TECHNIQUE(S) FOR SPIKE - SORTING

Christophe Pouzat

23rd May 2006

Laboratoire de Physiologie Cérébrale, CNRS UMR 8118
UFR Biomédicale de l'Université René Descartes (Paris V)
45, rue des Saints Pères
75006 Paris
France
http://www.biomedicale.univ-paris5.fr/physcerv/Spike-O-Matic.html
e-mail:christophe.pouzat@univ-paris5.fr

arXiv:q-bio.QM/0405012 v1 17 May 2004

Contents

1 Introduction **4**

2 The problem to solve **4**

3 Two features of single neuron data we would like to include in the spike-sorting procedure **6**

3.1 Spike waveforms *from a single neuron* are usually not stationary on a short time - scale . . . 6

3.1.1 An experimental illustration with cerebellar Purkinje cells . . . 6

3.1.2 A phenomenological description by an exponential relaxation . . . 6

3.2 Neurons inter - spike interval probability density functions carry a lot of information we would like to exploit 6

3.2.1 An experimental illustration from projection neurons in the locust antennal lobe . . . 6

3.2.2 A phenomenological description by a log - Normal *pdf* . . . 7

4 Noise properties **8**

4.1 Noise whitening . . . 8

5 Probabilistic data generation model **10**

5.1 Model assumptions . . . 10

5.2 Likelihood computation for single neuron data . . . 10

5.3 Complications with multi - neuron data . . . 13

5.3.1 Notations for the multi - neuron model parameters . . . 13

5.3.2 Configuration and *data augmentation* . . . 13

5.3.3 Posterior density . . . 14

5.4 Remarks on the use of the posterior . . . 14

5.5 The normalizing constant problem and its solution . . . 14

5.5.1 The problem . . . 14

5.5.2 Its solution . . . 15

6 Markov Chains **15**

6.1 Some notations and definitions . . . 16

6.2 The fundamental theorem and the ergodic theorem . . . 17

7 The Metropolis - Hastings algorithm and its relatives **20**

7.1 The Metropolis - Hastings algorithm . . . 21

7.1.1 Second fundamental theorem . . . 21

7.2 Metropolis - Hastings and Gibbs algorithms for multi - dimensional spaces . . . 21

7.2.1 An example: the Gibbs sampler for the parameters of the *ISI* density . . . 22

7.2.2 Generation of the amplitude parameters of our model . . . 23

7.2.3 Generation of the configuration . . . 24

7.2.4 Our complete MC step . . . 24

8 Priors choice **24**

9 The good use of the ergodic theorem. A warning. **25**
9.1 Autocorrelation functions and confidence intervals . . . 25
9.2 Initialization bias . . . 26

10 Slow relaxation and the Replica Exchange Method **26**

11 An Example from a simulated data set **27**
11.1 Data properties . . . 28
11.2 Algorithm dynamics and parameters estimates without the REM . . . 28
11.2.1 Initialization . . . 28
11.2.2 Energy evolution . . . 28
11.2.3 Model parameters evolution . . . 30
11.2.4 Model parameters estimates . . . 30
11.3 Algorithm dynamics and parameters estimates with the REM . . . 30
11.3.1 Making sure the REM "works" . . . 32
11.3.2 Posterior estimates with the REM . . . 32
11.3.3 Configuration estimate . . . 33
11.3.4 A more detailed illustration of the REM dynamics. . . . 34

12 Conclusions **35**

13 Exercises solutions **36**
13.1 Cholesky decomposition (or factorization) . . . 36
13.2 Bayesian posterior densities for the parameters of a log-Normal *pdf* . . . 36
13.3 Stochastic property of a Markov matrix . . . 37
13.4 Detailed balance . . . 38

1 Introduction

Most of this book is dedicated to the presentation of models of neuronal networks and of methods developed to work with them. Within the frameworks set by these models, the activity of populations of neurons is derived based on the intrinsic properties of "simplified" neurons and of their connectivity pattern. The experimentalist trying to test these models must start by collecting data for which model predictions can be made, which means he or she must record the activity of several neurons at once. If the model does in fact predict the compound activity of a local neuronal population (*e.g.*, a cortical column or hyper-column) on a relatively slow time scale (100 msec or more), techniques like intrinsic optical imaging [9] are perfectly adapted. If the model makes "more precise" predictions on the activity of individual neurons on a short time scale (~ msec) then appropriate techniques must be used like fast optical recording with voltage sensitive dyes [36] or the more classical extracellular recording technique [22].

We will focus in this chapter on the problems associated with the extracellular recording technique which is still a very popular investigation method. This popularity is partly due to its relative ease of implementation and to its low cost. The author is moreover clearly biased toward this technique being one of its users. We hope nevertheless that users of other techniques will learn something from what follows. We will explain how to make inferences about values of the parameters of a rather complex model from noisy data. The approach we will develop can be adapted to any problem of this type. What will change from an experimental context to another is the relevant data generation model and the noise model. That does not mean that the adaptation of the method to other contexts is trivial, but it is still doable.

The data generation model with which we will work in this chapter is qualitatively different from the ones which have been considered in the past [21, 34, 31, 30]. The reader unfamilliar with the spike-sorting problem and willing to get some background on it can consult with profit Lewicki's review [22]. To get an introduction on the actual statistical methods adapted to the models previously considered, we recommend the consultation of the statistical literature (*eg*, [3]) rather than the spike-sorting one, which tends, in our opinion, to suffer from a strong taste for ad-hoc methods.

2 The problem to solve

Extracellular recordings are typically a mixture of spike waveforms originating from a generally unknown number of neurons to which a background noise is superposed as illustrated on Fig 1. Several features can be used to distinguish spikes from different neurons [22] like the peak amplitude on a single of several recording sites (Fig 2), the spike width, a bi - or tri - phasic waveform, etc.

What do we want?

- Find the number of neurons contributing to the data.
- Find the value of a set of parameters characterizing the signal generated by each neuron (*e.g.*, the spike waveform of each neuron on each recording site).
- Acknowledging the classification ambiguity which can arise from waveform similarity and/or signal corruption due to noise, the probability for each neuron to have generated each event (spike) in the data set.
- A method as automatic as possible.
- A non ad - hoc method to answer the above questions. By non ad - hoc we mean a method based on an *explicit probabilistic* model for data generation.

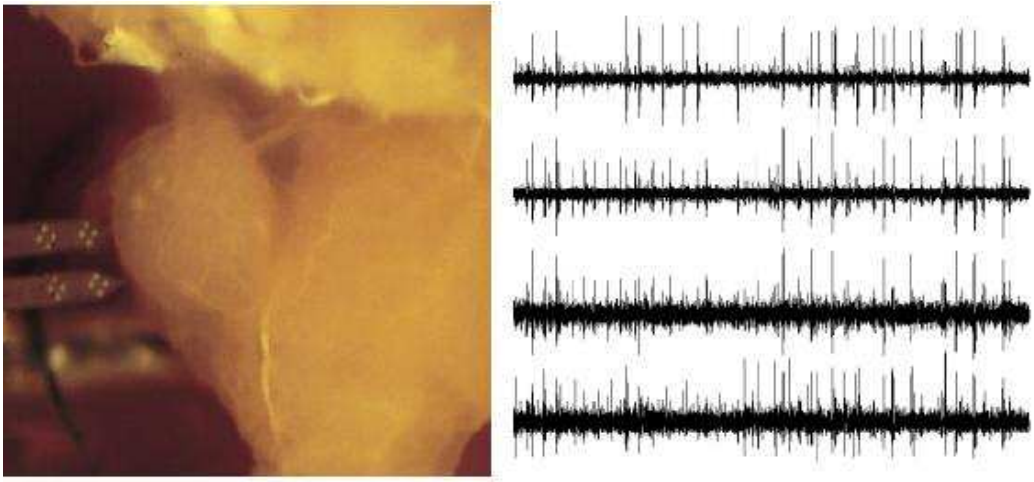

Figure 1: Example of a tetrode recording from the locust (*Schistocerca americana*) antennal lobe. Left, recording setting. The probe, a silicon substrate made of two shanks with 16 iridium deposits (the actual recording sites), can be seen on the left side of the picture. Each group of 4 recording sites is called a *tetrode* (there are therefore 4 tetrodes by probe). The recording sites are the bright spots. The width of the shanks is 80 μm , the side length of each recording site is 13 μm , the diagonal length of each tetrode is 50 μm , the center to center distance between neighboring tetrodes is 150 μm . The structure right beside the probe tip is the *antennal lobe* (the first olfactory relay of the insect), its diameter is approximately 400 μm . Once the probe has been gently pushed into the antennal lobe such that the lowest two tetrodes are roughly 100 μm below the surface one gets on these lowest tetrodes data looking typically as shown on the right part of the figure. Right, 1s of data from a single tetrode filtered between 300 and 5kHz.

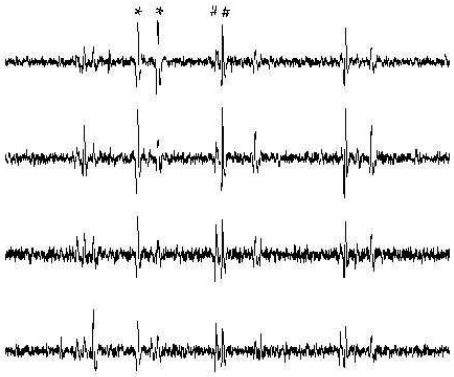

Figure 2: The last 200 ms of Fig. 1. Considering two pairs of spikes (** and ##) the interest of the tetrodes becomes clear. On the first recording site (top) the two spikes of the pair (**) look very similar and it would therefore be hard for an analyst having only the top recording site information to say if these two spikes originate from the same neuron or from two different ones. If now, one looks at the same spikes on the three other recording sites the difference is obvious. The same holds for the two spikes of the pair (##). They look similar on sites 3 and 4 but very dissimilar on sites 1 and 2.

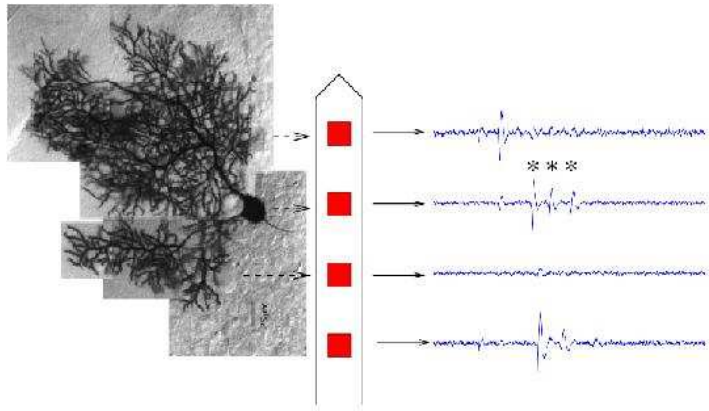

Figure 3: An example of a multi - electrode recording from a rat cerebellar slice. Left, picture of a slice (from a 38 days old rat) with a biocytin filled Purkinje cell. Middle, the tip of a Michigan probe drawn to scale. Right, 100 ms of data filtered between 300 and 5K Hz, Scale bar: 10 ms, Stars: a triplet of spike coming from a single Purkinje cell. (Delescluse and Pouzat, unpublished)

3 Two features of single neuron data we would like to include in the spike-sorting procedure

3.1 Spike waveforms *from a single neuron* are usually not stationary on a short time - scale

3.1.1 An experimental illustration with cerebellar Purkinje cells

One commonly observes that, when principal cells fire bursts of action potentials, the spike amplitude decreases[1] during the burst as illustrated on Fig. 3.

3.1.2 A phenomenological description by an exponential relaxation

Following [6] we will use an exponential relaxation to describe the spike waveform dependence upon the inter - spike interval:

$$\mathbf{a}\left(isi\right) = \mathbf{p} \cdot \left(1 - \delta \cdot \exp\left(-\lambda \cdot isi\right)\right), \tag{1}$$

where $\mathbf{p}$ is the vector of maximal amplitudes (or full waveforms) on the different recording sites, $\delta \in [0, 1]$, λ, measured in 1/s, is the inverse of the relaxation time constant.

3.2 Neurons inter - spike interval probability density functions carry a lot of information we would like to exploit

3.2.1 An experimental illustration from projection neurons in the locust antennal lobe

It is well known and understood since the squid giant axon study by Hodgkin and Huxley that once a neuron (or a piece of it like its axon) has fired an action potential, we must wait "a while" before the next action potential can be fired. This delay is dubbed the *refractory period* and is mainly due to inactivation of sodium channels and to strong activation of potassium channels at the end of the spike, meaning that we must wait for the potassium channels to de - activate and for sodium channel to de - inactivate. Phenomenologically it means that we should observe on the inter - spike interval (*ISI*) histogram from a single neuron a period without spikes (*i.e.*, the *ISI* histogram should start at zero and stay at zero for a finite time). In addition we can often find on *ISI* histograms some other features like a single mode[2] a "fast" rise and a "slower" decay as illustrated on Fig 4. The knowledge of the *ISI* histogram can in

[1] More generally the spike shape changes and basically slows down [13]. This is mainly due to sodium channels inactivation.
[2] That's the statistician's terminology for local maximum.

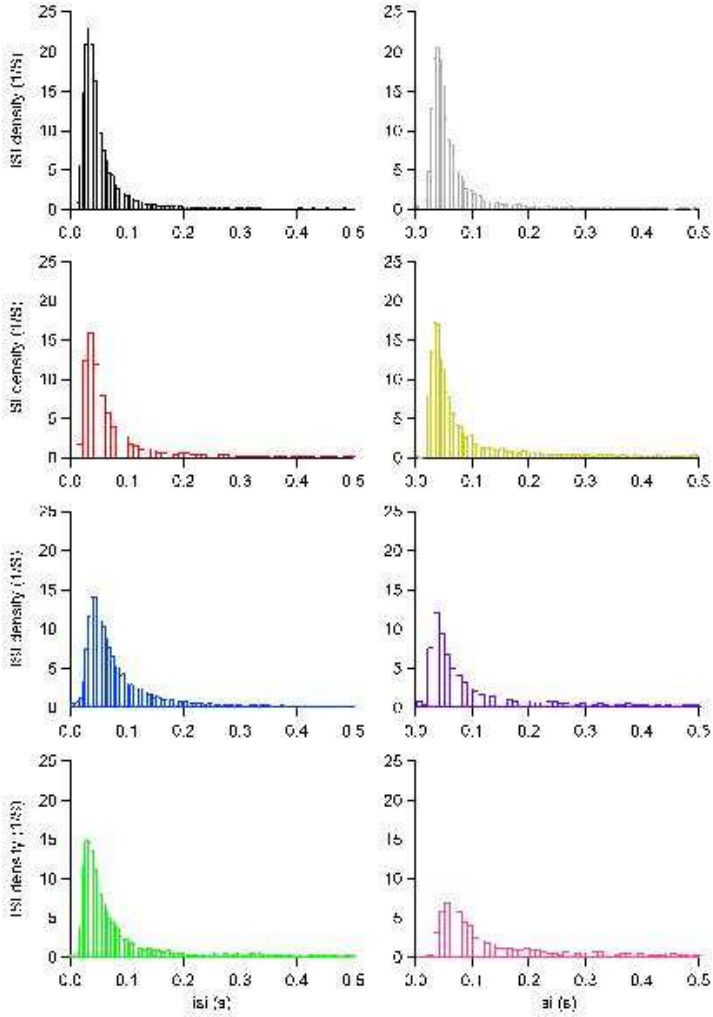

Figure 4: An example of *ISI pdf* estimates for 8 projection neurons simultaneously recorded in the locust antennal lobe (Pouzat, Mazor and Laurent, unpublished).

principle be used to improve spike - sorting because it will induce correlations between the labeling of successive spikes. Indeed, if in one way or another we can be sure of the labeling of a given spike to a given neuron, we can be sure as well that the probability to have an other spike from the same neuron within say the next 10 ms is zero, that this probability is high between 30 and 50 ms and high as well between 60 and 100 (you just need to integrate the *ISI* histogram to see that).

3.2.2 A phenomenological description by a log - Normal *pdf*

We will use a log - Normal approximation for our empirical *ISI* probability density functions (*pdf*s):

$$\pi\left(ISI = isi \mid S = s, F = f\right) = \frac{1}{isi \cdot f \cdot \sqrt{2\pi}} \cdot \exp\left[-\frac{1}{2} \cdot \left(\frac{\ln isi - \ln s}{f}\right)^2\right], \tag{2}$$

where S is a scale parameter (measured in seconds, like the *ISI*) and F is a shape parameter (dimensionless). Fig. 5 shows three log - Normal densities for different values of S and F. The lowest part of the figure shows that when we look at the density of the logarithm of the *ISI* we get a Normal distribution which explains the name of this density.

In the sequel, we will in general use π to designate proper probability density functions (*pdf* for continuous random variables) or probability mass functions (*pmf* for discrete random variables). By proper we mean the integral (or the sum) over the appropriate space is 1. We will (try to) use uppercases to designate random variables (*e.g.*, *ISI, S,* F) and lowercases to designate their realizations (*e.g.*, *isi*, s, f). We will use

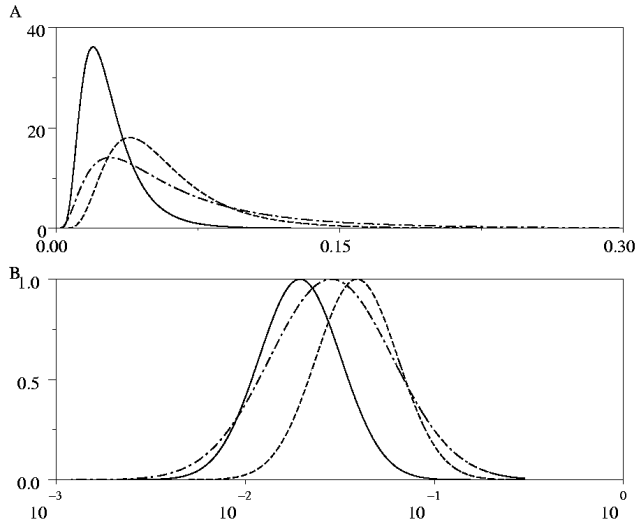

Figure 5: A, examples of log - Normal densities. plain: S = 0.025, F = 0.5; dashed: S = 0.05, F = 0.5; dot-dashed: S = 0.05, F = 0.75. B, peak normalized densities displayed with a logarithmic abscissa.

a Bayesian approach, which means that the model parameters (like S and F in Eq. 2) will be considered as random variables.

4 Noise properties

It's better to start with a reasonably accurate noise model and Fig. 6 illustrates how the noise statistical properties can be obtained from actual data. Once the empirical noise auto - and cross - correlation functions have been obtained, assuming the noise is stationary, the noise covariance matrix (Σ) is constructed as a block - Toeplitz matrix. There are as many blocks as the square of the number of recording sites. The first row of each block is the corresponding auto- or cross - correlation function, see [31]. Then, if Σ is a complete noise description, the *pdf* of a noise vector $\mathbf{N}$ is multivariate Gaussian:

$$\pi\left(\mathbf{N}=\mathbf{n}\right)=\frac{1}{(2\pi)^{\frac{D}{2}}}\cdot\left|\Sigma^{-1}\right|^{\frac{1}{2}}\cdot\exp\left(-\frac{1}{2}\cdot\mathbf{n}^{T}\Sigma^{-1}\,\mathbf{n}\right), \tag{3}$$

where D is the dimension of the space used to represent the events (and the noise), $\left|\Sigma^{-1}\right|$ stands for the determinant of the inverse of Σ, $\mathbf{n}^T$ stands for the transpose of $\mathbf{n}$.

4.1 Noise whitening

If the noise covariance matrix is known, it is useful to know about a transformation, *noise whitening*, which is easy to perform. It makes the equations easier to write and computations faster. If Σ is indeed a covariance matrix, that implies:

$$\mathbf{v}^{T}\Sigma^{-1}\,\mathbf{v}\geq 0,\forall\,\mathbf{v}\,\in\Re^{D}. \tag{4}$$

That is, the *Mahalanobis distance* ($\mathbf{v}^T\Sigma^{-1}\,\mathbf{v}$) is a proper distance which is stated differently by saying that the inverse of Σ is positive definite. Then one can show (Exercise 1 below) that there exists a unique lower triangular matrix A such that $AA^T=\Sigma^{-1}$ and transform vector $\mathbf{v}$ of Eq. 4 into $\mathbf{w}=A^T\mathbf{v}$. We then have:

$$\mathbf{w}^{T}\mathbf{w}=\mathbf{v}^{T}\Sigma^{-1}\,\mathbf{v}. \tag{5}$$

That is, we found a new coordinate system in which the *Mahalanobis distance* is the *Euclidean distance*. We say we perform *noise whitening* when we go from $\mathbf{v}$ to $\mathbf{w}$.

Assuming the noise is stationary and fully described by its second order statistical properties is not enough. One can check the validity of this assumption after whitening an empirical noise sample as illustrated in Fig. 7.

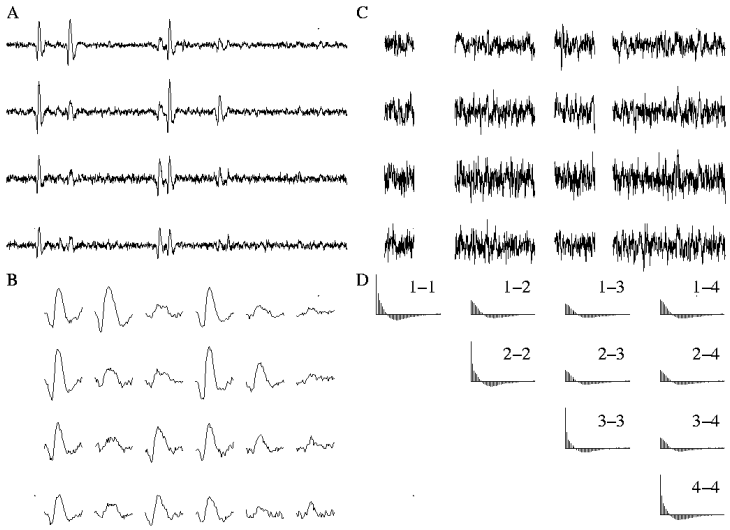

Figure 6: Procedure used to obtain the second order statistical properties of the recording noise illustrated with a tetrode recording from the locust antennal lobe. Putative "events" (B, the sweeps are 3 ms long) and "noise" (C) are separated from the complete data (A) after a threshold crossing and template matching detection of the events. The noise auto - and cross - correlation functions (D, 3 ms are shown for each function, the autocorrelations functions are on the diagonal, the crosscorrelation functions on the upper part) are computed from the reconstructed noise traces (C). Adapted from [31].

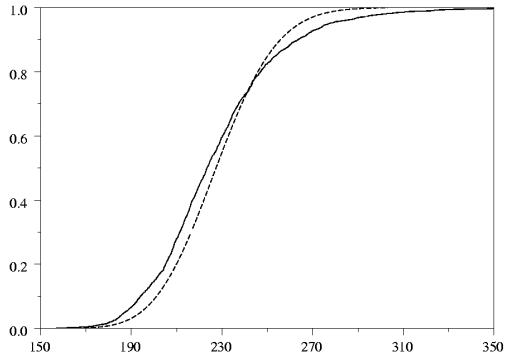

Figure 7: How good is a noise description based on a multivariate Gaussian *pdf*? A typical example (from locust antennal lobe) where Σ turns out to be a reasonable noise description. A sample of 2000 noise events $\{\mathbf{n}_1, \ldots, \mathbf{n}_{2000}\}$ was randomly selected from the reconstructed noise trace (Fig. 6C), the *Mahalanobis distance* to the origin: $\mathbf{n}_i^T \Sigma^{-1} \mathbf{n}_i$, was computed for each of them and the cumulative distribution of these distances (plain) was compared with the theoretical expectation (dashed) a χ^2distribution (with in that case 238 degrees of freedom). See [31] for details.

Exercise 1

Assume that $\Gamma = \Sigma^{-1}$ is a symmetric positive definite matrix. Show there exists a unique (in fact 2, one being obtained from the other by a multiplication by - 1) lower triangular matrix A (the *Cholesky* factor) such that:

$$AA^T = \Gamma. \tag{6}$$

In order to do that you will simply get the algorithm which computes the elements of A from the elements of Γ. See Sec. 13.1 for solution.

5 Probabilistic data generation model

5.1 Model assumptions

We will make the following assumptions:

1. The firing statistics of each neuron is fully described by its time independent inter - spike interval density. That is, the sequence of spike times from a given neuron is a realization of *a homogeneous renewal point process* [18]. More specifically, we will assume that the *ISI pdf* is log - Normal (Eq. 2).
2. The amplitude of the spikes generated by each neuron depends on the elapsed time since the last spike of this neuron. More specifically, we will assume that this dependence is well described by Eq. 1.
3. The measured amplitude of the spikes is corrupted by a Gaussian white noise which sums linearly with the spikes and is statistically independent of them. That is, we assume that noise whitening (Eq. 5) has been performed.

5.2 Likelihood computation for single neuron data

With a data set $\mathcal{D}' = \{(t_0, \mathbf{a}_0), (t_1, \mathbf{a}_1), \ldots, (t_N, \mathbf{a}_N)\}$, the likelihood is readily obtained (Fig. 8). One must first get the *isi* values: $i_j = t_j - t_{j-1}$, which implies one has to discard the first spike of the data set[3] to get the "effective" data set: $\mathcal{D} = \{(i_1, \mathbf{a}_1), \ldots, (i_N, \mathbf{a}_N)\}$. The likelihood is then[4]:

$$L(\mathcal{D} \mid \mathbf{p}, \delta, \lambda, s, f) = \prod_{j=1}^{N} \pi_{isi}(i_j \mid s, f) \cdot \pi_{amplitude}(\mathbf{a}_j \mid i_j, \mathbf{p}, \delta, \lambda), \tag{7}$$

where $\pi_{isi}(i_j \mid s, f)$ is given by Eq. 2 and:

$$\pi_{amplitude}(\mathbf{a}_j \mid i_j, \mathbf{p}, \delta, \lambda) = \frac{1}{(2\pi)^{\frac{D}{2}}} \cdot \exp\left\{-\frac{1}{2}\left\|\mathbf{a}_j - \mathbf{p} \cdot (1 - \delta \cdot \exp(-\lambda \cdot i_j))\right\|^2\right\}. \tag{8}$$

This last equation is obtained by combining Eq. 1 with our third model assumption (Gaussian white noise). For convenience we write $\pi_{isi}(i_j \mid s, f)$ for $\pi_{isi}(ISI_j = i_j \mid S = s, F = f)$ and $\pi_{amplitude}(\mathbf{a}_j \mid i_j, \mathbf{p}, \delta, \lambda)$ for $\pi_{amplitude}(\mathbf{A}_j = \mathbf{a}_j \mid ISI_j = i_j, \mathbf{P} = \mathbf{p}, \Delta = \delta, \Lambda = \lambda)$, where the ISI_j are considered independent and identically distributed (*iid*) random variables (conditioned on S and F) as well as the $\mathbf{A}_j$ (conditioned on ISI_j, $\mathbf{P}$, Δ and Λ).

[3] To avoid discarding the first spike we can assume periodic boundary conditions, meaning the last spike (N) "precedes" the first one (0), we then have: $i_0 = T - t_N + t_0$, where T is the duration of the recording which started at time 0.

[4] Purists from both sides (frequentist and Bayesian) would kill us for writing what follows (Eq. 7)... A frequentist would rather write:

$$L(\mathbf{p}, \delta, \lambda, s, f; \mathcal{D})$$

because for him the likelihood function is a random function (it depends on the sample through $\mathcal{D}$) and its arguments are the model parameters. The Bayesian, probably to show the frequentist that he perfectly understood the meaning of the likelihood function, would introduce a new symbol, say h and write:

$$h(\mathcal{D} \mid \mathbf{p}, \delta, \lambda, s, f) = L(\mathbf{p}, \delta, \lambda, s, f; \mathcal{D}).$$

But the likelihood is, within a normalizing constant, the probability of the data for given values of the model parameters. We will therefore use the heretic notation of Eq. 7.

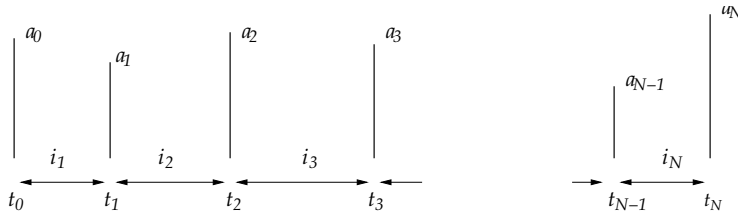

Figure 8: Illustration of the likelihood computation for data from a single neuron recorded on a single site. For simplicity we assume that the spike peak amplitude only is used.

The log - likelihood ($\mathcal{L} = \ln L$) can be written as a sum of two terms:

$$\mathcal{L}\left(\mathcal{D} \mid \mathbf{p}, \delta, \lambda, s, f\right) = \mathcal{L}_{isi}\left(\mathcal{D} \mid s, f\right) + \mathcal{L}_{amplitude}\left(\mathcal{D} \mid \mathbf{p}, \delta, \lambda\right), \tag{9}$$

where:

$$\mathcal{L}_{isi}\left(\mathcal{D} \mid s, f\right) = -N \cdot \ln f - \sum_{j=1}^{N} \left\{ \ln i_j + \frac{1}{2} \left[\frac{\ln\left(\frac{i_j}{s}\right)}{f} \right]^2 \right\} + Cst, \tag{10}$$

and:

$$\mathcal{L}_{amplitude}\left(\mathcal{D} \mid \mathbf{p}, \delta, \lambda\right) = -\frac{1}{2} \sum_{j=1}^{N} \left\| \mathbf{a}_j - \mathbf{p} \cdot \left(1 - \delta \cdot \exp\left(-\lambda \cdot i_j\right)\right) \right\|^2 + Cst. \tag{11}$$

The term $\mathcal{L}_{isi}$ is a function of the *ISI pdf* parameters only and the term $\mathcal{L}_{amplitude}$ is a function of the "amplitude dynamics parameters" only.

Exercise 2

Starting with the expression of the log - likelihood given by Eq. 10 , show that the values $\widetilde{s}$ and $\widetilde{f}$ of s and f maximizing the likelihood are given by:

$$\ln \widetilde{s} = \frac{1}{N} \sum_{j=1}^{N} \ln i_j, \tag{12}$$

and:

$$\widetilde{f} = \sqrt{\frac{1}{N} \sum_{j=1}^{N} \left[\ln\left(\frac{i_j}{\widetilde{s}}\right) \right]^2}. \tag{13}$$

Exercise 3

Remember that in a Bayesian setting your start with a prior density for your model parameters: $\pi_{prior}\left(s, f\right)$, you observe some data, $\mathcal{D}$, you have an explicit data generation model (*i.e.*, you know how to write the likelihood function), $L\left(\mathcal{D} \mid s, f\right)$ and you can therefore write using Bayes rule:

$$\pi\left(s, f \mid \mathcal{D}\right) = \frac{L\left(\mathcal{D} \mid s, f\right) \cdot \pi_{prior}\left(s, f\right)}{\int_{u,v} du\, dv\, L\left(\mathcal{D} \mid u, v\right) \cdot \pi_{prior}\left(u, v\right)} \tag{14}$$

The "Bayesian game" in simple cases is always the same, you take the numerator of Eq. 14, you get rid of everything which does not involve the model parameters (because that will be absorbed in the normalizing constant) and you try to recognize in what's left a known *pdf* like a Normal, a Gamma, a Beta, etc.

In the following we will assume that our prior density is uniform on a rectangle, that is:

$$\pi_{prior}\left(s, f\right) = \frac{1}{s_{max} - s_{min}} \cdot \mathcal{I}_{[s_{min}, s_{max}]}\left(s\right) \cdot \frac{1}{f_{max} - f_{min}} \cdot \mathcal{I}_{[f_{min}, f_{max}]}\left(f\right), \tag{15}$$

where $\mathcal{I}_{[x,y]}$is the indicator function:

$$\mathcal{I}_{[x,y]}(u) = \begin{cases} 1, \, if \, x \leq u \leq y \\ 0, \, otherwise \end{cases} \tag{16}$$

Detailed answers to the following questions can be found in Sec. 13.2.

Question 1 Using Eq. 8 and Eq. 2 show that the posterior density of $\ln S$ conditioned on $F = f$ and $\mathcal{D}$ is a truncated Gaussian with mean:

$$\overline{\ln i} = \frac{1}{N} \sum_{j=1}^{N} \ln i_j, \tag{17}$$

and un-truncated variance (that is, the variance it would have if it was not truncated):

$$\sigma^2 = \frac{f^2}{N}. \tag{18}$$

And therefore the posterior conditional *pdf* of the scale parameter is given by:

$$\pi(s \mid f, \mathcal{D}) \; \alpha \; \exp\left[-\frac{1}{2}\frac{N}{f^2}\left(\overline{\ln i} - \ln s\right)^2\right] \cdot \frac{1}{s_{max} - s_{min}} \cdot \mathcal{I}_{[s_{min}, s_{max}]}(s). \tag{19}$$

Question 2 Write an algorithm which generates a realization of S according to Eq. 19.

Question 3 We say that a random variable Θ has an *Inverse Gamma* distribution and we write: $\Theta \sim Inv - Gamma(\alpha, \beta)$ if the *pdf* of Θ is given by:

$$\pi(\Theta = \theta) = \frac{\beta^\alpha}{\Gamma(\alpha)} \theta^{-(\alpha+1)} \exp\left(-\frac{\beta}{\theta}\right). \tag{20}$$

Using Eq. 8 and Eq. 2 show that the posterior density of F^2 conditioned on $S = s$ and $\mathcal{D}$ is a truncated *Inverse Gamma* with parameters:

$$\alpha = \frac{N}{2} - 1$$

and

$$\beta = \frac{1}{2} \sum_{j=1}^{N} (\ln i_j - \ln s)^2.$$

Therefore the posterior conditional *pdf* of the shape parameter is:

$$\pi(f \mid s, \mathcal{D}) \; \alpha \; \frac{1}{f^N} \exp\left[-\frac{1}{2}\frac{\sum_{j=1}^{N}(\ln i_j - \ln s)^2}{f^2}\right] \cdot \frac{1}{f_{max} - f_{min}} \cdot \mathcal{I}_{[f_{min}, f_{max}]}(f). \tag{21}$$

Question 4 Assuming that your favorite analysis software (*e.g.*, Scilab[5] or Matlab) or your favorite C library (*e.g.*, GSL[6]) has a *Gamma* random number generator, write an algorithm which generates a realization of F from its posterior density Eq. 21. The *pdf* of a *Gamma* random variable Ω is given by:

$$\pi(\Omega = \omega) = \frac{\beta^\alpha}{\Gamma(\alpha)} \omega^{\alpha-1} \exp(-\beta \cdot \omega). \tag{22}$$

[5] http://www.scilab.org
[6] The Gnu Scientific Library: http://sources.redhat.com/gsl

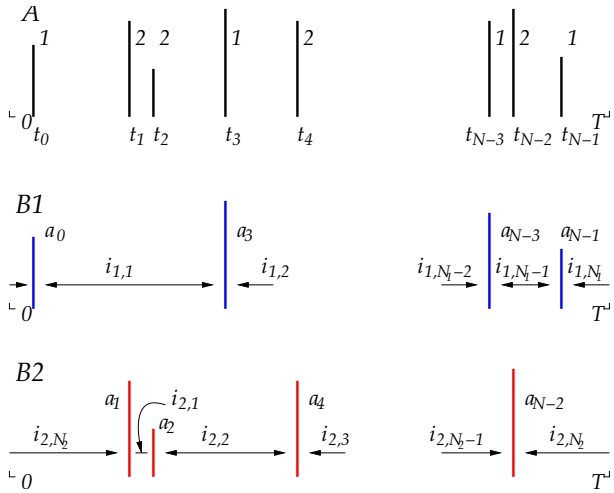

Figure 9: A simple example of sub-trains extraction. We consider a case with 2 neurons in the model and a single recording site. A, Spikes are shown with their labels in the present configuration and their occurrence times. The recording goes from *0* to *T*. B1, the sub-train attributed to neuron 1 has been isolated. The relevant *ISIs* are shown, as well as the amplitudes of the events. B2, same as B1 for the sub-train attributed to neuron 2.

5.3 Complications with multi - neuron data

5.3.1 Notations for the multi - neuron model parameters

In the following we will consider a model with K different neurons, which means we will have to estimate the value of a maximal peak amplitude parameter ($\mathbf{P}$), an attenuation parameter (Δ), the inverse of a relaxation time constant (Λ), a scale parameter (S) and a shape parameter (F) for each of the K neurons of our model. We will write Θ the random vector lumping all these individual parameters:

$$\Theta = (\mathbf{P}_1, \Delta_1, \Lambda_1, S_1, F_1, \ldots, \mathbf{P}_K, \Delta_K, \Lambda_K, S_K, F_K)\,. \tag{23}$$

5.3.2 Configuration and *data augmentation*

When we deal with multi - neuron data we need to formalize our ignorance about the origin of each individual spike. In order to do that, assuming we work with a model with K neurons, we will "attach" to each spike, j, in our data set ($\mathcal{D}$) a *label*, $l_j \in \{1, \ldots, K\}$ whose value corresponds to the number of the neuron which generated it. From this view point, a spike - sorting procedure is a method to set the labels values. But we don't know, at least until we have estimated the model parameters, what is the value of each l_j. In such an uncertain situation the best we can do is to consider l_j as a realization of a random variable L_j taking values in $\{1, \ldots, K\}$, then the best we can expect from a spike - sorting procedure is the *distribution* (or the *pmf*) *of* L_j for $j \in \{1, \ldots, N\}$.

Our data generation model will necessarily induce a dependence among the L_j because we take into account both the *ISI* densities and the spike waveform dynamics, we will therefore introduce an other random variable, the *configuration*, C defined as a vector of L_j that is:

$$C = (L_1, \ldots, L_N)^T\,. \tag{24}$$

The introduction of this random variable C is a very common procedure in the statistical literature where it is called *data augmentation*. The key idea being that we are given an *incomplete* data set ($\mathcal{D}$), which basically does not allow us to write the likelihood easily, and if we knew some extra properties of the data (like the *configuration*), we could write down the likelihood of the *augmented* (or *completed*) data in a straightforward way[7]. Indeed, with C introduced the likelihood of the augmented data ($L(\mathcal{D}, c \mid \theta)$) is obtained in two steps:

[7] C is also called a *latent* variable in the statistical literature. Statisticians, moreover, commonly use the symbol Z for our C. We have change the convention for reasons which will soon become clear.

1. Using the configuration realization, c, we can separate the "complete" data set into sub - trains corresponding to the different neurons of the model. This is illustrated with a simple case, $K = 2$ and a single recording site on Fig. 9. Here we have: $i_{1,1} = t_3 - t_0$, $i_{1,N_1-1} = t_{N-3} - t_{N-1}$, $i_{1,N_1} = T - t_{N-1} + t_0$, where periodic boundary conditions are assumed and where N_1 is the number of spikes attributed to neuron 1 in c. For neuron 2 we get: $i_{2,1} = t_2 - t_1$, $i_{2,2} = t_4 - t_2$, $i_{2,N_2} = T - t_{N-2} + t_1$, where N_2 is the number of spikes attributed to neuron 2 in c. Using Eq. 8 we can compute a "sub - likelihood" for each sub - train: $L_{l_j=1}(\mathcal{D}, c \mid \theta)$, $L_{l_j=2}(\mathcal{D}, c \mid \theta)$.

2. Because our data generation model does not include interactions between neurons, the "complete" likelihood is simply[8]:

$$L(\mathcal{D}, c \mid \theta) = \prod_{q=1}^{K} L_{l_j=q}(\mathcal{D}, c \mid \theta). \tag{25}$$

5.3.3 Posterior density

Now that we know how to write the likelihood of our *augmented* data set, we can obtain the posterior density of all our unknowns Θ *and* C applying Bayes rule:

$$\pi_{posterior}(\theta, c \mid \mathcal{D}) = \frac{L(\mathcal{D}, c \mid \theta) \cdot \pi_{prior}(\theta)}{Z}, \tag{26}$$

with the normalizing constant Z given by:

$$Z = \sum_{c \in \mathcal{C}} \int_{\theta} d\theta \, L(\mathcal{D}, c \mid \theta) \cdot \pi_{prior}(\theta), \tag{27}$$

where $\mathcal{C}$ is the set of all configurations.

5.4 Remarks on the use of the posterior

It should be clear that if we manage to get the posterior: $\pi_{posterior}(\theta, c \mid \mathcal{D})$, then we can answer any quantitative question about the data. For instance if we want the probability of a specific configuration c we simply need to integrate out θ:

$$\pi(c \mid \mathcal{D}) = \int_{\theta} d\theta \, \pi_{posterior}(\theta, c \mid \mathcal{D}). \tag{28}$$

If we are interested in the cross - correlogram between to neurons, say 2 and 4, of the model and if we formally write $Cross_{2,4}(c)$ the function which takes as argument a specific configuration (c) and returns the desired cross - correlogram, then the *best* estimate we can make of this quantity is:

$$\langle Cross_{2,4} \rangle = \sum_{c \in \mathcal{C}} \int_{\theta} d\theta \, \pi_{posterior}(\theta, c \mid \mathcal{D}) \cdot Cross_{2,4}(c), \tag{29}$$

where we use the symbol $\langle\rangle$ to designate averaging with respect to a proper *pdf*.

5.5 The normalizing constant problem and its solution

5.5.1 The problem

If we take a look at our normalizing constant Z (Eq. 27) we see a practical problem emerging. To compute Z we need to carry out a rather high dimensional integration (on θ) *and* a summation over every possible configuration in $\mathcal{C}$. That's where the serious problem arises because $\mathcal{C}$ contains K^N elements! Blankly stated, we have no way, in real situations where K varies from 2 to 20 and where N is of the order of 1000, to compute Z.

[8]Strictly speaking, this is true only if we can ignore spike waveforms superpositions.

Fine, if we can't compute directly Z, let us look for a way around it. First, we introduce an "energy function":

$$E(\theta, c \mid \mathcal{D}) = -\ln\left[L(\mathcal{D}, c \mid \theta) \cdot \pi_{prior}(\theta)\right]. \tag{30}$$

E is defined in such a way that given the data ($\mathcal{D}$), a value for the model parameters (θ) and a configuration (c) we can compute it.

Second, we rewrite our posterior expression as follows:

$$\pi_{posterior}(\theta, c \mid \mathcal{D}) = \frac{\exp\left[-\beta E(\theta, c \mid \mathcal{D})\right]}{Z}, \tag{31}$$

where $\beta = 1$.

Third, we make an analogy with statistical physics: E is an energy, Z a *partition function*, β is proportional to the inverse of the temperature ($(kT)^{-1}$) and $\pi_{posterior}(\theta, c \mid \mathcal{D})$ is the canonical distribution [29, 19].

Fourth, we look at the way physicists cope with the combinatorial explosion in the partition function...

5.5.2 Its solution

Physicists are interested in computing expected values (*i.e.*, things that can be measured experimentally) like the expected internal energy of a system:

$$\langle E \rangle = \sum_{c \in \mathcal{C}} \int_{\theta} d\theta \, \frac{\exp\left[-\beta E(\theta, c \mid \mathcal{D})\right]}{Z} \cdot E(\theta, c \mid \mathcal{D}), \tag{32}$$

or like the pair - wise correlation function between two molecules in a gas or in a fluid which will give rise to equations like Eq. 29. Of course they are very rarely able to compute Z explicitly so the trick they found fifty years ago [24] is to *estimate* quantities like $\langle E \rangle$ with *Monte Carlo* integration.

At this point to make notation lighter we will call *state* of our "system" the pair (θ, c) and we will write it as: $x = (\theta, c)$. We will moreover drop the explicit dependence on $\mathcal{D}$ in E. That is, we will write $E(x)$ for $E(\theta, c \mid \mathcal{D})$. Now by *Monte Carlo* integration we mean that we "create" a sequence of states: $x^{(1)}, x^{(2)}, \ldots, x^{(M)}$ such that:

$$\lim_{M \to \infty} \overline{E} = \langle E \rangle, \tag{33}$$

where:

$$\overline{E} = \frac{1}{M} \sum_{t=1}^{M} E\left(x^{(t)}\right), \tag{34}$$

is the empirical average of the energy. Showing how to create the sequence of states $\left\{x^{(t)}\right\}$ and proving Eq. 33 will keep us busy in the next section.

6 Markov Chains

Before embarking with the details of Monte Carlo integration let's remember the warning of one of its specialists, Alan Sokal [35]:

> Monte Carlo is an extremely bad method; it should be used only when all alternative methods are worse.

That being said it seems we are in a situation with no alternative method, so let's go for it.

Our first problem is that we can't directly (or independently) generate the $x^{(t)}$ from the posterior density because this density is too complicated. What we will do instead is generate the sequence as a realization of a *Markov Chain*. We will moreover build the transition matrix of the Markov chain such that regardless of the initial state $x^{(0)}$, we will have for any bounded function H of x the following limit:

$$\lim_{M \to \infty} \frac{1}{M} \sum_{t=1}^{M} H\left(x^{(t)}\right) = \langle H \rangle. \tag{35}$$

In principle we should consider here Markov chains in a (partly) continuous space because $x = (\theta, c)$ and θ is a continuous vector. But we will take the easy solution, which is fundamentally correct anyway, of saying that because we use a computer to simulate the Markov chain, the best we can do is approximate the continuous space in which θ lives by a *discrete and finite* space. We therefore split the study of the conditions under which Eq. 35 is verified in the general state space (a product of a discrete and a continuous space) into two questions[9]:

1. Under what conditions is Eq. 35 verified for a Markov chain defined on a *discrete and finite* state space?
2. What kind of error do we produce by approximating our general state space (a product of a discrete and of a continuous state space) by a discrete one?

We will answer here the first of these questions. For the second we will hope (and check) that our discrete approximation is not too bad.

6.1 Some notations and definitions

Now that we are dealing with a discrete and finite state space we label all our states x with an integer and our state space $\mathcal{X}$ can be written as:

$$\mathcal{X} = \{x_1, \ldots, x_\nu\}. \tag{36}$$

We can, moreover, make a unique correspondence between a random variable (*rv*), *X*, and its *probability mass function* π_X that we can view as a row vector:

$$\pi_X = (\pi_{X,1}, \pi_{X,2}, \ldots, \pi_{X,\nu}), \tag{37}$$

where:

$$\pi_{X,i} = Pr\left(X = x_i\right), \forall x_i \in \mathcal{X}. \tag{38}$$

Strictly speaking a *rv* would not be defined on a set like $\mathcal{X}$ but on a sample space $\mathcal{S}$ which would include every possible outcome, that is: all elements of $\mathcal{X}$ as well as every possible combination of them (like $x_i \cup x_j$, that we write $x_i + x_j$ if $i \neq j$ because then x_i and x_j are mutually exclusive) and the empty set Ø (corresponding to the impossible outcome). It is nevertheless clear that for a finite state space, knowing the *pmf* of a *rv* is everything one needs to compute the probability of any outcome (*e.g.*, $Pr\left(x_i + x_j\right) = \pi_{X,i} + \pi_{X,j}$). We will therefore in the sequel say that *rv*s are defined on the state space $\mathcal{X}$ acknowledging it is not the most rigorous way to speak.

Formal definition of a Markov chain A sequence of random variables $X^{(0)}, X^{(1)}, \ldots$, defined on a finite state space $\mathcal{X}$ is called a *Markov chain* if it satisfies the *Markov property*:

$$Pr\left(X^{(t+1)} = y \mid X^{(t)} = x, \ldots, X^{(0)} = z\right) = Pr\left(X^{(t+1)} = y \mid X^{(t)} = x\right). \tag{39}$$

If $Pr\left(X^{(t+1)} = y \mid X^{(t)} = x\right)$ does not explicitly depend on *t* the chain is said to be *homogeneous*.

An homogeneous Markov chain on a finite state space is clearly fully described by its initial random variable $X^{(0)}$ and by its *transition matrix*[10] $T(x, y) = Pr\left(X^{(t+1)} = y \mid X^{(t)} = x\right)$. This transition matrix must be *stochastic* which means it must satisfy (Sec 13.3):

$$\sum_{y \in \mathcal{X}} T(x, y) = 1, \forall x \in \mathcal{X}. \tag{40}$$

[9]The results which will be presented in the next section have of course equivalents in the general case. The reference which is always cited by the serious guys is [25].

[10]**Be careful here, the notations are, for historical reasons, misleading. We write down the matrix: $T(x, y)$ where *x* is the state from which we start and *y* the state in which we end. When we use the probability transition kernel notation $Pr(y \mid x)$ the starting state is on the right side, the end site is on the left side! In this section where we will discuss theoretical properties of Markov chains, we will use the matrix notation which can be recognized by the the "," separating the start end end states. When later on, we will use the kernel notation, the 2 states will be separated by the "|" (and the end will be on the left side while the start will be on the right).**

If m is the *pmf* of a *rv* defined on $\mathcal{X}$, we can easily show that the row vector n defined as:

$$n_j = \sum_{i=1}^{\nu} m_i T(x_i, x_j),$$

is itself a *pmf* (*i.e.*, $0 \leq n_i \leq 1\,, \forall\, i \in \{1, \ldots, \nu\}$ and $\sum_{i=1}^{\nu} n_i = 1$) and must therefore correspond to a *rv*. That justifies the following vector notation:

$$\pi_{X^{(t+1)}} = \pi_{X^{(t)}} T, \tag{41}$$

where $\pi_{X^{(t)}}$ and $\pi_{X^{(t+1)}}$ are the *pmf* associated to two successive *rv*s of a Markov chain.

Stationary probability mass function A *pmf* m is said to be stationary with respect to the transition matrix T if: $mT = m$.

Applied to our Markov chain that means that if the *pmf* of our initial *rv* ($\pi_{X^{(0)}}$) is stationary with respect to T, then the chain does not move (*i.e.*, $X^{(t+1)} \equiv X^{(t)}\,, t \geq 0$).

The notion of stationary *pmf* introduced we can restate our convergence problem (Eq 35). What we want in practice is a Markov transition matrix, T, which admits a single stationary *pmf* (which would be in our case $\pi_{posterior}$) and such that for any $\pi_{X^{(0)}}$ we have:

$$\lim_{t \to \infty} \pi_{X^{(t)}} = \pi_{posterior},$$

which means:

$$\forall\, x \in \mathcal{X}\,, \forall\, \epsilon > 0\,, \exists\, t\, \geq 0\, :\, |\pi_{posterior}\,(x) - \pi_{X^{(t)}}\,(x)| \leq \epsilon. \tag{42}$$

It is clear that if we knew explicitly a candidate transition matrix, T, we could test that the posterior *pmf* is stationary and then using eigenvalue decomposition we could check if the largest eigenvalue is 1, etc... See for instance: [2, 7, 33, 23, 27, 35, 29]. The problem is that we won't in general be able to write down explicitly T. We need therefore conditions we can check in practice and which ensure the above convergence. These conditions are *irreducibility* and *aperiodicity*.

Irreducible and aperiodic states (After Liu [23]) A state $x \in \mathcal{X}$ is said to be irreducible if under the transition rule one has nonzero probability of moving from x to any other state and then coming back in a finite number of steps. A state $x \in \mathcal{X}$ is said to be aperiodic if the greatest common divider of $\{t\, :\, T^t\,(x, x) > 0\}$ is 1.

It should not be to hard to see at this point that if one state $x \in \mathcal{X}$ is aperiodic, then all states in $\mathcal{X}$ must also be. Also, if one state $y \in \mathcal{X}$ is aperiodic and the chain is irreducible, then every state in $\mathcal{X}$ must be aperiodic.

Exercise 4

Prove the validity of Eq. 40 for a Markov chain on a finite state space $\mathcal{X}$ (Eq. 36). See Sec. 13.3 for a solution.

6.2 The fundamental theorem and the ergodic theorem

Lemma

Let $\left(X^{(0)}, X^{(1)}, \ldots\right)$ be an irreducible and aperiodic Markov chain with state space $\mathcal{X} = \{x_1, x_2, \ldots, x_\nu\}$ and transition matrix T. Then there exists an $M < \infty$ such that $T^n(x_i, x_j) > 0$ for all $i, j \in \{1, \ldots, \nu\}$ and $n \geq M$.

Proof

(After Häggström [11]) By definition of irreducibility and aperiodicity, there exist an integer $N < \infty$ such that $T^n(x_i, x_i) > 0$ for all $i \in \{1, \ldots, \nu\}$ and all $n \geq N$. Fix two states $x_i, x_j \in \mathcal{X}$. By the assumed irreducibility, we can find some $n_{i,j}$ such that $T^{n_{i,j}}(x_i, x_j) > 0$. Let $M_{i,j} = N + n_{i,j}$. For any $m \geq M_{i,j}$, we have:

$$Pr\left(X^{(m)} = x_j \mid X^{(0)} = x_i\right) \geq Pr\left(X^{(m-n_{i,j})} = x_i, X^{(m)} = x_j \mid X^{(0)} = x_i\right),$$

and:

$$Pr\left(X^{(m-n_{i,j})} = x_i, X^{(m)} = x_j \mid X^{(0)} = x_i\right) = Pr\left(X^{(m-n_{i,j})} = x_i \mid X^{(0)} = x_i\right) Pr\left(X^{(m)} = x_j \mid X^{(0)} = x_i\right)$$

but $m - n_{i,j} \geq 0$ implies that: $Pr\left(X^{(m-n_{i,j})} = x_i \mid X^{(0)} = x_i\right) > 0$ and our choice of $n_{i,j}$ implies that: $Pr\left(X^{(m)} = x_j \mid X^{(0)} = x_i\right) > 0$ therefore:

$$Pr\left(X^{(m)} = x_j \mid X^{(0)} = x_i\right) > 0.$$

We have shown that $T^m(x_i, x_j) > 0$ for all $m \geq M_{i,j}$. The lemma follows with:

$$M = \max\{M_{1,1}, M_{1,2}, \ldots, M_{1,\nu}, M_{2,1}, \ldots, M_{\nu,\nu}\}.$$

Remark It should be clear that the reciprocal of the lemma is true, that is, if there exists an $M < 0$ such that $T^n(x_i, x_j) > 0$ for all $i, j \in \{1, \ldots, \nu\}$ and $n \geq M$ then the chain is irreducible and aperiodic. A transition matrix which has all its elements strictly positive, or such that one of its powers has all its elements strictly positive, will therefore give rise to irreducible and aperiodic Markov chains.

We can now state and prove our first "fundamental" theorem which shows that irreducible and aperiodic Markov chains on finite state spaces enjoy *geometric convergence* to their *unique* stationary *pmf*.

Fundamental theorem

If an irreducible and aperiodic homogeneous Markov chain on a finite state space $\mathcal{X} = \{x_1, x_2, \ldots, x_\nu\}$ with a transition matrix $T(x, y)$ has $\pi_{stationary}$ as a stationary distribution then regardless of the initial *pmf* $\pi_{X^{(0)}}$ we have:

$$\lim_{t \to \infty} \pi_{X^{(t)}} = \pi_{stationary} \tag{43}$$

and there exists an $1 \geq \epsilon > 0$ such that for all $x \in \mathcal{X}$ and all $t > 1$ we have:

$$|\pi_{stationary}(x) - \pi_{X^{(t)}}(x)| \leq (1 - \epsilon)^t \tag{44}$$

Proof

Adapted from Neal [27]. We can without loss of generality assume that $T(x, y) > 0, \forall x, y \in \mathcal{X}$. If such is not the case for our initial transition matrix, we can always use the preceding lemma to find an M such that: $T'(x, y) = T^M(x, y) > 0$ and work with the Markov chain whose transition matrix is T'. Then the uniqueness of the stationary *pmf* regardless of the initial *pmf* will imply that the convergence of the "new" Markov chain holds for the initial Markov chain as well.

So let's define:

$$\epsilon = \min_{x, y \in \mathcal{X}} \frac{T(x, y)}{\pi_{stationary}(y)} > 0,$$

where we are assuming that $\pi_{stationary}(y) > 0, \forall y \in \mathcal{X}$ (if not we just have to redefine the state space to get rid of y and to take out row and column y from T). Then from the transition matrix definition and

the stationarity assumption we have: $\pi_{stationary}(y) = \sum_x \pi_{stationary}(x)\, T(x,y)\,,\ \forall\, y \in \mathcal{X}$ which with the above assumption gives:

$$1 = \sum_x \pi_{stationary}(x)\, \frac{T(x,y)}{\pi_{stationary}(y)} \geq \epsilon \sum_x \pi_{stationary}(x)\,,$$

it follows that: $\epsilon \leq 1$. We will now show by induction that the following equality holds:

$$\pi_{X^{(t)}}(x) = \left[1-(1-\epsilon)^t\right] \pi_{stationary}(x) + (1-\epsilon)^t\, r_t(x)\ ,\ \forall\, x \in \mathcal{X}, \tag{45}$$

where r_t is a *pmf*.

This equality holds for $t=0$, with $r_0 = \pi_{X^{(0)}}$. Let us assume it is correct for t and see what happens for $t+1$. We have:

$$\begin{aligned}
\pi_{X^{(t+1)}}(y) &= \sum_x \pi_{X^{(t)}}(x)\, T(x,y) \\
&= \left[1-(1-\epsilon)^t\right] \sum_x \pi_{stationary}(x)\, T(x,y) \ +\ (1-\epsilon)^t \sum_x r_t(x)\, T(x,y) \\
&= \left[1-(1-\epsilon)^t\right] \pi_{stationary}(y) \ +\ (1-\epsilon)^t \sum_x r_t(x)\ \left[T(x,y) - \epsilon\pi_{stationary}(y) + \epsilon\pi_{stationary}(y)\right] \\
&= \left[1-(1-\epsilon)^{t+1}\right] \pi_{stationary}(y) \ +\ (1-\epsilon)^t \sum_x r_t(x)\ \left[T(x,y) - \epsilon\pi_{stationary}(y)\right] \\
&= \left[1-(1-\epsilon)^{t+1}\right] \pi_{stationary}(y) \ +\ (1-\epsilon)^{t+1} \sum_x r_t(x)\, \frac{T(x,y) - \epsilon\pi_{stationary}(y)}{(1-\epsilon)} \\
&= \left[1-(1-\epsilon)^{t+1}\right] \pi_{stationary}(y) \ +\ (1-\epsilon)^{t+1}\, r_{t+1}(y)
\end{aligned}$$

where:

$$r_{t+1}(y) = \left[\sum_x r_t(x)\, \frac{T(x,y)}{(1-\epsilon)}\right] - \frac{\epsilon}{1-\epsilon}\, \pi_{stationary}(y)$$

One can easily check that: $r_{t+1}(y) \geq 0,\ \forall\, y \in \mathcal{X}$ and that: $\sum_y r_{t+1}(y) = 1$. That is, r_{t+1} is a proper *pmf* on $\mathcal{X}$.

With Eq 45 we can now show that Eq 43 holds:

$$\begin{aligned}
\left|\pi_{stationary}(x) - \pi_{X^{(t)}}(x)\right| &= \left|\pi_{stationary}(x) - \left[1-(1-\epsilon)^t\right] \pi_{stationary}(x) - (1-\epsilon)^t\, r_t(x)\right| \\
&= \left|(1-\epsilon)^t\, \pi_{stationary}(x) - (1-\epsilon)^t\, r_t(x)\right| \\
&= (1-\epsilon)^t\, \left|\pi_{stationary}(x) - r_t(x)\right| \\
&\leq (1-\epsilon)^t
\end{aligned}$$

Ergodic theorem

Let $\left(X^{(0)}, X^{(1)}, \ldots\right)$ be an irreducible and aperiodic homogeneous Markov chain on a finite state space $\mathcal{X} = \{x_1, x_2, \ldots, x_\nu\}$ with a transition matrix $T(x,y)$ and a stationary distribution $\pi_{stationary}$. Let a be a real valued function defined on $\mathcal{X}$ and let $\overline{a}_N$ be the empirical average of a computed from a realization $\left(x^{(0)}, x^{(1)}, \ldots\right)$ of $\left(X^{(0)}, X^{(1)}, \ldots\right)$, that is: $\overline{a}_N = \frac{1}{N}\sum_{t=0}^{N} a\left(x^{(t)}\right)$. Then we have:

$$\lim_{N\to\infty} \langle \overline{a}_N \rangle = \langle a \rangle_{\pi_{stationary}}\,,$$

where:

$$\langle a \rangle_{\pi_{stationary}} = \sum_{x\,\in\,\mathcal{X}} \pi_{stationary}(x)\ a(x)\,,$$

and:

$$\langle \overline{a}_N \rangle = \frac{1}{N} \sum_{t=0}^{N} \sum_{x\,\in\,\mathcal{X}} \pi_{X^{(t)}}(x)\ a(x)\,.$$

Proof

Using Eq. 45 in the proof of the fundamental theorem we can write:

$$\begin{aligned}
\langle \overline{a}_N \rangle &= \frac{1}{N} \sum_{t=0}^{N} \sum_{x \in \mathcal{X}} \pi_{X^{(t)}}(x)\, a(x) \\
&= \sum_{x \in \mathcal{X}} a(x)\, \frac{1}{N} \sum_{t=0}^{N} \pi_{X^{(t)}}(x) \\
&= \sum_{x \in \mathcal{X}} a(x) \left[\pi_{stationary}(x) \frac{1}{N} \sum_{t=0}^{N} \left(1 - \zeta^t\right) + \frac{1}{N} \sum_{t=0}^{N} \zeta^t r_t(x) \right] \\
&= \left[\sum_{x \in \mathcal{X}} \pi_{stationary}(x)\, a(x) \right] + \left[\sum_{x \in \mathcal{X}} a(x) \left(\frac{1}{N} \sum_{t=0}^{N} \zeta^t \left(r_t(x) - \pi_{stationary}(x) \right) \right) \right]
\end{aligned}$$

where ζ corresponds to $1 - \epsilon$ in Eq. 45 and therefore: $0 \leq \zeta < 1$. To prove the ergodic theorem we just need to show that the second term of the right hand side of the above equation converges to 0 as $N \to \infty$. For that it is enough to show that its modulus goes to 0:

$$\begin{aligned}
\left| \sum_{x \in \mathcal{X}} a(x) \left(\frac{1}{N} \sum_{t=0}^{N} \zeta^t \left(r_t(x) - \pi_{stationary}(x) \right) \right) \right| &\leq \sum_{x \in \mathcal{X}} |a(x)| \left(\frac{1}{N} \sum_{t=0}^{N} \zeta^t \left| r_t(x) - \pi_{stationary}(x) \right| \right) \\
&\leq \sum_{x \in \mathcal{X}} |a(x)| \frac{1}{N} \sum_{t=0}^{N} \zeta^t
\end{aligned}$$

We just need to recognize here the geometrical series $\sum_{t=0}^{N} \zeta^t$ which converges to $\frac{1}{1-\zeta}$ when $N \to \infty$. That completes the proof.

What did we learn in this section? For a given *pmf* π defined on a finite state space $\mathcal{X}$ if we can find an irreducible aperiodic Markov matrix T which admits π as its (necessarily only) stationary distribution, then *regardless* of our initial choice of *pmf* π_0 defined on $\mathcal{X}$ and regardless of the function $a : \mathcal{X} \to \Re$, an *asymptotically unbiased* estimate of $\langle a \rangle = \sum_{x \in \mathcal{X}} a(x)\, \pi(x)$ can be obtained by simulating the Markov chain to get a sequence of states: $\left(x^{(0)}, x^{(1)}, \ldots, x^{(N)}\right)$ and computing the empirical average: $\overline{a} = \frac{1}{N} \sum_{t=0}^{N} a\left(x^{(t)}\right)$.

7 The Metropolis - Hastings algorithm and its relatives

What we need now is a way to construct the transition matrix T and more specifically a method which works with an *unnormalized* version of the stationary density. We first introduce the notion of *detailed balance*.

Detailed balance definition We say that a *pmf* π defined on a (finite) state space $\mathcal{X}$ and a Markov transition matrix T satisfy the *detailed balance* if: $\forall x, y \in \mathcal{X},\ \pi(x)\, T(x, y) = \pi(y)\, T(y, x)$.

Detailed balance theorem If the *pmf* π defined on a (finite) state space $\mathcal{X}$ and a Markov transition matrix T satisfy the *detailed balance* then π is stationary for T.

Exercise 5

Prove the theorem. See Sec. 13.4 for solution.

7.1 The Metropolis - Hastings algorithm

7.1.1 Second fundamental theorem

Let π be a *pmf* defined on a (finite) state space $\mathcal{X}$ and T and G two Markov transition matrices defined on $\mathcal{X}$ satisfying:

$$\begin{aligned} T(x,y) &= A(x,y)\,G(x,y) \;\; if\; x \neq y \\ T(x,x) &= 1 - \sum_{y \in \mathcal{X}, y \neq x} T(x,y) \end{aligned}$$

where $G(y,x) = 0$ if $G(x,y) = 0$ and

$$\begin{aligned} A(x,y) &= \min\left(1, \frac{\pi(y)\,G(y,x)}{\pi(x)\,G(x,y)}\right), \; if\; G(x,y) > 0 \\ &= 1, \quad otherwise \end{aligned} \tag{46}$$

then

- π and T satisfy the detailed balance condition
- $\pi T = \pi$
- if G is irreducible and aperiodic so is T

Exercise 6

Prove the theorem.

THE MOST IMPORTANT COMMENT OF THESE LECTURES NOTES

> This theorem is exactly what we were looking for. It tells us how to modify an irreducible and aperiodic Markov transition (G) such that a *pmf* π of our choice will be the stationary distribution and it does that by requiring a knowledge of the desired stationary π only up to a normalizing constant, because Eq. 46 involves the ratio of two values of π.

G is often called the *proposal* transition and A the *acceptance* probability. The *Metropolis algorithm* [24] (sometimes called the $M(RT)^2$ algorithm because of its authors names) is obtained with a symmetrical G (*i.e.*, $G(x,y) = G(y,x)$). The above formulation (in fact a more general one) is due to Hastings [14]. The set of techniques which involves the construction of a Markov chain (with the Metropolis - Hastings algorithm) to perform Monte Carlo integration is called *Dynamic Monte Carlo* by physicists [35, 29, 19] and *Markov Chain Monte Carlo* (*MCMC*) by statisticians [10, 33, 23].

7.2 Metropolis - Hastings and Gibbs algorithms for multi - dimensional spaces

Talking of multi - dimensional spaces when we started by saying we were working with a discrete and finite one can seem a bit strange. It is nevertheless useful not to say necessary to keep a trace of the multi - dimensionality of our "initial" space (ideally all our model parameters: $\mathbf{P}_k, \Delta_k, \Lambda_k, S_k, F_k$ "live" in continuous spaces). If not it would be extremely hard to find our way on the map between the discrete approximations of all these continuous spaces and the genuine discrete and finite space on which our simulated Markov chain evolves.

We consider therefore now that our random variables are "vectors" and $X^{(t)}$ becomes $\mathbf{X}^{(t)}$. We can think of it as follows: $\mathbf{X}_1^{(t)}$ corresponds to $P_{1,1}^{(t)}$ the maximal peak amplitude of the first neuron in the model on the first recording site after t steps,..., $\mathbf{X}_4^{(t)}$corresponds to $P_{1,4}^{(t)}$ the maximal peak amplitude of the first neuron in the model on the fourth recording site (assuming we are using tetrodes), $\mathbf{X}_5^{(t)}$ corresponds to the parameter $\Delta^{(t)}$ of the first neuron, $\mathbf{X}_6^{(t)}$ corresponds to the parameter $\Lambda^{(t)}$ of the first neuron, $\mathbf{X}_7^{(t)}$

corresponds to the parameter $S^{(t)}$ of the first neuron, $\mathbf{X}_8^{(t)}$ corresponds the parameter $F^{(t)}$ of the first neuron, $\mathbf{X}_9^{(t)}$ corresponds to $P_{2,1}^{(t)}$ the maximal peak amplitude of the second neuron in the model on the first recording site, etc. The problem is that it turns out to be very hard to find a transition matrix G acting on object like these "new" *pmf*s: $\pi_{\mathbf{X}^{(t)}}$ such that the acceptance probabilities (Eq. 46) are not negligible. The way around this difficulty is to build the transition matrix T as a combination of component - wise transition matrices like T_1 which acts only on the first component of $\pi_{\mathbf{X}^{(t)}}$, T_2 which acts only on the second, etc. We just need to make sure that our combination is irreducible, and aperiodic (we assume here that we build each individual T_j such that $\pi_{posterior}$ is its stationary *pmf*). A way to do that is to construct each T_j such that it is irreducible and aperiodic on its "own" sub - space which is obtained practically by building the matrix such that it has a strictly positive probability to produce a move anywhere on its sub - space. Then two combinations of these T_js are typically used, the first one being:

$$T = w_1 T_1 + w_2 T_2 + \ldots + w_m T_m,$$

where m is the number of components of the random vectors and were the w_js are components of a *pmf* defined on the the set of coordinates $\{1, \ldots, m\}$. It should be clear that if each T_j is irreducible and aperiodic on its own sub - space, then T will be irreducible and aperiodic on the "complete" state space. Because each T_j is built such that $\pi_{posterior}$ is its stationary *pmf*, $\pi_{posterior}$ will be the stationary *pmf* of T as well. The concrete implementation of this scheme would go as follows: at each "sub - step" a coordinate j is randomly drawn from w, then a random move is performed using T_j. It is customary to call "Monte Carlo step" a sequence of m successive such "sub - steps" (that means that on average each model parameter will be "changed" during a Monte Carlo step).

The second combination is:

$$T = T_1 \times T_2 \times \ldots \times T_m, \tag{47}$$

that is, each model parameter is successively "changed". In the same way as above the irreducibility and aperiodicity of the T_js in their sub - spaces will give rise to an irreducible and aperiodic T on the "full" parameter space. The main difference is that detailed balance condition which be imposed by construction to the pairs T_j, $\pi_{posterior}$ is not kept by the pair T, $\pi_{posterior}$. We only have the stationarity property (which is enough to ensure convergence). Of course variations on that scheme can be used like using random permutations of the T_js (which would restore the detailed balance condition for the pair T, $\pi_{posterior}$). A "Monte Carlo step" for those schemes is obtained after the application of the complete series of T_js. See [7] for details.

7.2.1 An example: the Gibbs sampler for the parameters of the *ISI* density

The previous discussion seemed probably a bit abstract for most of the readers. In order to be more precise about what we meant we will start by considering the following "simple" situation. Let's assume that we are given a sample of 25 *isi*s: $\mathcal{D} = \{i_1, \ldots, i_{25}\}$ drawn independently from a log-Normal density with parameters: s_{actual}, f_{actual}. We are asked for a Bayesian estimation of values of s and f (assuming flat priors for these parameters like in Exercise 3):

$$\pi_{isi,posterior}\left(s, f \mid \mathcal{D}\right) \alpha\, L_{isi}\left(\mathcal{D} \mid s, f\right) \cdot \pi_{isi,prior}\left(s, f\right),$$

where L_{isi} is easily obtained from Eq. 10:

$$L_{isi}\left(\mathcal{D} \mid s, f\right) = \frac{1}{f^N} \exp\left[-\frac{1}{2}\frac{1}{f^2}\sum_{j=1}^{N}\left(\ln i_j - \ln s\right)^2\right] \tag{48}$$

We do not recognize any classical *pdf* in Eq. 48 and we choose to use a MCMC approach. Following the previous discussion we will try to build our transition matrix T as a "product" $T_S \times T_F$. Where T_S does only change s and T_F does only change f and where both are irreducible and aperiodic on their own sub-space. According to the second fundamental theorem we first need to find proposal transitions: $G_s\left(s_{now}, s_{proposed} \mid f, \mathcal{D}\right)$ and $G_f\left(f_{now}, f_{proposed} \mid s, \mathcal{D}\right)$. But questions 1 and 3 of Exercise 3 provide us with such proposal transitions. In fact these transitions are a bit special because they do not depend on the present value of the parameter we will try to change and because they indeed correspond to the posterior conditional density of this parameter. A direct consequence of the latter fact is that the acceptance probability (Eq. 46) is 1. An algorithm where the proposal transition for a parameter is the posterior conditional of this parameter is called a *Gibbs sampler* by statisticians and a *heat bath algorithm* by physicists.

We therefore end up with the following algorithm (using the results of Exercise 3):

1. Chose randomly $s^{(0)} \in [s_{min}, s_{max}]$ and $f^{(0)} \in [f_{min}, f_{max}]$.
2. Given $f^{(t)}$ draw:
$$s^{(t+1)} \sim \pi\left(\quad | \, f^{(t)}, \mathcal{D}\right)$$
where $\pi\left(\quad | \, f^{(t)}, \mathcal{D}\right)$ is defined by Eq. 19 (remark that $s^{(t+1)}$ is independent of $s^{(t)}$).
3. Given $s^{(t+1)}$ draw:
$$f^{(t+1)} \sim \pi\left(\quad | \, s^{(t+1)}, \mathcal{D}\right)$$
where $\pi\left(\quad | \, s^{(t+1)}, \mathcal{D}\right)$ is defined by Eq. 21.

Exercise 7

Simulate 25 *isi* following a log-Normal density with values of your choice for the pair s_{actual}, f_{actual}. Implement the algorithm and compare its output with the maximum likelihood based inference. That is with the maximal likelihood estimates for s and f (given by Eq. 12 & 13). You should compute as well the Hessian of the log-likelihood function at its maximum to get confidence intervals (see the chapter of E. Brown).

7.2.2 Generation of the amplitude parameters of our model

By "amplitude parameters" we mean here the following parameters: $\mathbf{P}, \Delta, \Lambda$. Given a data set from a single neuron: $\mathcal{D} = \{(i_1, \mathbf{a}_1), \ldots, (i_N, \mathbf{a}_N)\}$ (see Sec. 5.2) we now try to perform Bayesian inference on all its parameters: $\mathbf{P}, \Delta, \Lambda, S, F$. We again split our transition matrix T into parameter specific transitions: $T = T_{P_1} \times T_{P_2} \times T_{P_3} \times T_{P_4} \times T_\Delta \times T_\Lambda \times T_S \times T_F$. We have seen in the previous example how to get T_S and T_F. Following the same line we could try to build a Gibbs sampler for the other parameters as well. The problem is that the part of the Likelihood function which depends on the amplitude parameters (obtained from Eq. 11):
$$L_{amp}\left(\mathcal{D} \mid \mathbf{p}, \delta, \lambda\right) = \exp\left[-\frac{1}{2}\sum_{j=1}^{N} \left\|\mathbf{a}_j - \mathbf{p}\cdot\left(1 - \delta\cdot\exp\left(-\lambda\cdot i_j\right)\right)\right\|^2\right]$$
does not correspond to any know *pdf* even when considered as a function of a single parameter, say δ. The reader can notice that such would not be the case if we had $\delta = 0$ and if we knew it, see [30]. A robust solution to this problem is to use a piece-wise linear approximation of the posterior conditional as a proposal transition for each parameter (*e.g.*, $G_\Delta\left(\quad | \, \mathbf{p}, \lambda, s, f, \mathcal{D}\right)$) and then an acceptance probability as defined by Eq. 46. More specifically we can start with 101 regularly spaced "sampling values" of δ:
$$\delta \in \{\delta_0 = 0, \delta_1 = 0.01, \ldots, \delta_{99} = 0.99, \delta_{100} = 1\},$$
compute 101 values of:
$$L_{amp}\left(\mathcal{D} \mid \mathbf{p}, \delta_i, \lambda\right) \tag{49}$$
and define:
$$G_\Delta\left(\delta \mid \mathbf{p}, \lambda, s, f, \mathcal{D}\right) = \mathcal{N}_\Delta \cdot \left[L_{amp}\left(\mathcal{D} \mid \mathbf{p}, \delta_i, \lambda\right) + \frac{L_{amp}\left(\mathcal{D} \mid \mathbf{p}, \delta_{i+1}, \lambda\right) - L_{amp}\left(\mathcal{D} \mid \mathbf{p}, \delta_i, \lambda\right)}{\delta_{i+1} - \delta_i}\left(\delta - \delta_i\right)\right]$$
where $\mathcal{N}_\Delta$ ensures that G_Δ is properly normalized and where $\delta \in [\delta_i, \delta_{i+1}]$. The obvious problem with this approach is that we need to have a reasonably good piece-wise linear approximation of the corresponding posterior conditional *pdf* in order to get reasonable values for our acceptance probability. That means that when we start our Markov chain and we do not have a precise idea of the actual shape of this posterior conditional density we have to use a lot of sampling points. We spend therefore a lot of time computing terms like Eq. 49. A very significant increase of the algorithm speed can thus be obtained by using a first run with a lot of sampling points (say 101), then reposition fewer sampling point (say 13)[11] using

[11] We typically use 13 points because we want to have the 10 quantiles, the 2 extrema (here, δ_{min} and δ_{max}) allowed by our priors and an extra point corresponding to the smallest value obtained in our sample. The reader can "show" as an exercise the usefulness of this extra sample point. The easiest way to see the use of this point is to try without it and to remember that the piece-wise linear approximation *must be normalized*.

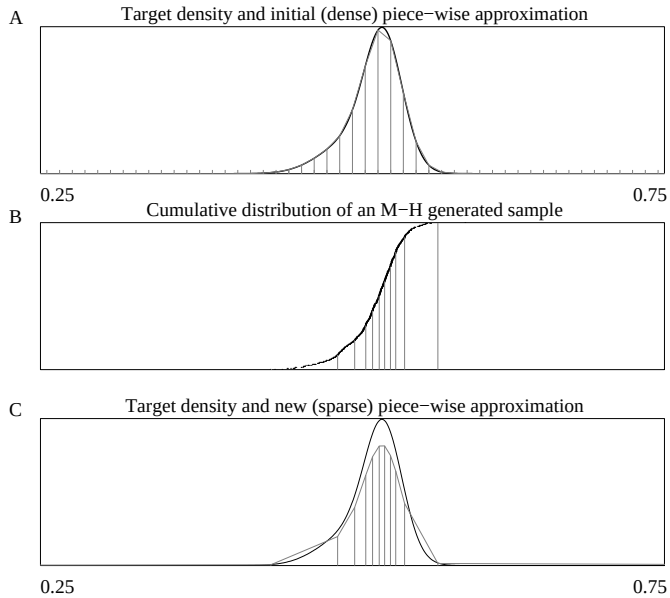

Figure 10: A, a simple one-dimensional target density (black curve, a Gaussian mixture: $0.3\mathcal{N}\left(0.5, 0.025^2\right) + 0.7\mathcal{N}\left(0.525, 0.015^2\right)$) together with its linear piece-wise approximation (gray curve) based on 100 discrete samples (the prior density is supposed to be flat between 0 and 1 and zero everywhere else). B, A sample of size 1000 is drawn using the piece-wise linear approximation as a proposal and a MH acceptance rule. The cumulative distribution of the sample is shown (black) together with the location of the 10 quantiles. C, Using the location of the 10 quantiles, the boundaries of the prior and the location of the smallest generated value, a sparser piece-wise linear approximation of the target density is built.

the output of the first run. That's explained in details in [32] and illustrated on Fig. 10. When dealing with multiple neurons data, Eq. 25 shows that the same approach can be immediately applied after the introduction of the configuration.

7.2.3 Generation of the configuration

This can be done (almost) exactly as for a "normal" Potts model [29, 35] and is left as an exercise to the reader (the answer can be found in [32]).

7.2.4 Our complete MC step

In Sec. 11 we will give an example of our algorithm at work with the following sequential MC step:

$$T_{l_1} \times \ldots \times T_{l_N} \times T_{P_{1,1}} \times \ldots \times T_{f_1} \times \ldots \times T_{P_{K,1}} \times \ldots \times T_{f_K} \tag{50}$$

8 Priors choice

We will assume here that we know "little" *a priori* about our parameters values and that the joint prior density $\pi_{prior}(\theta)$ can be written as a product of the densities for each component of θ, that is:

$$\pi_{prior}(\theta) = \prod_{q=1}^{K} \pi(f_q) \cdot \pi(s_q) \cdot \pi(\delta_q) \cdot \pi(\lambda_q) \cdot \pi(P_{q,1}) \cdot \pi(P_{q,2}) \cdot \pi(P_{q,3}) \cdot \pi(P_{q,4})$$

where we are assuming that four recording sites have been used. We will further assume that our signal to noise ratio is not better 20 (a rather optimistic value), that our spikes are positive and therefore the $\pi(P_{q,1\ldots4})$ are null below 0 and above +20 (remember we are working with normalized amplitudes). We

will reflect our absence of prior knowledge about the amplitudes by taking a uniform distribution between 0 and +20. The λ value reported by Fee et al [6] is 45.5 s^{-1}. λ must, moreover be smaller than ∞, we adopt a prior density uniform between 10 and 200 s^{-1}. δ must be ≤ 1 (the amplitude of a spike from a given neuron on a given recording site *does not* change sign) and ≥ 0 (spikes do not become larger upon short ISI), we therefore use a uniform density between 0 and 1 for δ. An inspection of the effect of the shape parameter f on the ISI density is enough to convince an experienced neurophysiologist that empirical unimodal ISI densities from well isolated neurons will have $f \in [0.1, 2]$. We therefore take a prior density uniform between 0.1 and 2 for f. The same considerations leads us to take a uniform prior density between 0.005 and 0.5 for s.

9 The good use of the ergodic theorem. A warning.

The ergodic theorem is the key theoretical result justifying the use of Monte Carlo integration to solve tough problems. When using these methods we should nevertheless be aware that the theorem applies only when the number of Monte Carlo steps of our algorithms go to infinity[12] and because such is never practically the case we will commit errors. We would commit errors even if our draws where directly generated from π_{post}. The difference between the MCMC/Dynamic MC based estimates and estimates based on direct samples ("plain MC") is that the variance of the estimators of the former have to be corrected to take into account the correlation of the states generated by the Markov chain. We explain now how to do that in practice, for a theoretical justification see Sokal [35] and Janke [17].

9.1 Autocorrelation functions and confidence intervals

We have to compute for each parameter, θ_i, of the model the normalized autocorrelation function (ACF), $\rho_{norm}(l;\theta_i)$, defined by:

$$\begin{aligned} \rho(l;\theta_i) &= \frac{1}{N_T - N_D - l} \cdot \sum_{t=N_D}^{N_T - l} (\theta_i^{(t)} - \bar{\theta}_i) \cdot (\theta_i^{(t+l)} - \bar{\theta}_i) \\ \rho_{norm}(l;\theta_i) &= \frac{\rho(l;\theta_i)}{\rho(0;\theta_i)} \end{aligned} \tag{51}$$

Where N_T is the total number of MC steps performed and N_D is the number of steps required to reach "equilibrium" (see Sec. 9.2, 11.2 & 11.3). Then we compute the *integrated autocorrelation time*, $\tau_{autoco}(\theta_i)$:

$$\tau_{autoco}(\theta_i) = \frac{1}{2} + \sum_{l=1}^{L} \rho(l;\theta_i) \tag{52}$$

where L is the lag at which ρ starts oscillating around 0. Using an empirical variance, $\sigma^2(\theta_i)$ of parameter θ_i, defined in the usual way:

$$\sigma^2(\theta_i) = \frac{1}{N_T - N_D - 1} \sum_{t=N_D}^{N_T} (\theta_i^{(t)} - \overline{\theta}_i)^2 \tag{53}$$

where $\overline{\theta}_i$ is defined like in Eq. 34. Our estimate of the variance, $Var\left[\overline{\theta}_i\right]$ of $\overline{\theta}_i$ becomes:

$$Var\left[\overline{\theta}_i\right] = \frac{2\,\tau_{autoco}(\theta_i)}{N_T - N_D - 1} \cdot \sigma^2(\theta_i) \tag{54}$$

In the sequel, the confidence intervals on our parameters estimates are given by the square root of the above defined variance (Table 2 & 3).

We can view the effect of the autocorrelation of the states of the chain as a reduction of our effective sample size by a factor: $2\,\tau_{autoco}(\theta_i)$. This gives us a first quantitative element on which different algorithms can be compared (remember that the MH algorithm gives us a lot of freedom on the choice of proposal transition kernels). It is clear that the faster the autocorrelation functions of the parameters fall to zero, the greater the statistical efficiency of the algorithm. The other quantitative element we want to consider

[12] This is not even true because our algorithm use *pseudo - random - number* generators which among many shortcomings have a *finite* period. See Chap 7 of [7], Chap 2 of [33] and [20].

is the computational time, τ_{cpu}, required to perform one MC step of the algorithm. One could for instance imagine that a new sophisticated proposal transition kernel allows us to reduce the largest τ_{autoco} of our standard algorithm by a factor of 10, but at the expense of an increase of τ_{cpu} by a factor of 100. Globally the new algorithm would be 10 times less efficient than the original one. What we want to keep as small as possible is therefore the product: $\tau_{autoco} \cdot \tau_{cpu}$.

9.2 Initialization bias

The second source of error in our (finite size) MC estimates is a bias induced by the state $(\theta^{(0)}, C^{(0)})$ with which the chain is initialized [35, 7]. The bad news concerning this source of error is that there is no general theoretical result providing guidance on the way to handle it, but the booming activity in the Markov chain field already produced encouraging results in particular cases [23]. The common wisdom in the field is to monitor parameters (and labels) evolution, and/or functions of them like the energy (Eq. 30). Based on examination of evolution plots (*eg*, Fig. 12 & 13) and/or on application of time-series analysis tools, the user will decide that "equilibrium" has been reached and discard the parameters values before equilibrium. More sophisticated tests do exist [33] but they wont be used in this chapter. These first two sources of error, finite sample size and initialization bias, are clearly common to all MCMC approaches.

10 Slow relaxation and the Replica Exchange Method

A third source of error appears only when the energy function exhibits several local minima. In the latter case, the Markov chain realization can get trapped in a local minimum which could be a poor representation of the whole energy function. This sensitivity to local minima arises from the local nature of the transitions generated by the MH algorithm. That is, if we use a sequential scheme like Eq. 50, at each MC time step, we first attempt to change the label of spike 1, then the one of spike 2, ..., then the one of spike N, then we try to change the first component of θ ($P_{1,1}$), and so on until the last component (σ_K). That implies that if we start in a local minimum and if we need to change, say, the labels of 10 spikes to reach another lower local minimum, we could have a situation in which the first 3 label changes are energetically unfavorable (giving, for instance, an acceptance probability, Eq. 46, of 0.1 per label change) which would make the probability to accept the succession of changes very low (0.1^3)... meaning that our Markov chain would spend a long time in the initial local minimum before "escaping" to the neighboring one. Stated more quantitatively, the average time the chain will take to escape from a local minimum with energy E_{min} grows as the exponential of the energy difference between the energy, E^*, of the highest energy state the chain has to go through to escape and E_{min}:

$$\tau_{escape} \, \alpha \, \exp\left[\beta \left(E^* - E_{min}\right)\right]$$

Our chains will therefore exhibit an Arrhenius behavior. To sample more efficiently such state spaces, the Replica Exchange Method (REM) [15, 26], also known as the Parallel Tempering Method [12, 37, 5], considers R replicas of the system with an increasing sequence of temperatures (or a decreasing sequence of β) and a dynamic defined by two types of transitions : usual MH transitions performed independently on each replica according to the rule defined by Eq. 50 and a replica exchange transition. The key idea is that the high temperature (low β) replicas will be able to easily cross the energy barriers separating local minima (in the example above, if we had a probability of 0.1^3 to accept a sequence of labels switch for the replica at $\beta = 1$, the replica at $\beta = 0.2$ will have a probability $0.1^{3 \cdot 0.2} \approx 0.25$ to accept the same sequence). What is needed is a way to generate replica exchange transitions such that the replica at β = 1 generates a sample from π_{post} defined by Eq. 31. Formally the REM consists in simulating, on an "extended ensemble", a Markov chain whose unique stationary density is given by:

$$\pi_{ee}\left(\theta_1, C_1, \ldots, \theta_R, C_R\right) = \pi_{post,\beta_1}\left(\theta_1, C_1\right) \ldots \pi_{post,\beta_R}\left(\theta_R, C_R\right) \tag{55}$$

where "*ee*" in π_{ee} stands for "extended ensemble" [16], R is the number of simulated replicas, $\beta_1 > \ldots > \beta_R$ for convenience and:

$$\pi_{post,\beta_i}\left(\theta_i, C_i\right) = \frac{\exp\left[-\beta_i E\left(\theta_i, C_i\right)\right]}{Z\left(\beta_i\right)} \tag{56}$$

That is, compared to Eq. 31, we now explicitly allow β to be different from 1. To construct our "complete" transition kernel we apply our previous procedure. That is, we construct it as a sequence of parameter,

label and inter-replica specific MH transitions. We already know how to get the parameter and label specific transitions for each replica. What we really need is a transition to exchange replicas, say the replicas at inverse temperature β_i and β_{i+1}, such that the detailed balance is preserved (Sec. 7):

$$\begin{aligned}
&\pi_{ee}\left(\theta_1, C_1, \ldots, \theta_i, C_i, \theta_{i+1}, C_{i+1}, \ldots, \theta_R, C_R\right) T_{i,i+1}\left(\theta_{i+1}, C_{i+1}, \theta_i, C_i \mid \theta_i, C_i, \theta_{i+1}, C_{i+1}\right) &= \\
&\pi_{ee}\left(\theta_1, C_1, \ldots, \theta_{i+1}, C_{i+1}, \theta_i, C_i, \ldots, \theta_R, C_R\right) T_{i,i+1}\left(\theta_i, C_i, \theta_{i+1}, C_{i+1} \mid \theta_{i+1}, C_{i+1}, \theta_i, C_i\right)
\end{aligned}$$

which leads to:

$$\begin{aligned}
\frac{T_{i,i+1}\left(\theta_i, C_i, \theta_{i+1}, C_{i+1} \mid \theta_{i+1}, C_{i+1}, \theta_i, C_i\right)}{T_{i,i+1}\left(\theta_{i+1}, C_{i+1}, \theta_i, C_i \mid \theta_i, C_i, \theta_{i+1}, C_{i+1}\right)} &= \frac{\pi_{ee}\left(\theta_1, C_1, \ldots, \theta_i, C_i, \theta_{i+1}, C_{i+1}, \ldots, \theta_R, C_R\right)}{\pi_{ee}\left(\theta_1, C_1, \ldots, \theta_{i+1}, C_{i+1}, \theta_i, C_i, \ldots, \theta_R, C_R\right)} \\
&= \frac{\pi_{post,\beta_i}\left(\theta_i, C_i\right) \cdot \pi_{post,\beta_{i+1}}\left(\theta_{i+1}, C_{i+1}\right)}{\pi_{post,\beta_i}\left(\theta_{i+1}, C_{i+1}\right) \cdot \pi_{post,\beta_{i+1}}\left(\theta_i, C_i\right)} \\
&= \exp\left[-\left(\beta_i - \beta_{i+1}\right) \cdot \left(E\left(\theta_i, C_i\right) - E\left(\theta_{i+1}, C_{i+1}\right)\right)\right]
\end{aligned}$$

Again we write $T_{i,i+1}$ as a product of a proposal transition kernel and an acceptance probability. Here we have already explicitly chosen a deterministic proposal (we only propose transitions between replicas at neighboring inverse temperatures) which gives us:

$$\frac{A_{i,i+1}\left(\theta_i, C_i, \theta_{i+1}, C_{i+1} \mid \theta_{i+1}, C_{i+1}, \theta_i, C_i\right)}{A_{i,i+1}\left(\theta_{i+1}, C_{i+1}, \theta_i, C_i \mid \theta_i, C_i, \theta_{i+1}, C_{i+1}\right)} = \exp\left[-\left(\beta_i - \beta_{i+1}\right) \cdot \left(E\left(\theta_i, C_i\right) - E\left(\theta_{i+1}, C_{i+1}\right)\right)\right]$$

It is therefore enough to take:

$$A_{i,i+1}\left(\theta_i, C_i, \theta_{i+1}, C_{i+1} \mid \theta_{i+1}, C_{i+1}, \theta_i, C_i\right) = \min\left\{1, \exp\left[-\left(\beta_i - \beta_{i+1}\right) \cdot \left(E\left(\theta_i, C_i\right) - E\left(\theta_{i+1}, C_{i+1}\right)\right)\right]\right\} \quad (57)$$

The reader sees that if the state of the "hot" replica (β_{i+1}) has a lower energy ($E\left(\theta_i, C_i\right)$) than the "cold" one, the proposed exchange is always accepted. The exchange can pictorially be seen as cooling down the hot replica and warming up the cold one. Fundamentally this process amounts to make the replica which is at the beginning *and* at the end of the replica exchange transition at the cold temperature to jump from one local minimum (θ_{i+1}, C_{i+1}) to another one (θ_i, C_i). That is precisely what we were looking for. The fact that we can as well accept unfavorable exchanges (*i.e.*, raising the energy of the "cold" replica and decreasing the one of the "hot" replica) is the price we have to pay for our algorithm to generate samples from the proper posterior (we are not doing optimization here).

In order for the replica exchange transition to work well we need to be careful with our choice of inverse temperatures. The typical energy of a replica (*i.e.*, its expected energy) increases when β decreases (Fig. 16A). We will therefore typically have a positive energy difference: $\Delta E = E_{hot} - E_{cold} > 0$ between the replicas at low and high β before the exchange. That implies that the acceptance ratio (Eq. 57) for the replica exchange will be typically smaller than 1. Obviously, if it becomes too small, exchanges will practically never be accepted. To avoid this situation we need to choose our inverse temperatures such that the typical product: $\Delta\beta \cdot \Delta E$, where $\Delta\beta = \beta_{cold} - \beta_{hot}$, is close enough to zero [28, 15, 16]. In practice we used pre-runs with an a priori too large number of βs, checked the resulting energy histograms and kept enough inverse temperatures to have some overlap between successive histograms (Fig. 16B).

In Sec. 11 we will perform replica exchange transitions between each pair β_i, β_{i+1} with an even, respectively odd, i at the end of each even, respectively odd, MC step. With this replica exchange scheme, each MC time step will therefore be composed of a complete parameter and label transition for each replica, followed by a replica exchange transition. This scheme corresponds to the one described by Hukushima and Nemoto [15]. A rather detailed account of the REM can be found in Mitsutake et al [26]. Variations on this scheme do exist [28, 4].

11 An Example from a simulated data set

We will illustrate the performances of the algorithm with a simple simulated data set. The data are supposed to come from 3 neurons which are exactly described by our underlying model. Such an illustration has in our opinion several advantages. Being simple it helps the reader to concentrate on the inner working of the algorithm. Because the data correspond to the underlying model hypothesis, our implementation of the MCMC method should give us back the parameters used to simulate the data, we are therefore performing here a simple and *necessary* test of our code. The data do moreover exhibit features (strong cluster overlap, Fig. 11B,C) which would make them unusable by other algorithms. A presentation of the algorithm performances with a much worse data set can be found in [32].

	neuron 1	neuron 2	neuron 3
P_1, P_2	15,9	8,8	6,12
δ	0.7	0.8	0.6
λ	33.33	40	50
s	25	30	18
f	0.5	0.4	1.0
$\langle isi \rangle$	28.3	32.5	29.7
#	1060	923	983

Table 1: Parameters used to simulate the neurons. The maximal peak amplitude values (P_i) are given in units of noise SD. The scale parameters (s) and mean *isi* ($\langle isi \rangle$) are given in ms. The bottom row indicates the number of events from each neuron. The correspondence between neuron number and color on Fig. 11: 1, green, 2, red, 3, blue.

11.1 Data properties

The parameters used to simulate the three neurons are given in Table 1. 30 seconds of data were simulated giving a total of 2966 spikes. The raw data, spike amplitude *vs* time on the two recording sites are illustrated on Fig. 11A1 & 11A2. Fig. 11B is a "Wilson plot" of the entire sample. Notice the *strong* overlap of points (spikes) arising from the 3 different neurons. Fig. 11C shows the theoretical iso-density contours for the clusters generated by each of the 3 neurons. Neuron 1 in Table 1 is green on the figure, neuron 2 is red and neuron 3 is blue. The reader can see that roughly 50% of the spikes generated by neuron 2 should fall in regions were neuron 1 or neuron 3 will also generate spikes. The theoretical densities associated with each neuron (cluster) *are not* 2-dimensional Gaussian. This can be most clearly seen for neuron 3 (blue iso-density contours) which has a "flat" summit. None of these densities is symmetrical with respect to its maximum along its principal axis (which is the axis going through the graph's origin and the neuron density maximum). Fig. 11D1 represents the amplitude dynamics of each of the three neurons, while Fig. 11D2 displays their respective ISI densities.

11.2 Algorithm dynamics and parameters estimates without the REM

11.2.1 Initialization

The algorithm being iterative we have to start it somewhere and we will use here a somewhat "brute force" initialization. We choose randomly with a uniform probability $\frac{1}{N}$ as many actual events as neurons in the model (K=3). That gives us our initial guesses for the $P_{q,i}$. δ is set to $\delta_{min} = 0$ for each neuron. All the other parameters are randomly drawn from their prior distribution (Sec. 8). The initial configuration is generated by labeling each individual spike with one of the K possible labels with a probability $1/K$ for each label (this is the $\beta = 0$ initial condition used in statistical physics).

11.2.2 Energy evolution

At each MC step, a new label is proposed and accepted or rejected for each spike and a new value is proposed and accepted or rejected for each of the (18) model parameters (Eq. 50). Our initial state is very likely to have a very low posterior probability. We therefore expect our system to take some time to relax from the highly disordered state in which we have initialized it to the (hopefully) ordered typical states. As explained in Sec. 9.2, we can (must) monitor the evolution of "system features". The most reliable we have found until now is the system's energy as shown on Fig. 12. Here one sees an "early" evolution (first 60000 MC steps) followed by a much slower one which *looks like* a stationary regime during the last 100000 steps[13].

[13]The time required to perform such simulations depends on the number of sampling points used for the piece-wise linear proposals of the amplitude parameters (Sec. 7.2.2). During the first 2000 steps we use 101 regularly spaced sampling points. We then need less than 5' to perform 1000 steps. After that, we reposition the sampling points and use only 13 of them and the time required to perform 1000 steps falls to 50 s (Pentium IV computer at 3.06 GHz).

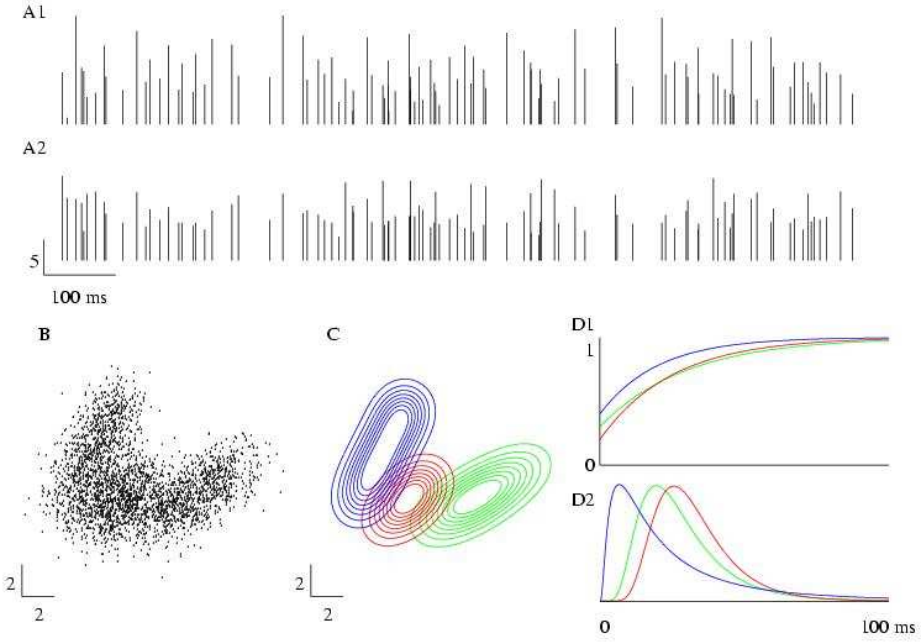

Figure 11: Simulated data with 3 active neurons recorded on a stereode (2 recording sites). 30 s of data were simulated resulting in 2966 spikes. A, An epoch with 100 spikes on site 1 (A1) and 2 (A2). Vertical scale bar in units of noise SD. B, Wilson plot of the 2966 spikes: the amplitude of each spike on site 2 is plotted against its amplitude on site 1. The scale bars corner is at the origin (0,0). C, Expected iso-density plots for each of the three neurons: neuron 1 (green), neuron 2 (red) and neuron 3 (blue). D1, Normalized spike amplitude *vs* ISI for each neuron (normalization done with respect to the maximal amplitude). Horizontal scale, same as D2. D2, ISI density of each of the 3 neurons (peak value of blue density: 35 Hz).

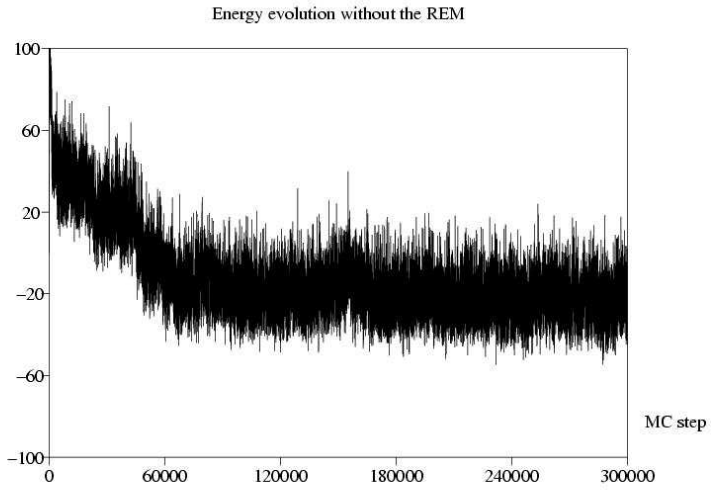

Figure 12: Energy evolution without the REM. Notice the step like decreases in the early part.

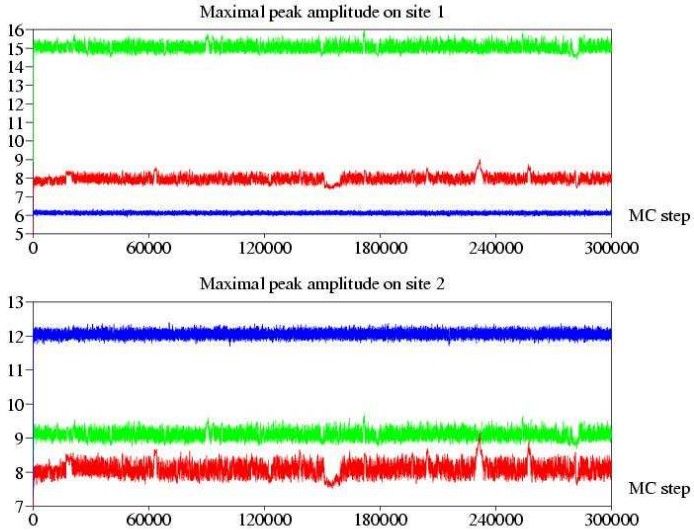

Figure 13: Evolution of the maximal peak amplitude on the two recording sites for the three neurons. Notice that an apparently stationary regime is reached quickly for those parameters, while the energy (Fig. 12) is still slowly decreasing.

11.2.3 Model parameters evolution

We can monitor as well the evolution of models parameters like the maximal peak amplitudes of the neurons[14] as shown on Fig. 13. The interesting, and typical, observation here is that the model parameters reach (apparent) stationarity much earlier than the energy (compare the first 60000 steps on Fig. 12 & 13). That means that most of the slow relaxation of the energy has a configurational origin. In other words, it comes from spike labels re-arrangements.

11.2.4 Model parameters estimates

Based on the energy trace (Fig. 12) and on the evolution of the model parameters (*e.g.,* Fig. 13) we could reasonably decide to keep the last 100000 MC steps for "measurements" and estimate the posterior (marginal) densities of our model parameters from these steps. We would then get for neuron 2 what's shown on Fig. 14. The striking feature of this figure is the time taken by the ACFs of the amplitude parameters ($P_1, P_2, \delta, \lambda$) to reach 0: 4000 steps. With the present data set we observe a similar behavior for the amplitude parameters of the first neuron but not for the third. That explains the un-acceptably low precision on the parameters estimates reported in Table 2 for neurons 1 & 2.

11.3 Algorithm dynamics and parameters estimates with the REM

The different behaviors of the model parameters, which relax to (apparent) equilibrium relatively fast ($2 \cdot 10^4$ to $3 \cdot 10^4$ MC steps) and of the energy which relaxes more slowly ($6 \cdot 10^4$ MC steps) suggests that the configuration dynamics is slower than the model parameters dynamics. It could therefore be worth considering a modification of our basic algorithm which could speed up the configuration (and if we are lucky the model parameters) dynamics like the Replica Exchange Method (Sec. 10). To illustrate the effect of the REM on our simple data set we have taken the first 2000 MC steps of the previous section and restarted[15] the algorithm augmenting our inverse temperature space from $\beta \in \{1\}$ to $\beta \in \{1, 0.95, 0.9, 0.85, 0.8, 0.75, 0.7, 0.65, 0.6, 0.55, 0.5\}$. The resulting energy trace for $\beta = 1$ and 30000 additional steps is shown on Fig. 15. The striking feature here is that the energy reaches after roughly $1.2 \cdot 10^4$ steps a *lower* energy than after $30 \cdot 10^4$ steps without the REM. Our REM scheme with 11 inverse

[14] We monitor in practice the evolution of every model parameter. We show here only the evolution of the maximal peak amplitude to keep this chapter length within bounds.

[15] When we introduce new β values from one run to the next, the initial state of the replica with the new β values is set to the final state of the replica with the closest β in the previous run. That is, in our example, at step 2001 all the replicas are in the same state (same parameters values, same configuration).

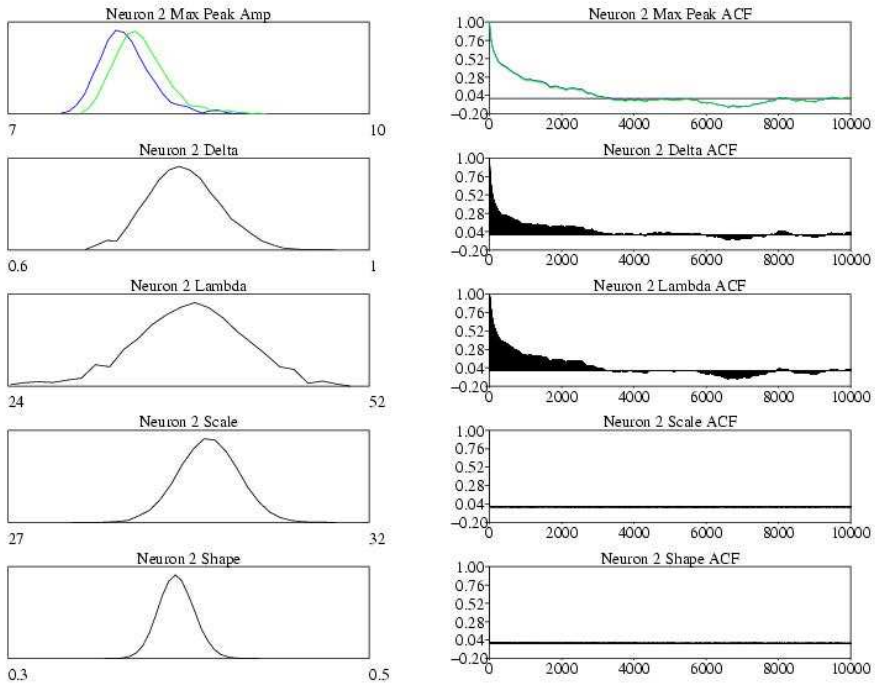

Figure 14: Marginal posterior estimates for the parameters of neuron 2. The left part shown the estimated density of each of the six parameters (for the peak amplitudes, the first site corresponds to the blue curve and the second to the green curve). The right side shows the corresponding ACFs (Sec. 9.1 and Eq. 51), the abscissa is the lag in MC steps.

	neuron 1	**neuron 2**	**neuron 3**
$\overline{P_1}$	15 ± 7	8 ± 7	6.1 ± 0.2
	(1071)	(1052)	(16)
$\overline{P_2}$	9 ± 4	8 ± 6	12.1 ± 0.5
	(945)	(930)	(27)
$\overline{\delta}$	0.7 ± 0.3	0.8 ± 1.2	0.58 ± 0.05
	(586)	(941)	(32)
$\overline{\lambda}$	33 ± 64	39 ± 137	47 ± 13
	(1250)	(1043)	(45)
$\overline{s}$	24.9 ± 0.1	29.8 ± 0.1	18.3 ± 0.5
	(6.5)	(7)	(1)
$\overline{f}$	0.51 ± 0.05	0.40 ± 0.05	1.01 ± 0.03
	(6.5)	(7)	(1)

Table 2: Estimated parameters values using the last 10^5 MC steps (among $3 \cdot 10^5$). The SDs are autocorrelation corrected (Sec. 9.1). The integrated autocorrelation time (Eq. 52) is given below each parameter between "()".

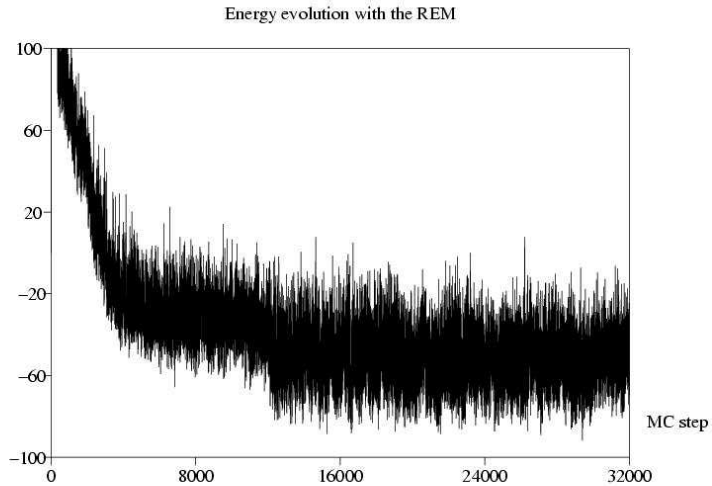

Figure 15: Energy evolution with the REM. The ordinate scale is the same as in Fig. 12.

temperatures implies that each step requires 11 times more CPU time than one step with a single inverse temperature[16]. From the energy evolution view point the REM is clearly more efficient than a single long simulation at $\beta = 1$. A more striking illustration of the REM effect on energy relaxation (on an other data set) can be found in [32].

11.3.1 Making sure the REM "works"

As explained in Sec. 10 the REM will work efficiently only if the energies explored by replicas at adjacent β exhibit enough overlap. Fig. 16 illustrates how the efficiency of the "replica exchange" can be empirically assessed [12, 37, 5, 26]. Energy traces at different β values are displayed on Fig.16A. The overlap between adjacent energy traces is already clear. The lowest trace ($\beta = 1$) is the trace of Fig. 15. Fig. 16B shows the energy histograms for the different β. Adjacent histograms clearly exhibit a significant overlap. A pictorial way to think of the REM is to imagine that several "boxes" at different pre-set inverse-temperatures are used and that there is one and only one replica per box at each step. After each MC step, the algorithm proposes to exchange the replicas located in neighboring boxes (neighboring in the sense of their inverse-temperatures) and this proposition can be accepted or rejected (Eq. 57). Then if the exchange dynamics works properly one should see each replica visit all the boxes during the MC run. More precisely each replica should perform a random walk on the available boxes. Fig. 16C shows the random walk performed by the replica which start at $\beta = 1$ at step 2001. The ordinate corresponds to the box index (see legend). Between steps 2001 and 32000, the replica travels through the entire inverse temperature range.

One of the shortcomings of the REM is that it requires more β to be used (for a given range) as the number of spikes in the data set increases because the width of the energy histograms (Fig. 16B) is inversely proportional to the square root of the number N of events in the sample [15, 12, 16]. The necessary number of β grows therefore as $\sqrt{N}$. The computation time per replica grows, moreover, linearly with N. We therefore end up with a computation time of the REM growing like: $N^{1.5}$.

11.3.2 Posterior estimates with the REM

We do not present here the Equivalent of Fig. 13 with the REM because the two figures would look too similar. The best way to see the effect of the REM on the model parameters dynamics is to look at the empirical *integrated autocorrelation time* (IAT) for each parameter as show in Table 3. When we compare these values with the ones obtained with our basic algorithm (without the REM), we see for instance that longest IAT is now 110 steps (parameter λ of the second neuron) while the longest IAT without the

[16]The computational overhead required accept or reject the proposed replica exchanges (Eq. 57) is negligible compared to the time required to update every spike label and every model parameter.

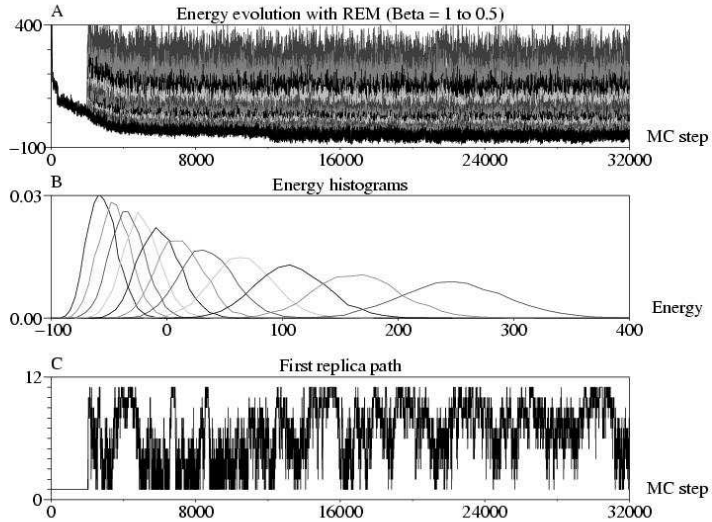

Figure 16: Test of the REM. A, Energy evolution for several β values (see text) during successive runs. 1 replica is used between steps 1 and 2000, 11 replicas are used between steps 2001 and 32000. B, Energy histograms at each β value (histograms computed from the 10000 last steps with 25 bins): the left histogram correspond to $\beta = 1$, the right one to $\beta = 0.5$. C, Path of the first replica in the temperature space. The ordinate corresponds to the β index (1 corresponds to $\beta = 1$ and 11 to $\beta = 0.5$).

REM was 1250 steps (parameter λ of the first neuron, Table 2). We therefore get *much better parameters estimates although the absolute sample size we use* 10^4 *is 10 times smaller than the one used previously*. This means that in this simple setting we were able without any fine tunning (we could have in fact used fewer inverse temperatures, we have not optimized the number of steps between replica exchange attempts) to get a much more efficient algorithm (in term of statistical efficiency and of relaxation to equilibrium) without extra computational cost.

11.3.3 Configuration estimate

As a quick way to compare estimated and actual configurations we can introduce what we will (abusively) call the "most likely" configuration estimate. First, we estimate the probability for each spike to originate from each neuron. For instance, if we discard the first $N_D = 22000$ on a total of $N_T = 32000$ steps we have

	neuron 1	**neuron 2**	**neuron 3**
$\overline{P_1}$	15 ± 2	8 ± 2	6.1 ± 0.1
	(62)	(91)	(12)
$\overline{P_2}$	9 ± 1	8 ± 2	12.1 ± 0.3
	(55)	(93)	(14)
$\overline{\delta}$	0.70 ± 0.07	0.8 ± 0.3	0.58 ± 0.02
	(25)	(93)	(9)
$\overline{\lambda}$	34 ± 19	40 ± 38	46 ± 7
	(60)	(110)	(20)
$\overline{s}$	24.9 ± 0.6	30.0 ± 0.4	18.3 ± 0.7
	(2)	(1.5)	(2.5)
$\overline{f}$	0.51 ± 0.02	0.40 ± 0.01	1.01 ± 0.03
	(2)	(1.5)	(2.5)

Table 3: Estimated parameters values using the last 10^4 MC steps (among $6 \cdot 10^4$). The SDs are autocorrelation corrected (Sec. 9.1). The integrated autocorrelation time (Eq. 52) is given below each parameter between "()".

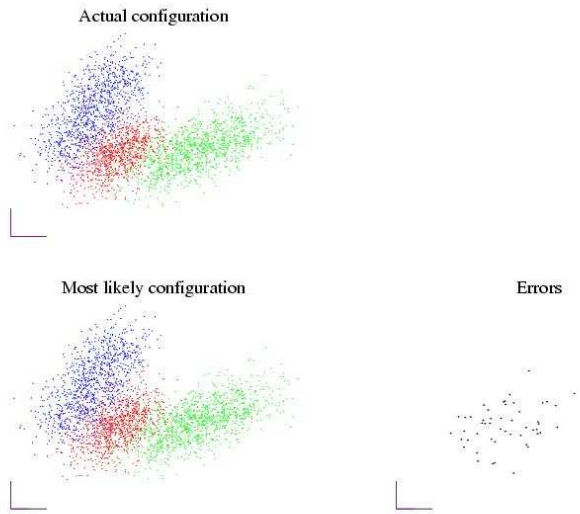

Figure 17: Comparison between the actual and estimated configurations. Top, the Wilson plot of Fig. 11B color coded to show the neuron of origin of each spike. Bottom left, the most likely configuration (see text) estimated from the last 10000 steps with the REM. Bottom right, the 50 errors (among 2966 events).

for the probability of the 100th spike to have been generated by the second neuron:

$$Pr(l_{100} = 2 \mid Y) \approx \frac{1}{10000} \cdot \sum_{t=22000}^{32000} \mathcal{I}_2 \left[l_{100}^{(t)} \right] \tag{58}$$

where $\mathcal{I}_q$ is the *indicator* function defined Eq. 16.

We then "force" the spike label to its most likely value. That is, if we have: $Pr(l_{100} = 1 \mid Y) = 0.05$, $Pr(l_{100} = 2 \mid Y) = 0.95$ and $Pr(l_{100} = 3 \mid Y) = 0$, we force l_{100} to 2. We proceed in that way with each spike to get our "most likely" configuration. Fig. 17 shows what we get using the last $10 \cdot 10^3$ MC steps of our run using the REM. The actual configuration is shown too, as well as the **50** misclassified events. We therefore get *1.7% misclassified spikes with our procedure*. We leave as an excercise to the reader to check that using an exact knowledge of the model parameters and the theoretical 2-dimensional Wilson plot densities one can get from them (Fig. 11C) we would get roughly 9% of the spikes misclassified. That would be the optimal classification we could generate using only the amplitude information. Our capacity to go beyond these 9% clearly stems from our inclusion of the ISI density in our data generation model. Using the last $10 \cdot 10^3$ MC steps of our run without the REM we generate 63 misclassifications, while we get "only" 59 of them with the last $100 \cdot 10^3$ steps[17].

11.3.4 A more detailed illustration of the REM dynamics.

When the posterior density one wants to explore is non isotropic a problem arise with the use of an MH algorithm based on component specific transitions. In such cases, the transitions proposed by the MH algorithm are not optimal in the sense that only "small" changes of the present parameters values are likely to be accepted. These "small" changes will lead to a "slow" exploration of the posterior density and therefore to "long" autocorrelation times. This is illustrated for our MH algorithm without the REM on the left side of Fig. 18. Here we show only the situation in the plane defined by the maximal peak amplitude on site 1 and the parameter λ of neuron 1 ($P_{2,1}, \lambda_1$) but the reader should imagine that the same holds in the 4 dimensional spaces defined by the 4 amplitude parameters of each neurons. The ACFs (Eq. 51) of the two parameters are shown. The last 1000 values of the parameters generated by the MH algorithm without the REM with the last 50 of them linked together by the broken line are shown. The movement of the "system" in its state space has clear Brownian motion features: small steps, random direction change from one step to the next, resulting in a "slow" exploration of the posterior density. If we now look at the

[17]This difference in the number of misclassification explains the energy difference observed in the two runs (Fig. 12 & 15). It means clearly that the run without the REM has not reached equilibrium.

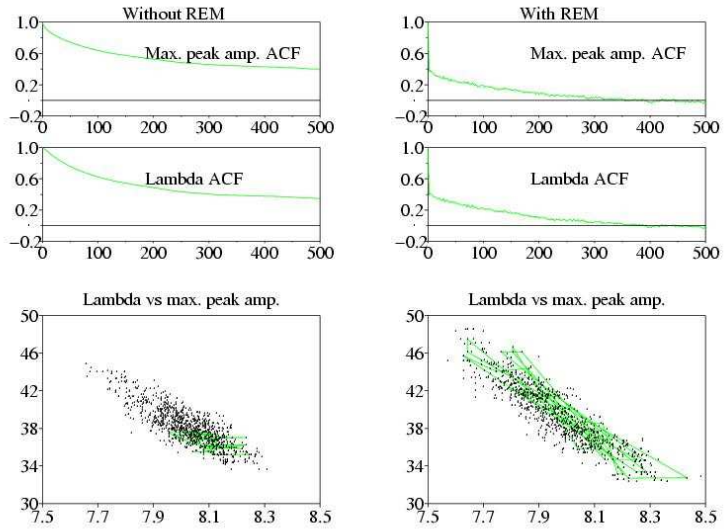

Figure 18: REM and parameters auto-correlation functions. The ACFs of the maximal peak amplitude of neuron 2 on site 1 ($P_{2,1}$) are shown together with the ACFs of the parameter λ of neuron 2. The ACFs without the REM (left side) were computed from the 100000 last iterations (Fig. 12). The ACFs with the REM (right side) were computed from the 10000 last iterations (Fig. 15). The ACFs with REM clearly fall faster than the ones without. The two graphs at the bottom give a qualitative explanation of why it is so. On each graph the last 1000 generated values for $P_{2,1}$ (abscissa) and λ_1 (ordinate) are shown (black dots). 50 values generated *successively* are linked together by the broken lines (green). The size of the jumps is clearly larger with REM than without.

system's dynamics using the REM, we see that the ACFs fall to zero much faster (right side of Fig. 18). This is easily understood by observing successive steps of the systems in the corresponding plane (λ vs $P_{2,1}$ graph, bottom). Large steps going almost from one end to the other of the marginal posterior density are now observed, meaning that the resulting Markov chain explores very efficiently the posterior density we want to study. This is clearly due to the fact that the high temperature replicas can make much larger steps with a still reasonably high acceptance probability. The replica exchange dynamics (cooling down "hot" replicas and warming up "cold" ones) does the rest of the job. This increased statistical efficiency (τ_{autoco} gets smaller) does in fact compensate for the increased computational cost (τ_{cpu} gets larger) of the REM (compare the integrated autocorrelation times in Table 2 & 3).

12 Conclusions

We have given a detailed presentation of a simple data generation model for extracellular recording. Although simple this model is already much more sophisticated than the ones previously used [21, 34, 31, 30], but this sophistication comes at a price: the usual algorithms presented in the spike-sorting literature cannot be used. To work with our model we had to find an analogy between our problem and the Potts spin glass model studied in statistical physics. This analogy allowed us to get a Dynamic MC/MCMC algorithm with which model parameters and configurations could be estimated *with confidence intervals*.

We have not explained here how one of the critical questions of spike-sorting, the number of neuron contributing to the data, can be tackled. We have recently described in [32] how this can be done with an ad-hoc method. The next step is clearly a Bayesian solution to this problem which fundamentally implies estimating the normalizing constant Z of Eq. 26 and 27 (which is the probability of the data). Nguyen et al [30] have already given a solution for a simple data generation model. This task of estimating Z could seem daunting but is luckily not. Dynamic MC and MCMC methods have been used for a long time which means that other people in other fields already had that problem. Among the proposed solutions, what physicists call thermodynamic integration [19, 8] seems very attractive because it is known to be robust and it requires simulations to be performed between $\beta = 1$ and $\beta = 0$. And that's precisely what we are already (partially) doing with the REM. Of course the estimation of Z is only half of the problem. The prior distribution on the number of neurons in the model has to be set properly as well. Increasing the

number of neurons will always lead to a decrease in energy which will give larger Z values (the derivative of the log of Z being proportional to the opposite of the energy [8]), we will therefore have to compensate for this systematic Z increase with the prior distribution.

Finally, although our method requires fairly long computation to be carried out, we hope we have convinced our reader that the presence of strong overlaps on Wilson plots does not necessarily precludes good sorting.

13 Exercises solutions

13.1 Cholesky decomposition (or factorization)

Write $a_{i,j}$ the elements of A and $\sigma_{i,j}$ the elements of Γ, compute:

$$AA^T = B$$

with: $b_{i,j} = \sum_{k=1}^{D} a_{i,k} a_{j,k}$ and identify individual terms, that is:

$$a_{1,1} = \sqrt{\sigma_{1,1}},$$

where the fact that A is lower triangular has been used.

$$a_{,j,1} = \frac{\sigma_{j,1}}{a_{1,1}}\ ,\ 2 \leq j \leq D.$$

$$a_{2,2} = \sqrt{\sigma_{2,2} - a_{2,1}^2}.$$

And keep going like that and you will find:

$$\begin{aligned} for\ i &= 1\ to\ D \\ for\ j &= i+1\ to\ D \\ a_{i,i} &= \sqrt{\sigma_{i,i} - \sum_{k=1}^{i-1} a_{i,k}^2} \\ a_{j,i} &= \frac{\sigma_{j,i} - \sum_{k=1}^{i-1} a_{j,k} a_{i,k}}{a_{i,i}} \end{aligned}$$

For details see [1].

13.2 Bayesian posterior densities for the parameters of a log-Normal *pdf*

Answer to question 1 We have:

$$\pi\left(s \mid f, \mathcal{D}\right) \alpha \exp\left[-\frac{1}{2}\frac{1}{f^2}\sum_{j=1}^{N}\left(\ln i_j - \ln s\right)^2\right] \cdot \frac{1}{s_{max} - s_{min}} \cdot \mathcal{I}_{[s_{min}, s_{max}]}\left(s\right).$$

If we introduce $\overline{\ln i}$ defined by Eq. 17 in the summed term we get:

$$\begin{aligned} \sum_{j=1}^{N}\left(\ln i_j - \ln s\right)^2 &= \sum_{j=1}^{N}\left(\ln i_j - \overline{\ln i} + \overline{\ln i} - \ln s\right)^2 \\ &= \sum_{j=1}^{N}\left(\ln i_j - \overline{\ln i}\right)^2 + 2\ \left(\overline{\ln i} - \ln s\right) \cdot \sum_{j=1}^{N}\left(\ln i_j - \overline{\ln i}\right) + N\ \left(\overline{\ln i} - \ln s\right) \end{aligned}$$

The first term does not depend on s, the second is zero, we therefore have:

$$\pi\left(s \mid f, \mathcal{D}\right) \alpha \exp\left[-\frac{1}{2}\frac{N}{f^2}\ \left(\overline{\ln i} - \ln s\right)^2\right] \cdot \frac{1}{s_{max} - s_{min}} \cdot \mathcal{I}_{[s_{min}, s_{max}]}\left(s\right).$$

Answer to question 2

1. Generate $U \sim Norm\left(\overline{\ln i}, \frac{f^2}{N}\right)$[18]
2. if $s_{min} \leq \exp u \leq s_{max}$, then $S = \exp u$
   otherwise, go back to 1

Answer to question 3 Using the first line of Answer 1 we get:

$$\pi\left(f \mid s, \mathcal{D}\right) \alpha \frac{1}{f^N} \exp\left[-\frac{1}{2}\frac{\sum_{j=1}^{N}\left(\ln i_j - \ln s\right)^2}{f^2}\right] \cdot \frac{1}{f_{max} - f_{min}} \cdot \mathcal{I}_{[f_{min}, f_{max}]}\left(f\right).$$

Identification does the rest of the job.

Answer to question 4 Using $\theta = \frac{1}{\omega}$ and the Jacobian of the change of variable, you can easily show that if $\Omega \sim Gamma\left(\alpha, \beta\right)$ then $\Theta = \frac{1}{\Omega} \sim Inv - Gamma\left(\alpha, \beta\right)$. The algorithm is simply:

1. Generate $\Omega \sim Gamma\left(\frac{N}{2} - 1, \frac{1}{2}\sum_{j=1}^{N}\left(\ln i_j - \ln s\right)^2\right)$
2. if $f_{min} \leq \sqrt{\frac{1}{\omega}} \leq f_{max}$, then $F = \sqrt{\frac{1}{\omega}}$
   otherwise go back to 1.

13.3 Stochastic property of a Markov matrix

$X^{(t+1)}$ being a *rv* defined on $\mathcal{X}$ we must have by definition:

$$Pr\left(X^{(t+1)} \subset \mathcal{X}\right) = 1,$$

where $X^{(t+1)} \subset \mathcal{X}$ should be read as: $X^{(t+1)} = x_1 + x_2 + \ldots + x_\nu$ (remember that the x_i are mutually exclusive events). We have therefore:

$$\begin{aligned}
Pr\left(X^{(t+1)} \subset \mathcal{X}\right) &= \sum_{i=1}^{\nu} Pr\left(X^{(t+1)} = x_i\right) \\
&= \sum_{i=1}^{\nu} Pr\left(X^{(t+1)} = x_i \mid X^{(t)} \subset \mathcal{X}\right) Pr\left(X^{(t)} \subset \mathcal{X}\right) \\
&= \sum_{i=1}^{\nu}\sum_{j=1}^{\nu} Pr\left(X^{(t+1)} = x_i \mid X^{(t)} = x_j\right) Pr\left(X^{(t)} = x_j\right) \\
&= \sum_{j=1}^{\nu} Pr\left(X^{(t)} = x_j\right)\left[\sum_{i=1}^{\nu} Pr\left(X^{(t+1)} = x_i \mid X^{(t)} = x_j\right)\right]
\end{aligned}$$

Which implies:

$$\sum_{j=1}^{\nu} Pr\left(X^{(t)} = x_j\right)\left[\sum_{i=1}^{\nu} Pr\left(X^{(t+1)} = x_i \mid X^{(t)} = x_j\right)\right] = 1$$

And that must be true for any $X^{(t)}$ which implies that[19]:

$$\sum_{i=1}^{\nu} Pr\left(X^{(t+1)} = x_i \mid X^{(t)} = x_j\right) = 1$$

[18] This notation means that the random variable U has a normal distribution with mean: $\overline{\ln i}$ and variance: $\frac{f^2}{N}$.

[19] If you're not convinced you can do it by *absurdum* assuming it's not the case and then find a $X^{(t)}$ which violate the equality.

13.4 Detailed balance

If $\forall x, y \in \mathcal{X},\ \pi(x)\,T(x,y) = \pi(y)\,T(y,x)$ then:

$$\begin{aligned}
\sum_{x \in \mathcal{X}} \pi(x)\,T(x,y) &= \sum_{x \in \mathcal{X}} \pi(y)\,T(y,x) \\
&= \pi(y) \sum_{x \in \mathcal{X}} T(y,x) \\
&= \pi(y)
\end{aligned}$$

Remember that T is stochastic.

References

[1] Bock RK and Krischer W (1998) *The Data Analysis BriefBook*. Springer - Verlag. http://physics.web.cern.ch/Physics/DataAnalysis/BriefBook/

[2] Brémaud P (1998) *Markov chains: Gibbs fields, Monte Carlo simulations and Queues*. New York: Springer - Verlag.

[3] Celeux G and Govaert G (1995) Gaussian parsimonious clustering models. *Pattern Recognition* **28**:781-793.

[4] Celeux G, Hurn M and Robert C (2000) Computational and inferential difficulties with mixture posterior distributions. *J American Statist Assoc* **95**:957-970.

[5] Falcioni M and Deem MW (1999) A biased Monte Carlo scheme for zeolite structure solution. *J Chem Phys* **110**:1754-1766.

[6] Fee MS, Mitra PP, and Kleinfeld D (1996) Variability of extracellular spike waveforms of cortical neurons. *J Neurophys* **76**: 3823 - 3833.

[7] Fishman GS (1996) *Monte Carlo. Concepts, Algorithms and Applications*. New York: Springer - Verlag.

[8] Frenkel D and Smit B (2002) *Understanding molecular simulation. From algorithms to applications*. San Diego: Academic Press.

[9] Frostig RD, Lieke EE, Tso DY and Grinvald A (1990) Cortical functional architecture and local coupling between neuronal activity and the microcirculation revealed by in vivo high resolution optical imaging of intrinsic signals. *PNAS* **87**: 6082 - 6086.

[10] Geman S, and Geman D (1984) Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **6**: 721 - 741.

[11] Häggström O (2002) *Finite Markov Chains and Algorithmic Applications*. Cambridge: Cambridge University Press.

[12] Hansmann UHE (1997) Parallel tempering algorithm for conformational studies of biological molecules. *Chem Phys Lett* **281**:140-150.

[13] Harris KD, Henze DA, Csicsvari J, Hirase H, and Buzsaki G (2000) Accuracy of tetrode spike separation as determined by simultaneous intracellular and extracellular measurements. *J Neurophys* **84**: 401 - 414.

[14] Hastings WK (1970) Monte Carlo sampling methods using Markov chains and their applications. *Biometrika* **57**: 92 - 109.

[15] Hukushima K and Nemoto K (1996) Exchange Monte Carlo and Application to Spin Glass Simulations. *J Phys Soc Japan* **65**:1604-1608.

[16] Iba Y (2001) Extended ensemble Monte Carlo. *Int J Mod Phys C* **12**:623-656.

[17] Janke W (2002) Statistical Analysis of Simulations: Data Correlations and Error Estimation. http://www.fz-juelich.de/nic-series/volume10

[18] Johnson DH (1996) Point process models of single - neuron discharges. *J Comput Neurosci* 3: 275 - 299.

[19] Landau D, and Binder K (2000) *A Guide to Monte Carlo Simulations in Statistical Physics*. Cambridge: Cambridge University Press.

[20] L'Ecuyer P (1994) Uniform random number generation. *Annals of Operations Research* **53**: 77 - 120.

[21] Lewicki MS (1994) Bayesian modeling and classification of neural signals. *Neural Comput* **6**:1005-1030.

[22] Lewicki MS (1998) A review of methods for spike-sorting: the detection and classification of neural action potentials. *Network: Comput Neural Syst* **9**: R53 - R78.

[23] Liu JS (2001) *Monte Carlo Strategies in Scientific Computing*. New York: Springer - Verlag.

[24] Metropolis N, Rosenbluth AW, Rosenbluth MN, Teller AH and Teller E (1953) Equations of state calculations by fast computing machines. *J Chem Phys* **21**: 1087 - 1092.

[25] Meyn SP and Tweedie RL (1996) *Markov Chains and Stochastic Stability*. New York: Springer - Verlag. A pdf version of the book (without figures) is available at the following address (Sean Meyn's web site): http://black.csl.uiuc.edu/~meyn/pages/TOC.html

[26] Mitsutake A, Sugita Y and Okamoto Y (2001) Generalized-ensemble algorithms for molecular simulations of biopolymers. *Biopolymers (Peptide Science)* **60**:96-123.

[27] Neal RM (1993) Probabilistic Inference Using Markov Chain Monte Carlo Methods. Technical Report CRG-TR-93-1, Department of Computer Science, Univ. of Toronto. http://www.cs.toronto.edu/~radford/papers-online.html

[28] Neal RM (1994) Sampling from multimodal distributions using tempered transitions. Technical Report No. 9421, Department of Statistics, Univ. of Toronto. http://www.cs.toronto.edu/~radford/papers-online.html

[29] Newman MEJ and Barkema GT (1999) *Monte Carlo Methods in Statistical Physics*. Oxford: Oxford University Press.

[30] Nguyen DP, Frank LM and Brown EN (2003) An application of reversible-jump Markov chain Monte Carlo to spike classification of multi-unit extracellular recordings. *Network: Comput Neural Syst* **14**: 61 - 82.

[31] Pouzat C, Mazor O, and Laurent G (2002) Using noise signature to optimize spike - sorting and to asses neuronal classification quality. *J Neurosci Methods* **122**: 43 - 57.

[32] Pouzat C, Delescluse M, Viot P and Diebolt J (2004) Improved spike-sorting by modeling firing statistics and burst-dependent spike amplitude attenuation: a Markov chain Monte Carlo approach. *J Neurophys* in press. DOI, 10.1152/jn.00227.2003.

[33] Robert CP, and Casella G (1999) *Monte Carlo Statistical Methods*. New York: Springer - Verlag.

[34] Sahani M (1999) Latent variable models for neural data analysis. PhD Thesis, California Institute of Technology: Pasadena.

[35] Sokal AD (1996) Monte Carlo methods in statistical mechanics: foundations and new algorithms. Cours de Troisième cycle de la Physique en Suisse Romande. http://citeseer.nj.nec.com/sokal96monte.html

[36] Wu JY, Cohen LB and Falk CX (1994) Neuronal activity during different behaviors in Aplysia: a distributed organization? *Science* **263**: 820-823.

[37] Yan Q and de Pablo JJ (1999) Hyper-parallel tempering Monte Carlo: Application to the Lennard-Jones fluid and the restricted primitive model. *J Chem Phys* **111**:9509-9516.