Default

```
03/05/2007 10:13    BROWN UNIVERSITY LIBRARY ANNEX   PROD SYSTEM    PG      2
RPT# 00061118

                         RETRIEVAL PACKING LIST


     Order: 008584
       For: QS Stored General Collection
   Shipped: 03/05/2007

   Item #: .i14043362
  Barcode: 31236071207610
    Title: Network : computation in neural systems
 Location: QS Stored General Collection
PickUp At: SC-Sciences Library
   Patron: FRANK WOOD
Patron ID: 21236007158608
      Notes: fwood@cs.brown.edu    Nguyen, D., Frank, L.M. and
             Brown, E.N. An application of reversible-jump MCMC
             to spike classification of multiunit extracellular
             recordings. Network: Computation in Neural
             Systems, 14,



      TOTAL COUNT: 1
```

Institute of Physics Publishing — Network: Computation in Neural Systems
Network: Comput. Neural Syst. **14** (2003) 61–82 — PII: S0954-898X(03)57776-4

An application of reversible-jump Markov chain Monte Carlo to spike classification of multi-unit extracellular recordings

David P Nguyen[1,2], **Loren M Frank**[1] **and Emery N Brown**[1]

[1] Neuroscience Statistics Research Laboratory, Department of Anesthesia and Critical Care, Massachusetts General Hospital, Division of Health Sciences and Technology, Harvard Medical School/Massachusetts Institute of Technology, Boston, MA 02114, USA
[2] Picower Center for Learning and Memory, RIKEN-MIT Neuroscience Research Center, Department of Brain and Cognitive Sciences, Massachusetts Institute of Technology, Cambridge, MA 02139, USA

E-mail: dpnguyen@mit.edu and brown@neurostat.mgh.harvard.edu

Received 20 May 2002, in final form 19 December 2002
Published 17 January 2003
Online at stacks.iop.org/Network/14/61

Abstract
Multi-electrode recordings in neural tissue contain the action potential waveforms of many closely spaced neurons. While we can observe the action potential waveforms, we cannot observe which neuron is the source for which waveform nor how many source neurons are being recorded. Current spike-sorting algorithms solve this problem by assuming a fixed number of source neurons and assigning the action potentials given this fixed number. We model the spike waveforms as an anisotropic Gaussian mixture model and present, as an alternative, a reversible-jump Markov chain Monte Carlo (MCMC) algorithm to simultaneously estimate the number of source neurons and to assign each action potential to a source. We derive this MCMC algorithm and illustrate its application using simulated three-dimensional data and real four-dimensional feature vectors extracted from tetrode recordings of rat entorhinal cortex neurons. In the analysis of the simulated data our algorithm finds the correct number of mixture components (sources) and classifies the action potential waveforms with minimal error. In the analysis of real data, our algorithm identifies clusters closely resembling those previously identified by a user-dependent graphical clustering procedure. Our findings suggest that a reversible-jump MCMC algorithm could offer a new strategy for designing automated spike-sorting algorithms.

1. Introduction

Spike sorting is the process through which action potentials from several simultaneously recorded neurons are separated and assigned to the neurons of origin (Lewicki 1998). The output of a spike-sorting analysis are spike train records of each neuron. These records will be the primary data for making inferences about the neural system being investigated. Because spike-sorting errors can have significant inferential consequences, the development of accurate spike-sorting algorithms is a central question in neuroscience data analysis.

The primary goal of a spike-sorting algorithm is to solve the challenging inverse problem of determining how many neurons are being simultaneously recorded and assigning the action potentials to the correct neuron. To accomplish these objectives several important subproblems must be addressed. These include modelling the action potential waveform, accounting for bursting and/or synchronous spiking activity; treating overlapping spike waveforms; and assessing the effects of misclassification errors and low signal-to-noise ratios.

Many spike-sorting algorithms solve the inverse problem by assuming a fixed number of source neurons and then devising a strategy to assign the action potentials to source neurons given this fixed number. While it does not address all the issues in the subproblems stated above, a common model used in many spike-sorting algorithms is the Gaussian mixture model. For a fixed number of neurons the mixture model is estimated using either maximum likelihood methods via the expectation–maximization (EM) algorithm (Sahani *et al* 1997, Shoham 2001) or by Bayesian methods (Lewicki 1994). In the EM algorithm, the probability of membership of each action potential to each neuron is estimated in the E-step. Then, in the M-step, conditional on the most probable assignment of each action potential, the parameters of each Gaussian component of the mixture are estimated. In a Bayesian analysis, the marginal posterior density of each spike belonging to each mixture component (neuron) is computed along with the marginal posterior density of the parameters for each mixture component. The number of source neurons in either of these fixed dimension approaches is determined by comparing the results of the algorithm across a range of possible values for the number of neurons.

In reality, the number of neurons in a spike-sorting problem is not a fixed, but is instead an unknown parameter that must be estimated as well. Recently in the statistics literature there has been substantial work on using Bayesian methods to estimate quantities of variable dimension using reversible jump Markov chain Monte Carlo (RJMCMC) methods (Green 1995, Robert and Casella 1999). A marginal probability density of the dimension is constructed along with those for the assignments of the spikes to different mixtures and those for the parameters of each mixture.

In this paper we use the RJMCMC algorithm to study the spike-sorting problem. We model the spike activity in a feature vector space as an anisotropic Gaussian mixture model and present as an alternative, a hybrid MCMC algorithm to simultaneously estimate the number of source neurons and to assign each action potential to a source. We review the concepts of MCMC and then present our algorithm in terms of the two MCMC techniques we use: the Gibbs sampler and the RJMCMC. We illustrate our algorithm in an example using simulated three-dimensional data and in a second example using real four-dimensional feature vectors extracted from tetrode recordings of rat entorhinal cortex neurons.

In our analysis of simulated data, our algorithm found the correct number of Gaussian clusters in the data and accurately captured the location of shape of each Gaussian cluster. In the real data example, where the Gaussian assumption was inaccurate, our algorithm located clusters resembling those previously identified by a user-dependent graphical clustering procedure.

2. Methods

MCMC refers to a set of techniques used to obtain random samples from the posterior density, which is the joint probability density of the unknown model parameters conditional on all the observed data and known parameters. The basic structure of MCMC is derived from the simple

rule defining a Markov chain: the probability of transitioning into any state is dependent on only the current state. A state can be regarded as some point in the model parameter space. The cornerstone of MCMC is the Metropolis–Hastings (MH) algorithm (Metropolis *et al* 1953, Hastings 1970), which is subsequently described. The elongation or realization of a Markov chain is governed by a MH transition kernel, which is embodied in the MH transition ratio (in the algorithmic description of MCMC). The MH transition ratio is directly related to the transition probability, which is the probability of jumping from the current state to a proposal state. The proposal state is obtained by sampling from a set of arbitrary proposal densities and is accepted with the probability dictated by the MH transition probability. If the proposal is rejected, the next state of the Markov chain is identical to the current state. By successively proposing and accepting or rejecting proposal states based on the MH transition kernel, the resulting Markov chain will consist of states that are distributed according to the posterior density. These samples can then be used to construct useful statistics about the model parameters for purposes of inference and parameter estimation.

We apply two special cases of the MH algorithm, the Gibbs sampler and the RJMCMC. By construction, the proposal densities used in the Gibbs sampler are the full conditional densities of the state parameters. Because the full conditional density of a parameter is proportional to the posterior density, samples drawn from the full conditional will be distributed exactly like samples drawn from the posterior density. Hence, the acceptance ratio is always 1. The RJMCMC facilitates dimension-changing transitions using a modified MH transition ratio that incorporates two concepts: the dimension-matching parameter and the state transformation function. The dimension-matching parameter is combined with the state of smaller dimensionality to satisfy the dimension-matching requirement, while the state transformation function deterministically relates together the current state, dimension-matching parameter, and proposal state. These two features allow the chain to be reversible, which is essential for convergence of the Markov chain to the posterior density. The reader is asked to refer to Green (1995) for the formulation of the theory and the proof of convergence of the RJMCMC, and Robert (1996), Diebolt and Robert (1994), Escobar and West (1995), and Gelfand *et al* (1990) for details on theory and implementation of the Gibbs sampler.

Our algorithm extends the algorithm of Richardson and Green (1997), which uses a univariate *Gaussian* basis density, to a multivariate anisotropic Gaussian basis density. The multivariate anisotropic Gaussian density allows all the eigenvalues of the covariance matrix to vary freely, in contrast to the spherical Gaussian density, thus allowing the mixture components to express ellipsoidal shapes. Our algorithm is a hybrid sampler that samples from the joint posterior density, $p(K, \Phi, Z|Y)$, where K is the number of components in the mixture, the set Φ contains the parameters for the mixtures with 1 to K_{max} components, and Z is the vector containing the data classifications. The hybrid sampler uses RJMCMC to sample the mixture size, K, and the Gibbs sampler to sample the mixture parameters conditional on a fixed value of K. The result of the MCMC is a set of predictive or marginal density functions that summarize the probability of each parameter with consideration for all of the data and possible values for other model parameters. Using these densities, we can infer the *most probable* mixture model specification to describe the neuronal activity.

Inference based on marginal density functions has several appealing features: multiple modes of the parameters are naturally observed; the existence of multiple solutions can be verified; and the evidence for a particular solution is always available. Moreover, the flexibility of the anisotropic Gaussian basis density allows the technique to be applied to a large range of data types. For example, the clusters of data can be spherical or ellipsoidal with varying sizes and orientations. The density can be used to model clusters of any dimensionality. In spike-sorting applications, this allows the procedure to operate on data recorded by a multi-electrode of any size.

2.1. Anisotropic Gaussian mixtures

Let $(0, T]$ denote the observation interval, $0 < t_1 < t_2 < \cdots < t_n < T$ be a set of N spike times, and let the set $Y = (y_n)_{n=1}^{N}$ be the set of feature vectors associated to the observed spike times. Assuming the observed data was produced from K neurons and the activity of each neuron in the feature vector space is Gaussian distributed, let $\Theta_K = (\theta_k)_{k=1}^{K}$ be the set of parameters that define the K Gaussians, and let $\theta_k = (w_k, \mu_k, \Sigma_k)$ be the weight, mean vector, and covariance matrix that describes the activity of the neuron with label k. Also, let the unobserved or latent data $Z = (z_n)_{n=1}^{N}$, $z_n \in [1, \ldots, K]$, be the set of labels that contains the identity of the source neuron for y_n, $n = 1, \ldots, N$. When the latent data is ignored, the mixture is expressed as

$$y_n \sim \sum_{k=1}^{K} w_k f_k(\cdot), \tag{1}$$

independently and identically for $n = 1, \ldots, N$, where

$$f_k(y|\mu_k, \Sigma_k) = \frac{1}{(2\pi)^{d/2}|\Sigma_k|^{1/2}} \exp\left[-\frac{1}{2}(y - \mu_k)^{\mathrm{T}}\Sigma_k^{-1}(y - \mu_k)\right]. \tag{2}$$

In (1), the weights of the mixture components $W = (w_k)_{k=1}^{K}$, $\sum w_k = 1$, set the relative responsibility of each mixture component for the production of y_n when z_n is unknown. If the latent data is known, the mixture can be expressed in a much simpler form,

$$y_n|z_n \sim f_{z_n}(\cdot). \tag{3}$$

For mixture component k, the set $G_k = (n|n \in [1, N], z_n = k)$ contains the index of each data point that was produced by component k and $g_k = \#G_k$ is the set size of G_k.

2.2. Bayesian estimation

The data classification problem arises when the classifications, Z, are unknown and are the subject of inference. The process of data classification consists of two steps: estimate the mixture model parameters Θ_K and classify the data by estimating Z. Optimization methods, such as EM, converge upon Θ_K and Z by iteratively maximizing the likelihood function (McLachlan and Krishnan 1997). The two main disadvantages of these methods are that K must be assumed prior to classification, and that (Θ_K, Z) may not lie at the global maximum of the likelihood surface. In contrast, a MCMC algorithm that is observed to mix well will have explored all combinations of (K, Θ_K, Z) under the support of the posterior and will, therefore, reveal the local and global maxima of the posterior density.

In our MCMC algorithm, one primary goal is to infer the value of K and then infer (Θ_K, Z) conditional on K. For inference on K, we use the full marginal posterior

$$p(K) = \int p(\Omega)\,\mathrm{d}\Omega_{|K|}. \tag{4}$$

The set $\Omega = (K, \Phi, Z)$, where $\Phi = (\cup_{k=1}^{K_{\max}}\{K \times \Theta_K\})$, contains the mixture size, the parameters for the mixture model defined by each value of K, and the latent data. The set $\Omega_{|K|}$ is defined as the set Ω with the parameter K omitted. The integrand in (4) is the joint posterior density

$$p(\Omega) = \frac{p(\Omega, Y)}{\int p(\Omega, Y)\,\mathrm{d}\Omega}, \tag{5}$$

where Y is the observed data. Since the specification of each full marginal requires the integrals in (4) and (5) to be evaluated over mixtures of differing sizes, an analytical solution is

intractable. However, with the use of MCMC methods, we can reconstruct (4) using samples drawn from $p(\Omega, Y)$. Moreover, the samples used to reconstruct (4) can also be used to reconstruct the conditional marginal posteriors $p(w_k|K)$, $p(\mu_k|K)$, $p(\Sigma_k|K)$, $p(z_n|K)$, for $k = 1, \ldots, K$ and $n = 1, \ldots, N$.

MCMC methods allow any density to be estimated up to a constant. This suggests that we can estimate the shape of the full posterior, $p(\Omega)$, and then normalize the area under the surface to get a true estimate of $p(\Omega)$, without ever evaluating the constant $\int p(\Omega, Y)\,\mathrm{d}\Omega$. The shape of $p(\Omega)$ is reconstructed from samples drawn from the joint density

$$p(\Phi, Y, Z) = p(K)p(W|K)p(Z|W, K)p(\Theta_K|K)p(Y|\Theta_K, Z). \tag{6}$$

The first four terms, which reflect our prior knowledge, are the prior densities for the mixture model parameters. The last term is the likelihood of the data given a particular mixture model specification. The fact that K is a parameter in the joint density implies that samples from (6) will be variates of differing dimensionalities. In our algorithm the Gibbs sampler is used to sample the mixture parameters conditional on a fixed K, while an embedded RJMCMC step samples K itself.

2.3. Gibbs sampler for anisotropic Gaussian mixtures with a known number of components

We state the full conditional distributions using the notation of Lavine and West (1992). The full conditionals were derived from (6) using the likelihood defined by (3) and the following prior densities:

$$W|K \sim \mathcal{D}_K(\alpha_1, \ldots, \alpha_K), \tag{7}$$

$$\mu_k|K, \Sigma_K \sim \mathcal{N}_d(m_{k,0}, h_{k,0}^{-1}\Sigma_k), \tag{8}$$

$$\Sigma_k|K \sim \mathcal{W}^{-1}(v_{k,0}, V_{k,0}), \tag{9}$$

$$z_n|K, W \sim p(z_n = k) = w_k. \tag{10}$$

W is the set of mixing weights, $\mathcal{D}_K(\cdot)$ is the K-variate Dirichlet probability density, $\mathcal{N}_d(\cdot)$ is the d-variate normal probability density, and $\mathcal{W}^{-1}(\cdot)$ is the inverse Wishart probability density. The Dirichlet distribution is simply the multivariate generalization of the beta distribution (Kotz *et al* 2000). The inverse Wishart distribution is a multivariate generalization of the inverse gamma distribution, which is often used as the conjugate prior for the variance parameter of the univariate Gaussian density (Andersen 1984). The subscript k refers to the mixture component with label k. It follows from the conjugateness of the priors and the likelihood that the full conditionals of the mixture parameters are

$$W|K, Y \sim \mathcal{D}_K(\alpha_1 + g_1, \ldots, \alpha_K + g_K), \tag{11}$$

$$\mu_k|K, \Sigma_k, Y \sim \mathcal{N}_d(m_k, h_k^{-1}\Sigma_k), \tag{12}$$

$$\Sigma_k|K, Y \sim \mathcal{W}^{-1}(v_k, V_k), \tag{13}$$

$$z_n|K, Y \sim P(z_n = j|Y) \propto w_j f(y_n|\mu_j, \Sigma_j, z_i = j). \tag{14}$$

The parameters used in (11)–(14), are defined as

$$m_k = \frac{g_k\bar{y}_k + h_{k,0}m_{k,0}}{g_k + h_{k,0}}, \tag{15}$$

$$h_k = g_k + h_{k,0}, \tag{16}$$

$$v_k = g_k + v_{k,0}, \tag{17}$$

$$V_k = V_{k,0} + S_k + \frac{g_k h_{k,0}}{h_k}(\bar{y}_k - m_k)(\bar{y}_k - m_k)^{\mathrm{T}}, \tag{18}$$

$$S_k = \sum_{j|z_j=k} (y_j - \bar{y}_k)(y_j - \bar{y}_k)^{\mathrm{T}}. \tag{19}$$

The parameters $V_{k,0}$, $h_{k,0}$, $\upsilon_{k,0}$, and $m_{k,0}$ are the same for all mixture components. $V_{k,0}$ and $m_{k,0}$ define a super-Gaussian proposal density that binds all of the proposals in the Gibbs chain to the vicinity of the data and provides some *a priori* structure to the mixture model. The parameter $h_{k,0}$ represents how much prior information should be incorporated in the full conditionals. The parameters of the inverse Wishart are V_k, the scale matrix, and υ_k, the number of degrees of freedom. The parameter $\bar{y}_k = \frac{1}{g_k}\sum_{j\in G_k} y_j$ is the empirical mean of the data in cluster k. The parameter, S_k, is defined to be the sum of the sample covariance matrices.

An entire state transition in the MCMC begins with the sampling of K and ends with an embedded Gibbs step. Equations (11)–(19) fully specify the half of the hybrid sampler that consists of the Gibbs sampler, while the following section outlines the half that contains the RJMCMC.

2.4. *RJMCMC for anisotropic Gaussian mixtures with unknown number of components*

RJMCMC address the model selection problem by allowing transitions to occur between states of different dimensionality in the MCMC according to the posterior density function. We accommodate the change in K between states by defining the parameter space of the variable dimension mixture as a union of multiple finite mixtures with different values of K, $\Phi = (\cup_{K\in\mathcal{K}}\{K\} \times \Theta_K)$, where $\mathcal{K} = (1, \ldots, K_{max})$. Furthermore, we define two types of MH dimension-changing moves, *split component* and *combine component*, which are randomly selected when proposing a change in parameter space dimension. Each dimension-changing move has an associated transformation that operates on the current state to produce a proposal state. The split-component move transforms the current state to a proposal state in a higher state space dimension, $\Theta_{K+1} = F_{split}(\Theta_K, U)$. The parameter U is referred to as the *dimension-matching parameter* with property $\dim(\Theta_k) + \dim(U) = \dim(\Theta_{k+1})$. The parameter U is sampled from the density $Q(U)$ to randomize the proposal state. The combine-component move and transformation, $F_{combine}(\cdot) \triangleq F^{-1}_{split}(\cdot)$, are for proposals to lower dimension state spaces. The transformation $F_{split}(\cdot)$ is required to be one-to-one and invertible. The proposed mixture size, $\hat{K}$, is accepted with probability α for $\hat{K} = K+1$ and α^{-1} for $\hat{K} = K-1$, where α is a MH acceptance probability.

The role of the split- and combine-component transformations is to provide a link between the current state and a proposal state of a different dimensionality. The split transformation takes a randomly chosen mixture component and splits the component into two components (figure 1). A line that joins the mean of the two new components will pass through the mean of the old component. The covariance matrices of the new components obey the laws of sufficient statistics; therefore, the split transformation preserves the statistics of the data. The dimension-matching parameter U matches the number of outputs with inputs so that invertibility is ensured. The combine transformation is simply defined as the inverse of the split transformation and occurs between two clusters that are relatively close to one another.

Given the time interval $(0, T]$, we define at discrete time t, $\Psi^t \times Z^t = (K^t \times \Theta^t_{K^t}) \times Z^t$ to be the state of the MCMC, which consists of a mixture model selected from the parameter space defined by Φ and the data classifications. The dimensionality of Ψ^t is given by

$$\dim(\Psi^t) = N + 1 + d + K^t\left(d + \frac{d(d+1)}{2}\right). \tag{20}$$

where d is the dimension of the data, and N is the number of data points.

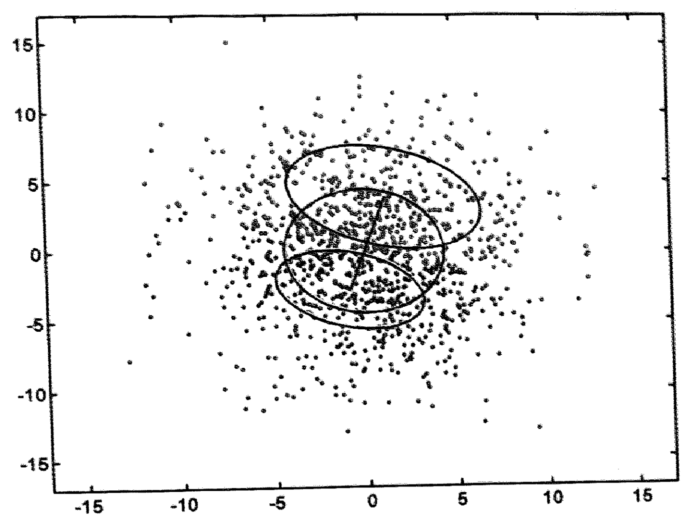


Figure 1. An illustration of the split/combine transformation. The line through the three ellipses is the called the split-line. The direction of the split-line is determined by the random vector $L_c u_2$. The components that are split from the original (centre) component have their means at the ends of the split-line. The observations are reallocated such that each set of new clusters lie on different sides of the plane defined by the direction of the split-line.

At iteration t, the state of the Markov chain can do one of three things depending on the proposed move and MH acceptance ratio: increase K^t by one, decrease K^t by one, or keep K^t. Assuming that the current state, Ψ^t, already has a high probability under the posterior, we would like to define $f_m(\cdot)$, such the proposed state, $\hat{\Psi}^{t+1}$, retains as much information from Ψ^t as possible in order to make the acceptance of $\hat{\Psi}^{t+1}$ more likely. This results in a nesting of mixture models, where the current state and the proposed state have common components, i.e. $\Psi^t \cap \hat{\Psi}^{t+1} \neq \emptyset$. For example, the split transformation only operates on one randomly chosen mixture component to produce two mixture components. Similarly, when moving to a smaller mixture, the combine transformation produces one mixture component by combining two existing components, while leaving the other components unaltered.

We define the split transformation, $f_{split}(\cdot) \equiv f(\cdot)$, as a set of individual parameter transformations:

$$\begin{aligned}
w_{s_1} &= w_c u_1, \\
w_{s_2} &= w_c(1-u_1), \\
\mu_{s_1} &= \mu_c - ((1-u_1)/u_1)^{1/2} L_c u_2, \\
\mu_{s_2} &= \mu_c + (u_1/(1-u_1))^{1/2} L_c u_2, \\
\Sigma_{s_1} &= \frac{1}{u_1} L_c (I_d - u_2 u_2^{\mathrm{T}}) L_c^{\mathrm{T}} \odot U_3, \\
\Sigma_{s_2} &= \frac{1}{1-u_1} L_c (I_d - u_2 u_2^{\mathrm{T}}) L_c^{\mathrm{T}} \odot (1_d - U_3).
\end{aligned} \tag{21}$$

where $\odot$ denotes the Hadamard product, 1_d is a $d \times d$ matrix full of ones, I_d is $d \times d$ identity matrix, L_c is the lower diagonal Cholesky factor of Σ_c, the subscript c is drawn from Unif$(1, K)$, $s_1 = c$, $s_2 = c+1$, and the dimension-matching parameters are sampled as

$$\begin{aligned} u_1 &\sim \text{beta}(2,2), \\ u_2(i) &\sim 0.5\mathcal{N}_1(0.5, 0.2) + 0.5\mathcal{N}_1(-0.5, 0.2), \\ U_3(i,j) &\sim \text{beta}(2,2), \\ U_3(j,i) &= U_3(i,j), \end{aligned} \tag{22}$$

for $i = 1, \ldots, d$ and $j \leqslant i$. The Hadamard product, $C = A \odot B$, is the product of the corresponding entries, $c_{ij} = a_{ij} \times b_{ij}$. The Cholesky factor is analogous to taking the square root of a matrix and is defined as $A = U_c U_c^{\mathrm{T}}$ for full-rank matrix A, where U_c is a lower triangular matrix. The split transformation is designed to randomly partition the mixture component, θ_c, into two components. The new centre s, μ_{s_1} and μ_{s_2}, are split along the axis defined by the random vector $L_c u_2$ and have relative distances, $((1-u_1)/u_1)^{1/2}$ and $(u_1/(1-u_1))^{1/2}$, from the original centre, μ_c. The dimension-matching variable, u_1, distributes the responsibility for the data to the new clusters. The symmetric matrix, U_3, contains entries that are individually drawn from the beta density and randomly rescales and reshapes the two new covariance matrices. The data indices in G_s, the set that defines the original cluster, are reallocated in a deterministic fashion; the plane defined by μ_c and $L_c u_2$ are used to partition the cluster points (figure 1).

We define the combine transformation, $f_{combine}(\cdot) \equiv f^{-1}(\cdot)$, as the inverse of the split transformation:

$$\begin{aligned} w_c &= w_{s_1} + w_{s_2}, \\ w_c \mu_c &= w_{s_1}\mu_{s_1} + w_{s_2}\mu_{s_2}, \\ w_c \Sigma_c &= w_{s_1}[(\mu_{s_1} - \mu_c)(\mu_{s_1} - \mu_c)^{\mathrm{T}} + \Sigma_{s_1}] + w_{s_2}[(\mu_{s_2} - \mu_c)(\mu_{s_2} - \mu_c)^{\mathrm{T}} + \Sigma_{s_2}], \\ u_1 &= w_{s_1}/w_c, \\ u_2 &= -\left[\frac{w_{s_1}}{w_{s_2}}\right]^{1/2} L_c^{-1}(\mu_{s_1} - \mu_c), \\ U_3 &= \frac{w_{s_1}}{w_c}\Sigma_{c_1} \oslash (L_c(I_d - u_2 u_2^{\mathrm{T}})L_c^{\mathrm{T}}, \end{aligned} \tag{23}$$

where $\oslash$ denotes an element-by-element division operator or the inverse of the Hadamard product, and the subscript s_1 is drawn from Unif$(1, K-1)$. The component s_2 is drawn from a distribution defined by the relative Mahalanobis distance of each component to s_1,

$$s_2 \sim P(s_2 = j|s_1) \propto (\mu_{s_1} - \mu_{s_2})^{\mathrm{T}}\Sigma_{s_1}^{-1}(\mu_{s_1} - \mu_{s_2}). \tag{24}$$

The index of the component produced by combining s_1 and s_2 is $c = \min(s_1, s_2)$. Since the combine move brings the state to a lower dimension, the dimension-matching parameter U is deterministically produced by the combine transformation.

The distribution for the dimension-matching parameters, u_1, u_2 and U_3, were chosen to produce a particular behaviour in the transformations, while taking into account the range of values for the parameters. The distribution for u_1, beta(2, 2), was chosen for its constrained support over [0, 1] and its symmetry. The symmetry implies that there is no preferable weight for one component over the other. The density for u_2 was chosen to produce split transformations with the highest possibility of being accepted; the density for u_2 produces the two new components near a principal axis of the original cluster. The components of U_3 are assigned a distribution of beta(2, 2) for the same reasons as u_1, i.e. the symmetry implies that there is no preference for one component to have a certain size or orientation over the other.

A split move, once proposed, is accepted with probability

$$\alpha_{split}(\Psi^t, \hat{\Psi}^{t+1}) = \min(1, R_{split}). \tag{25}$$

where

$$R_{split} = \frac{\pi_Y(\hat{\Psi}^{t+1})r_{split}(\hat{\Psi}^{t+1})}{\pi_Y(\Psi^t)r_{split}(\Psi^t)q(U^t)}\left|\frac{\partial f_{split}(\Psi, U)}{\partial(\Psi, U)}\right|_{\Psi=\Psi^t, U=U^t}, \tag{26}$$

and $\pi_Y(\Psi)$ is the target or posterior distribution, $r_m(x)$ is the probability of choosing move m while in state x, $q(\cdot)$ is the density from which U is drawn, and $|\partial f(x)/\partial x|$ is the Jacobian of $f_{split}(\cdot)$ evaluated at the current state. The combine move is accepted with probability

$$\alpha_{combine}(\Psi^t, \hat{\Psi}^{t+1}) = \min(1, R_{combine}^{-1}), \tag{27}$$

where $R_{combine}$ has the same form as R_{split} except that Ψ^t switches places with Ψ^{t+1}, and U^t is replaced by U^{t+1}. The full form of R_{split}, given the definition of the target distribution and the split/combine transformations is

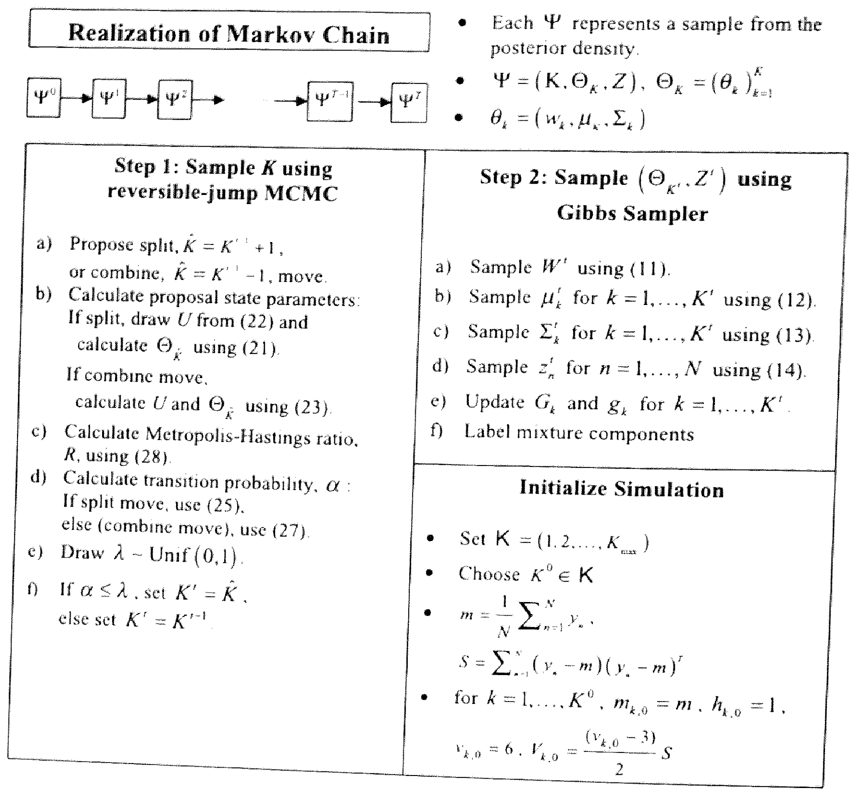
$$\begin{aligned} R_{split} = {} & \frac{p(Y|\hat{\Psi}^{t+1})}{p(Y|\Psi^t)}\frac{p(\hat{K}^{t+1})}{p(K^t)}\frac{w_{s_1}^{\alpha-1-g_{s_1}} w_{s_2}^{\alpha-1-g_{s_2}}}{w_c^{\alpha-1-g_c}\,\text{beta}(\alpha, K^t\alpha)} \\ & \times \frac{\mathcal{N}_d(\mu_{s_1}; m_{s_1}, S_{s_1})\mathcal{N}_d(\mu_{s_1}; m_{s_2}, S_{s_2})}{\mathcal{N}_d(\mu_c; m_c, S_c)} \\ & \times \frac{\mathcal{W}_d^{-1}(\Sigma_{s_1}; (g_{s_1} - 2(1+d))S_{s_1}, g_{s_1})\mathcal{W}_d^{-1}(\Sigma_{s_2}; (g_{s_2} - 2(1+d))S_{s_2}, g_{s_2})}{\mathcal{W}_d^{-1}(\Sigma_c; (g_c - 2(1+d))S_c, g_c)} \\ & \times \frac{P_{combine}(K+1)}{P_{split}(K)P_{alloc}}\frac{1}{g_{2,2}(u_1)[\prod_{j=1}^{d} p(u_2(j))][\prod_{j=1}^{d}\prod_{i=1}^{j} g_{2,2}(U_3(i,j))]} \\ & \times w_c[u_1(1-u_1)]^{-p(p+2)/2}|L_c||D_d^{\dagger}T D_d|. \end{aligned} \tag{28}$$

The first three lines represent the ratio of the posteriors, the fourth line is the ratio $r_{split}(\hat{\Psi}^{t+1})/(r_{split}(\Psi^t)q(U^t))$, and the last line is the determinant of the Jacobian of $f(\cdot)$. In the last line, $T = \text{diag}(\text{vec}(L(I_d - u_2u_2^{\mathrm{T}})L^{\mathrm{T}}))$, D_d is the duplication matrix for a $d \times d$ symmetric matrix, $D_d^{\dagger}$ is the pseudo-inverse of D_d, the vec(X) operator stacks the columns of matrix X into a single column, and the diag(X) operator converts the $r \times 1$ vector X into an $r \times r$ diagonal matrix. For more detailed information on these matrix operators, refer to Lutkepohl (1996) or Magnus and Neudecker (1999). The term $P_{split}(K)$ is the probability of proposing the split move while the mixture has K components. The term P_{alloc} is the probability of choosing the particular reallocation of data samples and the particular component(s) to split or combine.

On the third line of (28), the prior ratios for Σ use a different parametrization than what was proposed in (9). Prior information on Σ restricts the size and shape of the covariance matrices in the split and combine moves, causing trapping states to frequently occur. We found that the empirical covariance was best suited as a parameter for the Wishart prior ratio; the altered prior specification allowed proposed covariance matrices that truly reflected the structure of the data to be accepted. It is unclear how much effect this reparametrization will have on the limiting distribution of K; however, we can make the claim that the bias will be insignificant provided that $v_{k,0}$ is chosen to reflect a lack of prior information in the posterior.

2.5. *Bayesian computation*

Figure 2 summarizes our algorithm. The implementation is straightforward given facilities for sampling from the inverse Wishart and Dirichlet distributions. The mixture components are labelled according to their relative positions on a chosen axis. For example, if the chosen axis



Realization of Markov Chain

$$\begin{tikzcd} \boxed{\Psi^0} \arrow[r] & \boxed{\Psi^1} \arrow[r] & \boxed{\Psi^2} \arrow[r] & \cdots \arrow[r] & \boxed{\Psi^{T-1}} \arrow[r] & \boxed{\Psi^T} \end{tikzcd}$$

- Each Ψ represents a sample from the posterior density.
- $\Psi = (\mathrm{K}, \Theta_K, Z)$, $\Theta_K = (\theta_k)_{k=1}^{K}$
- $\theta_k = (w_k, \mu_k, \Sigma_k)$

Step 1: Sample *K* using reversible-jump MCMC

a) Propose split, $\hat{K} = K^{t-1} + 1$, or combine, $\hat{K} = K^{t-1} - 1$, move.
b) Calculate proposal state parameters: If split, draw U from (22) and calculate $\Theta_{\hat{K}}$ using (21). If combine move, calculate U and $\Theta_{\hat{K}}$ using (23).
c) Calculate Metropolis-Hastings ratio, R, using (28).
d) Calculate transition probability, α: If split move, use (25), else (combine move), use (27).
e) Draw $\lambda \sim \mathrm{Unif}(0,1)$.
f) If $\alpha \le \lambda$, set $K^t = \hat{K}$, else set $K^t = K^{t-1}$.

Step 2: Sample (Θ_{K^t}, Z^t) using Gibbs Sampler

a) Sample W^t using (11).
b) Sample μ_k^t for $k = 1, \ldots, K^t$ using (12).
c) Sample Σ_k^t for $k = 1, \ldots, K^t$ using (13).
d) Sample z_n^t for $n = 1, \ldots, N$ using (14).
e) Update G_k and g_k for $k = 1, \ldots, K^t$.
f) Label mixture components

Initialize Simulation

- Set $\mathrm{K} = (1, 2, \ldots, K_{\max})$
- Choose $K^0 \in \mathrm{K}$
- $m = \frac{1}{N}\sum_{n=1}^{N} y_n$,
  $S = \sum_{n=1}^{N} (y_n - m)(y_n - m)^T$
- for $k = 1, \ldots, K^0$, $m_{k,0} = m$, $h_{k,0} = 1$,
  $v_{k,0} = 6$, $V_{k,0} = \frac{(v_{k,0} - 3)}{2} S$

Figure 2. Summary of the MCMC algorithm for estimating the anisotropic Gaussian mixture model with an unknown number of components.

were the x-axis, the mixture component with the smallest x-value would be assigned a label of 1, while the mixture component with the largest x-value would be assigned a label of K^t.

The MCMC satisfies irreducibility since the chain can, after some iteration, move to any K and Θ_K in the support of the invariant distribution. Aperiodicity is satisfied since there is a nonzero probability of staying in some arbitrarily small neighbourhood of the previous state. By construction of the RJMCMC, the property of reversibility is satisfied. These conditions are sufficient to ensure the MCMC is sampling from the correct invariant distribution.

2.6. *Bayesian inference*

Once the Markov chain has been realized, the distribution of the states can be used to infer the most probable mixture size and mixture parameters given the observed data and the anisotropic Gaussian assumption. Since the parameters of the mixture model were sampled from their respective full conditionals, estimation of the full marginals using Rao–Blackwellization or kernel density estimation will produce very similar results. We chose the most computationally efficient version of the kernel density estimation, normalizing the histogram, to estimate the predictive densities.

When the dimensionality of the data is greater than 2, inferring mixture parameters from [illegible]-dimensional, multi-modal predictive densities can be challenging. We recommend a scheme for isolating single modes on the surface of the predictive densities, and using the states associated to the most massive mode for estimation of the parameters. The first step is to

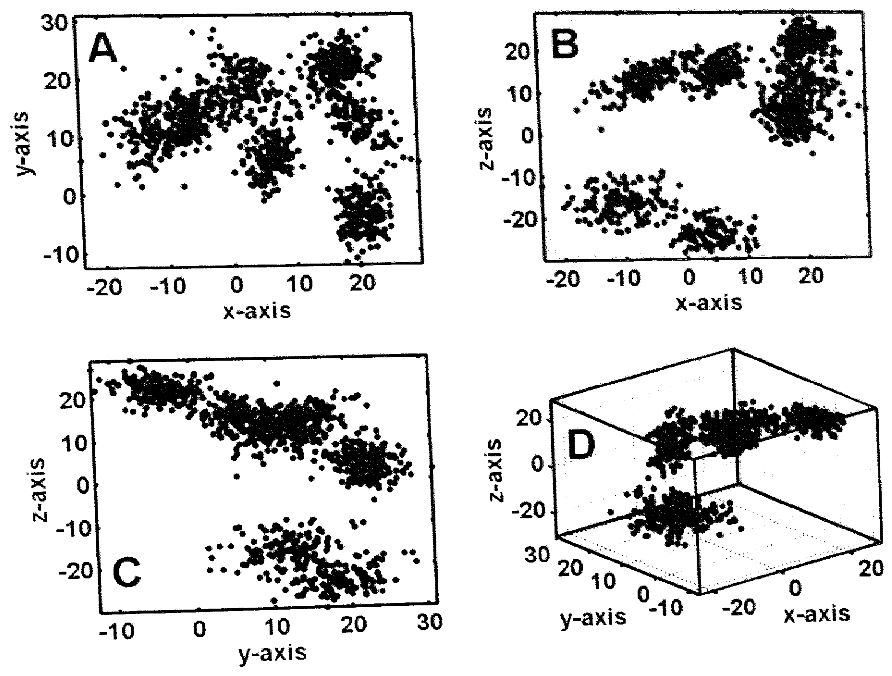


Figure 3. A set of 1000 simulated observations from a mixture containing seven anisotropic Gaussians.

estimate K from the mode of its normalized histogram. We then isolate the states of the Markov chain with K as the mixture size, and discard the irrelevant states. In the context of spike sorting, the parameter μ_k is very important as it determines the mean of the neural activity. Assume we are able to identify the states of the Markov chain where μ_k is confined to a finite and bounded area about the centre of cluster k. By averaging μ_k over these states, we can obtain an estimate of the centre of the cluster. Next, given μ_k is close to the cluster centre for a given state of the Markov chain, we can claim that w_k and Σ_k are also fluctuating about the true values for cluster k. Therefore, after isolating the states of the Markov chain that contribute to the largest mode in the predictive density of μ_k, we can estimate w_k and Σ_k via averaging methods. This approach uses only the relevant states in the MCMC to estimate Θ_K, thus, providing a reasonable basis for the resulting mixture model. An example of the process is illustrated in the inference procedure for example 2 in figure 8.

3. Examples

The following two examples demonstrate the applicability of our algorithm to spike sorting. The purpose of the first example is to provide the reader with some intuition about the performance of our algorithm and the inference process using a simple simulated data example. The simulated data set was produced with regard to our assumptions: clusters of activity are anisotropic Gaussian distributed and every data point belongs to a cluster (no outliers). The purpose of the second example is to illustrate concretely the characteristics of the MCMC algorithm. Only with a challenging data set, can we see the ways in which our algorithm will fail and succeed. We chose a real data set that contained considerable amounts of noise, widely dispersed outliers, and possibly non-Gaussian clusters.

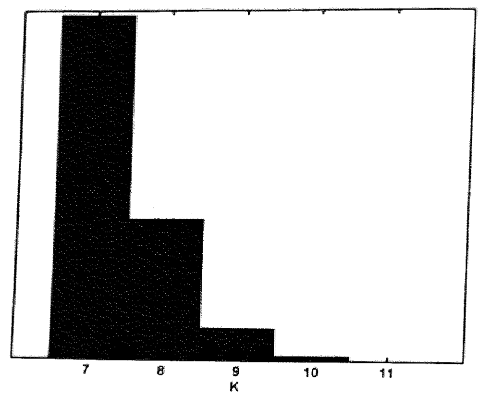


Figure 4. A histogram that estimates the full marginal posterior, $\pi_Y(K)$, from the states of the MCMC.

3.1. *Example 1: simulated data*

The 3D simulated data in figure 3 was produced with a total of seven Gaussian components. There are a total of 1000 samples in the dataset, with each mixture component responsible for at least 109 data points and at most 185 data points. The interesting features of this dataset include the variety of Gaussian shapes and overlapping of the components in each 2D projection.

We realized three Markov chains with lengths of 50 000. The components were labelled using the x-axis for each of the three chains. We treat the first 10 000 iterations of each chain as the burn-in period and retain every third iteration. The resulting chains or final chains have a length of 13 333. Figure 4 shows a histogram of the mixture sizes throughout the three chains.

We chose $K_f = 7$ and discarded any irrelevant states, e.g. states that did not have seven mixture components. The predictive densities for $\mu_k, k = 1, \ldots, 7$, were observed to be unimodal. Therefore, we can just average together to relevant states of the MCMC to get three different sets of mixtures, (Ψ_1, Ψ_2, Ψ_3), and data classifications, (Z_1, Z_2, Z_3). From each Z_q, we obtained the sets of data indices that define each cluster, $G_q^1, \ldots, G_q^7$. The components with the maximum overlap, $\max_{r,s,t} \#(G_1^r \cap G_2^s \cap G_3^t)$, are averaged together to obtain the final mixture, Ψ_f, and the final clusters are obtained by taking the intersection of the clusters, $G_f^r = (G_1^r \cap G_2^s \cap G_3^t)$, that were used to obtain Ψ_f.

Figure 5 shows the resulting clusters in different colours. The relationship between the true and estimated clusters can be easily seen in the overlap matrix

$$C_{T,B} = \begin{array}{c|ccccccc} {}_{\text{True}}\backslash^{\text{Bayes}} & \text{B1} & \text{B2} & \text{B3} & \text{B4} & \text{B5} & \text{B6} & \text{B7} \\ \hline \text{T1} & 0 & 0 & 0 & 0 & 1 & 109 & 1 \\ \text{T2} & 0 & 0 & 0 & 0 & 184 & 1 & 0 \\ \text{T3} & 0 & 0 & 0 & 151 & 0 & 0 & 0 \\ \text{T4} & 0 & 0 & 0 & 0 & 0 & 0 & 153 \\ \text{T5} & 0 & 0 & 109 & 0 & 0 & 0 & 0 \\ \text{T6} & 132 & 0 & 0 & 0 & 0 & 0 & 0 \\ \text{T7} & 0 & 159 & 0 & 0 & 0 & 0 & 0 \end{array}. \qquad (29)$$

where the element in the ith row and jth column is the number of data points that mixture component i from the original (true) mixture and j from the estimated (Bayesian) mixture

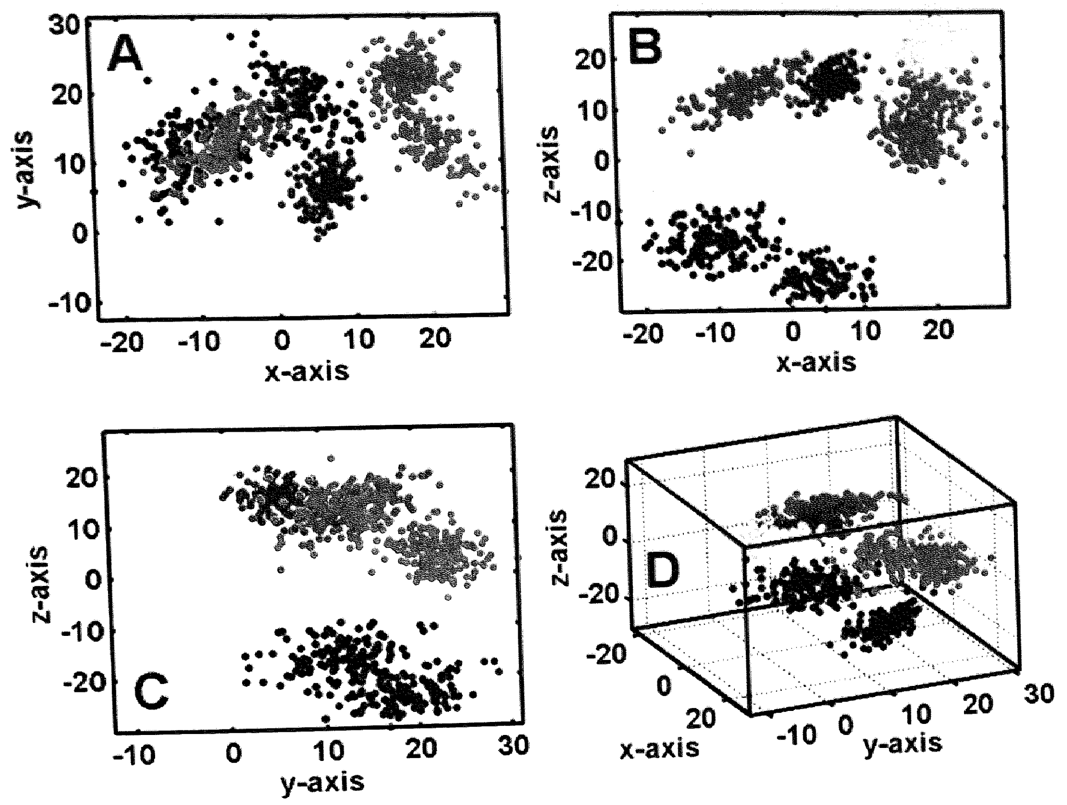


Figure 5. Feature vector clusters for simulated data, which were inferred from the predictive densities provided by the MCMC algorithm.

have in common. In (29), it is important to note that the labelling of the resulting clusters is not important, but rather which data points belong to the same cluster.

From the indices in (29), it is easy to see that the original clusters (except for three points) were successfully recovered by the MCMC algorithm but now have different labels. The three Markov chains were realized in parallel and on separate computers. The simulation required 6 h to finish, using Matlab 6.2 for Linux (Natick, MA) on computers equipped with Pentium III 900 MHz processors. The success of this example can primarily be attributed to the correct assumption about the anisotropic Gaussian structure. In the next example, this is not the case; the real data are extremely noisy and, due to pre-processing, can have clusters that lack the symmetry of a true anisotropic Gaussian cluster.

3.2. *Example 2: real data*

The real data used in experiment 2 were taken from the entorhinal cortex of a rat during a food foraging exercise in a U-shaped environment (Frank *et al* 2000). The recording device was a tetrode manufactured in a similar fashion to the method outlined in Gray *et al* (1995). The extracellular recordings were taken with 12 bit precision at 31.25 kHz. When any signal on the tetrode passed a threshold, eight samples before and 23 samples after the crossing were taken from each channel to create a 128-dimensional representation of the AP waveform.

Before the features were extracted, APs that did not cross a minimum threshold on all four channels were discarded. This ensures that the APs under consideration originated from neu-

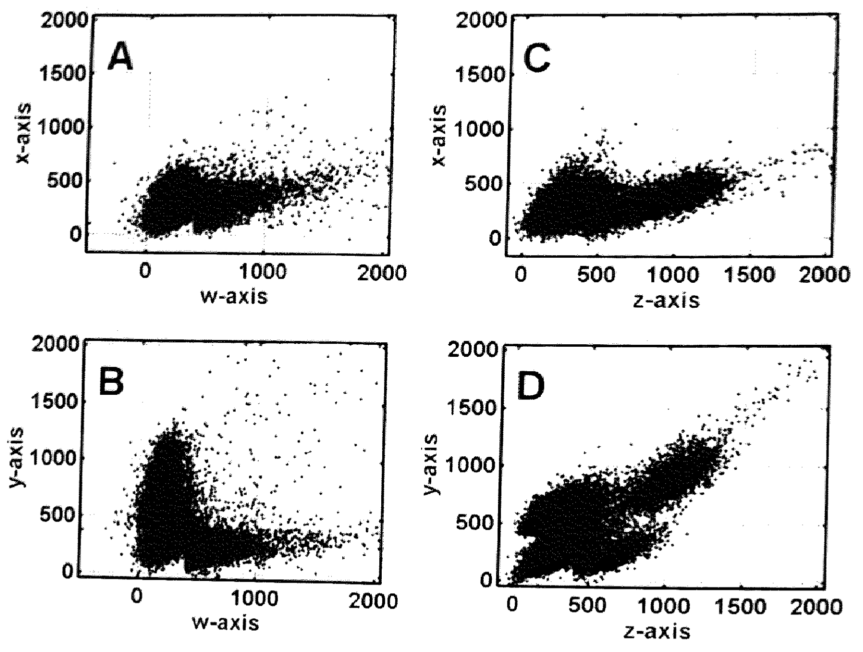


Figure 6. Feature vectors obtained by extracting the peak amplitude of action potential waveforms from tetrode recordings. There are a total of 30 000 4D points.

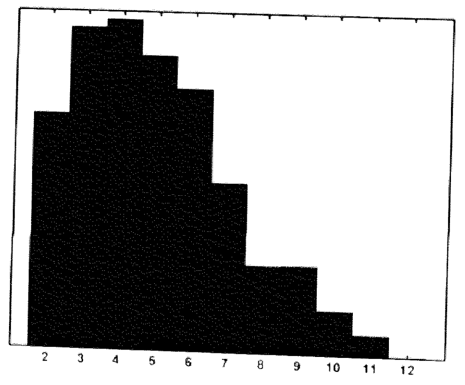


Figure 7. The predictive density for K. The mode at $K = 4$ is evidence that four neurons actively contributed to the data. This inference is based on the assumption that the neural activity of any individual neuron is anisotropic Gaussian distributed.

rons in a local sphere, centred about the tetrode. Array processing filters and noise filters were not used on the waveforms. Overlapping spike waveforms were neither detected nor decon- volved. From each AP waveform the maximum depolarization amplitudes were extracted from each channel to form a 4D feature vector set, $Y = (y_n)_{n=1}^{N}$, which served as the principal data used in our analysis (figure 6). We used maximum amplitude rather than the first principal com- ponent for purposes of illustration and comparison. Although principal components may serve to improve spike-sorting results, we feel that by using maximum amplitudes the MCMC algo- ithm will truly be pushed to its limits and reveal more of its characteristics. Moreover, the hand- sorting procedure was performed far in advance and used maximum amplitude as the feature.

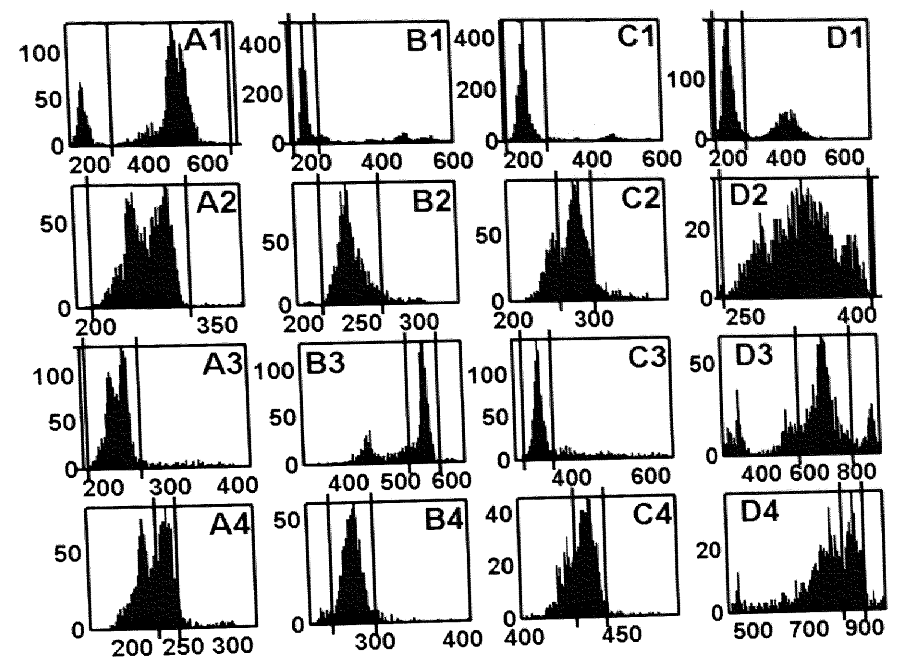


Figure 8. Inference of the mixture model using predictive densities of μ. Rows 1–4 correspond to the w, x, y, and z dimensions, respectively. Columns 1–4 correspond to different mixture components. Estimating the most probable value for w, μ, and Σ can be accomplished with a search for the highest peak on the high-dimensional surface of the posterior density. Instead, we find the states of the Markov chain that produced samples near the largest mode in the 4D probability surface of μ, and use those states to estimate w, μ, and Σ via averaging. We begin the search in the w-dimension, by drawing a boundary that contains the mode with the most mass. We use the samples inside the boundary to produce the predictive density for the x-dimension and isolate the most massive mode. This continues for the y and z dimensions. Notice that the number of samples used to define the predictive density decreases as we move to a lower row. As the boundaries are drawn and more samples are isolated, a hyper-rectangle is defined. The samples contained in this hyper-rectangle are traced back to the states of the Markov chain that produced them. These states are then averaged together to estimate w, μ, and Σ. Each mixture component is analysed independently; therefore, each mixture component will be estimated from a different set of states from the Markov chain.

The data set contains 30 000 data points. The computational efficiency of the MCMC is greatly reduced as the number of observations is increased, therefore we estimate the mixture parameters, K and Θ, using a thinned data set, which was obtained by retaining every 15th data point. The experiment was performed in a different manner from experiment 1; we labelled the mixture components according to their z-positions, we let $h_{k,0} = 20$ to reflect a lack of prior knowledge about the position of the mixture components, and due to time and computational constraints, we simulated only one chain with a length of 50 000.

The resulting histogram of the mixture components is shown in figure 7. We chose $K_f = 4$. The predictive densities for vector μ were observed to have multiple modes on a 4D surface. We located the mode with the largest mass by iteratively stepping through the predictive densities for each dimension of the vector μ and isolating the states that produced samples in areas of high probability for each dimension, figure 8. The effect is the creation of a hyper-rectangle that contains the largest mode in the predictive density of the vector μ. This step is fully explained in section 2.6 and the caption of figure 8. The samples of μ inside the hyper-rectangle were traced back to the states of the Markov chain. Those states were

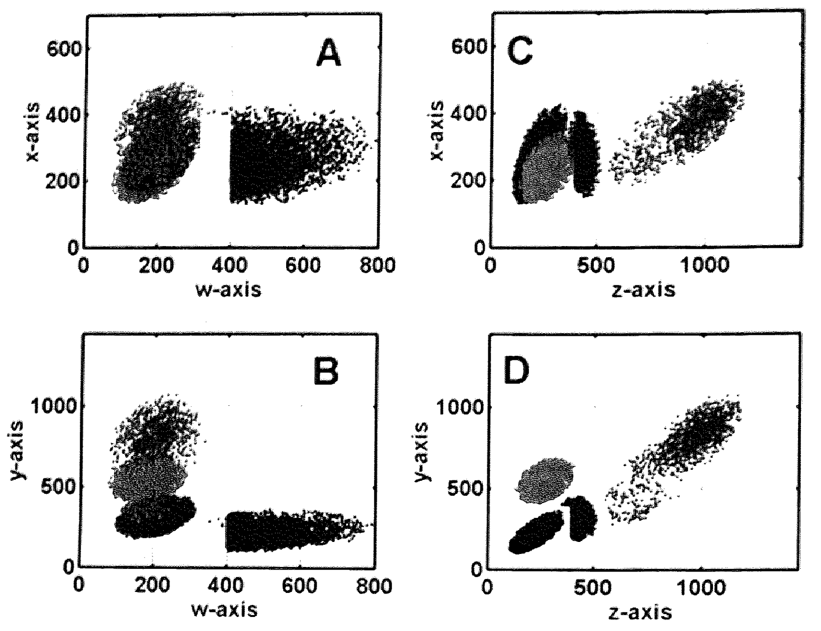


Figure 9. The clusters estimated from the real data with $K = 4$. The clusters were estimated from the process outlined in the caption of figure 7. The black, red, blue and green clusters contain 3798, 2639, 2702 and 1133 points, and correspond to columns 1–4 of figure 7, respectively. Although there are a total of six unique 2D projections for 4D data, we have chosen to display the clusters using only four projections. The colours of the clusters are assigned arbitrarily. The straight edge on the black and blue clusters is a result of spike detection via thresholding; the threshold for each channel was set at 400.

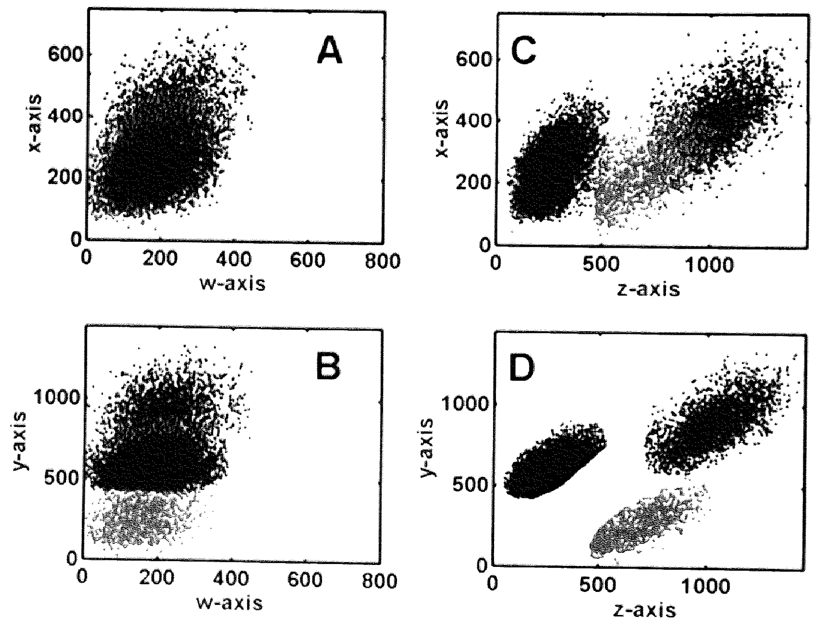


Figure 10. Hand-sorted spike clusters. The blue, red, and black clusters contain 2317, 997 and 6237 points, respectively. The colours of the clusters are arbitrarily assigned.

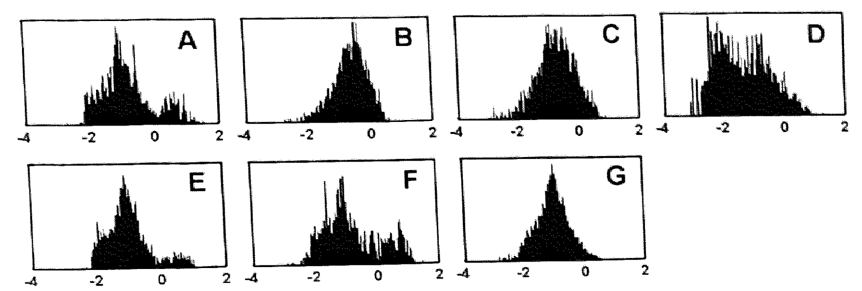


Figure 11. The distribution of $\log_{10}(\Delta t)$, where Δt is the inter-spike interval (ISI). These distributions are provided by the MCMC algorithm (top row, A = green, B = blue, C = red, D = black) and the hand-sorting procedure (bottom row, E = black, F = red, G = blue). The figure is organized such that clusters that are nearest in the feature vector space are contained in the same column. Panel (E) is a good example of the typical $\log_{10}(\Delta t)$ distribution for an entorhinal cortex (EC) neuron. Panel (E) has two distinguishing characteristics. The first is the non-uniform rise from 10 m (-2 tick mark) to 100 m (-1 tick mark). What this suggests is a strong tendency to fire in phase with the theta rhythm (8–12 Hz), which is a known property of EC neurons. The second characteristic feature of EC neurons is a tendency to be silent for relatively long periods of time. In panel (E), we see a significant amount of activity between 1 and 10 s. Only activity less than 3 ms should be considered indicative of corrupting noise.

then averaged together to produce the estimated mixture components. The components were then used to classify the 4D data using the method in appendix C. The resulting Bayesian estimated clusters (BECs) are shown in figure 9. An examination of the BECs reveal truncated Gaussian distributions for each cluster (data not shown), which is expected given the basis density function and method for discarding outliers.

We compare our results with the results of a hand-sorting procedure. The clusters in figure 10 were cut by hand using *X*-Clust, a user-dependent graphical data classification program. Examination of the hand-sorted clusters (HSCs) reveals truncated Gaussian and beta distributions for the individual dimensions of the components (data not shown). The colours given to the BECs and HSCs are arbitrary and have no significant meaning, *per se*. The number of overlapping spikes between the BECs and HSCs is shown in the indices of the overlap matrix

$$C_{B,H} = \begin{array}{c c} {}_{\text{Bayes}}\backslash^{\text{Hand}} & \begin{array}{ccc} H_{black} & H_{red} & H_{blue} \end{array} \\ \begin{array}{c} B_{black} \\ B_{red} \\ B_{blue} \\ B_{green} \end{array} & \left[\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 0 & 2173 \\ 0 & 0 & 0 \\ 972 & 67 & 0 \end{array}\right] \end{array}. \qquad (30)$$

We can see from $C_{B,H}$, that B_{green} and H_{black} have a large number of common spikes. Also, B_{red} and H_{blue} have a large number similar spikes. The components B_{green} and H_{red} have a small number of common spikes. The clusters B_{black} and B_{blue} have been entirely discarded by the hand-sorting procedure. The MCMC algorithm failed to recognize the H_{red} cluster. The fact that the BECs significantly overlap with HSCs cannot be translated into the BECs being valid. In order to confirm the validity of both the HSCs and BECs, we examine the ISI distributions for APs within each cluster.

Figure 11 contains plots of $\log_{10}(\Delta t)$ distributions for each HSC and BEC, where Δt is the interspike interval (ISI). Panel (E) is a good example of a typical $\log_{10}(\Delta t)$ distribution for an EC neuron. Using panel (E) as a template, we can conclude that clusters B_{black}, B_{red}, B_{blue} and H_{blue} are not valid clusters. After close examination, it is likely that cluster B_{black} is a real spike cluster that has been heavily contaminated with noise. This noise can easily be seen in figure 11, panel (D), at frequencies above 100 Hz. Because B_{green} and H_{black} correspond nicely in the feature vector space and in the shape of the $\log_{10}(\Delta t)$ distribution, we can regard B_{green} as a true spike cluster; however, from figure 9 it appears that B_{green} may contain false positive errors. Ultimately, the hand-sorting yielded two real clusters and one false cluster, and the Bayesian method yielded one real cluster and three false clusters.

What factors led to the poor clustering by the MCMC algorithm? The existence of B_{black} is an example of how noise can affect the MCMC algorithm. The MCMC algorithm is unable to distinguish a cluster of noise from true spike activity and, therefore, proceeds to cluster it. Outliers can degrade the performance of the MCMC algorithm appreciably. In fact, we can see the effect of outliers in the classification of B_{green} and B_{blue}. By construction of the mixture model, every observation needs to be assigned to a mixture component and mixture components cannot intersect. By assigning outliers to a mixture component, the size of the cluster will grow, the mean will tend to move away from the outliers, and the cluster will elongate in the direction of the outliers. This was the case for B_{green}. As B_{green} contained more outliers, it grew toward the origin and pushed B_{blue} away from its true centre. If there were no outliers, it is likely that B_{blue} would have corresponded nicely with H_{red}, and B_{green} would have an even greater correspondence with H_{black}.

Overall, given the challenging nature of the data set, the performance of our algorithm shows promise. Our algorithm found two clusters (B_{red} and B_{green}) that each contained overlaps with HSCs (H_{blue} and H_{black}), one noise-contaminated cluster (B_{black}), and one erroneous cluster (B_{blue}). We attribute the inability of our algorithm to locate the cluster corresponding to H_{red} to the influence of outliers on the ensemble position of the mixture components. Moreover, we observe that B_{green}, B_{red} and B_{blue} showed significant correspondence to H_{black}, H_{blue} and H_{red}, respectively, while B_{black} was ignored by the hand-sorting procedure. Therefore, based on these observations, we conclude that our algorithm is robust in determining the number of Gaussian components in the data, but lacks the ability to precisely locate each cluster when the data contain many outliers and the Gaussian assumption is incorrect.

4. Discussion

Our MCMC algorithm is an extension of the Richardson and Green (1997) algorithm for univariate Gaussian mixtures to multivariate, anisotropic Gaussian mixtures. Our MCMC algorithm is a hybrid sampler that uses RJMCMC to estimate the number of mixture components (neurons) and uses the Gibbs sampler to estimate the parameters of the mixture components. The MCMC algorithm allows us to estimate simultaneously the number of source neurons (clusters) and to assign each action potential waveform to a particular source. We illustrated our algorithm using both simulated and actual multi-unit tetrode data. The simulated example suggested that our MCMC algorithm produces predictive densities that are consistent with the true Gaussian structure in the data. The histogram for K peaks at 7 and the resulting mixture specification yields only three misclassified data points (figures 4 and 5). In example 2, our algorithm identified four spike clusters (figure 9). One of the four clusters consisted of mostly noise waveforms, whereas the three remaining clusters corresponded to HSCs that we had previously identified and used in data analyses.

Our algorithm, like those of Sahani *et al* (1997) and Shoham (2001), uses a mixture model to represent the action potential waveform. The main difference is that both the Sahani and Shoham approaches use the EM algorithm to perform parameter estimation for a fixed number of neurons. Sahani *et al* (1997) estimates the background noise in order to better estimate the covariance structure of the clusters. Shoham (2001) uses t-rather than Gaussian mixtures to model the waveforms, and uses deterministic annealing as a formal strategy to identify the

number of clusters. In the Bayesian procedure of Lewicki (1994), each neuron is represented by a piecewise-linear function that describes AP waveform shape, and the number of neurons is determined by the maximum of a Gaussian approximation to the marginal posterior density of different combinations of candidate waveforms. Fee *et al* (1996) also use a formal strategy to combine clusters. Their approach is initiated by randomly bisecting the waveform space until approximately ten times the expected number of final clusters is achieved. The abundant clusters are then combined based on a measure of connectivity. Fee *et al* (1996) and Sahani *et al* (1997) use ISI information to resolve waveform overlap and reduce misclassifications. This criterion for identifying false positives is both important and useful. We used it as well after the fact to identify misclassifications. Incorporating it explicitly into our algorithm is a topic of future research.

While the spike clusters are not necessarily Gaussian in nature (Shoham 2001), the Gaussian assumption allows our algorithm to identify dense clusters of feature vectors which we infer to arise from the same neuron. Hence, while the estimate of the number of clusters (source neurons) is likely to be correct, the accuracy of the actual spike sorting will vary.

Several factors can degrade the performance of our algorithm. From example 2, it is clear that noise can both contaminate clusters as well as lead to spurious separate clusters. Outliers in the data can also bias estimation of the mixture components. For example, when the outliers are high in amplitude, the cluster will grow in size, orient its major axis towards the outliers, and move its centre away from the origin. Overlapping waveforms can cause error, as the activities of two neurons are classified as arising from a single action potential. Thinning of the data to produce a smaller, yet perhaps non-representative data set may cause error. That is, non-representative thinning of the data can cause a change in the relative number of points observed from each neuron, which in turn may result in false negatives, or in clusters being undetected. Figures 9 and10 shows spike clusters that are not Gaussian in shape. This may be due to two factors: the clusters are truly non-Gaussian or the isolation of overlapping clusters results in truncated clusters. The choice of feature vectors can drastically change the shape, orientation and position of the clusters and, as a consequence, the spike classifications. Overlapping spike clusters caused by electrode drift or simply by neurons with similar observed activity can lead to either false negatives or false positive classifications.

We can address some of these sources of error. Background noise can be detected based on waveform shape and either removed by filtering (Bierer and Anderson 1999) or accounted for by modelling (Sahani *et al* 1997). Points that are obviously outliers can be removed prior to running the algorithm. A more principled thinning algorithm would be useful. For example, one could partition the feature vector space into hyper-cubes, count the number of points inside each cube prior to thinning, and impose the constraint that the relative proportion of points between cubes should be the same after thinning. Because data characteristics can drastically change depending on the brain region of the recording, the types of neurons and the hardware used, the process of choosing a set of features vectors is *ad hoc*. Because principal component analysis (PCA) is a second-order procedure it may provide plausible initial choices of feature vectors consistent with the Gaussian mixture assumption. One alternative to PCA is to develop a mapping strategy that projects the waveforms onto nonlinear manifold where a canonical representation can be defined (Tenenbaum *et al* 2000). A second alternative is to estimate the physical spatial location and orientation of the source neuron using a source-propagation model (Plonsey and Barr 2000). The choice of feature vectors can also be determined empirically by comparing choices with their ability to minimize apparent overlap between clusters.

The tradeoff between accuracy and practicality of our algorithm depends critically on the amount of data to be processed as well as the hardware and software used to implement it. When the dimension of the feature vector is increased by one or the number of data measurements increases by 500, our algorithm requires approximately 30% more computational time. The required computation time is also a function of the hardware, optimality of the source code, the coding language and the average number of clusters in the Markov chain. For large datasets, we use the thinning approach outlined in appendix C to obtain an approximate description of the mixture model from a subset of the data. We then use that approximate mixture to classify the entire dataset. This approach may result in a slight loss of accuracy in spike classification but will require less computation time. We found this approach to offer a reasonable compromise between accuracy and computational costs.

Several extensions of our MCMC algorithm are possible. First, the algorithm can be automated with a procedure that can find the global maximum in the multi-dimensional predictive density for μ, find the MCMC states in the vicinity of the maximum, and estimate the other mixture parameters from those states. Second, birth and death moves can be added to accelerate mixing. Implementation of these types of moves make the MCMC algorithm more practical for clustering feature vectors with dimensions larger than four. Third, in situations where the spatial relationship between the neurons and the multi-electrode are chronically stable, the algorithm can be extended to real-time application by estimating the mixture model from a learning period, treating the mixture model parameters as fixed for subsequent spikes, and clustering these spikes based on the estimated mixture model. Finally, a real-time update of the model parameters using dynamic Monte Carlo methods such as particle filters (Doucet *et al* 2001, Storvik 2002) can be interfaced with a real-time RJMCMC classification algorithm. This would allow the MCMC algorithm to adapt appropriately to the appearance and/or disappearance of spike clusters.

Acknowledgments

We thank Robert J Butera for his input and support throughout the course of this project. Support was provided in part by National Institute of Mental Health grants MH59733 and MH61637, National Institute of Drug Abuse grant DA015644, and National Science Foundation grant IBN-0081458 to ENB; National Institute of Mental Health grant MH65102 to LMF; and National Science Foundation grant DBI-9987074 to RJB.

Appendix A. Derivation of the Jacobian of the split transformation, equation (21).

The Jacobian matrix of the split transformation is

$$\frac{\partial\Psi_s}{\partial\Psi_c} = \begin{bmatrix} \frac{\partial w_{s_1}}{\partial w_c} & \frac{\partial w_{s_1}}{\partial u_1} & 0 & 0 & 0 & 0 \\ \frac{\partial w_{s_2}}{\partial w_c} & \frac{\partial w_{s_2}}{\partial u_1} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{\partial \mu_{s_1}}{\partial u_1} & \frac{\partial \mu_{s_1}}{\partial \mu_c} & \frac{\partial \mu_{s_1}}{\partial \mathrm{vech}(\Sigma_c)} & 0 \\ 0 & 0 & \frac{\partial \mu_{s_2}}{\partial u_1} & \frac{\partial \mu_{s_2}}{\partial \mu_c} & \frac{\partial \mu_{s_2}}{\partial \mathrm{vech}(\Sigma_c)} & 0 \\ 0 & 0 & 0 & \frac{\partial \mathrm{vech}(\Sigma_{s_1})}{\partial \mu_c} & \frac{\partial \mathrm{vech}(\Sigma_{s_1})}{\partial \mathrm{vech}(\Sigma_c)} & \frac{\partial \mathrm{vech}(\Sigma_{s_1})}{\partial \mathrm{vech}(U_3)} \\ 0 & 0 & 0 & \frac{\partial \mathrm{vech}(\Sigma_{s_2})}{\partial \mu_c} & \frac{\partial \mathrm{vech}(\Sigma_{s_2})}{\partial \mathrm{vech}(\Sigma_c)} & \frac{\partial \mathrm{vech}(\Sigma_{s_2})}{\partial \mathrm{vech}(U_3)} \end{bmatrix}$$

$$= \begin{bmatrix} Z & 0 \\ 0 & G \end{bmatrix}. \qquad (31)$$

Therefore, the determinant of the Jacobian is $|J_s| = |\partial\Psi_s/\partial\Psi_c| = |Z||G|$. We partition

G,

$$G = \begin{bmatrix} A & B \\ C & D \end{bmatrix} = \begin{bmatrix} A_1 & A_3 & B_1 & 0 \\ A_2 & A_4 & B_2 & 0 \\ 0 & C_3 & D_1 & D_3 \\ 0 & C_4 & D_2 & D_4 \end{bmatrix}, \qquad (32)$$

and write the determinant as $|G| = |A||D - CFB| = |A||H|$, where $F = A^{-1}$ and

$$H = \begin{bmatrix} H_1 & H_3 \\ H_2 & H_4 \end{bmatrix} = \begin{bmatrix} D_1 - C_3(F_3B_1 + F_4B_2), & D_3 \\ D_2 - C_4(F_3B_1 + F_4B_2), & D_4 \end{bmatrix}. \qquad (33)$$

Using matrix operations from Lutkepohl (1996) or Magnus and Neudecker (1999), we find that

$$|J_s| = |Z||A||H_4||H_1 - H_3H_4^{-1}H_2| = -w_c(u_1(1-u_1))^{-d(d+2)/2}|L||D_d^{\dagger}TD_d|, \qquad (34)$$

where $T = \mathrm{diag}(\mathrm{vec}(L(I_d - u_2u_2^{\mathrm{T}})L^{\mathrm{T}}))$ and I_d is the $d \times d$ identity matrix. To see that $|J_s|$ is not identically zero, consider the terms of $|J_s|$. L is full rank since it is the Cholesky factor of the full-rank covariance matrix, Σ. For any $p^2 \times p^2$ full-rank, diagonal matrix T, $D_d^{\dagger}TD_d$ is a full-rank diagonal matrix of size $r \times r$, where $r = d(d+1)/2$. We can now conclude that $|J_s|$ is not identically zero and the split/combine transformation is one-to-one and invertible.

Appendix B. 95% confidence ellipsoid of an *N*D anisotropic Gaussian distribution

The 95% confidence ellipsoid is parametrized using spherical coordinates, (θ, ϕ, ρ). Given the covariance matrix, Σ, of a *N*D Gaussian density, the ρ which defines the 95% confidence ellipsoid at the angles θ and ϕ is

$$\rho_{95\%}(\Sigma, \theta, \phi) = \left(\frac{c_{95\%}^{N}}{s^{\mathrm{T}}\Sigma^{-1}s}\right)^{1/2}, \qquad (35)$$

where $s = [\sin\phi\cos\theta, \sin\phi\sin\theta, \cos\phi]^{\mathrm{T}}$ and $c_{95\%}^{N}$ is the constant that defines the 95% confidence ellipse for Gaussian-distributed N-variate. The observation in Cartesian coordinates, $y \in \Re^3$, falls within the 95% confidence ellipse if the inequality

$$\rho_{95\%}(\Sigma, \theta_\Delta, \phi_\Delta) \geqslant \rho_\Delta \qquad (36)$$

is satisfied, where $\Delta = \frac{(y-\mu)}{\|y-\mu\|_2}$ and $(\theta_\Delta, \phi_\Delta, \rho_\Delta)$ is the transformation of Δ to spherical coordinates.

Appendix C. Classification of additional data

The computational efficiency of the MCMC algorithm is inversely related to the number of observations. We propose a three-step approach to clustering all the observation in the interval $(0, T]$ using only a subset of the observations. We assume that the properties of the mixture are stationary over the interval $(0, T]$.

The first step is to extract a set of observations, $Y_{(t_1,t_2]}$, where $0 \leqslant t_1 < t_2 \leqslant T$, such that the set $Y_{(t_1,t_2]}$ is sufficiently representative of the entire set $Y_{(0,T]}$ and the size of $Y_{(t_1,t_2]}$ leads to the desired efficiency. The second step is to use the MCMC algorithm (section 4) to obtain the mixture size and mixture parameters, $\hat{K}$ and $\hat{\Theta}$. The third step is to classify the entire data set using the following procedure:

```
(1) Initialize
    • $i = 1$.
(2) For $i \leqslant \#Y_{(0,T]}$
    • Set $z_i = k \mid \arg_k \max(p(k))$, where $p(k) = \hat{w}_k N(y_i; \hat{\mu}_k, \hat{\Sigma}_k)$.
    • If $y_i$ does not fall in the 95% confidence ellipsoid of the mixture component identified
      by $z_i$, set $z_i = 0$.
    • $i = i + 1$.
```

References

Andersen T W 1984 *An Introduction to Multivariate Statistical Analysis* (New York: Wiley) pp 244–79

Bierer S M and Anderson D J 1999 Multi-channel spike detection and sorting using an array processing technique *Neurocomputing* **26/27** 947–56

Diebolt J and Robert C P 1994 Estimation of finite mixture distributions through Bayesian sampling *J. R. Stat. Soc.* **56** 363–75

Doucet A, de Freitas N and Gordon N 2001 *Sequential Monte Carlo in Practice* (New York: Springer)

Escobar M D and West M 1995 Bayesian density estimation and inference using mixtures *J. Am. Stat. Assoc.* **90** 577–88

Fee M S, Mitra P P and Kleinfeld D 1996 Automatic sorting of multiple neuronal signals in the presence of anisotropic and non-Guassian variability *J. Neurosci. Methods* **69** 175–88

Frank L M, Brown E N and Wilson M A 2000 Trajectory encoding in the hippocampus and entorhinal cortex *Neuron* **27** 169–78

Gelfand A E, Hills S E, Racine-Poon A and Smith A F M 1990 Illustration of Bayesian inference in normal data models using Gibbs sampling *J. Am. Stat. Assoc.* **85** 972–85

Gray C M, Maldonado P E, Wilson M and McNaughton B 1995 Tetrodes markedly improve the reliability and yield of multiple single-unit isolation from multi-unit recordings in cat striate cortex *J. Neurosci. Methods* **63** 43–54

Green P 1995 Reversible jump Markov chain Monte Carlo computation and Bayesian model determination *Biometrika* **82** 711–32

Hastings W K 1970 Monte Carlo sampling methods using Markov chains and their applications *Biometrika* **57** 97–109

Kotz S, Balakrishnan N and Johnson N L 2000 *Continuous Multivariate Distributions* vol 1, 2nd edn (New York: Wiley) pp 485–91

Lavine M and West M 1992 A Bayesian method for classification and discrimination *Can. J. Stat.* **20** 451–61

Lewicki M S 1994 Bayesian modeling and classification of neural signals *Neural Comput.* **6** 1005–30

Lewicki M S 1998 A review of methods for spike sorting: the detection and classification of neural action potentials *Network: Comput. Neural Syst.* **9** R53–78

Lutkepohl H 1996 *Handbook of Matrices* (New York: Wiley)

Magnus J R and Neudecker H 1999 *Matrix Differential Calculus with Applications in Statistics and Econometrics* 2nd edn (New York: Wiley)

McLachlan G J and Krishnan T 1997 *The EM Algorithm and Extensions* (New York: Wiley)

Metropolis N, Rosenbluth A W, Rosenbluth M N, Teller A H and Teller E 1953 Equations of state calculations by fast computing machines *J. Chem. Phys.* **21** 1087–91

Plonsey R and Barr R C 2000 *Bioelectricity: A Quantitative Approach* 2nd edn (Boston, MA: Kluwer)

Richardson S and Green P 1997 On Bayesian analysis of mixtures with an unknown number of components *J. R. Stat. Soc.* **59** 731–92

Robert C P 1996 Mixtures of distributions: inference and estimation *Markov chain Monte Carlo in Practice* vol 24, ed W R Gilks, S Richardson and D J Spiegelhalter (New York: Chapman and Hall) pp 441–64

Robert C P and Casella G 1999 *Monte Carlo Statistical Methods* (New York: Springer)

Sahani M, Pezaris J S and Andersen R A 1997 On the separation of signals from neighboring cells in tetrode recordings *Neural Information Processing Systems (NIPS 1997)* vol 10, ed M Jordan, M Kearns and S Solla (Cambridge, MA: MIT Press) p 11

Shoham S 2001 Advances towards an implantable motor cortical interface *PhD Thesis* University of Utah, Salt Lake City, UT

Storvik G 2002 Particle filter for state-space models with the presence of unknown static parameters *IEEE Trans. Sig. Process.* **50** 281–9

Tenenbaum J B, de Silva V and Langford J C 2000 A global geometric framework for nonlinear dimensionality reduction *Science* 2319–23

INSTITUTE OF PHYSICS PUBLISHING NETWORK: COMPUTATION IN NEURAL SYSTEMS

Network: Comput. Neural Syst. **14** (2003) 83–102 PII: S0954-898X(03)57828-9

Likelihood approaches to sensory coding in auditory cortex

Rick L Jenison[1] **and Richard A Reale**

Departments of Psychology and Physiology, Waisman Center University of Wisconsin-Madison, 1202 W Johnson Street, Madison, WI 53706, USA

E-mail: rjenison@facstaff.wisc.edu

Received 19 March 2002, in final form 9 November 2002
Published 17 January 2003
Online at stacks.iop.org/Network/14/83

Abstract
Likelihood methods began their evolution in the early 1920s with R A Fisher, and have developed into a rich framework for inferential statistics. This framework offers tools for the analysis of the differential geometry of the full likelihood function based on observed data. We examine likelihood functions derived from inverse Gaussian (IG) probability density models of cortical ensemble responses of single units. Specifically, we investigate the problem of sound localization from the observation of an ensemble of neural responses recorded from the primary (AI) field of the auditory cortex. The problem is framed as a probabilistic inverse problem with multiple sources of ambiguity.

Observed and expected Fisher information are defined for the IG cortical ensemble likelihood functions. Receptive field functions of multiple acoustic parameters are constructed and linked to the IG density. The impact of estimating multiple acoustic parameters related to the direction of a sound is discussed, and the implications of eliminating nuisance parameters are considered. We examine the degree of acuity afforded by a small ensemble of cortical neurons for locating sounds in space, and show the predicted patterns of estimation errors, which tend to follow psychophysical performance.

1. Introduction

The process by which neural responses are translated into a perceptual decision or estimate can be viewed as a solution to a probabilistic inverse problem—that is, given a set of responses, what is the most likely state of the environment that evoked that response? Historically, Bayesians were the first to frame such problems in inferential statistics, and central to the Bayesian approach is the use of axiomatic prior probabilities. One aim in statistical inference is to infer something about the value of a parameter, which is unknown, based on observed

[1] Author to whom any correspondence should be addressed.