use our noise model to calculate the probability, $r_t^c = P(\rho_t^c = 1 | s_t^c, \mathbf{w}^c)$, of there being a spike of a certain cell at a certain spike time given that our convex problem found a spike of that cell at that spike time which is scaled by s_t^c, where s_t^c is part of the solution of our convex optimisation problem which we originally called r_t^c above. In other words, we are estimating the probability that there is a genuine spike in the data, but that the noise is responsible for scaling it by s_t^c. The formulas are

$$P(s_t^c | \rho_t^c, \mathbf{w}^c) \propto \exp\left[-\frac{1}{2\sigma^2}\|\rho_t^c \mathbf{w}^c - s_t^c \mathbf{w}^c\|^2\right] \tag{34}$$

then

$$\begin{aligned} P(\rho_t^c = 1 | s_t^c, \mathbf{w}^c) &= \frac{P(s_t^c | \rho_t^c = 1, \mathbf{w}^c) P(\rho_t^c = 1)}{\sum_{\rho_t^c} P(s_t^c | \rho_t^c, \mathbf{w}^c) P(\rho_t^c)} \\ &= \frac{P(s_t^c | \rho_t^c = 1, \mathbf{w}^c)\pi}{\sum_{\rho_t^c} P(s_t^c | \rho_t^c, \mathbf{w}^c)\pi^{\rho_t^c}(1-\pi)^{1-\rho_t^c}} \end{aligned} \tag{35}$$

where π stems from the prior in eq. (10). More abstractly we can write the calculated probabilities as a function of the scalings that we found, $r_t^c = g_{\mathbf{w}^c}(s_t^c)$, and it turns out that this function is a very steep sigmoidal centred at 0.5. So, on a completely different route we again arrive at thresholding the solution of optimisation.

However we motivate any kind of supplementary thresholding of optimisation results, it represents a jump to a different part in the optimisation space which may have, and in the convex case certainly has, a worse objective function value. Therefore, the very nice property of the EM-algorithm is lost that it is proven to increase free energy, or decrease error in our case, in every step. As a result, convergence of this algorithm with thresholding is not guaranteed. An example of the convergence behaviour can be seen in figure 1.

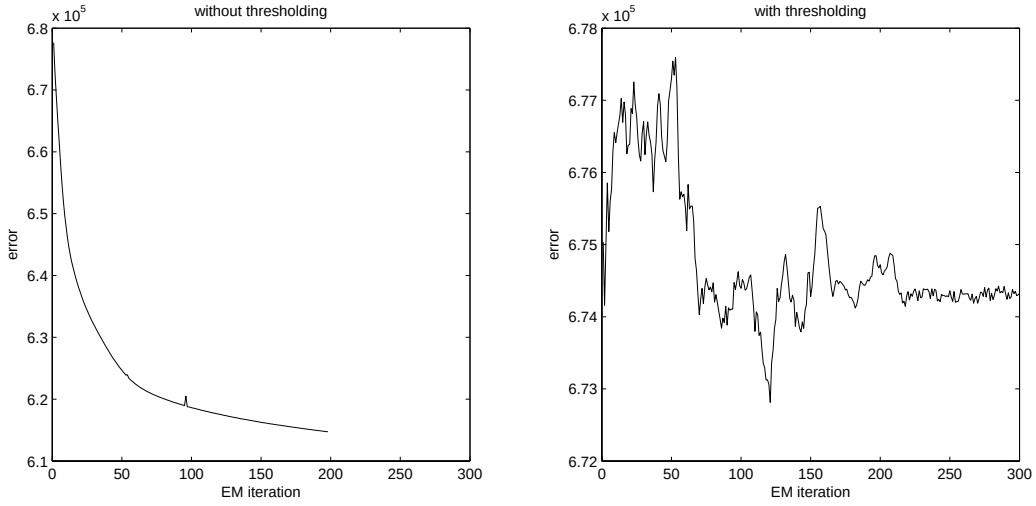

Figure 1: Two example runs of the convex problem algorithm without and with thresholding on the same data set. The runs have been stopped when the error changed by less than 0.01. Without thresholding the algorithm decreases the error in every step. This is not true for some steps in the plot. These are due to the inability of the MINQ QP-solver to find the global minimum of our convex problem in some cases. With thresholding the algorithm moves freely in error space until converging at a non-optimal point.

Since it is not at all clear anymore when the algorithm is best stopped except in retrospect, we are introducing a heuristic stopping criterion. We stop when the best achieved error has not been changed for the last 10 iterations and return the solution which produced this minimal error. This was chosen as a trade-off between running time and achieved minimal error

and is motivated by the informal observation that the performance of the solution as defined in section 3.3 does not significantly increase after the criterion is met, even if the error is further reduced.

Despite all shortcomings we find that the convex optimisation with thresholding is the best of the three algorithms in recovering spike shapes in the data, but still performs worse than clustering methods. we report detailed results in section 3, but before proceeding there we present details of the implementation and further considerations to improve the algorithm.

2.3 Reformulation as Quadratic Program

In the presentation of the algorithm we repeatedly derived quadratic subproblems (see eqs. (21), (29) and (33)). The variables we want to optimise, however, occur within a convolution, $(\mathbf{w}^c \otimes \mathbf{r}^c)_t$, and need to be separated to finally fit the quadratic programming formulation

$$\frac{1}{2}\mathbf{r}^\top \mathbf{H}\mathbf{r} + \mathbf{f}^\top \mathbf{r} \tag{36}$$

The convolution for discrete, finite vectors $\mathbf{w}^c$ and $\mathbf{r}^c$ which have elements $w_0, \ldots, w_{T_w}$ and $r_0, \ldots, r_{T_r}$ is defined as

$$(\mathbf{w}^c \otimes \mathbf{r}^c)_t = \sum_{i=t_0(t)}^{T(t)} w_i r_{t-i} \tag{37}$$

with $t_0(t)$ and $T(t)$ chosen such that the vector borders are not violated

$$t_0(t) = \max(0, t - T_r) \qquad T(t) = \min(T_w, t)$$

Using this definition we will now examplarily show how we can transform eq. (33) to the quadratic form of eq. (36). First we repeat eq. (33) and simultaneously substitute $\log[(1-\pi)/\pi]$ by a to ease the writing

$$\frac{1}{2\sigma^2}\sum_t \left(v_t - \sum_c (\mathbf{w}^c \otimes \mathbf{r}^c)_t \right)^2 + a \sum_c \sum_{t_r} r^c_{t_r}$$

Multiplying out the quadratic term gives

$$\frac{1}{2\sigma^2}\sum_t v_t^2 - 2v_t \sum_c (\mathbf{w}^c \otimes \mathbf{r}^c)_t + \left(\sum_c (\mathbf{w}^c \otimes \mathbf{r}^c)_t \right)^2 + a \sum_c \sum_{t_r} r^c_{t_r}$$

The term v_t^2 is constant in our optimisation problems, so we ignore it from here.

2.3.1 Linear Part

For now let us only look at the linear term of the quadratic

$$\frac{1}{\sigma^2}\sum_t -v_t \sum_c (\mathbf{w}^c \otimes \mathbf{r}^c)_t$$

Substituting in the definition of the convolution we get

$$\frac{1}{\sigma^2}\sum_t -v_t \sum_c \sum_{i=t_0(t)}^{T(t)} w_i^c r_{t-i}^c$$

This has to be transformed into

$$-\frac{1}{\sigma^2}\sum_c \sum_{t_r} g_{t_r}^c r_{t_r}^c$$

So finding $f_{t_r}^c$ corresponds to finding those v_t and w_i^c for which $t - i = t_r$ which is

$$g_{t_r}^c = \sum_{i=0}^{T_w} w_i^c v_{t_r+i} \tag{38}$$

You can imagine this as moving a window of the length of the spike shapes over the data and taking the inner product of this with the spike shapes, or alternatively as the centre part of the convolution between the reversed spike shapes and the data which corresponds to a matched filter on the data.

Now we can reintroduced the term from the prior

$$-\frac{1}{\sigma^2}\sum_c \sum_{t_r} g_{t_r}^c r_{t_r}^c + a\sum_c \sum_{t_r} r_{t_r}^c$$

and see that we finally get

$$\frac{1}{\sigma^2}\sum_c \sum_{t_r} f_{t_r}^c r_{t_r}^c = \frac{1}{\sigma^2}\sum_c \sum_{t_r} (a\sigma^2 - g_{t_r}^c) r_{t_r}^c \tag{39}$$

which is in vector form $\sum_c (\mathbf{f}^c)^\top \mathbf{r}^c/\sigma^2$.

2.3.2 Quadratic Part

For the quadratic term the procedure is a bit more complicated, but similar. We have

$$\frac{1}{2\sigma^2}\sum_t \left(\sum_c (\mathbf{w}^c \otimes \mathbf{r}^c)_t\right)^2 = \frac{1}{2\sigma^2}\sum_t \sum_{c',c''} \sum_{i,j=t_0(t)}^{T(t)} w_i^{c'} w_j^{c''} r_{t-i}^{c'} r_{t-j}^{c''}$$

Here we need to get to the form

$$\frac{1}{2\sigma^2}\sum_{c',c''}\sum_{t'_r,t''_r} r_{t'_r}^{c'} h_{t'_r,t''_r}^{c',c''} r_{t''_r}^{c''}$$

So we have to find those $w_i^{c'}$ and $w_j^{c''}$ for which we get $t - i = t'_r$ together with $t - j = t''_r$. For $t'_r = t''_r = t_r$ these are

$$h_{t_r,t_r}^{c',c''} = \sum_{i=0}^{T_w} w_i^{c'} w_i^{c''} = \mathbf{w}^{c'\top}\mathbf{w}^{c''}$$

Note that for $c' = c''$ this is a sum of squared terms which is either positive or zero. When $t'_r \neq t''_r$ the sums over i, j are offset. Let $\Delta t_r = t'_r - t''_r$, then

$$h_{t'_r,t''_r}^{c',c''} = \sum_{i=0}^{T_w-\Delta t_r} \begin{cases} w_i^{c'} w_{i+\Delta t_r}^{c''} & \Delta t_r > 0 \\ w_{i+\Delta t_r}^{c'} w_i^{c''} & \Delta t_r < 0 \end{cases}$$

Thus we see that the values of h are equal along the diagonals. Another way to look at this is to note that each column (or row) contains the convolution of the reversed $\mathbf{w}^{c'}$ with $\mathbf{w}^{c''}$ such that the centre of the convolution is at the diagonal of $\mathbf{H}^{c',c''}$, that is at $h_{t_r,t_r}^{c',c''}$.

2.3.3 Combining Cells

So far we have found the formulation

$$\frac{1}{\sigma^2}\left[\frac{1}{2}\left(\sum_{c',c''}\mathbf{r}^{c'\top}\mathbf{H}^{c',c''}\mathbf{r}^{c''}\right) + \sum_{c}\mathbf{f}^{c\top}\mathbf{r}^{c}\right]$$

Because we sum over cells, it is equivalent to concatenate the $\mathbf{f}^c$s and $\mathbf{H}^{c',c''}$s to form one $\mathbf{H}$ and $\mathbf{f}$ to get the final quadratic programming formulation

$$\frac{1}{\sigma^2}\left[\frac{1}{2}\mathbf{r}^\top\mathbf{H}\mathbf{r} + \mathbf{f}^\top\mathbf{r}\right]$$

with

$$\mathbf{r} = \begin{bmatrix} \mathbf{r}^1 \\ \vdots \\ \mathbf{r}^C \end{bmatrix} \qquad \mathbf{H} = \begin{bmatrix} \mathbf{H}^{1,1} & \dots & \mathbf{H}^{1,C} \\ \vdots & \ddots & \vdots \\ \mathbf{H}^{C,1} & \dots & \mathbf{H}^{C,C} \end{bmatrix} \qquad \mathbf{f} = \begin{bmatrix} \mathbf{f}^1 \\ \vdots \\ \mathbf{f}^C \end{bmatrix}$$

with $\mathbf{H}^{c',c''} = (\mathbf{H}^{c'',c'})^\top$ and C being the total number of cells in the problem.

We also presented problems (sections 2.2.1, 2.2.2) in which the quadratic terms $r_t^c r_t^c$ have been replaced by the linear term r_t^c. It is very easy to convert between these two settings as we only have to take the diagonal entries of $\mathbf{H}$ and set them to zero while also adding them to eq. (39) in the following way

$$f_{t_r}^c = a\sigma^2 - g_{t_r}^c + h_{t_r,t_r}^{c,c}.$$

This completes the reformulation of our particular quadratic problem into a form which can be used with standard QP-solvers. We tried out two such solvers, the Matlab (Optimization Toolbox) solver `quadprog` and freely available `MINQ` by Arnold Neumaier, University of Vienna[3]. Although `MINQ` is considerably faster than `quadprog` and usually gives correct results, sometimes the solution it finds depends on the seed that it has been given, even if the problem is convex. Due to this inconsistency the results we report are obtained with `quadprog` unless stated otherwise.

2.3.4 Spike Shapes

The above derivation has been done for optimising the spike times $\mathbf{r}$. However, since $(\mathbf{w}^c \otimes \mathbf{r}^c)_t = (\mathbf{r}^c \otimes \mathbf{w}^c)_t$, the same applies for the spike shapes $\mathbf{w}$ with the exception that we have no contribution from the prior in that case. Thus we get

$$f_{t_w}^c = -\sum_{i=0}^{T_r} r_i^c v_{t_w+i}$$

and

$$h^{c',c''}(t'_w, t''_w) = \sum_{i=0}^{T_r-\Delta t_w} \begin{cases} r_i^{c'} r_{i+\Delta t_w}^{c''} & \Delta t_w \geq 0 \\ r_{i+\Delta t_w}^{c'} r_i^{c''} & \Delta t_w < 0 \end{cases}$$

with $\Delta t_w = t'_w - t''_w$.

When $r_t^c r_t^c$ is again replaced by r_t^c, then this has only an effect on the diagonal elements $h_{t_w,t_w}^{c,c} = \mathbf{r}^{c\top}\mathbf{r}^c$ which have to be replaced by

$$h_{t_w,t_w}^{c,c} = \sum_{t_r} r_{t_r}^c$$

Note, however, that this has no effect, if r_t^c is binary.

[3] www.mat.univie.ac.at/~neum/software/minq/

2.4 Spike Detection

For a recording of 1 minute length which has been sampled at 30kHz we have to consider 1.8 million spike times for each cell ($r_t^c, t \in \{1, \ldots, 1.8e+6\}$). No ordinary QP-solver can handle such big problems. Usually, however, the recordings exhibit long stretches containing only noise and no foreground spikes. These parts of the signal are uninteresting for us, because we know that there are no spikes at the corresponding spike times, that is $r_t^c = 0$ for all t responsible for these parts. Therefore we want to skip stretches of pure noise and only consider patches which contain some activity.

This aim is very similar to spike detection done in combination with clustering approaches to spike sorting. The difference being that we do not have to choose a fixed-size window which is exactly positioned relative to the spike, but rather we only have to make sure that the window contains all spikes which might have an influence on each other.

The detection of spikes is usually reduced to the detection of time points at which the recorded potential crosses a certain threshold (see e.g. Sahani, 1999; Lewicki, 1998). An obvious candidate for the threshold is a multiple of the noise standard deviation, σ. Let us for the moment assume that we know it. Quiroga et al. (2004) found that a threshold of 4σ works well in practice. We use the same threshold on the absolute value of the recorded waveform.

To ensure that all spikes which overlap with each other fall into one window we require that all threshold crossings which lie within the length of a single spike must lie in the same window. The length of a single spike, T_w, is a free parameter of our model, but its choice is not so critical as long as it is large enough. We set it to something which is equivalent to 3ms or 2.67ms in accordance with the data that we got (see section 3.2).

When we found the set of threshold crossings which are close to each other in the sense just defined, we select the window containing them by choosing all time points starting at t_s time points before the first threshold crossing and ending at t_e time points after the last threshold crossing where $t_s = \frac{1}{3}T_w + 5$ and $t_e = \frac{2}{3}T_w + 3$. A single spike, for example, crosses the threshold with a section from its peak, about the same number of time points left and right of the peak; taken together, say, n_t threshold crossings. Our definition of the window then implies that the peak is at about 1/3 of the length of the spike and that we consider $5 + 3 + n_t$ spike times as possible times for the spike to have occurred. The numbers 5 and 3 have been arbitrarily chosen so as to increase the tolerance to noise changing the threshold crossings. They are not higher, because increasing the size of the window also increases the time until the QP-solvers find the optimum. Below we will denote all the windows

selected in that way as regions of interest.

We still need an estimate of the noise standard deviation. If we assume that spikes are relatively rare events, calculating the standard deviation of the complete signal might be a viable approach. Quiroga et al. (2004), however, show that with increasing spike rate the complete signal standard deviation increasingly over-estimates that of the noise. They propose to use $\sigma = \text{median}(|x|/0.6745)$, but we resort to an iterative approach. We calculate the standard deviation of the complete signal and use that to find a first set of regions of interest which then get excluded from the data points used to calculate a new estimate of the standard deviation. This is repeated until the estimate does not change anymore.

2.5 Constraining Solutions by Choice of Priors

We have seen in section 2.2.3 that the homogeneous Poisson process prior leads to a term in our objective function which facilitates sparse results. In this section we discuss changes to the terms contributed by the prior which might improve results for the convex problem.

2.5.1 Quadratic Sparsity

One problem is that small values are disproportionately preferred to large values. We can change the sparsity term from the homogeneous Poisson prior which was

$$a \sum_{c,t} s_t^c$$

to the quadratic term

$$a \sum_{c,t} s_t^c (1 - s_t^c) \tag{40}$$

which prefers both, values near 0 and values near 1, but penalises everything in between.

Amending the objective function with the new sparsity term hardly changes the implementation. In the QP-formulation only a diagonal matrix has to be subtracted from $\mathbf{H}$ to account for the additional quadratic term $-as^2$. So we are simply reducing the penalty, that we introduced by filling in the diagonal of $\mathbf{H}$, by a fixed amount and may even turn it into a reward.

It comes to mind that we loose convexity when we do this, but this depends on the properties of $\mathbf{H}$ and the size of a. The matrix that has to

be subtracted is $\mathbf{A} = a\mathbf{we}$. $\mathbf{H}$ can be decomposed into its eigenvalues and -vectors as $\mathbf{H} = \mathbf{V}\mathbf{D}\mathbf{V}^\top$. Then

$$\frac{1}{2\sigma^2}\mathbf{H} - \mathbf{A} = \frac{1}{2\sigma^2}\mathbf{V}\mathbf{D}\mathbf{V}^\top - a\mathbf{we} = \frac{1}{2\sigma^2}\mathbf{V}\mathbf{D}\mathbf{V}^\top - a\mathbf{V}\mathbf{V}^\top = \mathbf{V}(\mathbf{D} - 2\sigma^2 a\mathbf{we})\mathbf{V}^\top$$

Since $\mathbf{H}$ is positive semi-definite its eigenvalues in $\mathbf{D}$ are either zero or positive. Hence, for $\frac{1}{2\sigma^2}\mathbf{H} - \mathbf{A}$ to be positivesemi-definitee, that is for our problem to be convex, the smallest eigenvalue of $\mathbf{H}$ has to be greater than $2\sigma^2 a$. Unfortunately, $\mathbf{H}$ is usually not full rank, that means some eigenvalues are zero. This is the case especially for larger regions of interest which contain more than one spike. For them, any non-zero a makes the problem non-convex again.

We could interpret a as a free parameter. Then we can choose a as a trade-off between getting solutions that are more like the ones that we expect and running into local optima. We find that results of the algorithm with $a \in [2, 6]$ may be marginally better than results found with the original sparsity term ($a \approx 7$), but this depends on the data set at hand which is an indicator for problems with local minima. We also tried an annealing approach in which we had consecutive runs of the algorithm with increasing a which were seeded with the result from the previous run. The results were not better than using a small a only, probably because the problems with local minima become too severe for larger a.

Because the results are very similar for original and quadratic sparsity (with small a) we only report results for the original sparsity which can be interpreted as a homogeneous Poisson process prior. Note, however, that these experiments have been done before we lately noticed that `MINQ` does not always find the global optimum of a convex problem and we did not have the time to rerun them with `quadprog`.

2.5.2 Refractory Period

Neurons have an absolute refractory period. This means that the cells that we are considering do never spike in close succession where close succession is 1ms for most neurons. This criterion constrains the solutions that we expect to get and thereby reduces the space of possible spike times. So it should help us finding the correct spike times.

Again, we can extend the sparsity term that we got from our prior to facilitate a refractory period by penalising spikes that occur in the refractory period of another spike. A suitable formula is (neglecting a for simplicity)

$$\sum_{c,t} s_t^c \left[1 + A \sum_{\tau=-\mathcal{T},\tau\neq 0}^{\mathcal{T}} s_{t-\tau}^c\right] = \sum_{c,t} s_t^c + \sum_{c,t} \sum_{\tau=-\mathcal{T},\tau\neq 0}^{\mathcal{T}} A s_t^c s_{t-\tau}^c \tag{41}$$

where A determines the amount of penalty and $\mathcal{T}$ the size of the refractory period; both are free parameters which have to be chosen, but if we want an absolute refractory period we should set $A = \infty$.

Again, we have another quadratic term in the objective function which only needs to be added to the existing one. This time, however, the matrix that we have to add to $\mathbf{H}$ in a problem with only one cell has the form

$$\mathbf{A}^{1,1} = \begin{bmatrix} 0 & A & A & A & 0 & 0 & 0 \\ A & 0 & A & A & A & 0 & 0 \\ A & A & 0 & A & A & A & 0 \\ A & A & A & 0 & A & A & A \\ 0 & A & A & A & 0 & A & A \\ 0 & 0 & A & A & A & 0 & A \\ 0 & 0 & 0 & A & A & A & 0 \end{bmatrix}.$$

As shown in appendix A every matrix with only zeros in the diagonal is not convex. Also, the sum of a convex and a non-convex matrix is not convex. Therefore we get again caught in local minima. We could choose A very small to be as near to convex as possible, but we found that in these cases adding the refractory period has only limited effect. In all cases we did not find improved results by adding the refractory period.

The question arises whether there is a different formulation that facilitates a refractory period in our quadratic problem without making the problem non-convex. But the formulation of a refractory period, first, is intrinsically quadratic, because it is about the interaction between spikes and, second, should not restrict putting a spike independent of other spikes in the same way it restricts that for the interaction between two spikes. Hence we expect every formulation of a refractory period to have a Hessian matrix with a diagonal which makes it non-convex.

3 Results

3.1 Quasi-Deterministic Posteriors

In our algorithm we make the assumption that the posterior probability of spike times (eq. (14)) is deterministic which means that there is one setting of spike times which is certain and all others are impossible. As we are working in a noisy setting this assumption seems not justified, but we find that the idea is supported by our test data.

We took Quiroga's fourth data set with $\sigma = 0.1$ and extracted all spikes. The spike shapes in this data set are very similar, additionally we aligned

them such that the largest peak is at the same position for each cell. The result can be seen in figure 2.

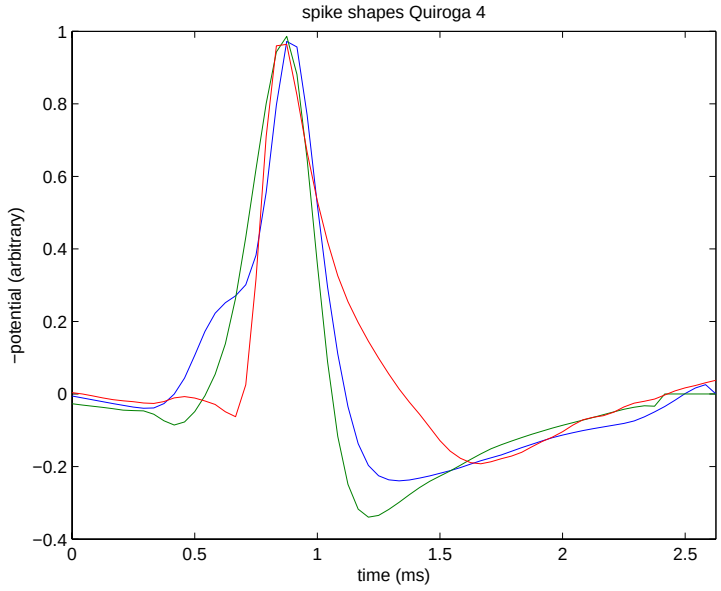

Figure 2: Aligned spike shapes from Quiroga shape set 4.

We restricted the calculation of the posterior to a window of 2 spike times before and after the actual spike which resulted in $2^{5\cdot 3} = 32768$ possible settings for the spike times (3 is the number of cells). This restriction is unproblematic for isolated spikes, because we know from the shape of the spikes that only spikes within the small window are able to explain the data. For overlapping spikes which fall in this window the same reasoning applies, but for overlapping spikes which are further apart the posterior can not be reliably computed on this restricted set of spike times. As a consequence, we do not have results on those overlapping spikes.

We find that the posterior of 82.2% (2136/2600) of isolated spikes have a single peak which is greater than 0.95, 16.9% (440/2600) have two peaks which are summed together greater than 0.95 and 0.9% (24/2600) have three peaks which have together 0.95 probability. For the closely overlapping spikes[4] we got 57.1% (8/14) with a single peak, 28.6% (4/14) with two peaks and 14.3% (2/14) with three peaks.

Here we used the original spike shapes which, of course, fit the data nicely, but does the situation change when the spike shapes are very different

[4]two spikes count as closely overlapping when they are at most two time steps apart

from those in the data? No. We tested with the spike shapes that we use to initialise our algorithm on data from Quiroga (see figure **??**). 69.7% of isolated spikes have a single dominant peak in the sense from above, 26% have two, 3.8% have three and 0.5% have four peaks. For closely overlapping spikes it is 1–50%, 2–35.7% and 3–14.3%.

Because the spike shapes are very similar in this data set, we expected that if there is a deviation from the hypothesis, then we find it here. With more different spike shapes it should be even clearer which combination of them is responsible for the data and therefore the posterior should become closer to deterministic. On the other hand, if the noise level is increased, the posterior should become less certain. Using Quiroga's fourth data set with $\sigma = 0.2$, we find an increase of uncertainty (isolated spikes: 1–25.2%,2–51.6%,3–15.2%,4–6.6%,5–1%,6–0.2%,7–0.1% overlapping: 1–10.7%,2–39.3%,3–25%,4–17.9%,5–7.1%)[5], but it is still only a very few number of spike time settings which dominate the posterior.

Although the posterior probability of spike times is not exactly deterministic, this result shows that it is very close to deterministic. There are only very few cases in which two settings of spike times have nearly equal probabilities. In these cases the assumption of determinism throws away the information about the uncertainty. Above all, however, the here presented results signify the extremity of the space in which we have to find the optimum. It does not really matter whether there are one, two, or three peaks in an otherwise flat space of 2^{many} values, it will always be hard to find them.

3.2 Synthetic Test Data

It is extremely hard to evaluate the result of a spike sorting algorithm on real extracellular recordings of neurons, because in this case we do not know the correct solution. There is the possibility to record simultaneously extracellular and intracellular from single cells to have some controlling information about the activity of single cells (see e.g. Harris et al., 2000; Wehr et al., 1999), but such experiments are extremely hard to conduct and consequently appropriate data sets are very rare.

Alternatively, we can generate test data ourselves which captures the most important properties of the signal that we want to model. The obvious advantage is that we know exactly the setting of parameters which generated the data, for example the spike times, and we can freely manipulate them. On the other hand, it may also be that the generated data misses an important

[5]Because this was a different data set, the total number of spikes changed: 2573 isolated spikes, 28 closely overlapping.

property of the real data, like correlations in the noise, which would influence the performance of a spike sorting algorithm.

Next we present how we generate our own synthetic data. These have a very simple noise model matching the choice in our model. Then we introduce another, freely available data set which has been produced with more care to fit real data more closely.

3.2.1 Inspired by a Data Set from Auditory Cortex

We have access to a data set recorded from the auditory cortex. With our synthetic data we aimed to reproduce properties of that data set, especially typical spike shapes and noise level.

The spike shapes extracellularly recorded from auditory cortex usually have a narrow, negative peak which ascends to the positive and is followed by a wider negative bow. As a convention, however, the signs are usually flipped such that we have a positive first peak in accordance with the potential within the cell which constitute the usual shape of an action potential. We are reproducing this particular shape with the product of a cosine and an exponential function in logarithmic time

$$w(t) = A\cos(t_l)\exp(-t_l/\omega)$$

where A sets the height of the first peak, ω the height of the second in relation to A, $t_l \propto \log(t\tau)$ and $t_l \in [-\pi/2, 5\pi/2]$ and τ is a time constant. Furthermore we are putting $t_l = 0$ at a third of the total length of the spike shape and $t_l = 5\pi/2$ at the end of the spike shape. The resulting shapes are very similar to the ones in the real data, although they obviously can not reproduce all peculiarities due to their simplified nature. A comparison between a single spike from real data and a synthetic one that closely matches the real shape can be seen in figure 3. We are producing random spike shapes by randomly choosing the parameters with $A \in [0.06, 0.11]$, $\omega \in [5, 15]$ and $\tau \in [0.1, 0.3]$.

We generate spike times from a discrete, homogeneous Poisson process with mean firing rates of 24Hz, 30Hz and 39Hz at a sampling rate of 30kHz. Subsequently we enforce an absolute refractory period of 1ms by shifting all spikes which occur within a refractory period to the end of the refractory period. The spike times are stored in ρ_t^c which is 0 unless cell c spikes at time t then $\rho_t^c = 1$.

The full synthetic recording is made by convolving $\boldsymbol{\rho}^c$ and $\mathbf{w}^c$, summing the result over cells c and adding uncorrelated Gaussian noise with standard deviation $\sigma = 0.008$.

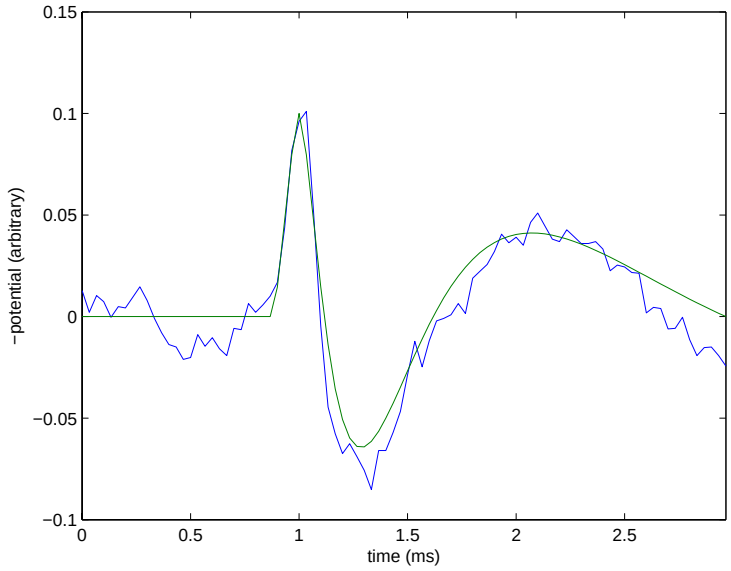

Figure 3: A real spike shape found in a data set from auditory cortex (blue) compared to a synthetic one (green, smooth) with parameters $A = 0.1$, $\omega = 7$ and $\tau = 0.18$. Although they do not completely match, the main characteristics are captured.

There is an additional source of spike variability which is due to sampling, because the actual time of spiking is very likely to fall between two samples. This means that the spike shapes are "microshifted" between samples by a random amount. In spike clustering this leads to problems in alignment which in turn lead to additional cluster variance (Lewicki, 1998; Sahani, 1999). We want to test the effect of microshifts on our algorithms and therefore also generate data with microshifts by upsampling existing data to $30 \cdot 32 = 960$kHz, introducing random shifts between 0 and 32 time steps on the spike times, convolving those with the interpolated spike shapes, downsampling to 30kHz and finally adding the noise.

3.2.2 Data Set from Quiroga

The background noise in extracellular recordings to the greatest part[6] consists of superimposed spikes from many background neurons, none of which can be identified individually. Given that those spikes have a certain shape as well, it is very unlikely that the resulting noise is uncorrelated. And indeed, Rutishauser et al. (2006) report that the autocorrelation of the background noise for their data set is significantly different from 0 until about 1.2ms and we observe similar autocorrelations in the data set from auditory cortex.

To reproduce the statistics of the noise most faithfully Quiroga et al. (2004) build their test data from "a database of 594 different average spike shapes compiled from recordings in the neocortex and basal ganglia". They generate background noise by superimposing randomly selected shapes from the database at random times and with random amplitudes. Then they add a train of three distinct spike shapes (see figure 4) from the database at times determined by a Poisson process with mean firing rate of 20Hz and 2ms refractory period. Noise and spike trains are scaled such that the noise standard deviation is one of $\{0.05, 0.1, 0.15, 0.2\}$ and the peak of the spikes is at 1. All of this is done at a sampling rate of 96kHz and the result is downsampled to 24kHz, thereby introducing random microshifts, too.

Rodrigo Quian Quiroga very kindly published all of the data sets from his paper on his website. We transformed them in our data format and use them to compare the performance of several spike sorting algorithms where performance is defined in the following section.

3.3 Measures of Performance

There are three main kinds of errors that a spike sorting algorithm can make. It can miss a spike completely, assign a spike to the wrong cell, or assign a

[6] after bandpass filtering

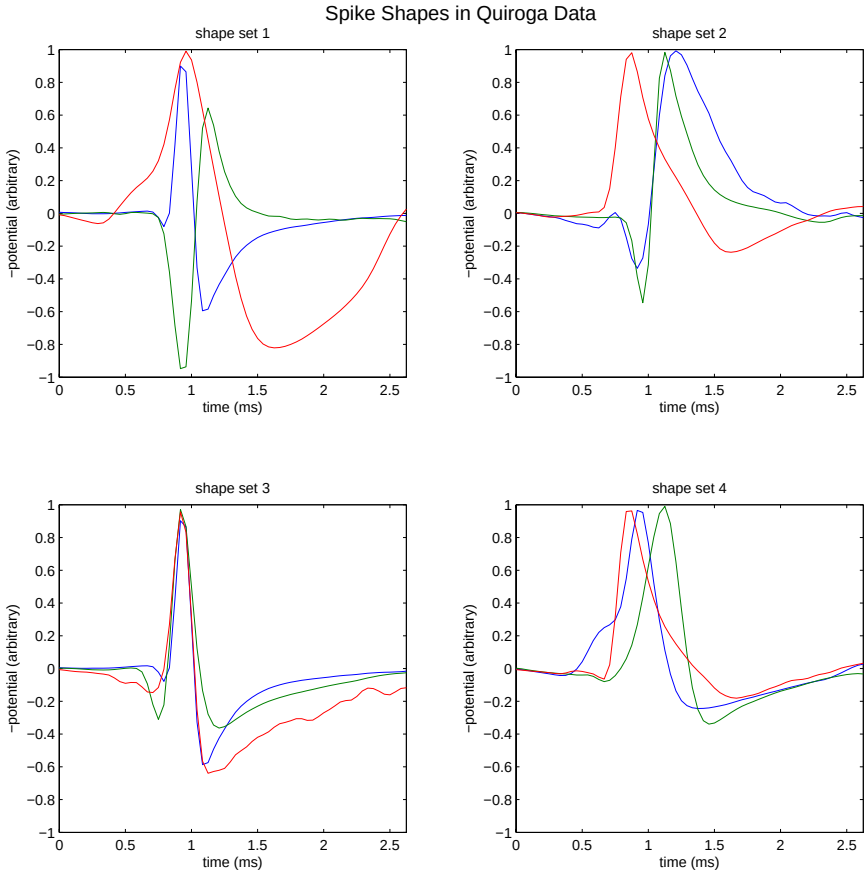

Figure 4: The 4 different sets of spike shapes that Quiroga used in his data.

spike to noise. We quantify these errors with recall, P_r, and precision P_p which are defined as

$$P_r = \frac{TP}{TP + FN} \qquad P_p = \frac{TP}{TP + FP} \tag{42}$$

where TP is the number of true positives, that is the number of spike times correctly assigned to 1 ($r_t^c = 1$ and $\rho_t^c = 1$), FN is the number of false negatives, that is the number of spike times falsely assigned to 0 ($r_t^c = 0$ and $\rho_t^c = 1$), and FP is the number of false positives, that is the number of spike times falsely assigned to 1 ($r_t^c = 1$ and $\rho_t^c = 0$). Then $TP + FN$ is the real total number of spikes in the data and $TP + FP$ is the total number of spikes reported by spike sorting. Thus, recall is a measure for errors by not finding spikes and precision is a measure for errors by assigning spikes where there are none. Note, however, that we do not make an explicit difference between not finding a spike and assigning it to the wrong cell. But when a spike is assigned to a wrong cell, this will reduce recall and precision, while not finding a spike will only reduce recall.

Because precision and recall are directly defined on the spike times which have a sub-millisecond resolution (e.g. 0.04ms at 24kHz), these measures are more sensitive to small shifts of spikes than we need. For example, when an algorithm finds a spike at t, but it actually is generated at $t+1$, this will add a false negative and a false positive without acknowledging the very close match at all. Therefore, we also allow for shifts of spike times up to a given number of samples s. As long as a spike of the correct cell is found within s shifts from the original spike time, it is counted as true positive. If there are two spikes of the correct cell within s shifts of the true spike, then one true and one false positive is counted.

In clustering the spike shapes are carefully aligned within a fixed window according to a certain feature such as the centre of mass of the most prominent peak. We do not have that. The presented algorithms are free to shift the spike shapes within their container which is only defined by shape length. Consequently, the spike shapes resulting from our algorithms may be shifted with respect to the spike shapes used to generate the data (the peaks are at different positions). Such a shift also means that the spike times are shifted with respect to the original spike times, but into the opposite direction, which is a consequence of the convolution of spike times and shapes in our model. We are accounting for these effects and establish cell identities between original data and algorithm outputs by searching for the best match between a shifted spike shape from the algorithm output and an original spike shape. The shift and cell identity of the best match is recorded and the spike times are adapted accordingly.

3.4 Influence of Data Parameters

We want to test the influence of firing rate, microshifts, spike shapes and number of cells on the performance of an algorithm which models spike time explicitly. We do this examplarily for the convex E-step with thresholding.

The data that we use has the following properties. It is based on three sets of spike shapes which are depicted in figure 5. Every set consists of 4 randomly generated shapes. For each set we produced spike trains with a length of about 1.66s (50,000 samples) and firing rates of first 24Hz, then 30Hz and then 39Hz. To obtain data sets with less than 4 cells we chose every permutation of 3, 2 and 1 cells and extracted the corresponding spike shapes and times from the data with 4 cells. This means that we have $3 \cdot 3$ data sets with 4 cells, $3 \cdot 4 \cdot 3$ with 3, $3 \cdot 6 \cdot 3$ with 2 and $3 \cdot 4 \cdot 3$ with 1, which makes 135 in total. Then we take these 135 data sets and make another 135 by adding microshifts.

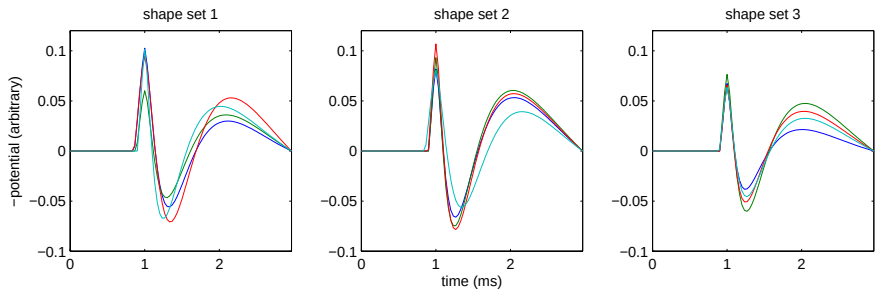

Figure 5: Spike shapes used for testing dependence of data parameters on performance.

The algorithm was initialised with a set of spike shapes which was generated in the same way as the spike shapes in the data, but with fixed parameters such that the initial shapes are quite different from each other. Then the algorithm was run until the stopping criterion was met.

The performance averaged over all 270 data sets is $P_r = 0.48 \pm 0.28$ and $P_p = 0.53 \pm 0.29$, where we introduce the convention: mean±standard deviation. For an allowable shift of $s = 1$ the numbers are $P_r = 0.65 \pm 0.27$ and $P_p = 0.72 \pm 0.27$. Performance does not increase considerably for larger values of s as figure 6 shows. In the following we will only report performance with $s = 1$.

The average performance for the data without microshifts is $[P_r, P_p] = [0.67 \pm 0.26, 0.74 \pm 0.27]$ while it is $[0.64 \pm 0.28, 0.71 \pm 0.27]$ for the data with microshifts. So there is only a small effect of the additional spike shape variability on the performance of our algorithm. There is also no big effect

of changing the firing rate (24Hz: $[0.66 \pm 0.26, 0.74 \pm 0.27]$, 30Hz: $[0.67 \pm 0.26, 0.74 \pm 0.28]$, 39Hz: $[0.67 \pm 0.26, 0.73 \pm 0.26]$), but data sets with spikes from shape set 1 in figure 5 are apparently easier to sort while those of set 3 are difficult (set 1: $[0.76 \pm 0.2, 0.87 \pm 0.19]$, set 2: $[0.75 \pm 0.22, 0.78 \pm 0.22]$, set 3: $[0.49 \pm 0.27, 0.55 \pm 0.28]$). It is surprising that shape set 2 has a considerably larger performance than shape set 3, because on first sight set 2 seems to have shapes which are more similar. The only explanations that we can think of at the moment are that this is either because of the low amplitudes of the shapes in set 3, or it is because the narrow first peak is the deciding factor which is more similar in set 3.

The number of cells has a considerable influence on performance. With one cell the values are $[0.87 \pm 0.27, 0.92 \pm 0.28]$ while with 4 cells the performance is much worse $[0.45 \pm 0.09, 0.53 \pm 0.11]$ (also see figure 7). Of course we had hoped that the performance is somewhat stable for an increase of cells, but this result is also expected, because the spike sorting problem becomes increasingly difficult with increasing number of cells. Especially, it becomes more and more difficult to differentiate between spike shapes when they are as similar as, for example, in shape sets 2 and 3 of figure 5.

3.4.1 Spike Shapes and Initial Spike Shapes

We wanted to know whether performance depends on the properties of the spike shapes in the data. To test this we made a single spike train and produced several data sets based on it by convolving the spike train with different shapes. The shapes are much simpler than in the study above and consist of a positive phase of a sine and a negative phase of a sine. The width of the first phase varies from narrow to wide as depicted in figure 8 in 9 steps and thereby also determines the width of the second phase. Additionally, we independently varied the amplitude of the second phase in 5 steps. With a medium setting of the width (31 samples) and amplitude (1) we get a complete sine waveform.

We ran the algorithm on all 45 data sets with the same initial shape as for the test data above. Similar shapes like this are shown in figure 5, compared to shapes in the data, however, it has a very small amplitude (see figure 8).

The results are shown in figure 9. In all cases the recall is equal or close to 1. So usually (nearly) all spikes are found. Precision, on the other hand, is highly dependent on the width of the first phase. With increasing width precision decreases dramatically. Because recall stays high, this means that the algorithm finds more spikes than there are in the data.

The shapes with narrow positive and wide negative phase are most similar to the initial shape. In these cases the algorithm recovers the shapes in the

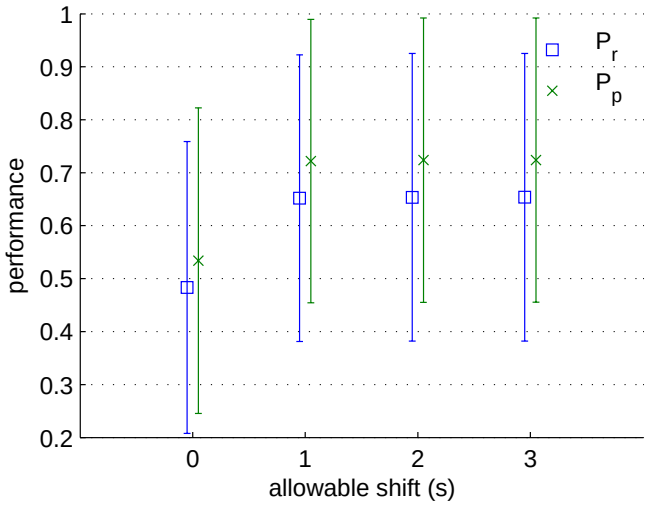

Figure 6: Average performance on generated test data dependent on allowable shift s. Errorbars indicate standard deviation. Increasing s above 1 has no effect for this data set.

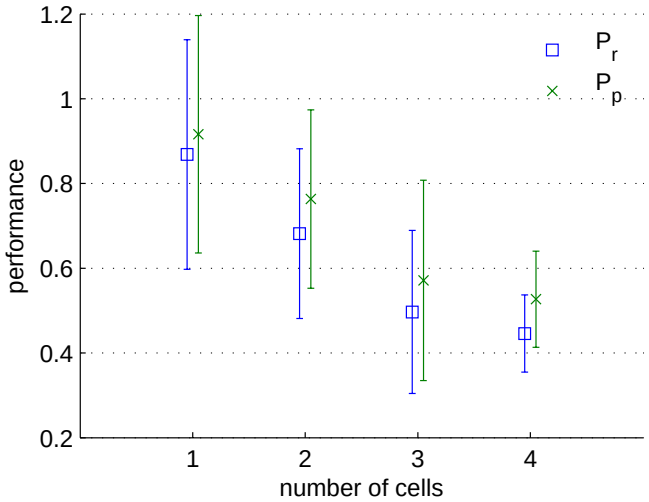

Figure 7: Average performance on generated test data dependent on the number of cells. Errorbars indicate standard deviation. Increasing the number of cells increases the difficulty of spike sorting and decreases performance.

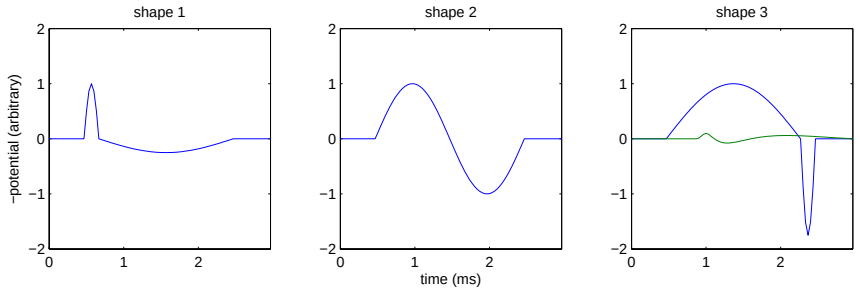

Figure 8: Samples of different spike shapes used to test the influence of shape on algorithm performance. shape 1: has sharpest first peak and lowest second peak amplitude, shape 2: medium width and medium amplitude, a symmetric shape (sine), shape 3: widest first peak and highest amplitude of second peak, green graph in panel of shape 3: shape used to initialise the algorithm in all cases.

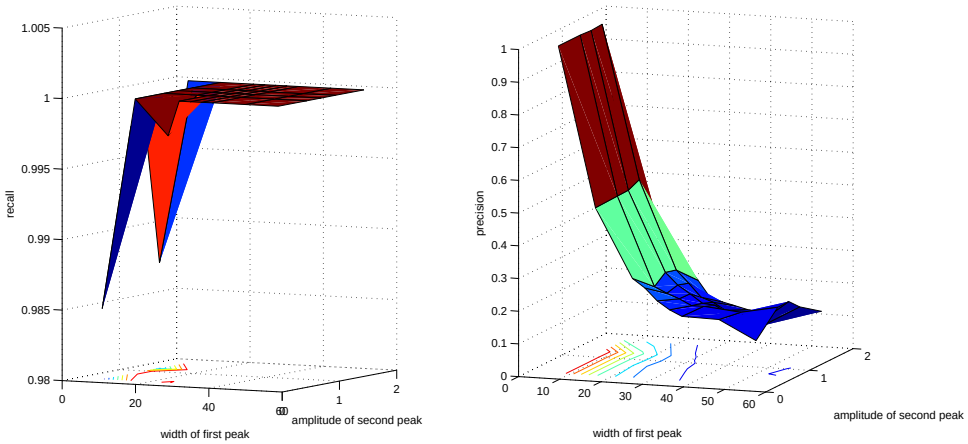

Figure 9: Performance when shapes in data and initial shapes become more different.

data very well, but with increasing width the algorithm never finds the right shape and converges to one which has a low overall amplitude. When shapes have low amplitudes, several spike times have to be set to 1 to fit the spike in the data which results in a low precision. By using an initial shape which is more similar to the ones in the data, the very good performance for narrow positive phases can be reproduced independent of the widths and amplitudes of the phases in the data. We conclude that not the properties of the spike shapes themselves are very imported, but good initialisation is.

3.5 Comparison of Algorithms

The performances of section 3.4 are not as good as expected, but how do performances compare between our algorithms and others in an equal setting?

We have presented three variations of one EM-based algorithm. They differ in the E-step which can be a genuine variational approximation (short: variational, section 2.2.1), a variational approximation assuming a deterministic posterior distribution (short: non-convex, section 2.2.2) and a convex optimisation based on the latter (short: convex, section 2.2.3). The convex optimisation is enhanced by thresholding. What happens without thresholding? Then the optimisation results, r_t^c, are free to take on values anywhere between 0 and 1 and those are then taken to estimate spike shapes. Having continuous r_t^c adds a lot more degrees of freedom to our algorithm and these are used to fit the noise completely. The result of the run from the left panel of figure 1 for one exemplary spike is shown in figure 10. The shapes are random spikes used to fit the noise by an intricate setting of spike times.

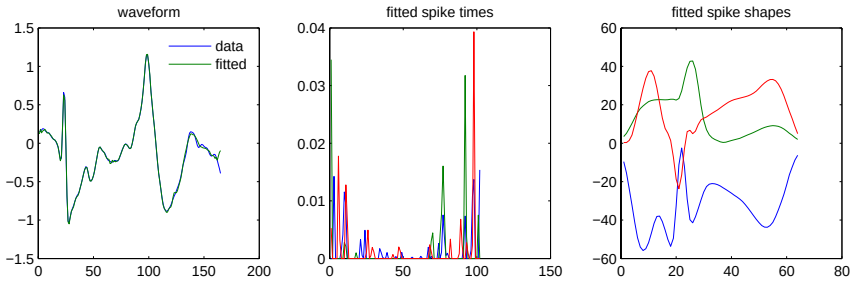

Figure 10: Convex E-step without thresholding fits the noise. left: data in blue and fit in green, middle: spike times, right: spike shapes, the data is from Quiroga shape set 3 with noise level $\sigma = 0.2$, the original spike shapes are shown in figure 4 panel 3

We also have implemented a simple matched filter algorithm for the E-

step. It calculates the similarity between the data and each spike shape as

$$g_{t_r}^c = \sum_{i=0}^{T_w} \frac{w_i^c}{\|\mathbf{w}^c\|} v_{t_r+i}$$

This is very similar to equation 38 in the reformulation of the quadratic problem (section 2.3). $\mathbf{G}$ then is normalised such that the maximum is at 1 and we select the maximum of each column in each row and put it into $\mathbf{f}$ so that $f_t = \max_c(g_t^c)$. We then find regions of interest in $\mathbf{f}$ by using the algorithm defined in 2.4 with $\sigma = 3/40$ and $T_w = 20$ samples. Within each region of interest we find the maximum in $\mathbf{G}$ and note its time and cell identity to set a corresponding spike time to 1. The ability of this algorithm to handle overlaps is limited by T_w. The smaller T_w, the more overlaps can be handled, but a smaller T_w also means that the matched filter is less tolerant to different spike shapes and may produce more false positives. Hence, other settings of the parameters might be better, but we did not have the time to explore them.

To compare these four E-steps we run the algorithm on the data sets of Quiroga et al. (2004) introduced in section 3.2.2. The result can be seen in figure 11. Note that the performances that we report here are with an allowable shift of $s = 2$, because this gave a bit better results. Variational and non-convex E-steps produce very similar but bad results. They have low recall (0.49 ± 0.16 and 0.48 ± 0.14, averaged over all data sets and noise levels), but much worse precision (0.16 ± 0.08 and 0.15 ± 0.08). The situation is similar to that of section 3.4.1 and points to problems with local minima. The matched filter is surprisingly good, but has more problems with very similar spike shapes like in shape sets 3 and 4 and achieves an average performance of $[0.48 \pm 0.3, 0.49 \pm 0.36]$. The convex E-step gives best results with $[0.62 \pm 0.26, 0.63 \pm 0.21]$ which means that on average it gets about 2 out of 3 spikes right, but it can be seen in the figure that the algorithm is better than this with less noise and worse with more noise.

All in all, the performances are not very good, but how do they relate to clustering algorithms? In figure 12 we compare performances of our algorithm with convex E-step to two freely available clustering algorithms: KlustaKwik (Harris et al., 2000) and Wave_Clus (Quiroga et al., 2004). It has to be acknowledged that clustering algorithms alone can not handle the task that we are solving. Their performance considerably depends on the spike detection that has to be done in a first step. We are using the spike detection that comes with Wave_Clus and thus the performance must be seen in the context of the combination of that spike detection and clustering. Additionally,

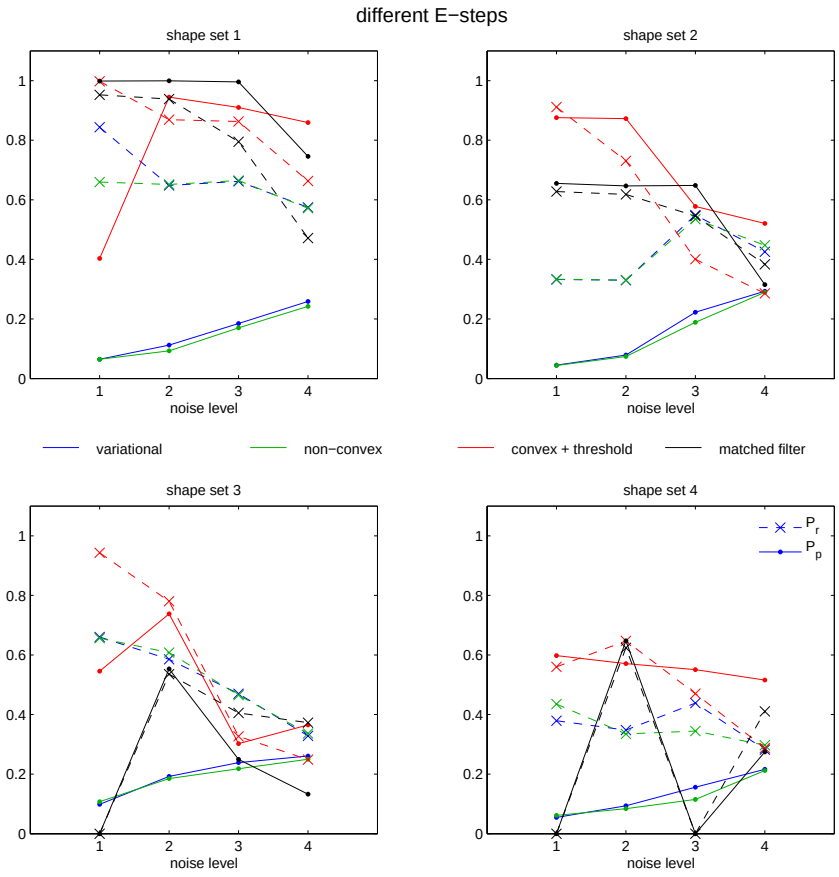

Figure 11: Comparing on data from Quiroga: variational, non-convex, convex with thresholding and matched filter E-steps. The noise levels correspond to $\sigma = 0.05$, $\sigma = 0.1$, $\sigma = 0.15$ and $\sigma = 0.2$.

clustering depends on the features that it is given. Instead of the original waveforms we give both clustering algorithms only the first three principal component scores. Note that Quiroga et al. (2004) report better performance with wavelet features, but these are not available to us. KlustaKwik fits a mixture of Gaussians and Wave_Clus implements superparamagnetic clustering.

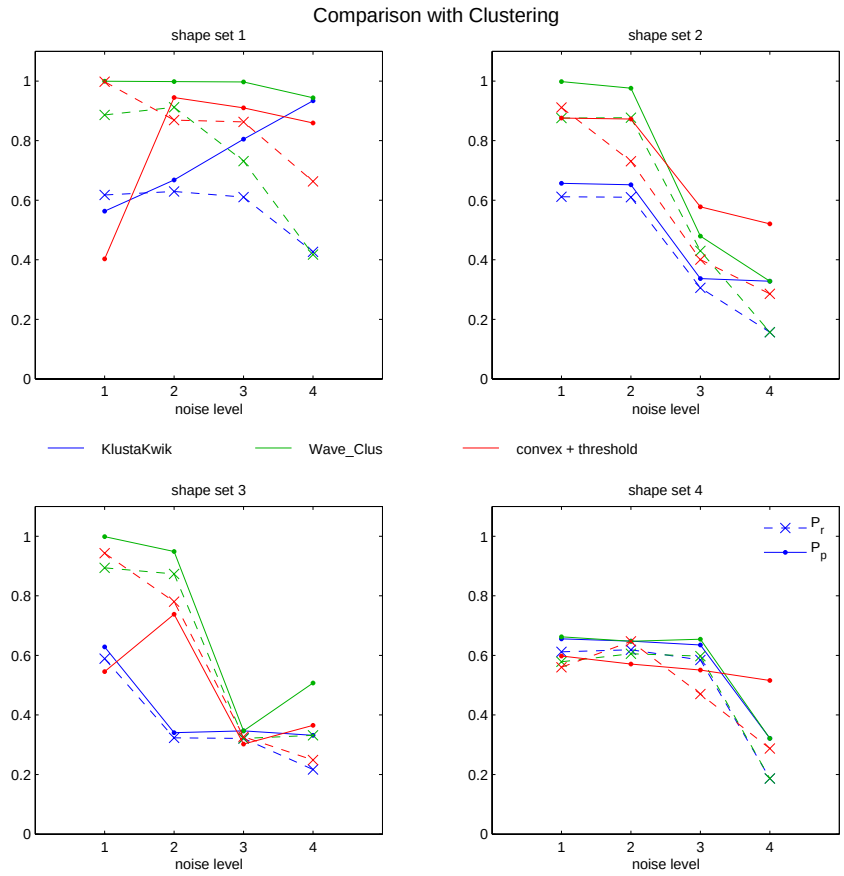

Figure 12: Comparison with clustering methods: convex problem with thresholding, KlustaKwik, Quiroga's Wave_Clus. The noise levels correspond to $\sigma = 0.05$, $\sigma = 0.1$, $\sigma = 0.15$ and $\sigma = 0.2$.

The average performance for KlustaKwik is $[0.46 \pm 0.18, 0.55 \pm 0.19]$, for Wave_Clus it is $[0.60 \pm 0.27, 0.74 \pm 0.27]$ and for convex with thresholding it is $[0.62 \pm 0.26, 0.63 \pm 0.21]$. Wave_Clus often has better recall and especially better precision than our algorithm, but it is not better in all cases.

3.5.1 Overlapping Spikes

Our algorithm is specially designed to cope with overlapping spikes, but does it do so? We calculate performance for spikes which occur at most 1.5ms apart[7]. The result is depicted in figure 13. For our algorithm with convex E-step the graphs barely change. This is also reflected in the average performance $[0.63 \pm 0.25, 0.64 \pm 0.20]$ which is approximately equal to the performance on all spikes. Therefore, the algorithm recognises overlapping spikes as well as single spikes. The main effect for KlustaKwik $[0.23 \pm 0.07, 0.50 \pm 0.16]$, Wave_Clus $[0.19 \pm 0.08, 0.66 \pm 0.24]$ and matched filter $[0.32 \pm 0.19, 0.55 \pm 0.34]$ is a drop in recall which was expected.

3.5.2 Initialisation with Clustering Result

Because clustering is quite good in extracting shapes from non-overlapping spikes in the data and is bad in telling overlapping spikes apart and our algorithm sometimes has a problem extracting the right shapes, it is an obvious idea to combine the two algorithms. We present results of such an experiment in figure 14. In the experiment we seeded our algorithm in the different variations with the result found by Wave_Clus. The results are again mixed. Although there is often a considerable improvement compared to the Wave_Clus performance, the performance on data sets with shape set 3 is actually degraded.

On the other hand it is very interesting that now the variational E-step is best with an average improvement in performance of $[0.17 \pm 0.18, -0.06 \pm 0.22]$, followed by the convex with subsequent non-convex E-step $[0.12 \pm 0.20, -0.05 \pm 0.19]$. Apparently, initialisation with a good estimate of spike shapes and times helps these algorithms to overcome local optima and to produce reasonable results. Finally even the matched filter $[0.08 \pm 0.16, -0.03 \pm 0.27]$ is better than the algorithm which worked best when not seeded: convex with thresholding $[0.01 \pm 0.19, -0.05 \pm 0.22]$.

There are other methods, however, that can handle overlapping spikes when given a good estimate of the single spike shapes (e.g. Lewicki, 1994) and it is unclear whether our algorithm is better than that.

4 Conclusion

We have presented a generative, probabilistic model for extracellular recordings of neurons. The model naturally handles overlapping spikes and it allows

[7]The algorithm result is the same as before, but we are restricting the analysis to overlapping spikes only.

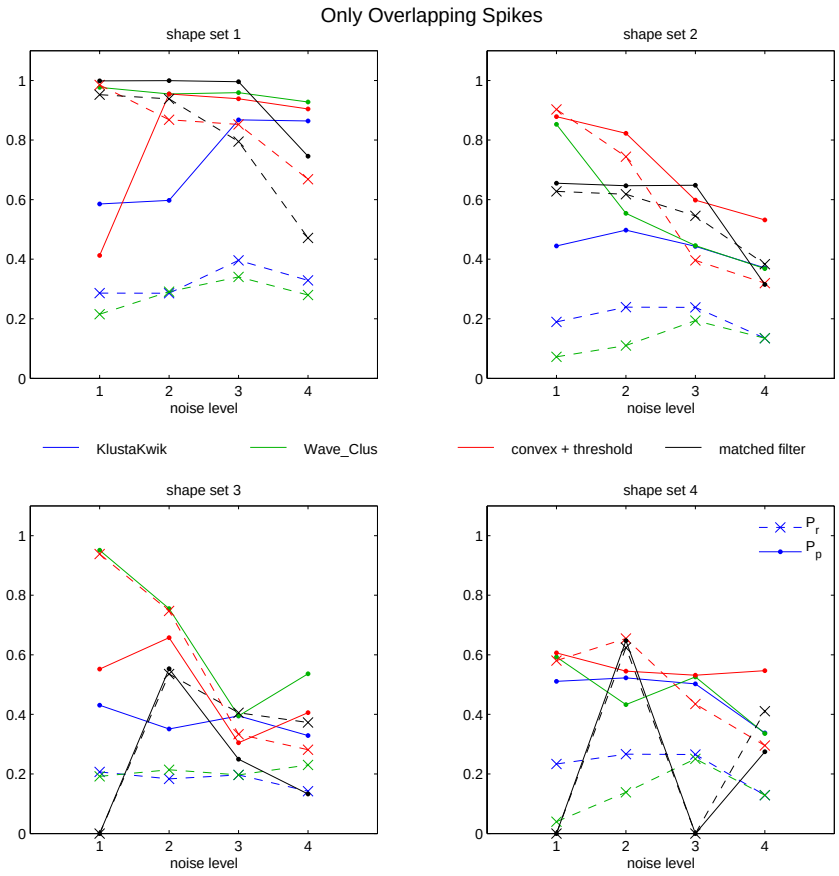

Figure 13: Performance only on overlapping spikes (at least 1.5ms) for KlustaKwik, Wave_Clus, convex and matched filter. Noise levels: $\sigma = 0.05$, $\sigma = 0.1$, $\sigma = 0.15$ and $\sigma = 0.2$.

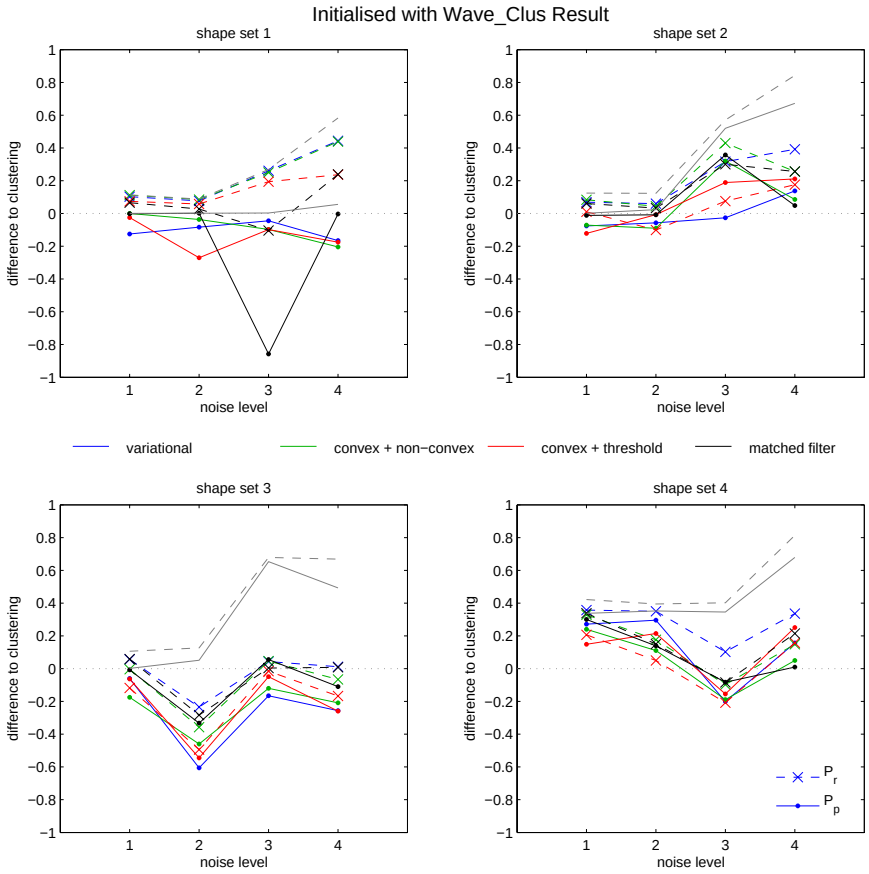

Figure 14: Using the result from Wave_Clus as initialisation. The difference in performance to the Wave_Clus result is shown for variational, convex with subsequent non-convex, convex with threshold and matched filter E-steps. Everything above 0 is an improvement, everything below degradation. The solid and dashed grey lines show the maximally possible improvement for precision and recall. Noise levels: $\sigma = 0.05$, $\sigma = 0.1$, $\sigma = 0.15$ and $\sigma = 0.2$.

us to learn spike times and spike shapes at the same time. The learning in the model is done with the expectation-maximisation algorithm, but to cope with computational difficulties we propose three approximate E-steps. While the variational approximation still is very close to the original EM algorithm (in our terms), it is hard to interpret the variables involved in our third approximation directly in terms of the expected values that we need for EM. This third approximation, however, involves a convex quadratic program (we denote it convex E-step) and therefore allows us to find a global optimum of our problem. This is a large advantage over the other two approximations which can only find local optima, because we have seen that the posterior distribution defines an optimisation space which is flat apart from very few single peaks.

Using the convex E-step alone for a couple of EM iterations leads to overfitting, because it allows that spike shapes are scaled to fit the data. This introduces a lot of additional degrees of freedom which are used to fit the noise. To counteract this effect we reduce the number of degrees of freedom by not allowing the scaling of spike shapes which is implemented with thresholding. The other approximations do that automatically, but often converge to the wrong optima as our results show.

None of the proposed algorithms, however, exhibits really good performance. An average performance of around 0.6 (convex E-step with threshold) for recall and precision on a data set which is very similar to real recordings is not very usable. This would for example mean that only a bit less than 2 out of 3 spikes are assigned to the correct cell while the random baseline is at 1/3. We have noticed two particular things that often lead to lower performance. First, recovered spike shapes are similar to the original ones, but have lower amplitudes such that the algorithm puts two, or more, of them right next to each other. Second, the algorithm recovers a shape which is actually the mean of two spike shapes and assigns spikes of both to the mean shape. The first problem might be solved by introducing a refractory period, but we have argued that we loose the convexity of our quadratic program when we do this which in turn leads to worse results. The second problem has no obvious solution except maybe for using different initial spike shapes.

Clustering algorithms do not perform a lot better, but they also can not really be compared to our algorithm, because they solve a slightly different problem. The matched filter can be compared to what we have done and it produces sometimes even better results than our more complicated E-steps. The good news, however, is that our results show that the convex E-step with thresholding automatically handles overlapping spikes which was the aim of the project in the first place.

There are several issues that we have not addressed. Most importantly,

the number of cells is a fixed, predetermined value in our algorithm, but in a real recording the number of cells in unknown. This issue could be overcome by using a cascade of runs with increasing number of cells and selecting the one which solution has the largest likelihood, but such an approach only works provided that the algorithm finds a good solution with every number of cells and this is not the case. Other extensions that could be made relatively easy are the change of the noise model from uncorrelated to correlated Gaussian and the inclusion of multiple recorded waveforms when the electrode had more than one wire (e.g. a tetrode).

Another disadvantage of the proposed E-steps is their assumption of a posterior distribution that is fully factorised. This assumption is equivalent to assuming that having a spike at a certain point in time given the data is independent of having a spike right next to it given the data, what is clearly not the case. We could, for example, try to recast our model as an hidden Markov model and include correlations between spike times into the dynamics. Then the Viterbi algorithm could be used to infer spike times in that model. In such a model it is also easier to introduce additional variability of the spike shape which is not included in our model. However, it is unclear whether this approach would scale well with increasing number of spike times considered in terms of computational expense.

The proposed algorithm is doing what it is supposed to do, but it does so not very well. It is unclear whether the exhibited performances are sufficient to give an experimenter a good view on the real situation behind an extracellular recording. Although this might be true in some cases, in others it surely is not.

A Matrices with All Zeros in the Diagonal are Non-Convex

A square matrix, $\mathbf{H}^{m\times m}$, which is convex only has eigenvalues which are equal or greater than zero.

$$\lambda_i \geq 0 \qquad i = 1, \ldots, m$$

The trace of a matrix is the sum of the values in its diagonal and the trace is equal to the sum of the matrix' eigenvalues.

$$\text{tr}(\mathbf{H}) = \sum_{i=1}^{m} h_{i,i} = \sum_{i=1}^{m} \lambda_i$$

Only the zero-matrix with $h^0_{i,j} = 0$, $i, j = 1, \ldots, m$ has all zero eigenvalues, $\lambda_i = 0$ for all i, because the product of the zero-matrix with any other matrix is equal to the zero-matrix, $\mathbf{H}^0\mathbf{M} = \mathbf{H}^0$ and hence any eigendecomposition with all zero eigenvalues is equal to the zero-matrix

$$\mathbf{V}\mathbf{H}^0\mathbf{V}^\top = \mathbf{H}^0$$

If $\mathbf{H}$ is a matrix with all zeros in the diagonal and non-zero entries off diagonal, then

$$\text{tr}(\mathbf{H}) = 0 = \sum_{i=1}^{m} \lambda_i.$$

Consequently, as $\mathbf{H} \neq \mathbf{H}^0$, $\mathbf{H}$ must have at least one negative eigenvalue. Therefore $\mathbf{H}$ is not convex.

References

I. Bar-Gad, Y. Ritov, E. Vaadia, and H. Bergman. Failure in identification of overlapping spikes from multiple neuron activity causes artificial correlations. *J Neurosci Methods*, 107(1-2):1–13, May 2001.

C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

K. D. Harris, D. A. Henze, J. Csicsvari, H. Hirase, and G. Buzski. Accuracy of tetrode spike separation as determined by simultaneous intracellular and extracellular measurements. *J Neurophysiol*, 84(1):401–414, Jul 2000.

M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37:183–233, 1999.

M. S. Lewicki. Bayesian modeling and classification of neural signals. *Neural Comput*, 6(5):1005–1030, 1994.

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

R. Q. Quiroga, Z. Nadasdy, and Y. Ben-Shaul. Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering. *Neural Comput*, 16(8):1661–1687, Aug 2004.

U. Rutishauser, E. M. Schuman, and A. N. Mamelak. Online detection and sorting of extracellularly recorded action potentials in human medial temporal lobe recordings, in vivo. *J Neurosci Methods*, 154(1-2):204–224, Jun 2006.

M. Sahani. *Latent Variable Models for Neural Data Analysis*. PhD thesis, California Institute of Technology, Pasadena, California, 1999.

S. Takahashi, Y. Anzai, and Y. Sakurai. Automatic sorting for multi-neuronal activity recorded with tetrodes in the presence of overlapping spikes. *J Neurophysiol*, 89(4):2245–2258, Apr 2003.

M. Wehr, J. S. Pezaris, and M. Sahani. Simultaneous paired intracellular and tetrode recordings for evaluating the performance of spike sorting algorithms. *Neurocomputing*, 26-27:1061–1068, 1999.