Latent Variable Models for Neural Data Analysis

Thesis by

Maneesh Sahani

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

California Institute of Technology

Pasadena, California

1999

(Submitted May 14, 1999)

Grau, teurer Freund, ist alle Theorie,
Und grün des Lebens goldner Baum.

Grey, dearest friend, is all of Theory,
And green, the golden Tree of Life.

Mephistopheles, in Goethe's *Faust.*

Acknowledgements

Many thanks are due to my advisors, Richard Andersen and John Hopfield. Richard has been tremendously supportive throughout, providing valuable advice and encouragement, even when my work has taken me outside the domains typically trodden by members of his group. Furthermore he has assembled an exceptional group of people in whose company I have worked for the last five years. John graciously provided an intellectual home away from home; as well as invaluable advice and mentoring on a number of thorny research and professional issues.

The work on spike sorting that is presented in chapter 5 is a component of a larger multiunit recording project within Richard Andersen's laboratory on which John Pezaris and I have collaborated for the past five years. This has been a tremendously fruitful and rewarding collaboration. I could not have wished for a more able or more generous colleague. The analysis of chapter 6 was also a collaborative effort, in this case with Jennifer Linden. This too was an extremely rewarding effort; in both the intellectual and personal domains my debt to Jennifer is immeasurable.

A number of critical conversations have helped to shape the work that appears here. In the first place, the fact that I chose to study neuroscience at all is largely due to the encouragement of Jim Bower, Christof Koch and Bill Bialek. Sam Roweis introduced me to the EM algorithm which forms the core thread running throughout this thesis. Bill Bialek steered me towards the simple greedy approach to spike detection that appears in section 5.13. Tali Tishby brought earlier work on deterministic annealing to my attention, from which the REM algorithm of chapter 3 grew. Sanjoy Mahajan convinced me to take a closer look at the possibility of modeling bursting with hidden Markov models (section 5.10.2). Alan Barr provided a piece of inspirational advice at a critical juncture (see the Preface). Ken Miller and his group at UCSF have been working on the problem of spike sorting over the last few years as well, and we have enjoyed very many useful exchanges of ideas.

Besides these specific points, innumerable conversations with my peers at Caltech have contributed to my understanding involved. Special thanks are due to Sam Roweis and Flip Sabes, who provided crucial support as I made my initial faltering steps into the realm of machine learning and statistics. Members of the Hopfield group, particularly Sam Roweis, Carlos Brody, Erik Winfree and Sanjoy Mahajan, provided an exceptional environment in which to develop my skills in theoretical work. Larry Snyder in the Andersen lab was primarily responsible for my introduction to the practical details of electrophysiology.

A number of people read earlier versions of the chapters that appear in this dissertation: Flip

Sabes, Jennifer Linden, Sam Roweis, John Pezaris and Murray Jarvis. Their feedback improved the clarity of the exposition considerably. All confusions, and certainly any errors, that remain are entirely my responsibility.

Finally, I must acknowledge a great debt to the very many people who have contributed to the stimulating intellectual environment that I have enjoyed during my postgraduate years at Caltech. These include all of the members of the Andersen and Hopfield groups, the faculty and students of the CNS program; members of the Sloan Centres for Theoretical Neuroscience, both at Caltech and elsewhere; and many students in the Biology, Electrical Engineering and other departments with whom I have enjoyed the opportunity to interact.

Abstract

The brain is perhaps the most complex system to have ever been subjected to rigorous scientific investigation. The scale is staggering: over 10^11 neurons, each making an average of 10^3 synapses, with computation occurring on scales ranging from a single dendritic spine, to an entire cortical area. Slowly, we are beginning to acquire experimental tools that can gather the massive amounts of data needed to characterize this system. However, to understand and interpret these data will also require substantial strides in inferential and statistical techniques. This dissertation attempts to meet this need, extending and applying the modern tools of latent variable modeling to problems in neural data analysis.

It is divided into two parts. The first begins with an exposition of the general techniques of latent variable modeling. A new, extremely general, optimization algorithm is proposed — called Relaxation Expectation Maximization (REM) — that may be used to learn the optimal parameter values of arbitrary latent variable models. This algorithm appears to alleviate the common problem of convergence to local, sub-optimal, likelihood maxima. REM leads to a natural framework for model size selection; in combination with standard model selection techniques the quality of fits may be further improved, while the appropriate model size is automatically and efficiently determined. Next, a new latent variable model, the mixture of sparse hidden Markov models, is introduced, and approximate inference and learning algorithms are derived for it. This model is applied in the second part of the thesis.

The second part brings the technology of part I to bear on two important problems in experimental neuroscience. The first is known as spike sorting; this is the problem of separating the spikes from different neurons embedded within an extracellular recording. The dissertation offers the first thorough statistical analysis of this problem, which then yields the first powerful probabilistic solution. The second problem addressed is that of characterizing the distribution of spike trains recorded from the same neuron under identical experimental conditions. A latent variable model is proposed. Inference and learning in this model leads to new principled algorithms for smoothing and clustering of spike data.

Preface

In the course of mid-graduate school angst about the direction of my research, I was offered the following advice by Professor Alan Barr. It is all too easy in multidisciplinary work, he warned, to take simple ideas from one field and use them to impress researchers in another. While sometimes producing useful work, this is not the most rewarding way to cross the boundaries between fields. Instead, he urged me to strive to make solid contributions to both areas of research, so that workers in either might appreciate the advances made in the study of their own subject. It is my hope that the research described within this dissertation comes some distance towards this standard. In particular, I hope that both the statistician (or machine learning theorist) and the experimental biologist will find within these pages original ideas and contributions of interest to them.

In acknowledgement of this goal, the dissertation is arranged in two parts, very nearly equal in length. The first is entirely concerned with the statistical modeling of data; the techniques described, including the novel proposals, are of general applicability. Nonetheless, the development of these techniques has been driven by the desire to solve very specific problems in biology. The second part presents models for the analysis of two different types of neural data; in both cases, this development draws heavily on the statistical tools presented in part I.

The statistical content of this dissertation is as follows.

- Chapter 1 begins with a review of a significant portion of the theory of statistical modeling. I treat both parameter estimation and model selection in the general case. The focus then shifts to latent variable models in particular, and the Expectation–Maximization (EM) algorithm for maximum-likelihood estimation. In one way or another, this algorithm will form the basis for most of the original work in the dissertation. The free energy formulation of EM is described along with resulting extensions, such as the generalized and incremental variants.

- Chapter 2 introduces the problem of data clustering, and begins by reviewing some older algorithms. This leads to a restatement of the clustering objective as a problem in statistical modeling, thereby arriving at the simplest of latent variable models, the mixture model. The standard EM algorithm for the mixture is derived. Finally, I raise and address a number of important practical issues that arise in the use of mixture models for clustering.

  Both chapters 1 and 2 contain, for the most part, reviews of the existing literature. The only novel contribution is to be found in the detailed analysis of the problem of outliers in clustering (section 2.7.1); and even this follows closely suggestions that have appeared before.

- In chapter 3, I introduce the **Relaxation Expectation–Maximization** (REM) algorithm. This is perhaps the single most significant and widely applicable of the original contributions to statistics. It provides a relaxation-based generalization of any EM algorithm, frequently leading to good maximum-likelihood solutions without convergence difficulties due to local maxima. Furthermore, it may be combined with standard model selection techniques to yield a novel framework called **cascading model selection**. This approach further improves the quality of the maxima found by REM, and also allows for the appropriate model size (for example, the number of components in a mixture) to be determined in parallel with the parameter optimization.

- Chapter 4 opens with a brief review of the standard hidden Markov model (HMM) development. I then introduce a special case of HMM, the **sparse hidden Markov model** (SHMM), in which most of the output symbols assume a single, null, value. The development of algorithms specific to this model forms the original content of this chapter. It is shown that inference and learning in such models can be accelerated as a result of the sparse structure. The primary algorithmic interest in the SHMM, however, lies in the fact that a good approximate learning scheme can be derived for mixtures of such models. In particular, I derive an EM algorithm for such a mixture, with a constrained E-step given by the novel **coupled forward–backward** algorithm.

- Some novel statistical ideas also appear in the course of the second part of the dissertation. Section 5.7.2 introduces both a scheme to approximate the optimal linear discriminant space for certain types of clustered data, without knowledge of the cluster memberships; and also **robust principal component analysis**, a version of the usual principal component analysis in which outlier points are gracefully discounted. Section 5.12 discusses incremental and adaptive versions of EM for mixtures and for SHMMs. Chapter 6 examines the inhomogeneous Polya point process in some detail. Finally, a novel **Monte-Carlo goodness of fit procedure** is proposed and applied in section 6.4.

In the second half of the dissertation, the primary focus shifts to biology, and original solutions are proposed to two important problems in experimental neuroscience.

- Chapter 5 addresses the problem of **spike sorting**; that is, distinguishing between the spike waveforms that arise from different neurons in an extracellular recording. Historically, this has, for the most part, been done by hand, and with relatively little understanding of the statistical properties of the data. No thorough statistical treatment of this subject has been presented, and in this dissertation I seek to rectify the omission.

  - A number of important signal processing issues that arise in the handling of spike data,

and that are often neglected, are addressed. These include the appropriate thresholding of multi-channel data (section 5.5); the alignment of spike waveforms to reduce discretization "noise" (section 5.7.1); and the reduction in dimensionality of the waveform space so as to maintain the greatest separability between clusters (section 5.7.2).

- A novel latent variable model schema is proposed for the generating process, which captures the expected structure of the variation in spike shapes. Learning the parameters (that is, the firing statistics and distribution of spike waveforms) in this schema can be decoupled from the inference of the exact spike times: the first occurs in a mixture model in which overlapped spikes are simply treated as outliers, the second in a matched-filter-based scheme where overlaps can be correctly resolved.
- Three specific models within the schema are examined, and learning and inference algorithms for these derived. The first (section 5.8) presumes that all of the variability in observed spike shape is due to the addition of the background process: spikes from cells too distant to be distinguished and noise from electrical sources. The more sophisticated examples provide models for the intrinsic variability of spike waveforms. In particular, in section 5.10.2, an instance of the sparse hidden Markov model is used to explicitly model the change in spike waveform during a burst.
- The problem of on-line adaptation of the parameters of these models is discussed. This adaptation allows the algorithm to track slow drift in the spike waveforms that might occur due to motion of the electrode in neural tissue over a long timescale.
- Finally, a greedy, approximate filtering scheme is proposed for spike-time inference. This scheme is well-suited to real-time operation on parallel signal processors.

These investigations result in a new tool-box of statistical techniques which can be applied to automatically resolve an extracellular recording into its constituent spikes. Such techniques are crucial, both to the reliable scaling of scientific data acquisition to the hundreds of cells or more, as well as to the realization of the biomedical engineering dream of neural prosthetic devices.

- In chapter 6, I turn to the problem of characterizing the distribution of spike times invoked in a cortical neuron by constant experimental conditions. The model examined is an old one in the statistical literature, but appears to be new to this particular application: it is the randomly scaled inhomogeneous Poisson process, or Polya process. This choice is inspired by recent experiments which have suggested that, at least in part, the super-Poisson variability in firing rate can be accounted for by slow changes in the overall excitability of a neuron or cortical area. Combined with a Gaussian-process prior that enforces slow variation in firing

rate, learning in such a model leads to a statistically rigorous method for the smoothing of spike trains. Through Monte-Carlo simulations, it is shown that while the model is not exact, it is reasonably able to describe the data. A mixture of Polya processes can also be used as the basis for the clustering of spike trains, a problem which has been tackled unsatisfactorily by a number of authors in the past. The proposed approach avoids many of the difficulties which have hampered previous efforts, and it shown that this procedure leads to a believable grouping of real data.

The material in chapter 6 was developed in collaboration with J. Linden, and more extensive applications demonstrating the value of this simple model appear in her doctoral dissertation.

Contents

List of Figures

Part I

Statistics

Chapter 1 Latent Variable Models

1.1 Statistical Modeling

We are given a set of observations $\mathcal{X} = \{x_i \mid i = 1 \ldots |\mathcal{X}|\}$. The data x_i may be multivariate and are not necessarily independent. We are interested in learning about the nature of the process that gave rise to these data. In particular, we believe that by investigating the relationships that exist between the various components of the x_i, or between the different x_i, we can arrive a succinct description of the data, and that this description will reveal the structure of the generating process. In this quest we shall follow a path that lies at the intersection of two fields: unsupervised learning and density estimation.

In the machine learning literature, the project that we have laid out is known as **unsupervised learning**. We shall focus on a subset of the machine learning techniques, defined by our belief that the underlying generative process is **stochastic**, where we seek to learn an explicit probabilistic model that describes the data. This will exclude from our purview some effective techniques, for example the Kohonen and ART networks; in general, however, there are probabilistic formulations that very closely resemble each of these, and so we expect the loss not to be too serious. In return, we gain access to a powerful collection of probabilistic analysis tools.

Thus, we seek a description of the probability distribution (or density, for continuous observations) function $\mathsf{P}(\mathcal{X})$[1]. As such, our objectives are similar to those of the field of **density estimation**. However, it is not the resultant distribution (or density) function that holds our interest, but rather the structure of the function and what that structure reveals about the process that generated the data. Thus, we will not pursue many useful, "non-parametric" techniques of density estimation on the basis that these will give us little insight into the underlying process.

It is important to note that the general task of density estimation – given data $\mathcal{X}$, estimate $\mathsf{P}(\mathcal{X})$ – is not well formed unless something is known *a priori* about the probability function. This prior knowledge may be as simple as a belief that the function must be smooth, but in the absence of any prior, any scheme for ranking two candidate distributions will fail at least as often as it will succeed. This point is made clearly by Wolpert (1996). In our case, the prior knowledge, dictated by scientific experience and intuition, will go towards the selection of one or more families of parameterized probability functions $\mathsf{P}_\theta(\mathcal{X})$. θ here denotes a set of parameters, each of which

[1] We shall use the notation $\mathsf{P}(\cdot)$ generically for probability distribution and density functions. The exact nature of the function should be clear from context and the arguments provided, when this is not so we shall identify particular functions with a subscript such as $\mathsf{P}_\theta(\cdot)$

may be discrete or continuous. There are two central problems to be addressed in the project of statistical modeling: the first, called **learning** or **fitting**, is to estimate a suitable set of parameters $\widehat{\theta}$, or, if one is of the Bayesian persuasion, a posterior distribution over the parameters $\mathsf{P}(\theta \mid \mathcal{X})$, that is appropriate for the observed data. The second, **model comparison**, is to choose from among a group of candidate models the one which is better supported by, or more probable given, the data. It is worth noting that in the strict Bayesian viewpoint there is no difference between these operations: we can simply introduce a **hyper-parameter** that identifies which model is to be used and then infer its posterior distribution. However, we are interested in the properties of the particular model that best describes the data, and so although we might accept a distribution over parameters, we insist on identifying a single best model.

1.2 Parameter Estimation

We are given a set of observations $\mathcal{X}$, along with a parameterized family of probability functions $\mathsf{P}_\theta(\mathcal{X})$. We would like to infer an "optimal" value of the parameters such that the corresponding function describes the data best. There are many competing definitions of "optimal" in this context.

It will be simplest to survey these definitions by starting from the Bayesian viewpoint. In the Bayesian framework, the parameters θ are treated as random variables, to be handled on a similar footing to the observations $\mathcal{X}$. In this case we can more aptly write our family of distributions as $\mathsf{P}_M(\mathcal{X} \mid \theta)$, where the subscript M identifies the particular model. Bayes' rule then allows us to find a **posterior** distribution of the θ,

$$\mathsf{P}_M(\theta \mid \mathcal{X}) = \frac{\mathsf{P}_M(\mathcal{X} \mid \theta)\,\mathsf{P}_M(\theta)}{\mathsf{P}_M(\mathcal{X})} \tag{1.1}$$

The function $\mathsf{P}_M(\theta)$ denotes the probability associated with particular value of the parameters under the model M *a priori* – that is, without reference to any observations. It is called the **prior** distribution. Similarly, $\mathsf{P}_M(\theta \mid \mathcal{X})$ gives the probability of the parameter values θ in the context of the observed data. This is the *a posteriori* or simply **posterior** distribution. The term $\mathsf{P}_M(\mathcal{X} \mid \theta)$ is the familiar function that describes the distributions within our model, however in the context of parameter estimation by (1.1) it is best viewed as a function of θ, rather than of $\mathcal{X}$. In this context it is given a different name; it is called the **likelihood** of the parameters in light of the data, and will be written $\mathcal{L}_\mathcal{X}(\theta)$ to emphasize the exchange of rôles between θ and $\mathcal{X}$. It is important to note that the the numerical value of the probability of data $\mathcal{X}$ under parameters θ, $\mathsf{P}_\theta(\mathcal{X})$ or $\mathsf{P}(\mathcal{X} \mid \theta)$, is identical to that of the likelihood of parameters θ given data $\mathcal{X}$, $\mathcal{L}_\mathcal{X}(\theta)$. The difference is only one of interpretation. The final term in (1.1) is the denominator $\mathsf{P}_M(\mathcal{X})$. This is also given a name, but one that will only really be relevant when we discuss model selection below. It is called the

evidence for the model M, or else the **marginal likelihood**, since it is obtained by integrating the likelihood with respect to θ. From the point of view of parameter estimation from observations it is usually of little importance, as it has a constant value with no dependence on the parameters.

In the strict Bayesian point of view the equation (1.1) represents all that there is to be said about parameter estimation. Once we know the posterior distribution of the parameters we have exactly expressed the complete extent of our knowledge about their value. In this view, any attempt to provide a single parameter estimate as a description of the situation must give up some useful information. This is most apparent if we ask how the parameter estimate is to be used. Typically, we are interested in predicting the value of some statistic that is dependent on the parameters; it might, for example, be the next data point to be drawn from the distribution. In this case we need to integrate over the posterior (this will also be true for model selection, treated below). Let us call the statistic that we wish to predict k. The probability distribution that describes our prediction will be

$$\mathsf{P}_M(k \mid \mathcal{X}) = \int d\theta\, \mathsf{P}_M(k \mid \theta)\, \mathsf{P}_M(\theta \mid \mathcal{X}) \tag{1.2}$$

Here we see the practical difficulty in the strict Bayesian point of view. For many models, this integral is impossible to compute exactly. One approach taken is to approximate the integral by a Monte-Carlo sampling technique such as the Gibbs or Metropolis samplers, or by various so-called "hybrid" Monte-Carlo methods (Gelfand and Smith 1990; Smith and Roberts 1993; Neal 1996). Such methods are asymptotically exact, although the number of samples needed to reach the asymptotic distribution can be probitively large.

In practice, we often use a single estimate of the values of the parameters. This approach may be understood from one of two points of view. In the first case, we will argue below that a suitable choice of estimate can, under certain circumstances, actually provide a reasonable approximation to the correct Bayesian predictor. In the second, it may be that the problem we are trying to solve requires a single estimate. If that is so, the problem will have introduced (perhaps implicitly) a **loss-function**, which we can then optimize to obtain the appropriate estimate.

In many cases the posterior distribution is very strongly peaked at its modal value, written θ^{MP} for *maximum a posteriori*. In this case we may assume that the only significant contribution to the integral comes from parameters very near the peak, and we may assume that the value of $\mathsf{P}_M(x \mid \theta)$ is approximately constant for these values of θ. Armed with these assumptions, along with the knowledge that $\int d\theta\, \mathsf{P}_M(\theta \mid \mathcal{X}) = 1$, we can make the approximation

$$\int d\theta\, \mathsf{P}_M(x \mid \theta)\, \mathsf{P}_M(\theta \mid \mathcal{X}) \approx \mathsf{P}_M\left(x \mid \theta^{\mathsf{MP}}\right) \tag{1.3}$$

That is, calculations made by simply plugging in the MAP estimator in the parameterized density approximate the Bayesian results. In general, this approximation improves with the number of

available data. The MAP value is also important in other, more accurate, approximations to the posterior which are based on the Laplace or saddle-point integral. In these techniques, the posterior is approximated by a Gaussian whose mean lies at the posterior mode and whose covariance is in the inverse of the Hessian of the posterior with respect to the parameters, evaluated at the mode (MacKay 1992). We will treat these in greater detail when we discuss model selection.

The MAP estimator maximizes the posterior (1.1). The denominator on the right hand side of Bayes' rule does not depend on θ, and so the maximization applies only to the numerator $\mathsf{P}_M(\mathcal{X} \mid \theta)\,\mathsf{P}_M(\theta)$. In many situations we may choose to neglect the prior and maximize only the first factor, the likelihood. This yields the maximum-likelihood or ML estimate, θ^{ML}. The ML estimate occupies an extremely prominent position in the classical (non-Bayesian) approach to statistics. In particular, the ML estimate can be shown to be asymptotically efficient, that is, as the sample size grows the expected square error of the ML estimate approaches the fundamental lower bound on such errors (known as the Cramér-Rao bound). In the presence of a "vague" prior (for example, a uniform prior in cases where this is well defined) the ML estimate enjoys all the properties of MAP estimator discussed above.

The MAP estimator can be seen to minimize the expected value of a probability-of-error loss function, which penalizes all errors equally. For continuous parameters we define the loss by the limit as $\epsilon \to 0$ of the function taking the value 0 in an ϵ-ball around the true parameter values and 1 elsewhere. An alternative loss function penalizes errors by the square of the departure from the true value. Minimizing the expected value of this loss leads to the minimum-square-error (MSE) estimator. The fact that the second moment of any distribution is smallest about its mean implies that the MSE estimator is the mean of the posterior. Finding this value may well involve integration of the posterior, with all its attendent pratical difficulties. The result about the asymptotic efficiency of the ML estimator quoted above implies that as the number of data grow larger the mode and mean of the posterior must converge.

We have argued that the MAP and ML parameter estimates are of considerable importance in statistical theory, either as legimate goals in their own part, or as inputs to approximations of Bayesian integrals. In much of this dissertation we shall focus on maximum-likelihood techniques, tacitly assuming a vague prior. In almost all cases, (in particular, in the EM algorithm that we shall encounter shortly and which will resurface throughout this dissertation) the techniques that we will discuss can easily be adapted in the presence of a strong prior to yield a MAP estimate.

1.3 Model Selection

We now consider the situation in which we do not have a single parameterized family of probability functions, but rather must choose between alternative families with different (and perhaps different

numbers of) parameters. These families might be very closely related. For example, we will discuss clustering models at some length in chapter 2, where the data are presumed to arise from some number of distinct distributions, one for each cluster. In this case we shall need to determine the appropriate number of clusters, given the data. This is a model selection problem.

Hyperparameters and stacked generalization

One approach to the model selection problem is to ignore it. We can combine the models into a single family, with a **hyperparameter** that selects between them. The parameter set is then the union of the parameters of the different models, along with the hyperparameter. In the case of **nested** models, where one family is a proper subset of the other, this is almost the same as selecting the most complex model with the addition of the new hyperparameter. If we proceed with the full Bayesian predictive procedure (1.2) this is, in fact, the correct approach. In the case of clustering, for example, we should use an unbounded number of clusters (Neal 1991). However, with such models, the posterior distribution will tend to be far more complex than with a single, continuously parameterized family. In particular, we expect modes corresponding to the MAP estimator for each model, along with the corresponding value of the hyperparameter. In the face of sufficient data one of these modes is likely to dominate, in which case we will have selected one model after all. With less data, we generally need to integrate this posterior, for example when making predictions, by Monte-Carlo means (Neal 1991).

A related approach, now termed **stacked generalization**, was proposed by Stone (1974) and has recently been explored by Wolpert (1992) and Breiman (1996). We can explicitly write the marginal of the predictive density over the model selection hyperparameter. If the models are labelled $\mathcal{M}_i$ this is

$$\mathsf{P}(k \mid \mathcal{X}) = \sum_i \mathsf{P}(\mathcal{M}_i \mid \mathcal{X}) \, \mathsf{P}_{\mathcal{M}_i}(k \mid \mathcal{X}) \tag{1.4}$$

where the rightmost factor is the predictive distribution derived from the ith model. Thus, the predictive distribution is the weighted sum of the predictions made by the different models. The weighting factor for the ith model is given by Bayes' rule,

$$\mathsf{P}(\mathcal{M}_i \mid \mathcal{X}) \propto \mathsf{P}(\mathcal{X} \mid \mathcal{M}_i) \, \mathsf{P}(\mathcal{M}_i) \tag{1.5}$$

that is, it is proportional to the product of the evidence or marginal likelihood $\mathsf{P}(\mathcal{X} \mid \mathcal{M}_i) = \mathsf{P}_{\mathcal{M}_i}(\mathcal{X})$ and the prior probability of the model. The weights are normalized to add to one.

Choosing one model: the dangers of maximum likelihood

Such combined model approaches are attractive in situations where the goal is predictive, and the true family is unknown. In the case of statistical modeling as we have laid it out, however, we are often interested in identifying the particular model that is best supported by the data. In the example of clustering, one of our goals may well be to determine how many classes are present. If we are content with a probabilistic answer, then the marginal likelihood, or evidence, described above, indicates the relative probabilities of each model, as long as the prior weighting of each model is equal. If not, we may elect to choose the most probable model, thereby tacitly assuming a zero-one loss function as in the case of the MAP parameter estimate. In the following discussion we shall assume the latter point of view, arguing for the selection of a single, most probable model; however most of the approximations we will discuss can equally well be used to estimate the posterior probabilities of various models and thus used in techniques such as stacked generalization.

Note that choosing the model with the greatest *marginal* likelihood is different from choosing the model with the greatest *maximum* in the likelihood, which might have been the naïvely favoured policy. In general, more complex models will exploit the greater flexibility of their parameterizations to achieve higher likelihood maxima on the same data; however, such models will be able to explain all sorts of different data by adjusting their parameters appropriately, and can thus only assign a relatively low probability to any particular data set. In other words, the complexity is penalized in the integral, as the region of parameter space that assigns high likelihood to the data is likely to be proportionately smaller. Thus, the Bayes approach leads to the selection of the *simplest* model, within the group considered, that is adequate to explain the data; as a result this approach has been compared with the philosophical "razor" of William of Ockham.

We can express the difficulty with maximum-likelihood model choice in another way. The maximal likelihood for a given model, represents the suitability of one particular member of the model family to describe the data. The member chosen depends critically on the data provided. If the model is complex, and two equivalent, independent samples from the same probability distribution are available, the member functions chosen in the two cases may be very different. In either case, the function may well be far from the true density.

An example appears in figure 1.1. To produce this figure, one dimensional data, shown as filled half-circles on the lower axis, were generated from the Gaussian density shown by the solid line. These data were fit by two different models: one, a simple Gaussian density with mean and variance estimated from the data; the other a three-component mixture of Gaussians (basically the weighted sum of three Gaussian densities). Both models were fit by maximum likelihood estimation (the details of fitting the mixture model will be discussed in a subsequent chapter). The optimal estimates are shown: the simple Gaussian estimate is plotted with dashes; the more complex mixture estiamte with dashes and dots — the faint dotted lines show the shapes of the three mixture components.

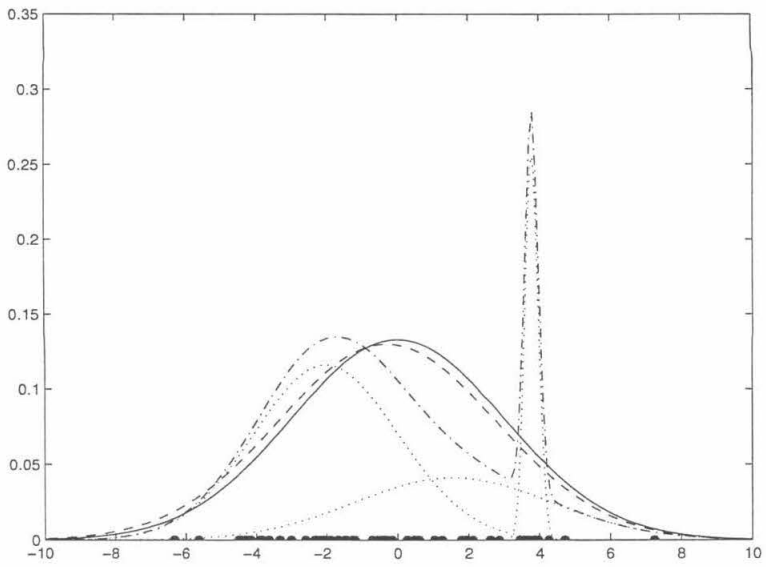


Figure 1.1: The dangers of over-fitting with a complex model.

The mixture model has a higher likelihood than the simpler one. In this case, the log likelihood per point for the simple model is -2.54, while that of the mixture model is -2.41. In part, this increase in likelihood has been achieved by adapting to the cluster of data that appears near the value 4, assigning high probability to this region. Different data, unlikely to cluster near 4, will probably yield a very different estimate.

It is obvious by inspection that the simple model has approximated the true density with greater accuracy. This tendency of complex models to fit the peculiarities of the given sample, rather than the underlying density funtion, is called **over-fitting**.

Bayesian analysis

We consider two candidate models, $\mathcal{M}_0$ and $\mathcal{M}_1$, to be used to describe the data $\mathcal{X}$. The two models have, respectively, p_0 and p_1 parameters, with $p_0 \leq p_1$. The parameter vectors will be written θ_0 and θ_1. In some cases we shall consider **nested** models, where the family of functions allowed under $\mathcal{M}_1$ is a proper superset of the functions available in $\mathcal{M}_0$. In this case we shall further assume that $\mathcal{M}_0$ can be obtained from $\mathcal{M}_1$ by fixing the values of $p_1 - p_0$ parameters, and that the remaining p_0 parameters of $\mathcal{M}_1$ are identical to the parameters of $\mathcal{M}_0$. Thus, $\mathcal{M}_1$ is to be thought of as the more complex model, and, in the nested case, may be a direct generalization of $\mathcal{M}_0$. The Bayesian model selection procedure (sometimes called **empirical Bayes**) dictates that we select model $\mathcal{M}_1$ if and only if the **posterior odds** in favour of $\mathcal{M}_1$, $\mathsf{P}(\mathcal{M}_1 \mid \mathcal{X}) / \mathsf{P}(\mathcal{M}_0 \mid \mathcal{X})$ are greater than one.

Using Bayes' rule, we can write this as

$$\frac{\mathsf{P}\left(\mathcal{M}_1 \mid \mathcal{X}\right)}{\mathsf{P}\left(\mathcal{M}_0 \mid \mathcal{X}\right)} = \frac{\mathsf{P}_{\mathcal{M}_1}\left(\mathcal{X}\right)}{\mathsf{P}_{\mathcal{M}_0}\left(\mathcal{X}\right)} \times \frac{\mathsf{P}\left(\mathcal{M}_1\right)}{\mathsf{P}\left(\mathcal{M}_0\right)} \tag{1.6}$$

The second term on the right hand side of this expression is the **prior odds** of $\mathcal{M}_1$ being correct; the first term, which is the ratio of the marginal likelihoods, is called the **Bayes factor**. It is convenient to work with logarithms, and so the empirical Bayes criterion for selecting $\mathcal{M}_1$, in the face of equal prior probabilities for the two models (prior odds = 1), is

$$\log B_{10} = \log \mathsf{P}_{\mathcal{M}_1}\left(\mathcal{X}\right) - \log \mathsf{P}_{\mathcal{M}_0}\left(\mathcal{X}\right) > 0 \tag{1.7}$$

These are the same marginal likelihoods that appeared in the denominator of (1.1). While they do not play much of a rôle in parameter estimation, they can be seen to be vital to model selection.

The marginal likelihood is an integral over the parameter vector θ_i for the model $\mathcal{M}_i$,

$$\mathsf{P}_{\mathcal{M}_i}\left(\mathcal{X}\right) = \int d\theta_i \, \mathsf{P}_{\mathcal{M}_i}\left(\mathcal{X} \mid \theta_i\right) \mathsf{P}_{\mathcal{M}_i}\left(\theta_i\right) \tag{1.8}$$

As in the case of predictions using the posterior (1.2) this integral is often difficult to compute. Analytic solutions can be found for simple exponential family models, including multivariate normal linear regression models, with so-called **conjugate priors** on the parameters (these being priors chosen in part for the simplicity of the resulting integral). In the general case we need to estimate the integral via Monto-Carlo techniques (which we will not discuss here, but see Gelfand and Smith (1990), Smith and Roberts (1993) and Neal (1996)) or else employ analytic approximations which, while they may be asymptotically exact, yield biased estimates with realistic sample sizes.

Approximations to the Bayes factor

A simple and widely used approximation is called **Laplace's method** (Tierney and Kadane 1986; MacKay 1992). Let us write $\Phi(\theta)$ for the logarithm of the integrand in (1.8), the unnormalized posterior over the parameters. We have dropped the subscript i for simplicity. We can expand $\Phi(\theta)$ in a Taylor series about its maximum, which falls at θ^{MP}.

$$\Phi(\theta) = \Phi(\theta^{\mathsf{MP}}) + \nabla\Phi(\theta^{\mathsf{MP}}) \cdot (\theta - \theta^{\mathsf{MP}}) + \frac{1}{2}(\theta - \theta^{\mathsf{MP}})^T \nabla\nabla\Phi(\theta^{\mathsf{MP}})(\theta - \theta^{\mathsf{MP}}) + \ldots \tag{1.9}$$

where the notation $\nabla\nabla\Phi$ denotes the Hessian matrix of second derivatives $[\partial^2\Phi/\partial\theta_i\partial\theta_j]$ and should not be confused with the Laplacian, $\nabla^2\Phi = \mathrm{Tr}\left[\nabla\nabla\Phi\right]$. As θ^{MP} lies at a maximum of Φ, the gradient there is 0 and the linear term in the expansion vanishes. We ignore the terms of higher order than quadratic, a choice tantamount to approximating the posterior by a Gaussian, and write

$(K^{\mathsf{MP}})^{-1} = -(\nabla\nabla\Phi(\theta^{\mathsf{MP}}))^{-1}$ for the covariance of the approximation. The integral of (1.8) is then

$$\mathsf{P}_{\mathcal{M}_i}(\mathcal{X}) \approx \left|K_i^{\mathsf{MP}}/2\pi\right|^{-1/2} \exp \Phi_i(\theta_i^{\mathsf{MP}}) = \left|K_i^{\mathsf{MP}}/2\pi\right|^{-1/2} \mathsf{P}_{\mathcal{M}_i}\left(\mathcal{X} \mid \theta_i^{\mathsf{MP}}\right) \mathsf{P}_{\mathcal{M}_i}\left(\theta_i^{\mathsf{MP}}\right) \tag{1.10}$$

where we have reintroduced the model subscript. The log Bayes factor of (1.7) is thus approximated by

$$\log B_{10} \approx \Lambda_{10}^{\mathsf{MP}} + \Pi_{10}^{\mathsf{MP}} + \frac{1}{2}\log\frac{\left|K_0^{\mathsf{MP}}/2\pi\right|}{\left|K_1^{\mathsf{MP}}/2\pi\right|} \tag{1.11}$$

where $\Lambda_{10}^{\mathsf{MP}}$ is similar to the log likelihood ratio statistic for classical model comparison, although evaluated at the MAP estimates, and Π_{10}^{MP} is the difference in the log priors of the MAP estimators for the two models. Note that this is different to the log of the prior odds of $\mathcal{M}_1$, which we have assumed to be 0. The priors in this case are not the priors of the models, but rather the priors of the *parameters* of each model, evaluated at the maximum of the posterior. In general, the more complex model may be expected to spread its prior more thinly over a larger parameter space, and thus to provide a smaller prior density at any particular point. Thus, we expect the term Π_{10}^{MP} to be negative, *penalizing* the likelihood ratio. Similarly, the determinant of the Hessian of the more complex model is likely to be larger (if the parameters are all estimated with roughly equivalent error e and we rotate to a diagonal basis we see that it will scale as $(1/e)^{p_i}$) and so the ratio of $|K|$ will be less than one, also penalizing the likelihood. The Laplace approximation is asymptotically correct, with, under certain regularity conditions, relative error of order $O(N^{-1})$ where N is the number of observations (Kass *et al.* 1990).

In the discussion of parameter estimation, we argued that we would remain agnostic on the nature of the prior and choose the maximum-likelihood estimator, which is likely to be close to the MAP value for vague priors. Can we reduce (1.11) from the same standpoint? Assuming the prior is vague, and that θ^{ML} is close to θ^{MP}, we can approximate $\Lambda_{10}^{\mathsf{MP}}$ by the more conventional likelihood ratio, Λ_{10}, evaluated at the respective maxima of the likelihoods. Also, the prior will not have strong curvature, and so the Hessian of the log unnormalized posterior, evaluated now at θ^{ML} will be dominated by the likelihood term. Thus we can replace K_i^{MP} by the **observed information matrix** $K_i = -\nabla\nabla\ell_{\mathcal{X}}\left(\theta_i^{\mathsf{ML}}\right)$. This gives us

$$\log B_{10} \approx \Lambda_{10} + \Pi_{10}^{\mathsf{ML}} + \frac{1}{2}\log\frac{\left|K_0/2\pi\right|}{\left|K_1/2\pi\right|} \tag{1.12}$$

where Π_{10}^{ML} is the log ratio of priors evaluated at the maximum likelihood parameter values. This approximation exhibits relative error $O(N^{-1/2})$.

At first glance, it would seem that we cannot dispense with the term Π_{10}^{ML} as it reflects the difference in dimensionality of the two models and provides an important penalty. However, consideration of the asymptotic behaviour of (1.12) reveals that for large data sets it may be neglected. If we have

N data points, the likelihood ratio takes the form $\sum_{n=1}^{N} \log\left(\mathsf{P}_{\mathcal{M}_1}\left(x_n \mid \theta_1^{\mathsf{ML}}\right)/\mathsf{P}_{\mathcal{M}_0}\left(x_n \mid \theta_0^{\mathsf{ML}}\right)\right)$ and will therefore grow with N. A similar argument applies to the Hessian of the log-likelihood, so that the magnitude of the final term of (1.12) grows as $\log N$. Thus the term Π_{10}^{ML}, which is constant with changes in the number of data can be asymptotically neglected.

We can further simplify the ratio of Hessians that appears in the final term of (1.12). With N data points, we have

$$\begin{aligned}
\log|K_i/2\pi| &= \log\left|-\tfrac{1}{2\pi}\sum_{n=1}^{N}\nabla\nabla\mathsf{P}_{\mathcal{M}_i}\left(x_n \mid \theta_i\right)\right| \\
&\approx \log\left|N\hat{K}/2\pi\right| \\
&= \log\left((N/2\pi)^{p_i}\left|\hat{K}\right|\right) \\
&= p_i(\log N - \log 2\pi) + \log\left|\hat{K}\right| \qquad (1.13)
\end{aligned}$$

where $\hat{K}$ is the expected value with respect to the distribution of x of the one-point Hessian $\nabla\nabla\mathsf{P}_{\mathcal{M}_i}\left(x \mid \theta_i\right)$ evaulated at θ_i^{ML}. Again we drop the terms that do not grow with N, and obtain

$$\log B_{10} \approx \Lambda_{10} - \frac{1}{2}(p_1 - p_0)\log N \qquad (1.14)$$

This approximation was introduced by Schwartz (1978) with a far more rigorous derivation in the case of multivariate linear regression with an exponential family noise distribution, and was extended by Haughton (1988) to regression on curves. The heuristic approach we have adopted here suggests that it should be useful for many model families, and indeed it is used quite widely. It is referred to in the literature as the Schwartz criterion, or as the Bayesian Information Criterion, BIC.

In general the BIC approximation to the Bayes factor introduces relative errors of order $O(1)$. Some authors attempt to reduce the BIC error in the context of particular models by approximating the constant (with respect to N) term that we have neglected. One approach, practical in this modern day of the computer, is to determine a suitable value of the constant empirically by simulating and fitting data from known distributions. Other authors pay close attention to the definition of the number N. In the above, we simply took it to be the total count of data; on other hand, from the derivation it is clear that it is really the growth rate of the Hessian. In some models, the parameters are local and are only affected by data that fall within a small region. The clustering models of chapter 2, for example, fall into this category. In this case it may be argued that N is not the total number of data, but rather the average number of data falling into each cluster. In practice, however, all of these corrections are of order $O(1)$ and, provided that the number of data are large, the BIC alone has been found to produce reasonable results. We shall see, however, that in the context of latent variable models care must be taken in the choice of the number of parameters (Geiger *et al.* 1998). We will postpone our discussion of this issue, along with treatment of another approximate

Bayes technique for latent variable models introduced by Cheeseman and Stutz (1996). Instead, we shall proceed to investigate another class of model selection methods based on direct estimates of the probability of over-fitting.

Validation

We have motivated much of our development of model selection criteria by the notion of predictive accuracy. One approach, then, is to try to measure the predictive performance of the various models directly by observing the probability they assign to data outside the observations used for training. This approach is called **validation**. In its simplest form the process of validation involves the division of the set of observations $\mathcal{X}$ into two parts, the **training data** for which we will continue to use the symbol $\mathcal{X}$, and the validation or **test data** for which we will write $\mathcal{V}$. The posterior over parameters for each model (or the parameter estimates) are obtained on the training data, and the models are ranked by the probability that they assign to the validation set

$$V_i = \int d\theta_i \, \mathsf{P}_{\mathcal{M}_i}(\mathcal{V} \mid \theta_i) \, \mathsf{P}_{\mathcal{M}_i}(\theta_i \mid \mathcal{X}) \approx \mathsf{P}_{\mathcal{M}_i}\left(\mathcal{V} \mid \theta_i^{\mathsf{MP}}\right) \tag{1.15}$$

The intuition behind this approach is appealing, but it is often a fairly noisy criterion. We usually have only a limited amount of data available, and the necessity to divide it in two means that both the estimate of the parameters, and the estimate of the expected off-training set error are likely to be noisy. Once we have chosen a model by validation, we can combine the training and validation data sets and then reestimate the parameters to improve our predictions. However, the noise due to small $\mathcal{X}$ and $\mathcal{V}$ may lead to the incorrect model being selected.

In the simplest validation procedure there is a single training set and a single validation set. However, we could equally well train on $\mathcal{V}$ and test on $\mathcal{X}$. This would yield two independent estimates of the off-training-set performance of a particular model. The average of the two will thus have smaller variance than any one of them. In general, we can split up the data set into N_V disjoint subsets. One by one, we take each of these subsets, call it validation data, train on its complement in the data set, and validate the resulting model. Thus we obtain N_V independent estimates of V_i, which we can average to reduce the error in the estimate by $O(1/\sqrt{N_V})$. This simple improvement on the basic validation scheme is called **cross-validation**. In the extreme case where $N_V = N$, the number of data, the term **leave-one-out** cross-validation is applied.

Non-Bayesian Penalties

The spirit of such validation techniques, along with approximations similar to those made during the Bayesian treatment above, can also be used to obtain alternative likelihood penalization schemes similar to the BIC. The goal here is to estimate by how much the observed training likelihood is

likely to differ from the likelihood of the validation set.

Let us suppose that the true distribution of the data is some distribution $\mathsf{P}_*(\cdot)$, which we are attempting to fit with a family $\mathsf{P}_\theta(\cdot)$. Let θ^* represent the parameters that come closest to the true distribution in the sense of the Kullback-Leibler divergence, that is

$$\theta^* = \underset{\theta}{\operatorname{argmin}}\, \mathsf{KL}[\mathsf{P}_*||\mathsf{P}_\theta] = \underset{\theta}{\operatorname{argmin}} \int dx\; \mathsf{P}_*(x) \log \frac{\mathsf{P}_*(x)}{\mathsf{P}_\theta(x)} \tag{1.16}$$

If the true distribution is actually a member of the parametric family then the minimum KL divergence will, of course, be 0. Asymptotically, the maximum likelihood estimator will approach θ^*. When discussing parameter estimation we made the well known observation that the maximum likelihood estimator is asymptotically efficient, which holds when the true distribution falls within the parameterized family. This result can be extended to the general case.

The ML estimator given data $\mathcal{X}$ has the property that $\nabla \ell_{\mathcal{X}}\left(\theta^{\mathsf{ML}}\right) = 0$. Assuming that θ^{ML} is close to θ^*, we can make a linear approximation to the gradient at θ^*

$$\nabla \ell_{\mathcal{X}}(\theta^*) \approx \nabla \ell_{\mathcal{X}}\left(\theta^{\mathsf{ML}}\right) + (\theta^* - \theta^{\mathsf{ML}}) \nabla\nabla \ell_{\mathcal{X}}\left(\theta^{\mathsf{ML}}\right) = (\theta^* - \theta^{\mathsf{ML}}) K \tag{1.17}$$

where K is the observed information matrix, as before. Thus the error $\theta^* - \theta^{\mathsf{ML}} \approx K^{-1} \nabla \ell_{\mathcal{X}}(\theta^*)$ Asymptotically, the expected value of the difference is 0. To calculate the variance we note that for iid data $\mathcal{E}[K] = N\hat{K}$ where N is the number of observations and $\hat{K}$ is the expected one-point Hessian. We write $\hat{J} = \mathcal{V}ar\left[\nabla \ell_{x_i}(\theta^*)\right]$ as the more conventional definition of the Fisher information, the variance of the one-point log likelihood gradient, so that $\mathcal{V}ar\left[\nabla \ell_{\mathcal{X}}(\theta^*)\right] = N\hat{J}$, and so

$$\mathcal{V}ar\left[\theta^* - \theta^{\mathsf{ML}}\right] \approx \frac{1}{N} \hat{K}^{-1} \hat{J} \hat{K}^{-1} \tag{1.18}$$

The expectations and variances are all with respect to the true density $\mathsf{P}_*(\cdot)$. If this is the same as $\mathsf{P}_{\theta^*}(\cdot)$ then the two definitions of the information are equivalent and $\hat{J} = \hat{K}$, so that the mean square error approaches the standard Crámer–Rao bound $1/N\hat{J}$.

Given the asymptotic behaviour of the ML estimate, we can ask what likelihood we will assign to a validation point, v generated from the true distribution $\mathsf{P}_*(v)$. We expand around the "correct" validation value at θ^*.

$$\ell_v\left(\theta^{\mathsf{ML}}\right) \approx \ell_v(\theta^*) + (\theta^{\mathsf{ML}} - \theta^*)^T \nabla_\theta \ell_v(\theta^*) + \frac{1}{2}(\theta^{\mathsf{ML}} - \theta^*)^T \nabla\nabla_\theta \ell_v(\theta^*)(\theta^{\mathsf{ML}} - \theta^*) \tag{1.19}$$

$$= \ell_v(\theta^*) + (\theta^{\mathsf{ML}} - \theta^*)^T \nabla_\theta \ell_v(\theta^*) + \frac{1}{2}\mathrm{Tr}\left[\nabla\nabla_\theta \ell_v(\theta^*)(\theta^{\mathsf{ML}} - \theta^*)(\theta^{\mathsf{ML}} - \theta^*)^T\right] \tag{1.20}$$

If we now take the expectation with respect to the true distribution of the training data *and* of v, we can take the expected gradient at θ^* to be 0. Also, since v is independent of $\mathcal{X}$ and therefore of

θ^{ML}, we can factor the expectation within the trace.

$$\begin{aligned}
\mathcal{E}\left[\ell_v\left(\theta^{\mathsf{ML}}\right)\right] &= \mathcal{E}\left[\ell_v\left(\theta^*\right)\right] + \frac{1}{2}\mathrm{Tr}\left[\mathcal{E}\left[\nabla\nabla_\theta \ell_v\left(\theta^*\right)\right]\mathcal{E}\left[(\theta^{\mathsf{ML}} - \theta^*)(\theta^{\mathsf{ML}} - \theta^*)^T\right]\right] \\
&= \mathcal{E}\left[\ell_v\left(\theta^*\right)\right] - \frac{1}{2}\mathrm{Tr}\left[\hat{K}\mathcal{V}ar\left[(\theta^{\mathsf{ML}} - \theta^*)\right]\right] \\
&= \mathcal{E}\left[\ell_v\left(\theta^*\right)\right] - \frac{1}{2}\mathrm{Tr}\left[\hat{K}\frac{1}{N}\hat{K}^{-1}\hat{J}\hat{K}^{-1}\right] \\
&= \mathcal{E}\left[\ell_v\left(\theta^*\right)\right] - \frac{1}{2N}\mathrm{Tr}\left[\hat{J}\hat{K}^{-1}\right] \qquad (1.21)
\end{aligned}$$

This expression shows the approximate bias in the validation likelihood. On the training data we can expand $\ell_{\mathcal{X}}\left(\theta^*\right)$ around θ^{ML} (where the gradient is always 0) to obtain

$$\mathcal{E}\left[\ell_{\mathcal{X}}\left(\theta^*\right)\right] = \mathcal{E}\left[\ell_{\mathcal{X}}\left(\theta^{\mathsf{ML}}\right)\right] - \frac{1}{2}\mathrm{Tr}\left[\hat{J}\hat{K}^{-1}\right] \qquad (1.22)$$

Now, the expected values of the log-likelihoods at the fixed point θ^* are equal (up to a factor of the number of training data, N). Thus, we obtain

$$\mathcal{E}\left[\ell_v\left(\theta^{\mathsf{ML}}\right)\right] = \frac{1}{N}\left(\mathcal{E}\left[\ell_{\mathcal{X}}\left(\theta^{\mathsf{ML}}\right)\right] - \mathrm{Tr}\left[\hat{J}\hat{K}^{-1}\right]\right) \qquad (1.23)$$

This can be viewed as a prediction of the expected difference between the validation likelihood and the training likelihood. We might therefore rank models according to their training likelihoods penalized by the trace term.

This is the NIC (Network Information Criterion) of Murata *et al.* (1991, 1993, 1994). To use it we replace the expected values of the information measures $\hat{J}$ and $\hat{K}$ by their observed values,

$$\mathrm{NIC} = \ell_{\mathcal{X}}\left(\theta^{\mathsf{ML}}\right) - \mathrm{Tr}\left[JK^{-1}\right] \qquad (1.24)$$

with K the observed information and $J = \sum_i (\nabla \ell_{x_i}\left(\theta^{\mathsf{ML}}\right))^2/(N-p)$ where p is the number of parameters. If the true distribution lies within the parameterized family then $\hat{J} = \hat{K}$ and we can replace the trace penalty by the number of parameters p. This is the AIC of Akaike (1974). Akaike used AIC as an abbreviation for *An Information Criterion*, although it is usually taken to stand for the *Akaike Information Criterion*.

1.4 Graphical Representations

In most experiments we measure more than one variable simultaneously. Thus the observations x_i that we have described above are usually multivariate. It is often useful to partition the observations into a number of distinct random variables, each of which may still be multivariate. For example,

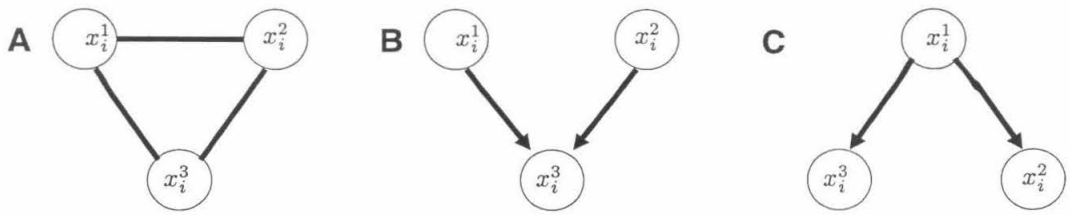


Figure 1.2: Graphical representation of conditional independence.

we may make measurements with different instruments and regard the output of each instrument, whether a single number or a vector, as a random variable. The advantage to such a partition is that it is often possible to write the parameterized model distribution $\mathsf{P}_\theta\left(x_i\right)$ more easily in terms of the partitioned variables. Why would this be so?

Let us consider a case where the observation x_i is partitioned into three random variables x_i^1, x_i^2, x_i^3. In general any probability function of the x_i may be written in conditional form:

$$\mathsf{P}\left(x_i\right) = \mathsf{P}\left(x_i^3 \mid x_i^2, x_i^1\right) \mathsf{P}\left(x_i^2 \mid x_i^1\right) \mathsf{P}\left(x_i^1\right) \tag{1.25}$$

However, it might be that x_i^2 is independent of x_i^1 and so we replace the second term on the right above with just $\mathsf{P}\left(x_i^2\right)$. Another possibility is that x_i^3 is conditionally independent of x_i^2 given x_i^1 so that we can write $\mathsf{P}\left(x_i^3 \mid x_i^1\right)$ in place of the first right hand term. This might seem like only a notational convenience, but, in fact, if we are to parameterize the probability distribution we have saved ourself some parameters. The factorized function is *simpler* (in the sense of model selection) than before.

The factorized structure of the distribution can be shown graphically as in figure 1.2. In panel A the case of no conditional or marginal independencies is shown as a fully connected undirected graph. Panel B represents the marginal independence of x_i^1 and x_i^2. Panel C represents the conditional independence of x_i^2 and x_i^3. Each of the latter two cases is represented by a **directed acyclic graph** or DAG.

It should be noted that the connection between probabilistic models and DAGs is far from cosmetic. An important and deep theory is available connecting reasoning about the probability distribution with algorithmic manipulations of the graph (Pearl 1988; Lauritzen 1996). However, we shall not exploit this theory at all; for us the graph will simply be a convenient tool for visualization.

When representing parameterized probability functions $\mathsf{P}_\theta\left(x_i\right)$ we will find it useful to introduce nodes in our graphical representation corresponding to the parameters. Since we have factorized our probability functions, we need to partition the parameters θ into the groups appropriate for each

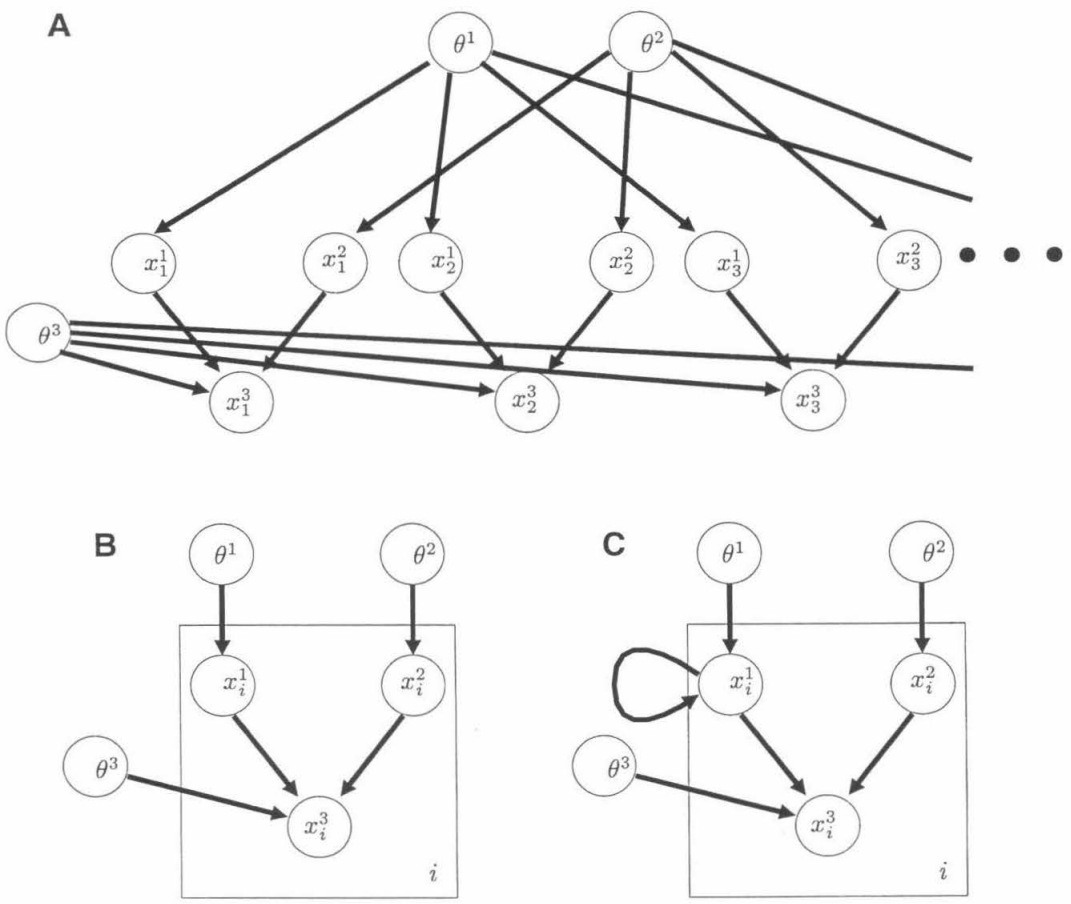


Figure 1.3: Graphical representations of repeated observation models.

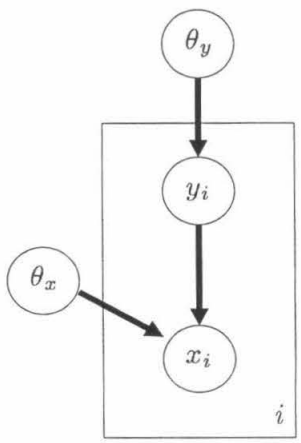
factor function. In general, we might write

$$\mathsf{P}_\theta\left(x_i\right) = \mathsf{P}_{\theta^3}\left(x_i^3 \mid x_i^2, x_i^1\right) \mathsf{P}_{\theta^2}\left(x_i^2 \mid x_i^1\right) \mathsf{P}_{\theta^1}\left(x_i^1\right) \tag{1.26}$$

where θ is the union of $\theta^r, r = 1 \ldots 3$. Figure 1.3A illustrates the representation. Whereas before it was sufficient to show the variables involved in a single observation i, with the implicit information that each observation is independent and identically distributed, we now need to make clear that the parameters are chosen exactly once and have the same value over all observations, whereas each observation has its own set of random variables x_i^r. This time the fact that the x_i^r are independent (conditioned on the parameters) is shown explicitly by the lack of edges between nodes at different values of i.

We can condense the representation as shown in Figure 1.3B[2]. The rectangle represents a single

[2]To the best of my knowledge, this representation was introduced in the computer program BUGS from the MRC biostatistics unit at Cambridge (Thomas 1994; Spiegelhalter *et al.* 1996).



$$\mathsf{P}_{\theta_x}(x_i \mid y_i)\,\mathsf{P}_{\theta_y}(y_i)$$

Figure 1.4: Graphical representation of a latent variable model.

observation with an index indicated its lower right hand corner; variables that appear within the rectangle are repeated across observations, while the parameters which are chosen only once for all observations appear outside it. As before, the lack of edges between nodes at differing i indicates that the observations are independent. Now, our decision to represent all the functions $\mathsf{P}_\theta(x_i)$ by a single subgraph indicates further that they are identical.

If the observations are not independent, say there are correlations between the variables x_i^1 at different i, we may represent this fact by an edge that crosses out of, and then back into, the rectangle, as in figure 1.3C. However, we cannot show the limits of this interaction. For example, if x_i^1 is generated by a Markov process, so that x_i^1 is conditionally independent of $x_1^1 \ldots x_{i-2}^1$ given x_{i-1}^1 we need the expanded time view of figure 1.3A, with additional edges for the Markovian dependence, to distinguish this from the other possible cross-observation dependency structures.

1.5 Latent Variables

We have seen that it can be useful to partition the observed variables so as to simplify the expression of the probability function by exploiting the conditional dependency structure of the problem. Another manipulation that can assist in this simplification is the introduction of **latent variables**. These are variables which are not observed. The parameters, of course, are also not observed; the latent variables are different in that they are presumed to be instantiated once for every observation, that is there is a latent y_i for each observation x_i. In graphical terms, the simplest latent variable model is sketched in figure 1.4. Note that the latent variable node appears within the rectangle.

In a latent variable model we can add a third operation to our pair of learning and model

selection, **inference**. This will refer to the estimation of value of the latent variables y_i given known parameters and the observations x_i. The difference from fitting, that is, estimating the parameters, is simply one of scale.

Again, it has been shown that certain algorithmic manipulations on the graph that defines the latent variable model can yield the correct form of inference (Pearl 1988). For most of the models we shall discuss, however, inference will be a simple matter of the application of Bayes' rule:

$$\mathsf{P}_\theta\left(y_i \mid x_i\right) = \frac{\mathsf{P}_\theta\left(x_i \mid y_i\right)\mathsf{P}_\theta\left(y_i\right)}{\mathsf{P}_\theta\left(x_i\right)} \tag{1.27}$$

1.6 The Expectation–Maximization Algorithm

How do we go about learning the parameter values of a latent variable model? It is possible to define a likelihood function for the parameters by integrating over the latent variables[3].

$$\ell_{\mathcal{X}}\left(\theta\right) = \log \int d\mathcal{Y}\, \mathsf{P}_\theta\left(\mathcal{X} \mid \mathcal{Y}\right)\mathsf{P}_\theta\left(\mathcal{Y}\right) \tag{1.28}$$

where the integral is over all the y_i in the set $\mathcal{Y}$. However, in many cases this likelihood is quite difficult to optimize in closed form. Gradient- or Hessian-based numerical optimization schemes can be very effective for a number of problems. In the case of latent variable models, however, another algorithm exists that is frequently more straightforward and of comparable efficiency. This is the **Expectation–Maximization** (or **EM**) algorithm (Dempster *et al.* 1977). Quite complicated models may be fit efficiently by use of EM (Xu and Jordan 1996).

We shall first lay out the steps of the EM algorithm and only then offer two (informal) proofs of its validity. The second of these proofs will also provide the justification for various extensions.

If we had, in fact, observed the variables y_i we would be able to write the **joint data log likelihood**

$$\ell_{\mathcal{X},\mathcal{Y}}\left(\theta,\theta\right) = \log \mathsf{P}_\theta\left(\mathcal{X} \mid \mathcal{Y}\right) + \log \mathsf{P}_\theta\left(\mathcal{Y}\right) \tag{1.29}$$

This likelihood is often much easier to manipulate than the true likelihood of (1.28), since it avoids the awkward log-of-integral (or log-of-sum) expression. It will be the starting point for EM.

To begin the EM algorithm we provide seed guesses for the parameters. We will label successive outputs of the iterations by the iteration number in the superscript. Thus, the initial guesses will be called θ^0. At the nth iteration we estimate new values of the parameters by the following two steps.

E-step: Find the **expectation** of the joint data log-likelihood under the distribution of the y_i given

[3]In this general introduction we shall assume that the y_i are continuous, but discrete latent variables may be handled in the same fashion with the integral replaced by a sum.

the $n-1$th parameter estimates and the observations.

$$Q^n(\theta) = \mathcal{E}_{\mathcal{Y}|\mathcal{X};\theta^{n-1}}\left[\ell_{\mathcal{X},\mathcal{Y}}(\theta)\right] \tag{1.30}$$

M-step: Then **maximize** this expected joint data log-likelihood with respect to the parameters to obtain the new estimates.

$$\theta^n = \operatorname{argmax} Q^n(\theta) \tag{1.31}$$

Why does EM work? Let us consider the effect of the iterations on the true log-likelihood function given in (1.28). In each iteration we start with parameters θ^{n-1} and estimate new parameters θ^n. For notational simplicity we will write $\mathsf{P}_{n-1}(\cdot)$ for the various probability functions with parameters θ^{n-1} and similarly for $\mathsf{P}_n(\cdot)$. The resulting log-likelihood is

$$\ell_{\mathcal{X}}(\theta^n) = \log \int d\mathbf{y}\ \mathsf{P}_n(\mathcal{Y})\,\mathsf{P}_n(\mathcal{X} \mid \mathcal{Y}) \tag{1.32}$$

We introduce a factor of $\dfrac{\mathsf{P}_{n-1}(\mathcal{Y} \mid \mathcal{X})}{\mathsf{P}_{n-1}(\mathcal{Y} \mid \mathcal{X})}$ within the integral and rearrange to obtain

$$\ell_{\mathcal{X}}(\theta^n) = \log \int d\mathbf{y}\ \mathsf{P}_{n-1}(\mathcal{Y} \mid \mathcal{X}) \left(\frac{\mathsf{P}_n(\mathcal{Y})\,\mathsf{P}_n(\mathcal{X} \mid \mathcal{Y})}{\mathsf{P}_{n-1}(\mathcal{Y} \mid \mathcal{X})}\right) \tag{1.33}$$

We can now use Jensen's inequality (see, for example, Cover and Thomas (1991)) applied to the convex function $\log(\cdot)$ to exchange the logarithm and integral. In this context, Jensen's inequality states that, for positive weights α_i that sum to 1,

$$\log(\sum_i \alpha_i x_i) \geq \sum_i \alpha_i \log(x_i) \tag{1.34}$$

We can generalize this for a positive continuous weight function with unit integral, in our case $\mathsf{P}_{n-1}(\mathcal{Y} \mid \mathcal{X})$, to obtain

$$\begin{aligned} \ell_{\mathcal{X}}(\theta^n) &\geq \int d\mathbf{y}\ \mathsf{P}_{n-1}(\mathcal{Y} \mid \mathcal{X})\ \log\left(\frac{\mathsf{P}_n(\mathcal{Y})\,\mathsf{P}_n(\mathcal{X} \mid \mathcal{Y})}{\mathsf{P}_{n-1}(\mathcal{Y} \mid \mathcal{X})}\right) && (1.35) \\ &= \int d\mathbf{y}\ \mathsf{P}_{n-1}(\mathcal{Y} \mid \mathcal{X})\ \log\left(\mathsf{P}_n(\mathcal{Y})\,\mathsf{P}_n(\mathcal{X} \mid \mathcal{Y})\right) - \\ &\quad \int d\mathbf{y}\ \mathsf{P}_{n-1}(\mathcal{Y} \mid \mathcal{X})\ \log\left(\mathsf{P}_{n-1}(\mathcal{Y} \mid \mathcal{X})\right) && (1.36) \end{aligned}$$

Thus the quantity on the right hand side of (1.36) is a lower bound on the likelihood at the nth iteration. The first term is readily identified as the quantity $Q^n(\theta)$ from our statement of the EM algorithm (1.30). The second term has no dependence on θ^n. Thus by maximizing $Q^n(\theta)$ as dictated by the m-step (1.31) we are maximizing a lower bound on the likelihood. Further, we know that the

maximum must be $\geq \ell_{\mathcal{X}}\left(\theta^{n-1}\right)$ since we can obtain that value by simply putting $\theta^n = \theta^{n-1}$. Thus we can be sure that as long as the EM algorithm does not converge, the likelihood of the model must increase.

We need also to show that when the EM algorithm does converge, we have reached a maximum of the true likelihood. This proof appears in (Dempster *et al.* 1977), and we will not reproduce it. Instead, we will follow Neal and Hinton (1998) and take a slightly different view of the algorithm; this approach will yield the necessary second component of the proof.

1.7 Free Energy and EM

Let us define a more general form of the function Q in (1.30) by taking the expectation with respect to an arbitrary probability function $p(\mathcal{Y})$, in place of the particular probability $\mathsf{P}_{n-1}\left(\mathcal{Y} \mid \mathcal{X}\right)$.

$$Q(p,\theta) = \mathcal{E}_p\left[\ell_{\mathcal{X},\mathcal{Y}}\left(\theta\right)\right] \tag{1.37}$$

We can then introduce a function that we will call the **free energy** by analogy with statistical mechanics,

$$F(p,\theta) = Q(p,\theta) + H(p) \tag{1.38}$$

where $H(p) = -\mathcal{E}_p\left[\log p\right]$ is the entropy of p. This function is familiar from above; it is the right hand side of (1.36) with the arbitrary function p replacing $\mathsf{P}_{n-1}\left(\mathcal{Y} \mid \mathcal{X}\right)$. Furthermore, in arriving at that expression our choice of weighting function to use in Jensen's inequality was arbitrary, so F also bounds the likelihood $\ell_{\mathcal{X}}\left(\theta\right)$ below. In drawing the physical analogy we should note that our F should, in fact, be regarded as the negative of the conventional free energy, which is consistent with the fact that we are interested in maximizing F, while physical systems evolve to minimize their free energy.

We observe (Neal and Hinton 1998) that, if θ is held constant, the free energy is, in fact, maximized by choosing $p(\mathcal{Y}) = \mathsf{P}_\theta\left(\mathcal{Y} \mid \mathcal{X}\right)$. To see this, we maximize the quantity

$$\begin{aligned} L_\theta(p) &= F(p,\theta) - \lambda \int d\mathcal{Y}\, p(\mathcal{Y}) && (1.39) \\ &= \int d\mathcal{Y}\, p(\mathcal{Y}) \left(\ell_{\mathcal{X},\mathcal{Y}}\left(\theta\right) - \log p(\mathcal{Y}) - \lambda\right) && (1.40) \end{aligned}$$

where λ is a Lagrange multiplier enforcing the normalization constraint. From the theory of the calculus of variations (Mathews and Walker 1970) we find that at the maximum with respect to p the functional derivative of the integrand must be 0 (this is a trivial special case of the Euler-Lagrange

equations). Thus the maximum occurs when

$$\begin{aligned} 0 &= \frac{\partial}{\partial p}\left(p(\mathcal{Y})\left(\ell_{\mathcal{X},\mathcal{Y}}(\theta) - \log p(\mathcal{Y}) - \lambda\right)\right) \\ &= \left(\ell_{\mathcal{X},\mathcal{Y}}(\theta) - \log p(\mathcal{Y}) - \lambda\right) - \frac{p(\mathcal{Y})}{p(\mathcal{Y})} \end{aligned} \tag{1.41}$$

and so

$$p(\mathcal{Y}) = e^{-\lambda-1}\mathcal{L}_{\mathcal{X},\mathcal{Y}}(\theta) = e^{-\lambda-1}\mathsf{P}_\theta(\mathcal{X},\mathcal{Y}) \tag{1.42}$$

The requirement that p be normalized determines the multiplier λ and yields $p(\mathcal{Y}) = \mathsf{P}_\theta(\mathcal{Y} \mid \mathcal{X})$.

Thus we obtain a new interpretation of the EM algorithm.

E-step: Maximize F with respect to p holding θ constant.

M-step: Maximize F with respect to θ holding p constant.

We can now sketch the proof that if F achieves a local maximum at p^*, θ^* then $\ell_{\mathcal{X}}(\theta)$ achieves a local maximum at θ^* (Theorem 2 of Neal and Hinton (1998)). We first note that if $p(\mathcal{Y}) = \mathsf{P}_\theta(\mathcal{Y} \mid \mathcal{X})$ then

$$\begin{aligned} F(\mathsf{P}_\theta(\mathcal{Y} \mid \mathcal{X}), \theta) &= Q(\mathsf{P}_\theta(\mathcal{Y} \mid \mathcal{X}), \theta) + H(\mathsf{P}_\theta(\mathcal{Y} \mid \mathcal{X})) \\ &= \mathcal{E}_{\mathcal{Y}|\mathcal{X};\theta}\left[\ell_{\mathcal{X},\mathcal{Y}}(\theta)\right] - \mathcal{E}_{\mathcal{Y}|\mathcal{X};\theta}\left[\log \mathsf{P}_\theta(\mathcal{Y} \mid \mathcal{X})\right] \\ &= \mathcal{E}_{\mathcal{Y}|\mathcal{X};\theta}\left[\frac{\log \mathsf{P}_\theta(\mathcal{Y},\mathcal{X})}{\log \mathsf{P}_\theta(\mathcal{Y} \mid \mathcal{X})}\right] \\ &= \mathcal{E}_{\mathcal{Y}|\mathcal{X};\theta}\left[\log \mathsf{P}_\theta(\mathcal{X})\right] \\ &= \log \mathsf{P}_\theta(\mathcal{X}) \\ &= \ell_{\mathcal{X}}(\theta) \end{aligned} \tag{1.43}$$

Thus, writing $p^*(\mathcal{Y})$ for $\mathsf{P}_{\theta^*}(\mathcal{Y} \mid \mathcal{X})$, we have $\ell_{\mathcal{X}}(\theta^*) = F(p^*, \theta^*)$. Suppose there is some θ^{**} ϵ-close to θ^* at which the log-likelihood is larger, and that p^{**} is the corresponding $\mathsf{P}_{\theta^{**}}(\mathcal{Y} \mid \mathcal{X})$. Then it must be that $F(p^{**}, \theta^{**}) > F(p^*, \theta^*)$. But, assuming that $\mathsf{P}_{\theta^*}(\mathcal{Y} \mid \mathcal{X})$ varies continuously with θ^*, if θ^{**} is ϵ-close to θ^* then p^{**} is δ-close to p^*. This violates the assumption that F achieves a local maximum at p^*, θ^*, and so there can be no such θ^{**} close to θ^* with larger likelihood. Thus $\ell_{\mathcal{X}}(\theta^*)$ is a local maximum. A similar argument can be made for the global maximum (and we don't even need the continuity assumption).

1.8 Generalizations of EM

This formulation does not just provide straightforward access to the above proof; it also justifies a number of generalizations of the EM algorithm. The first actually follows from the argument

following (1.36) and appeared in (Dempster *et al.* 1977). This is the generalized M-step. As long as, at each iteration, the function Q is increased relative to its value at θ^{n-1}, all of the guarantees of increasing the likelihood are maintained. We do not need to maximize Q at each iteration, we can instead just take a step in the direction of its gradient (provided we are guaranteed that Q will indeed be maximized at convergence – see the comments below). This variant is called **gradient** or **generalized** EM (usually written GEM):

E-step: Find the **expectation** of the joint data log-likelihood under the distribution of the y_i given the $n-1$th parameter estimates and the observations. (This is unchanged.)

$$Q^n(\theta) = \mathcal{E}_{\mathcal{Y}|\mathcal{X};\theta^{n-1}}\left[\ell_{\mathcal{X},\mathcal{Y}}(\theta)\right] \tag{1.44}$$

GM-step: Change θ in the direction of the gradient of Q.

$$\theta^n = \theta^{n-1} + \eta \nabla_\theta Q^n(\theta^{n-1}) \tag{1.45}$$

where η is some learning rate parameter chosen in accordance with the usual principles of gradient ascent. Clearly, this is useful when Q cannot be maximized in closed form. In such situations it is usually computationally more efficient to use GEM rather than numerically optimizing Q in each M-step.

The free energy formulation suggests an alternative generalization. In principle, we could make a corresponding generalized E-step, and choose a function p different from $\mathsf{P}_{n-1}(\mathcal{Y} \mid \mathcal{X})$, provided it increases the free energy. We must be careful, however. We have shown that when the free energy reaches a local maximum, so does the likelihood. If we generate functions p by an algorithm that can converge even though F is not at a true local maximum, our guarantees of maximal likelihood evaporate. Such a situation arises when the functions p are restricted in functional form so that for most values of θ the function $\mathsf{P}_\theta(\mathcal{Y} \mid \mathcal{X})$ does not lie within the family of possibilities. In this case we can at best optimize F on the surface of constraint defined by the function family. An example is found in the Helmholtz machine (Dayan *et al.* 1995). The wake-sleep learning algorithm (Hinton *et al.* 1995) for the Helmholtz machine involves exactly such a constrained generalized E-step where the estimate p must be the output of a sigmoid belief network. As a result, it cannot guarantee convergence to the maximum likelihood parameters.

A similar caution, of course, can apply to generalized M-steps too. The usual choice of a gradient M-step, however, *is* guaranteed to converge to a local stationary point of F.

One example of an approximate E-step that maintains the convergence properties is provided by Neal and Hinton (1998). This is the incremental E-step, applicable when the x_i and y_i are independent. In this case, we can restrict the functions p to the family of functions with the form

$p(\mathcal{Y}) = \prod_i p_i(y_i)$ since the independence of the y_i guarantees that the optimal p will be in the family. We can now write

$$\begin{aligned} F(p,\theta) &= \sum_i F_i(p_i,\theta) \\ &= \sum_i \mathcal{E}_{p_i}\left[\ell_\theta\left(x_i,y_i\right)\right] + H(p_i) \end{aligned} \tag{1.46}$$

and maximize each component F in turn. The incremental EM algorithm now proceeds from initial guesses θ^0 and p_i^0 so:

IE-step: Choose some i. Maximize $F_i(p_i,\theta^{n-1})$ and leave the remaining $p_j, j \neq i$ unchanged.

$$\begin{aligned} p_i^n(y_i) &= \mathsf{P}_{n-1}\left(y_i \mid x_i\right) \\ p_j^n(y_j) &= p_j^{n-1}(y_j) \end{aligned} \tag{1.47}$$

M-step: Maximize F with respect to θ holding p constant.

In practice, for many distributions of interest, the M-step can be performed from sufficient statistics of the data, which are efficiently updated with respect to p_i (Neal and Hinton 1998). We shall, in fact, use a similar approach to track non-stationary mixture distributions efficiently.

Chapter 2 Clustering and Mixture Models

2.1 Clustering of Data

We have laid out our overall goal as follows: given a group of observations $\mathcal{X} = \{x_i \mid i = 1 \ldots N\}$, x_i not necessarily univariate or independent, discover the structure of the stochastic process from which the data arose. In this chapter we will investigate one particular form of structure: we will examine ways to discover if the data fall naturally into distinct clusters of points.

Clustering has a long history of essentially *ad hoc* techniques (Duda and Hart 1973; Jain and Dubes 1988). In recent years, however, considerable progress has been made with various statistically well-founded techniques. In our treatment of the problem we will pass very quickly to one particular statistical model, the **mixture**, which will be seen to be a particularly simple example of a latent variable model.

In general, the clustering problem assumes that the observations are independent and identically distributed (iid), and further that some measure of dissimilarity between observations is available. This measure may be quite general; there is no need for it be symmetric, to obey the triangle inequality, or even to be always nonnegative. Many of the techniques which work with these weak assumptions are fundamentally **agglomerative**, that is they form the data into progressively larger clusters by merging together smaller groups that display significant similarity. We shall not discuss such algorithms; many examples are reviewed by Jain and Dubes (1988).

Probabilistic models require well-defined measures in the space of observations, which in turn require a defined metric. Thus, we will examine clustering problems where the similarity measure obeys all the requirements of a metric. Indeed, we will go further and assume that each of our observations defines a point in $\mathbb{R}^D$, and that the similarity measure is simply the Euclidean distance between the points. In particular, this assumption allows us to speak of distances to points that were not observed, and thus to speak quantitatively of the *process* that generated the data, something not always possible in the extremely general spaces.

In this early treatment we shall also assume that the number of clusters, M, is known. Once we have achieved a properly probabilistic framework, the problem of determining the number of clusters will be reduced to that of model selection and so the techniques of the previous chapter will become applicable.

A particularly straightforward criterion for the assignment of D-dimensional observations $\{x_i\}$ to M clusters is as follows. We associate with each cluster a central point $\mu_m \in \mathbb{R}^D, m = 1 \ldots M$,

and then require that the sum of the squared distances from each point to the center of its assigned cluster be minimal. For this to be the case, it is clear that μ_m must be the mean of the observations assigned to the mth cluster, hence this approach is often referred to as the **k-means** clustering criterion (McQueen 1967). (The 'k' in k-means refers to the number of clusters, a quantity for which we have chosen the symbol M.)

The clustering is fully specified by the location of the μ_m, since the assignments of the x_i are then determined by which mean is closest. How are we to find the optimal locations of the μ_m? Iterative algorithms to do this have been known since the 60's. The basic approach was provided by Forgy (1965) (this approach is also known, in the related vector quantization literature, as the Lloyd–Max algorithm). We begin with an initial, random partitioning of the data into M sets. The μ_m are placed at the means of these data sets. We then iterate the following two steps

1. Re-assign all data points to the closest μ_m.

2. Move each μ_m is the mean of its assigned data.

This basic iteration (which, as we will see, is quite reminiscient of the EM algorithm) is what we shall call the k-means algorithm.

A number of variants of this basic approach have been suggested. For completeness, we mention them here; no details will be provided and we will not encounter them again in this dissertation, preferring instead the probabilistic approach described below. A more complete review is available in Duda and Hart (1973), Jain and Dubes (1988) or Ripley (1996).

The ISODATA algorithm (Hall and Ball 1965; see also Duda and Hart 1973) introduces an additional step to the iteration above, in which the number of clusters may be adjusted. Hartigan and Wong (1979) re-assign only one data point at a time, updating the means each time a point changes hands. McQueen (1967) gives an incremental algorithm, in which data are considered one-by-one in a single pass and the corresponding cluster mean updated after each assignment. Adaptive resonance theory (ART) (Carpenter and Grossberg 1987a, 1987b, 1990) provides a similar scheme within a "neural" framework; rather than choosing the closest mean, the data point is compared to each in a set order and the assignment is made to the first cluster for which the data point falls within a distance threshold. In addition, the distortion measures involved in ART are not exactly the squared-distance measures of the other techniques.

2.2 A Statistical Interpretation

As presented, the k-means and related algorithms appear *ad hoc*, but in fact they can be given a statistical interpretation (Scott and Symons 1971). We note that the sum of squared distance from μ_m is (up to a normalization constant) the negative log-likelihood of the model that the data are

generated by an isotropic (that is, identity covariance matrix) multivariate Gaussian distribution with mean μ_m. Thus, we can introduce the following likelihood function

$$\mathcal{L}_{\mathcal{X}}\left(\{\mu_m\}, \mathcal{Y}\right) = \prod_i G(x_i - \mu_{y_i}) \tag{2.1}$$

where $\mathcal{Y} = \{y_i\}$ is a set of assignment variables taking values between 1 and M, which tell us in which cluster the each observation falls, while $G(\cdot)$ denotes a standard multivariate Gaussian density with mean 0 and covariance I. The values of $\{\mu_m\}$ and $\mathcal{Y}$ which maximize this likelihood are precisely the solutions to the k-means sum-of-squares criterion. We have therefore converted our clustering problem into maximum-likelihood estimation.

This viewpoint also allows us to easily generalize the sum-of-squares criterion. In place of the isotropic Gaussian, we might choose Gaussians with arbitrary covariance matrices, so that each cluster is ellipsoidal but can have a different size and orientation. Indeed, we can in general choose any parameterized family of densities, and require that each cluster be represented by one of them (Scott and Symons 1971; Banfield and Raftery 1993). The likelihood is then

$$\mathcal{L}_{\mathcal{X}}\left(\theta, \mathcal{Y}\right) = \prod_i \mathsf{P}_{\theta_{y_i}}\left(x_i\right) \tag{2.2}$$

where the $\theta_m, m = 1 \ldots M$ parameterize the densities. If we are to retain the intuitive notion of a cluster being spatially compact we would expect the densities to all be well localized. Algorithms to maximize these likelihoods are exactly analogous to the procedures we discussed above in what we now see was the isotropic Gaussian case.

In this framework we maximize the likelihood with respect to both the density parameters and the assignment variables simultaneously. This is appropriate if our goal is to group the data at hand, as is often the case. However, the project we laid out was to discover the nature of the process that generated the data. The process is characterized only by the density parameters, along with the probability distribution of the y_i. The particular choices of the y_i are not important, and indeed we wish to maximize not the likelihood (2.2), but its marginal taken over all the possible assignments $\mathcal{Y}$. This leads to the mixture model.

2.3 Mixture Models

The **mixture model** is perhaps the simplest example of a latent variable statistical model. It consists of a single observed vector variable and one discrete scalar latent variable. Both observations and latent variables are iid. This model is represented by the graph in figure 2.1a, using all the

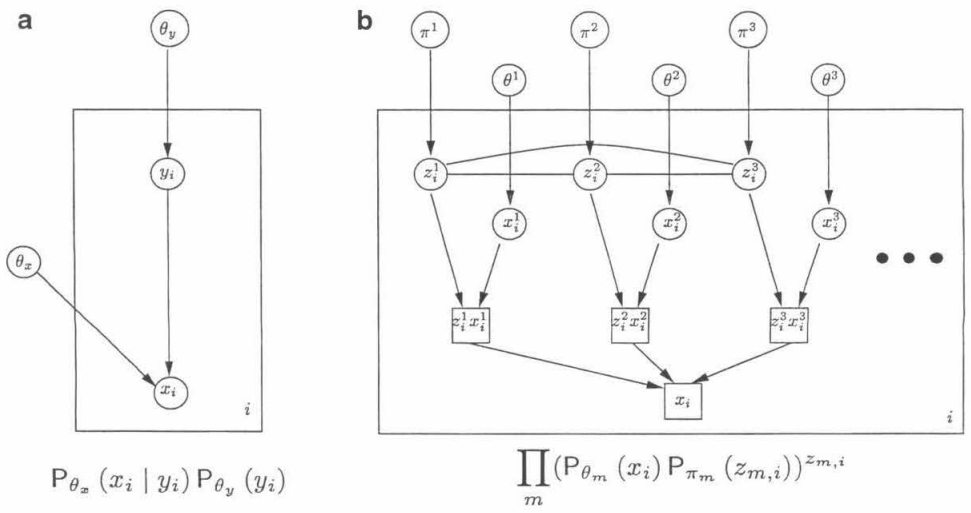


Figure 2.1: A mixture model.

conventions introduced in section 1.4. The marginal density of the ith observation x_i is

$$\mathsf{P}_\theta(x_i) = \sum_{y_i} \mathsf{P}_\theta(y_i)\,\mathsf{P}_\theta(x_i \mid y_i) \tag{2.3}$$

where the sum is taken over all the possible values the latent variable might assume. The choice of discrete values available to y_i is arbitrary, although the number of such values is not. We will write M for the number of distinct values the latent variable can take, and will assume that these values lie in the range $1 \ldots M$. The distribution function of the y_i is unconstrained, and so is parameterized by the probabilities associated with each value (strictly, by the probabilities of the first $M-1$ values). We will write π_m for $\mathsf{P}_\theta(y_i = m)$ and $\mathsf{P}_{\theta_m}(x_i)$ or even just $\mathsf{P}_m(x_i)$ for $\mathsf{P}_\theta(x_i \mid y_i = m)$. We can then rewrite the marginal density thus,

$$\mathsf{P}_\theta(x_i) = \sum_{m=1}^{M} \pi_m \mathsf{P}_{\theta_m}(x_i) \tag{2.4}$$

where the parameter set $\theta = \{\pi_1 \ldots \pi_M, \theta_1 \ldots \theta_M\}$.

Why the name "mixture model"? The latent variable can be viewed as a gate that, for each observation, selects one of the densities $\mathsf{P}_m(\cdot)$, from which the x_i is then drawn. Thus, the resultant set of observations is formed by mixing together sets of data drawn from each of the **component** densities $\mathsf{P}_m(\cdot)$. The relative sizes of these sets are defined by the **mixing parameters** π_m.

2.4 EM for Mixtures

The EM algorithm for mixture distributions has a particularly appealing form. The log-likelihood function for the parameters is

$$\ell_{\mathcal{X}}(\theta) = \sum_i \log \sum_{m=1}^{M} \pi_m \mathsf{P}_{\theta_m}(x_i) \tag{2.5}$$

which has the log-of-sum structure common to latent variable models. The joint data log likelihood is

$$\ell_{\mathcal{X},\mathcal{Y}}(\theta) = \sum_i \log \pi_{y_i} \mathsf{P}_{\theta_{y_i}}(x_i) \tag{2.6}$$

Written in this way, it is hard to manipulate. For this reason we will first re-express the mixture density in a way more conducive to application of EM.

In place of the single M-valued latent variable y_i we introduce a set of M binary-valued indicator latent variables $z_{m,i}$. For any observation, the one of these corresponding to the value of y_i takes the value 1, while the others are all 0. This version of the model is drawn in figure 2.1b. The $z_{m,i}$ are all dependent on each other. A random variable $x_{m,i}$ is drawn from the mth component distribution and multiplied by the value of $z_{m,i}$. All of these products are summed to produce the final observation. The square nodes in the graph represent deterministic combinations of random variables.

Armed with the variables $z_{m,i}$ we can rewrite the joint data log-likelihood

$$\ell_{\mathcal{X},\mathcal{Z}}(\theta) = \sum_i \sum_m z_{m,i} \log \pi_m \mathsf{P}_{\theta_m}(x_i) \tag{2.7}$$

with only one term in the inner sum being non-zero. The fact that this expression is linear in the $z_{m,i}$ makes the E-step of the EM algorithm quite straightforward.

$$\begin{aligned} Q^n(\theta) &= \mathcal{E}_{\mathcal{Z}|\mathcal{X},\theta^{n-1}}\left[\ell_{\mathcal{X},\mathcal{Y}}(\theta)\right] \\ &= \mathcal{E}_{\mathcal{Z}|\mathcal{X},\theta^{n-1}}\left[\sum_i \sum_m z_{m,i} \log \pi_m \mathsf{P}_{\theta_m}(x_i)\right] \\ &= \sum_i \sum_m \mathcal{E}_{z_{m,i}|x_i,\theta^{n-1}}\left[z_{m,i}\right] \log \pi_m \mathsf{P}_{\theta_m}(x_i) \\ &= \sum_i \sum_m r_{m,i}^n \log \pi_m \mathsf{P}_{\theta_m}(x_i) \end{aligned} \tag{2.8}$$

where we have written $r_{m,i}^n$ for $\mathcal{E}_{z_{m,i}|x_i,\theta^{n-1}}[z_{m,i}]$. The variable $z_{m,i}$ is binary, and so its expected value is just the probability that it assumes the value 1, which it does when the gating variable y_i

is equal to m. Thus,

$$\begin{aligned} r_{m,i}^n = \mathcal{E}_{z_{m,i}|x_i,\theta^{n-1}}\left[z_m^i\right] &= \mathsf{P}_{\theta^{n-1}}\left(y_i = m \mid x_i\right) \\ &= \frac{\mathsf{P}_{\theta^{n-1}}\left(x_i \mid y_i = m\right)\mathsf{P}_{\theta^{n-1}}\left(y_i = m\right)}{\mathsf{P}_{\theta^{n-1}}\left(x_i\right)} \\ &= \frac{\pi_m^{n-1}\mathsf{P}_{\theta_m^{n-1}}\left(x_i\right)}{\sum_l \pi_l^{n-1}\mathsf{P}_{\theta_l^{n-1}}\left(x_i\right)} \end{aligned} \tag{2.9}$$

In other words, the number $r_{m,i}^n$ is the posterior probability that the ith observation was generated from mth component, under the $(n-1)$th iteration of the parameters. It is called the **responsibility** of the mth component for the ith observation. In clustering terms it can be thought of as the degree to which observation x_i is associated with cluster m.

We can also say some general things about the M-step without knowing the form of the component densities. Rewriting (2.8), we have

$$Q^n(\theta) = \sum_m \log \pi_m \sum_i r_{m,i}^n + \sum_m \sum_i r_{m,i}^n \log \mathsf{P}_{\theta_m}\left(x_i\right) \tag{2.10}$$

and so the maximization with respect to π_m and θ_m can proceed separately. We can find the new values of the π_m directly. We impose the constraint $\sum \pi_m = 1$ using a Lagrange multiplier λ and differentiate to obtain

$$\left.\frac{\partial}{\partial \pi_m}\right|_{\pi_m^n} \left(\sum_m \log \pi_m \sum_i r_{m,i}^n - \lambda \sum \pi_m\right) = \sum_i \frac{r_{m,i}^n}{\pi_m^n} - \lambda = 0 \tag{2.11}$$

and so π_m^n is proportional to $\sum_i r_{m,i}^n$. The normalization constraint then gives us

$$\pi_m^n = \frac{\sum_i r_{m,i}^n}{|\mathcal{X}|} \tag{2.12}$$

where the denominator is the number of observations and we have used the fact that $\sum_m r_{m,i}^n = 1$.

We cannot, of course, solve for the θ_m^n without knowing the forms of the component densities, but even here we can make a little headway. First, note that the θ_m (unlike the π_m) are independent of each other, and so we can maximize with respect to each component separately. Furthermore, the only term in (2.10) that depends on θ_m is $\sum_i r_{m,i}^n \log \mathsf{P}_{\theta_m}\left(x_i\right)$. Now, if we were to fit the mth component density alone to all of the observations, we would find the parameters by maximizing the log-likelihood $\sum_i \log \mathsf{P}_{\theta_m}\left(x_i\right)$. Thus, we can interpret the M-step as fitting each of the component distributions to all of the observations, weighting the contribution of the ith datum to the log-likelihood by the responsibility $r_{m,i}^n$.

Here, then, is the EM algorithm for mixture distributions:

E-step: Calculate the responsibilities at the nth iteration

$$r_{m,i}^{n} = \frac{\pi_m^{n-1} \mathsf{P}_{\theta_m^{n-1}}(x_i)}{\sum_l \pi_l^{n-1} \mathsf{P}_{\theta_l^{n-1}}(x_i)} \tag{2.13}$$

M-step: Estimate the new mixing parameters

$$\pi_m^n = \frac{\sum_i r_{m,i}^n}{|\mathcal{X}|} \tag{2.14}$$

and the new component distribution parameters

$$\theta_m^n = \underset{\theta_m}{\operatorname{argmax}} \sum_i r_{m,i}^n \log \mathsf{P}_{\theta_m}(x_i) \tag{2.15}$$

2.5 Applications of Mixture Models

We have introduced the mixture model from the point of view of clustering. The component densities are thus taken to represent different physical processes, the observed data being a mixture of points generated by these processes. The mixture-model likelihood and the EM algorithm used to optimize it, differ in focus from the clustering likelihood of (2.2) and the k-means algorithms: the mixture parameter estimates describe the generating process, while the sum-of-squares and related methods find the best grouping of the observed data. In general, if we consider many sets of data that generated by mixing the outputs of the same group of processes, we expect the mixture parameter estimates to exhibit much tighter variance than their clustering analogues. In situations where we expect to classify new data, or to make predictions, it is clear that the former approach is to be preferred.

The difference may also be viewed in another way. The likelihood of (2.2) dictates a "hard" clustering scheme — the solution involves an explicit assignment of observations into clusters. In contrast, fitting the mixture model describes a "soft" or "fuzzy" clustering scheme where observations are not, in fact, classified, but are partially associated with clusters through the responsibilities. We might intuitively expect these techniques to yield different answers. Fuzzy clustering schemes have been proposed, without the probabilistic interpretation, within the theory of fuzzy sets (Backer 1978; Bezdek 1981).

The clustering view of mixture modeling is only really meaningful in situations where the component densities are reasonably well separated. In such cases the likelihood landscape generally exhibits sharp maxima to which EM converges quickly.

Mixture models can also be employed in situations where the component densities overlap for the purposes of density estimation. The mixture density (2.4) can be quite complex, even when

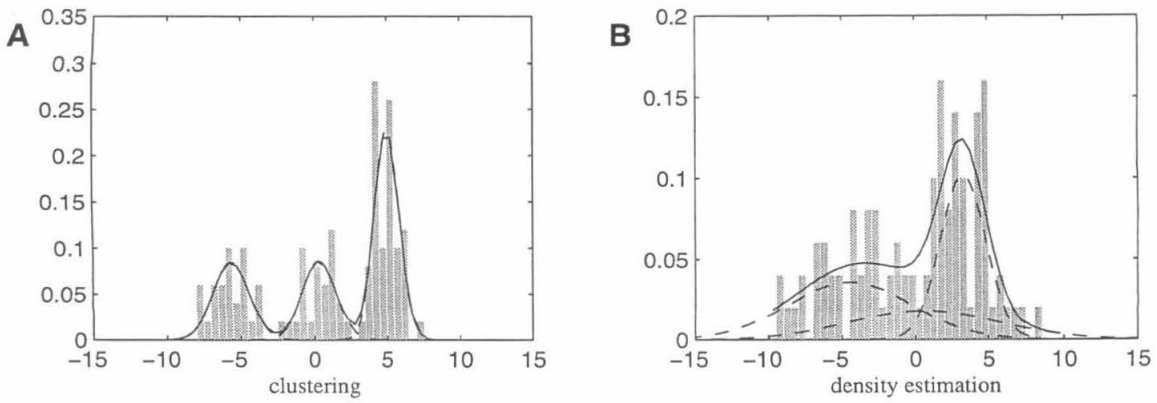


Figure 2.2: Two views of a mixture model.

the component distributions are relatively simple. As a result, complicated densities can be "non-parametrically" fit, with mixtures of Gaussians for instance, by the EM algorithm. From this viewpoint, there is no significance to the gating variable or to the component distributions – there is only one process with a complicated density and the mixture is just a convenient and flexible representation of the unknown density function. Indeed, one could view the familiar kernel-estimation technique as a particular case of a mixture model used in this way. The two views of the mixture model are illustrated in figure 2.2 where mixture models (the scaled components are shown by the dashed lines, the resulting mixture density by the solid lines) are fit to different types of one-dimensional data (histogrammed and shown by the grey bars).

We should make a short observation on our choice of the EM algorithm for learning the mixture model. If the component distributions overlap considerably it has been argued (Redner and Walker 1984) that the convergence of the EM algorithm to the optimal parameters of the mixture is slow (first order) and that superlinear methods should be preferred. However Redner and Walker (1984) themselves point out, and Xu and Jordan (1996) later elaborate, that the convergence of the *likelihood* of the mixture model is rapid, and that the mixture density approximates the true density quite quickly under EM. Thus, when the mixture model is used for clustering and thus the estimates of parameters are of importance, the components are likely to be reasonably well separated and therefore EM will converge well; while in the density estimation case, the criterion of importance is the convergence of the density estimate, and again this is rapid under EM.

2.6 Mixtures of Gaussians

A particularly fruitful mixture model, both in the context of clustering and of density estimation, arises when the components are (possibly multivariate) Gaussian densities. The parameters θ_m are

then a mean vector μ_m and a covariance matrix Σ_m. The log-likelihood of the model is

$$\ell_{\mathcal{X}}(\theta) = \sum_i \log \sum_{m=1}^{M} \pi_m |2\pi\Sigma_m|^{-1/2} e^{-\frac{1}{2}(x-\mu_m)^T\Sigma_m^{-1}(x-\mu_m)} \tag{2.16}$$

The joint data log-likelihood with the indicator latent variables (2.7) is then

$$\ell_{\mathcal{X},\mathcal{Z}}(\theta) = \sum_i \sum_m z_{m,i}\left(\log \pi_m - \frac{1}{2}\log|2\pi\Sigma_m| - \frac{1}{2}(x_i-\mu_m)^T\Sigma_m^{-1}(x_i-\mu_m)\right) \tag{2.17}$$

where the exchange of the logarithm and the sum has eliminated the exponentials. The E-step is as for a generic mixture distribution (2.13), in this case given by

$$r_{m,i}^n \propto \pi_m^{n-1} \left|2\pi\Sigma_m^{n-1}\right|^{-1/2} e^{-\frac{1}{2}(x-\mu_m^{n-1})^T(\Sigma_m^{n-1})^{-1}(x-\mu_m^{n-1})} \tag{2.18}$$

with the responsibilities normalized so as to sum to 1. In the M-step, the estimation of the mixing parameters is as for the generic mixture (2.14). The estimation of the mth component parameters is achieved by maximizing

$$Q_m^n(\theta) = -\sum_i r_{m,i}^n \left(\frac{1}{2}\log|2\pi\Sigma_m| + \frac{1}{2}(x_i-\mu_m)^T\Sigma_m^{-1}(x_i-\mu_m)\right) \tag{2.19}$$

Differentiating and equating to 0 we obtain

$$\begin{aligned} \left.\frac{\partial Q_m^n}{\partial \mu_m}\right|_{\mu_m^n} &= -\sum_i r_{m,i}^n (\Sigma_m^n)^{-1}(x_i-\mu_m^n) = 0 \\ \mu_m^n &= \frac{\sum_i r_{m,i}^n x_i}{\sum_i r_{m,i}^n} \end{aligned} \tag{2.20}$$

and (differentiating with respect to $R_m = \Sigma_m^{-1}$)

$$\begin{aligned} \left.\frac{\partial Q_m^n}{\partial R_m}\right|_{R_m^n} &= \sum_i r_{m,i}^n \left(\frac{1}{2}(R_m^n)^{-1} - \frac{1}{2}(x_i-\mu_m^n)(x_i-\mu_m^n)^T\right) = 0 \\ \Sigma_m^n &= \frac{\sum_i r_{m,i}^n (x_i-\mu_m^n)(x_i-\mu_m^n)^T}{\sum_i r_{m,i}^n} \end{aligned} \tag{2.21}$$

Thus the mean is updated to the responsibility-weighted mean of the observations, and the covariance to their responsibility-weighted covariance. This is a particularly elegant and fast update.

2.7 Practical Issues

We have argued that in situations where predictive power is desired, or where the parameters of the generating model are to be estimated as accurately as possible, the mixture model approach

to clustering is to be preferred. Can we then blindly fit (with the EM algorithm) a basic mixture model to solve all clustering problems that confront us? Unfortunately, we will find that a number of practical issues need to be examined quite closely before we can achieve robust and repeatable parameter estimates.

We shall raise the issues one by one, discussing briefly some of the possible solutions to them as we proceed. The order is arbitrary, and some of the more basic and serious points are not discussed until last. In chapter 3 we will discuss in depth an elaboration of the EM algorithm which provides a new way to address a number of these issues.

2.7.1 Outliers

It is often the case that some of the data under consideration do not fall into any of the data clusters. These **outliers** may be caused by measurement errors, such as sensor artifacts or data mis-entry, or may be due to an additional data generating process which is diffuse and for which no model is available. The outliers may have a considerable effect on the estimates of the cluster parameters. For example, in a mixture of Gaussians clustering algorithm, the estimate of the mean for each Gaussian component is disproportionally sensitive to data from the tails of the distribution. The outliers fall far from all of the Gaussian clusters but nevertheless must be assigned to one or the other of them. As such, they will perturb the estimates of the means.

We can resolve this problem by introducing an additional generative component in the mixture which can take responsibility for the outliers[1]. This component density must be far more diffuse that the cluster densities, and must perturb the component density estimates as little as possible.

The most suitable choice for the outlier component probability is found in the uniform density. More precisely,

$$\mathsf{P}_O(x_i) = \begin{cases} \frac{1}{\|A\|} & \text{if} \quad x_i \in A \\ 0 & \text{if} \quad x_i \notin A \end{cases} \tag{2.22}$$

for some region A. This choice correctly embodies (in the Bayesian sense) our utter lack of knowledge of the distribution from which the outliers are drawn. Furthermore, it tends to minimize the pertubation in the cluster parameter estimates. We will make this assertion more precise in the particular case of Gaussian clusters.

Without loss of generality, we consider data drawn from a single Gaussian cluster, with mean μ and covariance Σ, corrupted by the addition of some outliers. We fit a model that has two components: one Gaussian and the other uniform. For simplicity in this analysis, assume that any outliers fall far from the center of the cluster and, as a result, have negligible responsibility assigned to the Gaussian. Under this assumption, the outliers themselves do not disturb the estimates of

[1] Banfield and Raftery (1993) take a similar approach in the context of hard clustering, introducing a Poisson distribution for outlier generation

the Gaussian parameters. However, the density of the uniform component within the region of the cluster is not negligible, and so responsibility for points that were, in fact, generated from the Gaussian is shared between the Gaussian and the uniform component. How will this sharing affect the estimates of the parameters of the Gaussian?

Consider the transform $\Sigma^{-1/2}$ applied to the data space. Both the Gaussian and the Uniform densities enjoy the property of mapping to another member of their respective families under a linear transformation, so that the nature of the mixture is unchanged. In this space, the data that belong to the cluster will be distributed according to a unit Gaussian (one with a covariance matrix equal to the identity). Without loss of generality, take the mean to be 0. We write $\tilde{\mu}$ and $\tilde{\Sigma}$ for the estimated mean and covariance, respectively, of the Gaussian component. Let the value of the uniform density in this space be $\tilde{u}$. The mixing probabilities are π_g and π_u for the Gaussian and uniform components respectively.

The following system of equations must hold at the maximum likelihood parameter values,

$$\begin{aligned} r_{g,i} &= 1 - \frac{\pi_u \tilde{u}}{\left(\pi_u \tilde{u} + \pi_g \left|2\pi\tilde{\Sigma}\right|^{-1} \exp\left(-\frac{1}{2}(\tilde{x}_i - \tilde{\mu})^T \tilde{\Sigma}^{-1}(\tilde{x}_i - \tilde{\mu})\right)\right)} \\ \tilde{\mu} &= \frac{\sum_i r_{g,i} x_i}{\sum_i r_{g,i}} \\ \tilde{\Sigma} &= \frac{\sum_i r_{g,i}(x_i - \tilde{\mu})(x_i - \tilde{\mu})^T}{\sum_i r_{g,i}} \end{aligned} \tag{2.23}$$

It is difficult to derive expressions for the estimates $\tilde{\mu}$ and $\tilde{\Sigma}$ directly, however we can make some arguments based on the symmetry of the situation. The data within the cluster are generated from a spherically symmetric distribution. Neglecting edge effects, the uniform density is also completely symmetric. Thus, on the average, there cannot be any directional bias to the estimates. This means that the expected value of $\tilde{\mu}$ must be 0, since any other value would break symmetry. Similarly, the expected value of $\tilde{\Sigma}$ must be isotropic, and will generally be slightly smaller than the true covariance in the transformed space I. These comments are about the *expected* values of the estimates, particular values of the estimates will be different based on the particular data instances being fit.

What do these results tell us about the estimated Gaussian in the original space? The linear transform $\Sigma^{1/2}$ maps from the whitened space to the original one. Since expectations are linear functions, the expected values of the parameter estimates are simply the transforms of the corresponding values in the whitened space. The estimated mean is thus distributed around the true value of the mean. The expected value of the covariance estimate is slightly smaller than the true covariance, but has the same shape in the sense of the same eigenvectors, and eigenvalue ratios.

It is important to note that this invariance came as a result of the uniform density being sub-

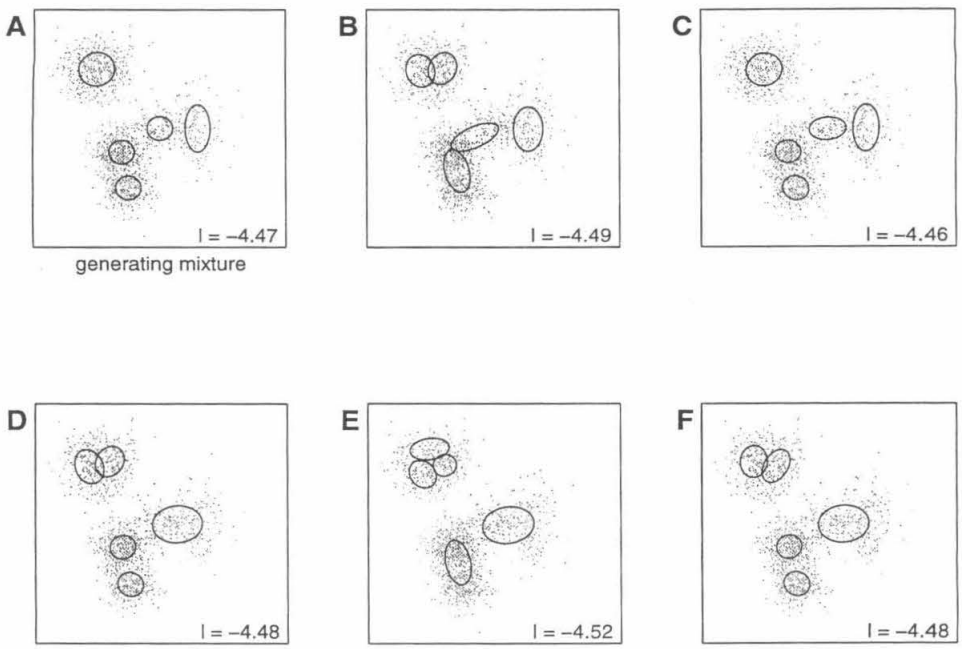


Figure 2.3: Multiple maxima in the mixture likelihood

stantially symmetric under any linear transform. Any other distribution would have had to have been carefully crafted to be symmetric. Furthermore, we would have to know a good deal about the cluster distribution to do so. With many, differently shaped, clusters only the uniform density will suffice.

2.7.2 Multiple maxima

The likelihood surface associated with a typical mixture model tends to exhibit multiple maxima. Trivially, given locally optimal parameters $\{\pi_m, \theta_m\}$, another maximum can be identified by retaining the same numerical values but permuting the component indices. In this case, the different maxima are equivalent in all practical senses and any one of them provides an equally good fit. Unfortunately, the system also exhibits non-trivial multiplicity.

Figure 2.3 illustrates the problem. Two-dimensional data are generated from the Gaussian mixture shown in A (each Gaussian in the mixture is represented by its 1-*sigma* contour). Panels B–F show the results of 5 separate fits to these data. The average log likelihood per point for each model (including the generating model) is recorded in the bottom right corner. Each model is the result of an EM optimization, and each optimization has converged. The difference between the results lies in the initial values of the parameters which are used to seed the EM process. (As an aside note that the best optimum (C) has a larger log-likelihood than the generating model — the

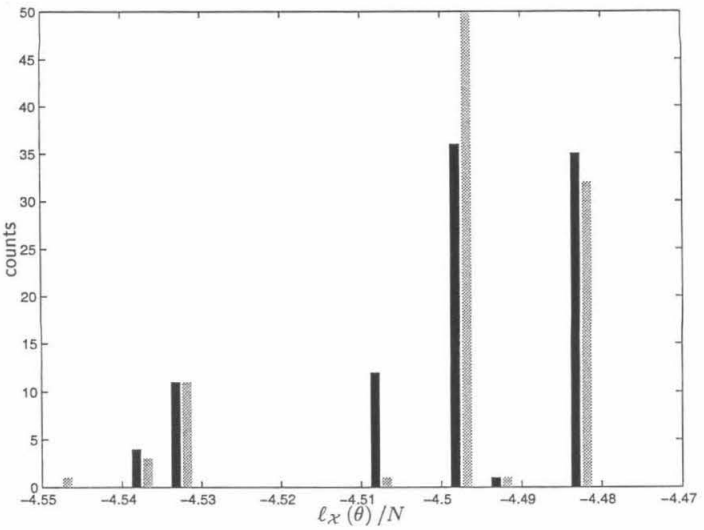


Figure 2.4: Likelihoods obtained from random restarts

data have permitted a small degree of over-fitting).

How are these initial values chosen? One generic approach, that does not depend on the type of component densities, is to randomly assign responsibilities for each data point and then derive the initial parameters using the M-step update rules. In large data sets, this approach tends to make the initial parameter values for each component virtually identical. This initial condition is similar to that of the REM algorithm to be discussed in chapter 3, however applying it in the standard EM context does not seem to be efficient. Convergence from such an initial point tends to be slow, and is no more reliable at finding a good maximum than the other techniques mentioned below.

An alternative approach, particularly useful in the case of mixtures of Gaussians (or the similar, well-localized, densities that are commonly used for clustering), is to pick a single covariance matrix (scale parameter) and initialize the means (location parameters) to randomly chosen data points. This is the method that was used to generate the fits in figure 2.3. We can refine the technique slightly by using these initial locations as the seed for a k-means clustering algorithm, and then using the output of that algorithm to provide the initial values of location parameters of the mixture model. K-means algorithms are also sensitive to the seed parameter values, but often less so than the full mixture, and so this initial stage tends to stabilize the estimates slightly. Nevertheless, experiments (an example appears in figure 2.4, to be described more completely below) suggest that in many situations the improvement is only very slight.

In general, optimization problems of this sort are known to be NP-hard, and so no entirely reliable, efficient solution can be found. Various approximate approaches are well-known in the optimization literature, and most may be adapted to the present problem. We will not dicuss most

of these here, instead referring the reader to the books by Hertz *et al.* (1991), for general techniques and McLachlan and Krishnan (1996) for EM specific approaches. One general method, simulated annealing (Kirkpatrick *et al.* 1983), will be described briefly in chapter 3, although we will not elaborate on the application of this approach to mixture models. However, the principal subject of chapter 3, relaxation EM, is extremely pertinant to this issue and application to mixture models will be discussed in some detail.

For the moment, we note one quite straightforward approach, which is often remarkably effective. This is simply to choose a number of random starting conditions by one of the means described above, maximize the mixture likelihood starting from each of these initial values, and then choose the result that provides the largest likelihood. Figure 2.4 shows a histogram of the different values of the log-likelihood per point obtain by running 100 optimizations on the data of figure 2.3. The dark bars show the results when the EM algorithm started directly from randomly chosen parameter values; the lighter bars show the results obtained when a simple k-means algorithm was run first. On the basis of this experiment, we conclude that approximately one-third of the random selected conditions yield the best maximum (given either initialization). Thus, in only 10 restarts of the algorithm, the probability of finding the best optimum is 0.985. Of course, this probability will be dependent on the problem being examined: an appropriate number of restarts will need to be determined through simulation for each new type of problem.

2.7.3 The number of clusters

In general, when presented with a clustering problem we have no *a priori* information about how many different clusters we will encounter. This number, along with the optimal parameters to describe each cluster, must be estimated from the available data. This is a classic example of the general problem of model selection, which was addressed at some length in section 1.3. All of the analysis of that section applies to the present problem, and the methods described there are frequently employed.

In this section we will add another result to the battery of approximations to the marginal likelihood. This new approximation, introduced by (Cheeseman and Stutz 1996), is peculiar to mixture models and related latent variable models. In the following chapter, we shall introduce a novel framework, cascading model selection, for the efficient application of these various techniques.

The Cheeseman-Stutz criterion

The marginal likelihood for a mixture model with M components is given by

$$\mathsf{P}_M(\mathcal{X}) = \int d\theta\, \mathsf{P}_M(\theta) \prod_{i=1}^{N} \left(\sum_{m=1}^{M} \pi_m \mathsf{P}_{\theta_m}(x_i) \right) \tag{2.24}$$

Even if the individual cluster likelihood $\mathsf{P}_{\theta_m}(x_i)$ can be integrated with respect to θ_m, the overall integral proves to be intractable due to the M^N terms that appear once the product is distributed over the sum.

On the other, hand, if the latent variable values (expressed as the indicators $z_{m,i}$) were known, the marginal likelihood in this case could be written in a simpler form (compare the joint log-likelihood (2.7))

$$\mathsf{P}_M(\mathcal{X}) = \int d\theta\, \mathsf{P}_M(\theta) \prod_{i=1}^{N} \prod_{m=1}^{M} (\pi_m \mathsf{P}_{\theta_m}(x_i))^{z_{i,m}} \tag{2.25}$$

$$= \int d\theta\, \mathsf{P}_M(\theta) \prod_{m=1}^{M} \pi_m^{(\sum_i z_{i,m})} \prod_{i=1}^{N} (\mathsf{P}_{\theta_m}(x_i))^{z_{i,m}} \tag{2.26}$$

This integral is more likely to be tractable. If the prior factors over the different cluster parameters θ_m the expression above reduces to the product of the marginal likelihoods of each cluster, given only the data assigned to that cluster.

Cheeseman and Stutz (1996) propose that we use this form, with the indicator values $z_{m,i}$ replaced by their expected values at the optimum, $r^*_{m,i}$, as the basis for an approximation of the true integral. In fact, direct substitution of the responsibilities into (2.26) will under-estimate the correct integral; however, the size of the error can be estimated from the mismatch between the value of the approximate integrand and the true likelihood at the estimated parameter values, θ^*. The complete approximation is

$$\mathsf{P}_M(\mathcal{X}) \approx \frac{\prod_{i=1}^{N} \left(\sum_{m=1}^{M} \pi^*_m \mathsf{P}_{\theta^*_m}(x_i) \right)}{\prod_{m=1}^{M} \pi_m^{R^*_m} \prod_{i=1}^{N} (\mathsf{P}_{\theta_m}(x_i))^{r^*_{i,m}}} \int d\theta\, \mathsf{P}_M(\theta) \prod_{m=1}^{M} \pi_m^{R^*_m} \prod_{i=1}^{N} (\mathsf{P}_{\theta_m}(x_i))^{r^*_{i,m}} \tag{2.27}$$

where we have written $R^*_m = \sum_i r^*_{m,i}$.

Chapter 3 Relaxation Expectation–Maximization

In chapter 2 we noted a number of practical difficulties that arise in the use of the Expectation–Maximization (EM) algorithm to find maximum likelihood fits of mixture models. Two among these were the sensitivity to initial conditions and the computational overhead involved in carrying out model selection. In this chapter we shall introduce a modified EM algorithm which addresses both of these issues in a natural fashion. Our modifications will rely on the statistical mechanics notion of **relaxation**.

3.1 Annealing and Relaxation

3.1.1 Simulated annealing

Relaxation methods are well known in data analysis, primarily due to the popularity of the **simulated annealing** technique for the solution of non-convex optimization problems (Kirkpatrick *et al.* 1983). This being the most common example, we will review it briefly so as to provide a point of departure for our discussion.

The objective is to find the global minimum of a function $E(x)$. The approach taken is to simulate the motion (in x space) of a thermally excited particle under the influence of a potential energy landscape given by $E(x)$. In principle, at zero temperature the particle will be found at the global minimum. Of course, in practice, if it starts at a position far from the lowest energy point it will most likely travel to a local minimum and come to rest there. At higher temperatures, the particle will travel rapidly all over the landscape, spending relatively more time in regions where the function $E(x)$ is minimal. The annealing procedure lowers the simulated temperature gradually. As the temperature falls, the bias towards regions of lower energy increases, while the particle is still able to cross barrier regions of higher energy. If the rate of cooling is sufficiently gradual, these two tendencies — the attraction to regions of low energy and the thermal activation to cross energy barriers — combine in such a way as to inevitably leave the particle at the global minimum once the temperature reaches 0. Cooling schedules which guarantee this result can be shown to exist in principle (Geman and Geman 1984); however, they invariably take impractically long. Fortunately, less than perfect cooling schedules usually yield good results.

This physical picture of the optimization process is appealing, but it is difficult to build intuition for why the trade-off between activation energy and attraction to potential wells should work out so conveniently. Also, while it will be valuable to contrast this view with the "deterministic annealing"

or relaxation procedure we will discuss later, it is not the most convenient starting point for the development of the new approach. Therefore we reexamine the algorithm from a more statistical viewpoint.

3.1.2 Annealed sampling

The fundamental logic behind annealing schemes is best illustrated by the simulated annealing of Markov chain Monte-Carlo (MCMC) samplers (Neal 1993; Bertsimas and Tsitsiklis 1993). The objective here is to sample from some complicated target probability function $\mathsf{P}(x)$. For convenience, we will introduce an energy function given, up to an arbitrary additive constant, by $E(x) = -\log \mathsf{P}(x)$. The density is thus given by the Boltzmann equation $\mathsf{P}(x) = \frac{1}{Z}\exp(-E(x))$, for some normalizing constant Z. We are able to evaluate $E(x)$ for any point x, but the energy does not have a simple functional form that makes direct sampling by analytic means tractable. The MCMC sampling approach constructs an ergodic Markov-chain[1] over the target space such that the stationary distribution of the chain is $\mathsf{P}(x)$. In other words, we obtain a scheme for making probabilistic transitions from one point in the space to another in a memory-less (Markov) fashion, and such that, in the long run, the probability of visiting some point x is exactly $\mathsf{P}(x)$. A number of schemes to construct a suitable Markov chain exist, the most prominent being the Gibbs sampling and the Metropolis algorithms. The details of the process are unimportant for our purposes; we seek only to gain an intuitive picture of the value of annealing; the reader interested in more detail is referred to the excellent review by Neal (1993).

When using an MCMC sampler, we need to begin the chain at some point in the domain, say x_0. Since we cannot sample directly from the target density, this point must be chosen from an arbitrary density, probably quite different to the target one. Let us say this initial density is uniform on the domain of interest, although the argument is not crucially dependent on this choice. The density of the next point, call it x_1, is then the product of this uniform distribution and the transition density of the Markov chain, marginalized over x_0, $\mathsf{P}_1(x_1) = \int dx_0\ \mathsf{P}_0(x_0)\,\mathsf{P}(x_1 \mid x_0)$. (For discrete domains we can picture multiplying a vector representing the uniform distribution by a transition matrix.) The resultant density will also be far from the target, as will the densities of many subsequent samples. Thus, our necessarily poor choice of $\mathsf{P}_0(x_0)$ results in a "burn-in" period of incorrectly distributed samples. The typical length of this period is related to the mismatch between the initial distribution and the target (or stationary) distribution, and to the magnitude of the non-unit eigenvalues of the transition operator, which set the decay rate of the non-stationary modes in $\mathsf{P}_0(\cdot)$. In general, the mixing time cannot easily be calculated, but in experiments with practical examples it is often impractically long.

The difficulty is that in many problems $\mathsf{P}_0(x_0)$ is likely to ascribe a relatively large mass to

[1] The basic theory of Markov chains will be reviewed in section 4.1.1.

regions where the target function is vanishingly small, and furthermore, has small log-gradients. For domains of high dimensionality, the probability of falling in such regions can approach 1. The structure of the usual MCMC samplers (in particular, a feature called **detailed balance** which is needed to guarantee ergodicity) results in the sampler executing an almost unbiased random walk within that region until it finally emerges into a region of higher probability.

How can annealing help reduce this burn-in period? We create a sequence of probability functions $\mathsf{P}_0(x)$, $\mathsf{P}_1(x)$, ..., $\mathsf{P}(x)$ which starts with the uniform distribution and ends in the target. In the case of the Boltzmann distribution this sequence is easily constructed using a "inverse-temperature" parameter, β. We choose a sequence of β_i, starting with 0 and ending in 1, and write $\mathsf{P}_i(x) = \frac{1}{Z(\beta_i)} \exp(-\beta_i E(x))$, where $Z(\beta_i)$ is the partition function. By analogy with statistical physics, these densities correspond to the canonical distributions of a system with energy E cooled through a sequence of temperatures $T = 1/\beta$. We now choose an initial point from $\mathsf{P}_0(x)$ as before, but then use the MCMC sampler corresponding to the density $\mathsf{P}_1(x)$, with $0 < \beta_1 \ll 1$, rather than the target sampler. The mismatch between these two distributions is small by construction, and so this Markov chain will soon achieve the stationary distribution for $\mathsf{P}_1(x)$. Once enough time has elapsed to make convergence likely, we switch to sampling from $\mathsf{P}_2(x)$, where the same argument about quick convergence holds. Eventually, we reach the target distribution (at $\beta = 1$). In many situations, the total burn-in time for all of the annealing steps is much smaller than the burn-in encountered stepping directly to the target.

What does all this have to do with the physical picture of optimization by simulated annealing that we saw before? The Metropolis sampling algorithm used in some MCMC simulations has its origins in the physical simulation of particle motion, and, indeed, is precisely the simulation algorithm used by Kirkpatrick *et al.* (1983). If we extend to temperatures close to 0 ($\beta \gg 1$) the sequence of distributions discussed above, virtually all of the probability mass becomes concentrated near the global energy minimum. Provided the MCMC sampler is maintained in equilibrium, then, samples drawn in this limit will be arbitrarily close to the optimum. This is precisely the simulated annealing optimization algorithm.

3.1.3 Relaxation

We have examined the simulated annealing algorithm from two different points of view. In the first, the underlying energy landscape was fixed by the function to be optimized, while the motion of a thermally active particle in the landscape was simulated at steadily decreasing temperatures. In the second, the energy landscape was transformed from a flat initial condition to the target function *and beyond*, while samples were drawn from the corresponding Boltzmann distribution. This gradual transformation of the energy surface is called **relaxation**; for this reason, simulated annealing is also known as **stochastic relaxation**.

Optimization within a relaxation framework need not be stochastic. Let us focus on the energy functions themselves rather than on the implied Boltzmann densities. We can construct a sequence of functions, $E_0(x) \ldots E(x)$ such that the first function $E_0(x)$ is easily optimized — it might, for example, have a single extremum — while the final function is the target. Our goal in constructing this sequence is for the global optimum of the ith function $E_i(x)$ to lie within the domain of convergence of the global optimum of the next function $E_{i+1}(x)$. We then pass along the sequence of functions, optimizing each one by a hill-climbing (or, for minima, descending) algorithm, which is seeded with the location of the previous optimum. Thus, we hope to track the global optimum from $E_0(x)$, where it was easily found, to $E(x)$. Unfortunately, unlike the case of stochastic relaxation, there is no simple strategy that is guaranteed to provide a suitable sequence of functions in the case of such deterministic relaxation, even with exponentially long relaxation schedules, and indeed schemes devised for particular classes of energy (say mixture likelihoods) may not work even in all examples of that class. Nevertheless, in practice, this approach often does yield good results.

3.2 Deterministic Annealing

One example of a non-stochastic relaxation process has been called **deterministic annealing**. This algorithm was introduced by Rose *et al.* (1990) as a maximum entropy approach to clustering and vector quantization, following earlier work on **elastic net** algorithms for the traveling salesman problem (Durbin and Willshaw 1987; Durbin *et al.* 1989; Simic 1990; Yuille 1990). In this form, the algorithm is strongly motivated by physical analogy. Below, we will see that it can be generalized beyond its statistical physics origins, to yield a powerful procedure that can be applied to any problem in which the EM algorithm is used for learning. We shall refer to the generalization as Relaxation Expectation–Maximization, reserving the term "deterministic annealing" for the original formulation.

Rose *et al.* view clustering as a squared-distance distortion minimization operation. They introduce a cost function, $E_m(x_i)$, describing the distortion due to association of the the ith data point with the mth cluster. We shall take this cost to be the squared Euclidean distance $E_m(x_i) = ||\mu_m - x_i||^2$, although other distortions may be considered. The cost of adopting a particular set of cluster parameters $\theta = \{\mu_m\}$ *and* a particular assignment of points to clusters, represented by indicator variables $\mathcal{Z} = \{z_{m,i}\}$, is given by

$$E(\theta, \mathcal{Z}) = \sum_i \sum_m z_{m,i} E_m(x_i) \tag{3.1}$$

We have chosen notation different from that of Rose *et al.* (1990) in order to highlight the similarity to the mixture model development in chapter 2. This cost, $E(\theta, \mathcal{Z})$, may be viewed as the energy

of a microstate, identified by the pair $(\theta, \mathcal{Z})$, of a physical system and we may proceed by analogy to statistical physics (as we will see below, this analogy is not vital; the results follow directly from the maximum-likelihood framework and the EM algorithm). We expect the system to display a distribution over microstates $\mathsf{P}(\theta, \mathcal{Z})$. For a fixed average energy, E, this distribution will maximize the entropy under the constraint $\mathcal{E}[E(\theta, \mathcal{Z})] = E$ (see, for example, Kittel and Kroemer (1980)). We can find this maximizer by the method of Lagrange multipliers, optimizing the entropy $H = -\int d\theta \ \sum_{\mathcal{Z}} \mathsf{P}(\theta, \mathcal{Z}) \log \mathsf{P}(\theta, \mathcal{Z})$ while enforcing the constraint $E - \int d\theta \ \sum_{\mathcal{Z}} \mathsf{P}(\theta, \mathcal{Z}) E(\theta, \mathcal{Z}) = 0$ with the multiplier β. Doing so, we obtain the well-known Boltzmann distribution

$$\mathsf{P}_\beta(\theta, \mathcal{Z}) \propto e^{-\beta E(\theta, \mathcal{Z})} \tag{3.2}$$

The value of the multiplier β can be obtained by solving for the constraint energy. Rose *et al.* argue, as we have, that the distribution of interest in the case of modeling or prediction problems is not the joint, but rather the marginal

$$\mathsf{P}_\beta(\theta) = \sum_{\mathcal{Z}} \mathsf{P}(\theta, \mathcal{Z}) \propto \prod_i \sum_m e^{-\beta E_m(x_i)} \tag{3.3}$$

For the case of the squared distance cost, this is seen to be the same as the likelihood of a mixture of Gaussians with mixing probabilities $\pi_m = \frac{1}{M}$ and covariances $\Sigma_m = \frac{1}{2\beta} I$.

Given this "likelihood", they proceed to derive heuristically re-estimation equations similar to those of the EM algorithm (written here for the squared error distortion metric):

$$\begin{aligned} r_{i,m} &\leftarrow e^{-\beta E_m(x_i)} / \textstyle\sum_l e^{-\beta E_l(x_i)} \\ \mu_m &\leftarrow \textstyle\sum_i r_{i,m} x_i / \sum_i r_{i,m} \end{aligned} \tag{3.4}$$

We have again chosen notation to emphasize the connection to our previous development. The deterministic annealing algorithm then involves varying the value of the parameter β from 0 to a final value chosen either through some knowledge of the expected final distortion (due, say, to a known noise-floor), or else by a validation-based stopping criterion (or else by operator fiat). At each step the re-estimations (3.4) are iterated to convergence.

The intuitions that underlie this algorithm can be used to obtain similar solutions to a number of other problems (Rose *et al.* 1993; Buhmann and Kuhnel 1993; Miller *et al.* 1996; Kloppenburg and Tavan 1997; Rao *et al.* 1997; Rao *et al.* 1999). Many of these are reviewed by Rose (1998). In general, however, each such problem presents the need for a fresh derivation. Furthermore, it is not always clear how best to generalize the approach to some problems. For example, Kloppenburg and Tavan (1997) provide an extension to a mixture of multivariate Gaussians with arbitrary covariances; but they are forced to introduce multiple annealing parameters, leaving serious questions about the

choice of relative annealing schedules.

In the next section we will encounter a generalized relaxation method which subsumes the various deterministic annealing algorithms, and allows extremely straightforward generalization.

3.3 REM-1

In this section, we will develop a novel relaxation scheme within the framework of the EM algorithm, to obtain an algorithm that we call the first **Relaxation Expectation–Maximization** algorithm[2] (REM-1).

In section 1.7 we introduced a free-energy F, a function of the model parameters, θ, and a probability distribution on the latent variables, p,

$$F(p,\theta) = Q(p,\theta) + H(p) = \mathcal{E}_p\left[\ell_{\mathcal{X},\mathcal{Y}}(\theta)\right] - \mathcal{E}_p\left[\log p(\mathcal{Y})\right] \tag{3.5}$$

We showed that if this function achieved a maximum at (θ^*, p^*) the true model likelihood (marginalized over the latent variables) achieved a maximum at θ^*. This allowed us to interpret the EM algorithm as an alternation of optimization steps, maximizing F first with respect to p, and then with respect to θ. This view of EM forms the basis for our relaxation scheme.

Let us introduce an annealing parameter β so as to construct a family of free-energy functions,

$$F_\beta(p,\theta) = \beta Q(p,\theta) + H(p) \tag{3.6}$$

The analogy to statistical mechanics inherent in the term "free-energy" is maintained by this choice (modulo an overall minus sign). We may view β as the inverse of a (dimensionless) temperature, in which case it enters into the free-energy definition in the physically appropriate position. When β takes the value 1 (that is, $T = 1$) we recover the original free-energy, which is the target function whose maximum we seek. On the other hand, when β is 0 ($T \to \infty$) F is equal to the entropy $H(p)$. In general, there is a single, easy to find, global maximum of this entropy. For discrete latent variables, for example, it is achieved by the uniform distribution. For the case of the mixture model, in which the latent variables indicate with which cluster each point is associated, and we see that F_0 is maximized by associating all of the points uniformly with all of the clusters. The $\beta = 0$ case does not constrain the parameters θ at all, however it is convenient to choose θ as before, maximizing Q with p fixed at its maximum-entropy value.

Thus, the sequence of functions $F_{\beta_i}(p,\theta)$, $0 = \beta_0 < \beta_1 < \cdots < \beta_R = 1$ satisfies at least two of the conditions we desired for a relaxation progression: it starts with an easily maximized function

[2]The same formulation has been independently proposed under the name "Deterministic Annealing Expectation Maximization" by Ueda and Nakano (1998). A slightly different development, which we call REM-2, will appear below.

and ends with the target. To be sure of finding the global maximum of the target function we need another condition to be satisfied: the global maximum of each function in the sequence must lie within the basin of attraction of the global maximum of the next function. Provided that the location of global maximum changes continuously with β, this can be assured by choosing sufficiently small annealing steps.[3] Unfortunately, we will see below that even for the particularly simple example of the mixture model, the maximum does not move smoothly. In general it is not guaranteed that REM will find the global maximum of the target. However, in many common examples it does find a good maximum.

Any hill-climbing technique may be used to find the optimum of each succeeding free-energy in the relaxation sequence; however, we choose to employ the same approach as in the EM algorithm, alternately optimizing with respect to p and θ, in each case holding the other variable fixed. Note first that, for fixed p, the relaxation factor β has no effect on the optimal value of θ. Thus, the M-step of the algorithm is exactly as for the normal EM algorithm. The E-step, however, does differ.

We showed previously (1.42) that the target free-energy is maximized with respect to p (for fixed θ) by choosing $p(\mathcal{Y}) = \mathsf{P}_\theta(\mathcal{Y} \mid \mathcal{X})$. In the case of the relaxation free-energies we can proceed in the same fashion as we did at that point. We introduce a Lagrange multiplier λ enforcing the constraint $\int d\mathcal{Y}\, p(\mathcal{Y}) = 1$ and obtain

$$\begin{aligned} 0 &= \frac{\partial}{\partial p}\left(F_\beta(p,\theta) - \lambda \int d\mathcal{Y}\, p(\mathcal{Y})\right) \\ &= \frac{\partial}{\partial p}\left(\int d\mathcal{Y}\, p(\mathcal{Y})(\beta \ell_{\mathcal{X},\mathcal{Y}}(\theta) - \log p(\mathcal{Y}) - \lambda)\right) \end{aligned} \tag{3.7}$$

from which, by the calculus of variations,

$$\begin{aligned} 0 &= \frac{\partial}{\partial p}\left(p(\mathcal{Y})(\beta \ell_{\mathcal{X},\mathcal{Y}}(\theta) - \log p(\mathcal{Y}) - \lambda)\right) \\ &= (\beta \ell_{\mathcal{X},\mathcal{Y}}(\theta) - \log p^*(\mathcal{Y}) - \lambda) - \frac{p^*(\mathcal{Y})}{p^*(\mathcal{Y})} \end{aligned} \tag{3.8}$$

and so

$$p^*(\mathcal{Y}) = e^{-\lambda-1}(\mathcal{L}_{\mathcal{X},\mathcal{Y}}(\theta))^\beta = e^{-\lambda-1}(\mathsf{P}_\theta(\mathcal{X},\mathcal{Y}))^\beta \tag{3.9}$$

But $\mathsf{P}_\theta(\mathcal{X},\mathcal{Y}) = \mathsf{P}_\theta(\mathcal{X} \mid \mathcal{Y})\,\mathsf{P}_\theta(\mathcal{Y})$ and so

$$p^*(\mathcal{Y}) = \frac{1}{Z(\beta)}(\mathsf{P}_\theta(\mathcal{X} \mid \mathcal{Y})\,\mathsf{P}_\theta(\mathcal{Y}))^\beta \tag{3.10}$$

[3] This assertion can be proved by noting that a global maximum must have at least an ϵ-sized basin of attraction and that continuity guarantees that there exists some δ so that for a δ-sized step in β the change in global maximum is smaller than this ϵ.

with $Z(\beta)$ and appropriate normalizing constant.

Thus we obtain the steps of the REM-1 algorithm, repeated until $\beta = 1$.

R-step: Increment β according to the relaxation schedule.

Repeat the following EM steps until convergence:

E-step: Maximize F_β with respect to p holding θ fixed.

$$p(\mathcal{Y}) \leftarrow \frac{1}{Z(\beta)}(\mathsf{P}_\theta(\mathcal{X} \mid \mathcal{Y})\,\mathsf{P}_\theta(\mathcal{Y}))^\beta \tag{3.11}$$

M-step: Maximize F_β with respect to θ holding p fixed.

$$\theta \leftarrow \operatorname{argmax} \mathcal{E}_p[\ell_{\mathcal{X},\mathcal{Y}}(\theta)] \tag{3.12}$$

Relationship to deterministic annealing

The deterministic annealing algorithm for vector quantization described in section 3.2 is easily seen to arise from REM-1 applied to a simple mixture model. Consider an M-component model in which each component is a Gaussian with identity covariance matrix and mean μ_m. We will refer to this as a mixture of unit Gaussians. Any model in which the all of the components are known to share the covariance matrix Σ can be transformed to this canonical form by multiplying each data vector by the whitening matrix $\Sigma^{-1/2}$. The relaxation free-energy for such a model is

$$F_\beta(p,\theta) = \beta \sum_i \sum_m r_{m,i}(\log \pi_m - \frac{1}{2}||x_i - \mu_m||^2) - \sum_i \sum_m r_{m,i} \log r_{m,i} \tag{3.13}$$

where the distribution p is expressed in terms of the responsibilities $r_{m,i}$. For notational simplicity we have left out the normalization factor from the Gaussian. For a model with fixed, equal, covariances this factor does not change and careful inspection reveals that it does not survive in any of our eventual results.

The REM-1 iterations for such a model are easily seen to be given by

$$\begin{aligned} r_{i,m} &\leftarrow \tfrac{1}{Z_i}\pi_m^\beta e^{-\frac{1}{2}\beta\|x_i-\mu_m\|^2} \\ \pi_m &\leftarrow \textstyle\sum_i r_{i,m}/|\mathcal{X}| \\ \mu_m &\leftarrow \textstyle\sum_i r_{i,m}x_i / \sum_i r_{i,m} \end{aligned} \tag{3.14}$$

If we further constrain the mixing probabilities to remain equal, that is, $\pi_m = 1/M$, we obtain exactly the iterations of (3.4).

Note that in the case of the fixed mixing probabilities, the relaxation likelihoods correspond to true likelihoods for other models, in this case, a mixture of Gaussians with covariance $\beta^{-1}I$. This

allows us to interpret the relaxation procedure as the successive optimization of a sequence of models with shrinking covariances. This is actually a special case and for the majority of models no such equivalence holds. Given even the simple step of allowing unconstrained mixing probabilities, the iterations (3.14) do not correspond to EM for any model.

It is instructive to note that the maximization of the free-energy with respect to p, which is motivated in REM entirely by the maximum likelihood considerations of chapter 1, may indeed be interpreted as a maximization of the entropy of p under a "constraint" set by the expected joint log-likelihood and enforced by a Lagrange multiplier. This is in accordance with the physical analogy of Rose *et al.* (1990), although it is obtained directly without resort to the physics.

Yuille *et al.* (1994) remarked on a connection between the heuristic optimization steps usually employed within deterministic annealing solutions and the EM algorithm. However, they seem to regard EM simply as an optimization technique embedded within the physically motivated deterministic annealing framework. Notably, they appear to have failed to observe the deep connection between the free-energy formulation of EM and the relaxation procedures of deterministic annealing; in particular, they make no mention of the availability of a simple generalization of any EM algorithm to yield a relaxation (or "annealing") procedure.

3.4 Phase Transitions in REM

An important feature of deterministic annealing and relaxation EM is best illustrated in a simple example. We will use the mixture of unit Gaussians described in the preceding section. We will write $(r^*_{m,i}, \pi^*_m, \mu^*_m)$ for the optimum of the relaxation free-energy. Clearly, these values satisfy the recurrence relations

$$r^*_{m,i} = \frac{\pi^{*\beta}_m e^{-\frac{1}{2}\beta\|x_i - \mu^*_m\|^2}}{\sum_l \pi^{*\beta}_l e^{-\frac{1}{2}\beta\|x_i - \mu^*_l\|^2}} \tag{3.15}$$

$$\pi^*_m = \frac{\sum_i r^*_{m_i}}{|\mathcal{X}|} \tag{3.16}$$

$$\mu^*_m = \frac{\sum_i r^*_{m,i} x_i}{\sum_i r^*_{m,i}} \tag{3.17}$$

When $\beta = 0$ the relaxation E-step finds the maximum entropy distribution over the latent variables. For a mixture distribution, where the latent variables are discrete, this is the uniform distribution and

$$r^*_{m,i} = \mathsf{P}\left(z_{m,i} = 1 \mid x_i\right) = \frac{1}{M} \tag{3.18}$$

In this limit the relaxation free-energy is independent of θ and so the M-step is unconstrained. However, we can choose it to maximize $Q(\theta, p^*)$ where p^* is the maximum entropy distribution described above, thereby preserving consistency with the $\beta > 0$ case. As the responsibilities for each

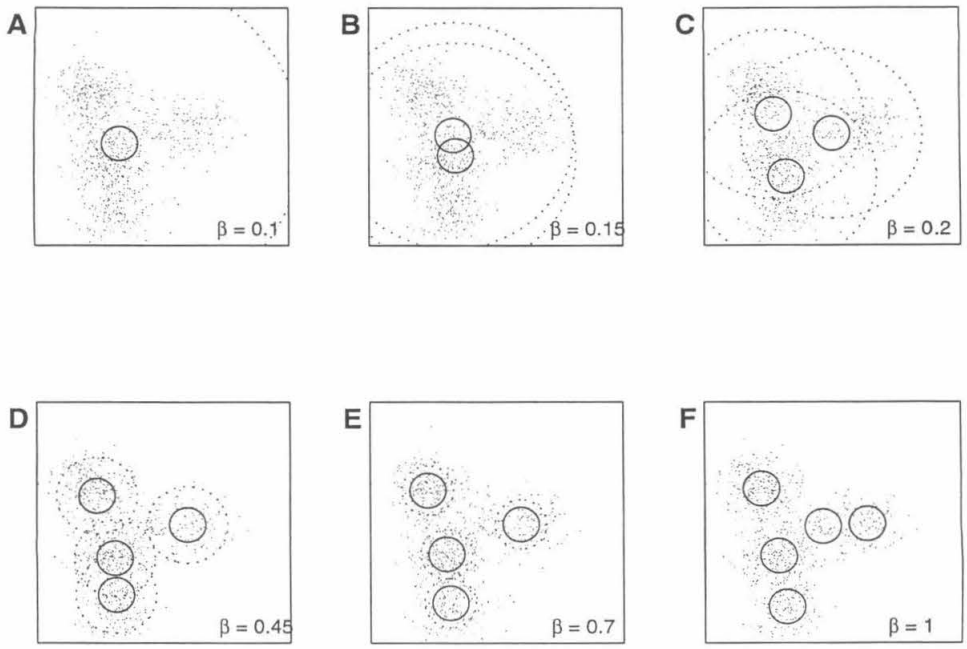


Figure 3.1: Phase transitions in REM-1 for fixed-variance Gaussians

data point are shared equally between all of the components, the maximizing μ_m are all identical. The solution in the $\beta = 0$ case, then, has all the components located at the overall mean of the data.

A remarkable fact is that even as the temperature decreases (that is, β increases) this solution remains the global maximum of the likelihood for some range of temperatures. Once the relaxation process reaches a critical temperature, the solution undergoes a **phase transition** and the former stationary point (where all the components are identical) ceases to be a maximum. A new maximum appears, usually dividing the components into two groups, so that all of the components assume one of only two distinct parameter values. As the system cools further, the optimal solution again continues with only two distinct component values, although the values of those components may change. Eventually, though, it undergoes another phase transition and more distinct components are observed.

Figure 3.1 shows an example of the optimal mixtures at various stages of relaxation. We fit two dimensional data, shown by the scattered points, by a mixture of five unit Gaussians. Each panel of the figure shows the mixture at a different temperature. The inner, solid, circle shows the 1σ boundary of the Gaussian; the outer, dashed, circle shows the effective variance ($\beta^{-1}I$) boundary. In the first few diagrams, fewer than five components are visible due to the exact coincidence of the means.

3.4.1 Critical temperatures

In the case of this simple model it is possible to calculate the critical temperatures at which the mixture will undergo a phase transition.

Suppose we were to start the EM algorithm with parameters θ^0 in which two (or more) of the components were identical. Without loss of generality we shall take these two be the first two components, setting $\mu_1^0 = \mu_2^0$ and $\pi_1^0 = \pi_2^0$. At each E-step the responsibilities of these two components for each of the data points will be the same. Thus, at the M-step they will both be updated in exactly the same way, and will remain identical. The EM algorithm will thus preserve the duplication, and will converge to a stationary point with $\mu_1^* = \mu_2^*$ and $\pi_1^* = \pi_2^*$.

Is this stationary point a maximum, or merely a saddle point? The stability of the solution θ^* can be evaluated by examining the value of the Hessian of the free-energy at that point. In fact, we know that for any parameter value, F_β is maximized with respect to the $r_{m,i}$ by the relaxation E-step. Thus, we need only evaluate the Hessian within the surface of constraint set by the equation (3.11). With the responsibilities chosen optimally, we can reduce the free-energy thus,

$$\begin{aligned}
\ell_\beta(\theta) &= F_\beta\left(\frac{\pi_m^\beta e^{-\frac{1}{2}\beta\|x_i-\mu_m\|^2}}{\sum_l \pi_l^\beta e^{-\frac{1}{2}\beta\|x_i-\mu_l\|^2}}, \theta\right) \\
&= \beta \sum_i \sum_m r_{m,i} \log\left(\pi_m e^{-\frac{1}{2}\|x_i-\mu_m\|^2}\right) - \sum_i \sum_m r_{m,i} \log r_{m,i} \\
&= \sum_i \sum_m r_{m,i} \log\left(\frac{\pi_m^\beta e^{-\frac{1}{2}\beta\|x_i-\mu_m\|^2}}{r_{m,i}}\right) \\
&= \sum_i \sum_m r_{m,i} \log \sum_l \pi_l^\beta e^{-\frac{1}{2}\beta\|x_i-\mu_l\|^2} \\
&= \sum_i \log \sum_l \pi_l^\beta e^{-\frac{1}{2}\beta\|x_i-\mu_l\|^2} \qquad (3.19)
\end{aligned}$$

where, in the last step we have used the fact that $\sum_m r_{m,i} = 1$. This form is quite similar to the log-likelihood of the underlying model. We refer to it as the **relaxation log-likelihood**. Precisely the same relationship exists between the relaxation free-energy and the relaxation log-likelihood as does between the true free-energy and log-likelihood.

Evaluation of the Hessian of $\ell_\beta(\theta)$ proves to be notationally challenging. Rose (1998) suggests an alternative which is more tractable and which we shall adopt. We consider a perturbation $\epsilon\delta_m$ applied to each of the means μ_m^* respectively, with $\delta_m = 0$ for all but the identical components. We then evaluate the derivative $\frac{d^2}{d\epsilon^2}\ell_\beta(\{\pi_m^*\}, \{\mu_m^* + \epsilon\delta_m\})$ at the point in question. This is equivalent to finding the projection of the Hessian on the direction defined by the perturbation δ_m.

We begin with the first derivative.

$$\frac{d}{d\epsilon}\sum_i \log \sum_l \pi_l^{*\beta} e^{-\frac{1}{2}\beta\|x_i-\mu_l^*-\epsilon\delta_l\|^2} = \sum_i \sum_l \frac{\pi_l^{*\beta} e^{-\frac{1}{2}\beta\|x_i-\mu_l^*-\epsilon\delta_l\|^2}}{\sum_k \pi_k^{*\beta} e^{-\frac{1}{2}\beta\|x_i-\mu_k^*-\epsilon\delta_k\|^2}} \beta\delta_l^T(x_i - \mu_l^* - \epsilon\delta_l)$$

$$= \sum_i \sum_l \beta r_{l,i} \delta_l^T (x_i - \mu_l^* - \epsilon\delta_l) \tag{3.20}$$

with the responsibilities evaluated at the perturbed θ. We note that when $\epsilon = 0$ we can write this derivative as $\beta \sum_l \delta_l^T \left(\sum_i r_{l,i}^* x_i - \mu_l^* \sum_i r_{l,i}^* \right)$ which is always zero by (3.17). This simply verifies that parameters which satisfy the recurrence relations (3.15)–(3.17) are indeed stationary points of the relaxation log-likelihood.

The second derivative is

$$\frac{d}{d\epsilon} \sum_i \sum_l \beta r_{l,i} \delta_l^T (x_i - \mu_l^* - \epsilon\delta_l) = \sum_i \sum_l \left(\beta \frac{dr_{l,i}}{d\epsilon} \delta_l^T (x_i - \mu_l^* - \epsilon\delta_l) - \beta r_{l,i} \|\delta_l\|^2 \right) \tag{3.21}$$

with the derivative of the responsibility given by

$$\begin{aligned}
\frac{dr_{l,i}}{d\epsilon} &= \frac{d}{d\epsilon} \left(\frac{\pi_l^{*\beta} e^{-\frac{1}{2}\beta \|x_i - \mu_l^* - \epsilon\delta_l\|^2}}{\sum_k \pi_k^{*\beta} e^{-\frac{1}{2}\beta \|x_i - \mu_k^* - \epsilon\delta_l\|^2}} \right) \\
&= \frac{\pi_l^{*\beta} e^{-\frac{1}{2}\beta \|x_i - \mu_l^* - \epsilon\delta_l\|^2} \beta \delta_l^T (x_i - \mu_l^* - \epsilon\delta_l)}{\sum_k \pi_k^{*\beta} e^{-\frac{1}{2}\beta \|x_i - \mu_k^* - \epsilon\delta_l\|^2}} - \\
&\quad \frac{\pi_l^{*\beta} e^{-\frac{1}{2}\beta \|x_i - \mu_l^* - \epsilon\delta_l\|^2} \sum_j \pi_j^{*\beta} e^{-\frac{1}{2}\beta \|x_i - \mu_j^* - \epsilon\delta_l\|^2} \beta \delta_j^T (x_i - \mu_j^* - \epsilon\delta_j)}{\left(\sum_k \pi_k^{*\beta} e^{-\frac{1}{2}\beta \|x_i - \mu_k^* - \epsilon\delta_l\|^2} \right)^2} \\
&= \beta r_{l,i} \left(\delta_l^T (x_i - \mu_l^* - \epsilon\delta_l) - \sum_j \beta r_{i,j} \delta_j^T (x_i - \mu_j^* - \epsilon\delta_j) \right)
\end{aligned} \tag{3.22}$$

Combining these equations we arrive at

$$\begin{aligned}
\frac{d^2}{d\epsilon^2} \ell_\beta (\theta^\epsilon) &= \beta^2 \sum_i \sum_l r_{l,i} \left(\delta_l^T (x_i - \mu_l^* - \epsilon\delta_l) \right)^2 - \beta^2 \sum_i \left(\sum_l r_{l,i} \delta_l^T (x_i - \mu_l^* - \epsilon\delta_l) \right)^2 \\
&\quad -\beta \sum_i \sum_l r_{l,i} \|\delta_l\|^2
\end{aligned}$$

and so, evaluating at $\epsilon = 0$ and exploiting the facts that $\delta_l = 0$ for $l > 2$ and that the means and responsibilities of components 1 and 2 are identical by construction.

$$\begin{aligned}
\frac{d^2}{d\epsilon^2} \ell_\beta (\theta^*) &= \beta \sum_l \delta_l^T \left(\beta \sum_i r_{l,i}^* (x_i - \mu_l^*)(x_i - \mu_l^*)^T - \sum_i r_{l,i}^* \right) \delta_l \\
&\quad -\beta^2 \sum_i \left(r_{1,i}^* (x_i - \mu_1^*)^T \sum_l \delta_l \right)^2
\end{aligned} \tag{3.23}$$

The second term in this expression, a sum of squares, is always non-negative. We can force it to 0 by choosing the perturbations so that $\sum_l \delta_l = 0$. The first part will be negative for all choices of δ as long as the matrix $\beta \sum_i r_{l,i}^* (x_i - \mu_l^*)(x_i - \mu_l^*)^T - \sum_i r_{l,i}^*$ is negative definite. Let $\sigma_{l,s}$ be the sth

eigenvalue of the matrix $\sum_i r^*_{l,i}(x_i - \mu^*_l)(x_i - \mu^*_l)^T / \sum_i r^*_{l,i}$. The condition for negative definiteness is thus

$$\beta < \frac{1}{\max(\sigma_{l,s})}; l = \{1, 2\} \tag{3.24}$$

This condition is both necessary and sufficient for the solution θ^* with components 1 and 2 identical to be a stable maximum. We have shown that if it holds then the derivative of (3.23) is negative for any choice of δ_m. If it fails we can choose δ_1 and δ_2 pointing in opposite directions along the eigenvector corresponding to the largest $\sigma_{l,s}$ so as to obtain a positive Hessian.

Thus, a critical temperature is reached whenever the temperature β^{-1} becomes smaller than the leading eigenvalue of the covariance of the data assigned to any of the mixture's components. If we interpret the parameter β^{-1} as the effective scale of the covariance matrix of each Gaussian, this result is intuitively appealing. When the observed covariance of the data assigned to a component becomes larger than the component can "handle", a transition to more distinct component centers occurs.

3.4.2 Model-size

It is tempting to interpret the phase transition structure of relaxation models as indicating a progressive change in the underlying model-size (for example, the number of components in a mixture). Take the mixture model shown in figure 3.1, for example. Initially, only one distinct set of component parameters exists, and we might think of the mixture as containing only that one component. As the relaxation progresses, each phase transition introduces more distinct component values. We would like to view these as new components being added to the mixture, thus growing the underlying model-size.

Unfortunately, under the REM-1 algorithm (as well as the basic deterministic annealing algorithm), such an interpretation does not hold up. In the ground-state ($\beta = 1$) mixture likelihood, if two components, say the first two, have identical parameters, so that $\mathsf{P}_1(x_i) = \mathsf{P}_2(x_i)$, they may be replaced by a single component with the same parameters and mixing proportion $\pi_1 + \pi_2$ without any change in the likelihood. This is made clear by inspection of the likelihood

$$\ell_{\mathcal{X}}(\theta) = \sum_i \log \sum_m \pi_m \mathsf{P}_m(x_i) \tag{3.25}$$

In particular, if the larger model is at a maximum in the likelihood, then the smaller one will be too.

This convenient behaviour does not carry through to higher temperatures. Recall the form of the relaxation log likelihood

$$\ell_{\mathcal{X},\beta}(\theta) = \sum_i \log \sum_m \pi_m^\beta \mathsf{P}_m(x_i)^\beta \tag{3.26}$$

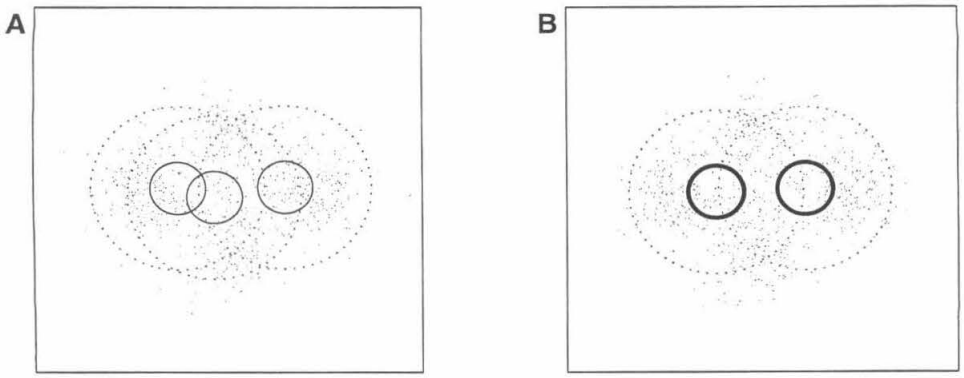


Figure 3.2: Inequivalence of different size models

Clearly, with $\beta < 1$ we cannot replace the identical components as before, since $\pi_1^\beta + \pi_2^\beta \neq (\pi_1 + \pi_2)^\beta$. Nor can we simply set the mixing proportion of the new component to $(\pi_1^\beta + \pi_2^\beta)^{1/\beta}$, since this violates the normalization of $\mathsf{P}_\theta(y_i)$. In general, then, the relaxation likelihood changes between the two models. Furthermore, a maximum in the more complex model may not correspond to a maximum in the simpler one, indeed the number of distinct component values in the two models may not be the same.

Figure 3.2 illustrates the point. Panel A shows a maximum in the relaxation likelihood of a three-component mixture of unit Gaussians at the stage $\beta = 0.3$. Panel B shows the optimal configuration, at the same temperature, of a four-component mixture, which was constructed by replacing the rightmost component of the mixture of panel A with two identical Gaussians. Both visible contours in B represent two identical components (indicated by the dark lines — other than this the representation of the components is as in figure 3.1). Thus, the duplication of one component has, in effect, driven the relaxation of the mixture in reverse, to a smaller phase.

Thus, the view of the model changing in size during the relaxation process cannot be maintained consistently under REM-1.

A further issue emerges from this analysis. Consider the mixture of figure 3.2B, where a four component mixture is being fit, but where only two distinct component values are visible. How do we know how to distribute these duplicated components? Clearly, each choice will yield a different intermediate solution; but the final result may also be affected since subsequent phase transitions will be constrained by the availability of components. We would like to be able to introduce the additional component wherever it is needed, but we cannot "move" the component around without changing the likelihood landscape. The result is that the choice of how to group the various components, a choice that must be made at each phase transition, will affect the outcome of the relaxation process.

Both of these issues can be rectified by the introduction of a variant of the basic relaxation

algorithm, which we call REM-2.

3.5 REM-2

It is instructive to examine the structure of the relaxation free-energy of REM-1 for clues to the origin of the inequivalence of different model-sizes described above. Recall that the term $Q(p,\theta)$ is the expected value of the joint data log-likelihood under the distribution p. Using the fact that $\ell_{\mathcal{X},\mathcal{Y}}(\theta) = \log\left(\mathsf{P}_\theta\left(\mathcal{X} \mid \mathcal{Y}\right)\mathsf{P}_\theta\left(\mathcal{Y}\right)\right)$ we can write the free-energy of (3.6) as

$$F_\beta(p,\theta) = \beta\mathcal{E}_p\left[\log \mathsf{P}_\theta\left(\mathcal{X} \mid \mathcal{Y}\right)\right] + \beta\mathcal{E}_p\left[\log \mathsf{P}_\theta\left(\mathcal{Y}\right)\right] - \mathcal{E}_p\left[\log p\right] \tag{3.27}$$

If we introduce a new hidden state, we increase the entropy of the latent variables. However, provided the new state is identical to some old one, the cross-entropy $-\mathcal{E}_p\left[\log \mathsf{P}_\theta\left(\mathcal{Y}\right)\right]$ decreases by the same amount. When $\beta = 1$, then, such an addition has no net effect on the free-energy. However, at higher temperatures the free-energy increases with the introduction of the new state. The size of this increase depends on both p and θ and so the location of the maxima of the free-energy may also change, as we saw above.

This formulation suggests a resolution of the difficulty. We introduce a slightly different relaxation free-energy which will form the basis of our second Relaxation Expectation–Maximization algorithm (REM-2).

$$\begin{aligned} F'_\beta(p,\theta) &= \beta\mathcal{E}_p\left[\log \mathsf{P}_\theta\left(\mathcal{X} \mid \mathcal{Y}\right)\right] + \mathcal{E}_p\left[\log \mathsf{P}_\theta\left(\mathcal{Y}\right)\right] - \mathcal{E}_p\left[\log p\right] \\ &= \beta Q'(p,\theta) - \mathsf{KL}[p(\mathcal{Y})||\mathsf{P}_\theta\left(\mathcal{Y}\right)] \end{aligned} \tag{3.28}$$

Here $\mathsf{KL}[f||g]$ stands for the Kullback-Leibler divergence between the distributions f and g. This form no longer enjoys the analogy with the familiar free-energy of statistical physics. Nonetheless, from the point of view of optimization it provides just as valid a relaxation progression as does the more traditional form.

Again, we optimize each free-energy in the relaxation sequence using the EM approach of alternate optimizations with respect to p and with respect to θ. The E-step is derived in the same manner as before. We introduce a Lagrange multiplier λ enforcing the constraint $\int d\mathcal{Y}\, p(\mathcal{Y}) = 1$ to obtain

$$\begin{aligned} 0 &= \frac{\partial}{\partial p}\left(F'_\beta(p,\theta) - \lambda\int d\mathcal{Y}\, p(\mathcal{Y})\right) \\ &= \frac{\partial}{\partial p}\left(\int d\mathcal{Y}\, p(\mathcal{Y})(\beta\log \mathsf{P}_\theta\left(\mathcal{X} \mid \mathcal{Y}\right) + \log \mathsf{P}_\theta\left(\mathcal{Y}\right) - \log p(\mathcal{Y}) - \lambda)\right) \end{aligned} \tag{3.29}$$

from which, by the calculus of variations,

$$\begin{aligned} 0 &= \frac{\partial}{\partial p}\left(p(\mathcal{Y})(\beta \log \mathsf{P}_\theta(\mathcal{X} \mid \mathcal{Y}) + \log \mathsf{P}_\theta(\mathcal{Y}) - \log p(\mathcal{Y}) - \lambda)\right) \\ &= (\beta \log \mathsf{P}_\theta(\mathcal{X} \mid \mathcal{Y}) + \log \mathsf{P}_\theta(\mathcal{Y}) - \log p^*(\mathcal{Y}) - \lambda) - \frac{p^*(\mathcal{Y})}{p^*(\mathcal{Y})} \end{aligned} \tag{3.30}$$

and so

$$p^*(\mathcal{Y}) \propto \mathsf{P}_\theta(\mathcal{Y})\left(\mathsf{P}_\theta(\mathcal{X} \mid \mathcal{Y})\right)^\beta \tag{3.31}$$

The multiplier λ ensures that p is correctly normalized.

At first glance it might seem that the M-step, involving the maximization of $\beta\mathcal{E}_p[\log \mathsf{P}_\theta(\mathcal{X} \mid \mathcal{Y})] + \mathcal{E}_p[\log \mathsf{P}_\theta(\mathcal{Y})]$ will be different from standard EM and REM-1. In most models, however, the parameters θ can be partitioned into two disjoint and independent sets, one responsible for the distribution of the latent variables and the other for the conditional of the observables given the latent variables. If this is the case, F'_β can be optimized with respect to each of these sets separately, and clearly the resulting update rules will be exactly as in standard EM.

Now, when $\beta = 0$, this free-energy is optimized by any choice of p and θ for which $p(\mathcal{Y}) = \mathsf{P}_\theta(\mathcal{Y})$. Although p need not be the maximum entropy distribution, the resulting parameter values are very similar to the initial conditions for REM-1. In particular, the distribution p must be independent of the observations $\mathcal{X}$. For the mixture model, for example, we have $r_{m,i} = \pi_m$, which implies that each component is fit with equal weight given to all of the data (although that weight may be different for the different components) and so all the component parameters are identical. For consistency with REM-1, and in the spirit of maximum entropy statistical methods where unknown distributions are assumed to be maximally uncertain, we will adopt the convention that the initial choice of parameters governing $\mathsf{P}_\theta(\mathcal{Y})$ does indeed maximize the entropy of the latent variables under the constraints of the model. This is merely a convention, though. Any initial choice of $\mathsf{P}_\theta(\mathcal{Y})$, provided every possible outcome has non-zero probability, will produce the same results.

In figure 3.3 the REM-2 algorithm is used to fit a 5-component mixture to the same data as was used in figure 3.1. This figure illustrates the fact that REM-2 exhibits the same type of phase transition structure as we saw previously in REM-1. Indeed, we can follow through the analysis of section 3.4.1 and find that exactly the same condition for stability holds, except that now the responsibilities that appear in (3.23) are those of the new algorithm

$$r_{m,i} = \frac{\pi_m e^{-\frac{1}{2}\beta\|x_i - \mu_m\|^2}}{\sum_l \pi_l e^{-\frac{1}{2}\beta\|x_i - \mu_l\|^2}} \tag{3.32}$$

(note that the mixing probabilities π_m are not raised to the power β). This results is a small change in the actual values of the critical temperatures between the two algorithms on the same data set;

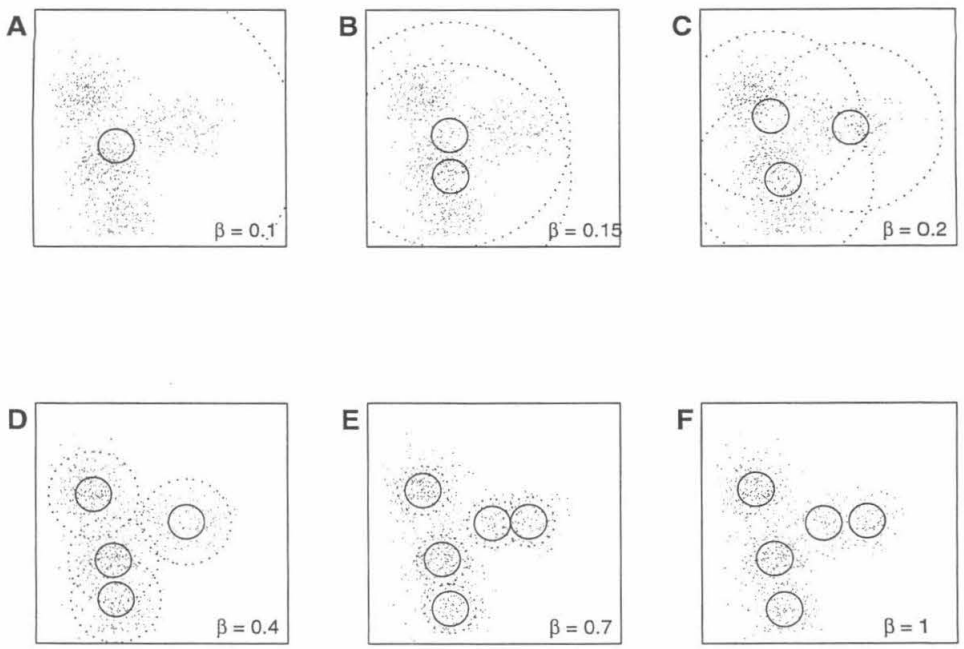


Figure 3.3: Phase transitions in REM-2 for fixed-variance Gaussians

an example of this is evident in a comparison of figures 3.3 and 3.1.

We can verify that the issues raised in section 3.4.2 are resolved by REM-2 by consideration of the implied relaxation likelihood for a mixture model.

$$\begin{aligned}
\ell_\beta(\theta) &= F'_\beta(\{r_{i,m}\},\theta) \\
&= \beta \sum_i \sum_m r_{m,i} \log \mathsf{P}_m(x_i) + \sum_i \sum_m r_{m,i} \log \pi_m - \sum_i \sum_m r_{m,i} \log r_{m,i} \\
&= \sum_i \sum_m r_{m,i} \log \frac{\pi_m \mathsf{P}_m(x_i)^\beta}{r_{m,i}} \\
&= \sum_i \sum_m r_{m,i} \log \sum_l \pi_l \mathsf{P}_m(x_i)^\beta \\
&= \sum_i \log \sum_l \pi_l \mathsf{P}_m(x_i)^\beta
\end{aligned} \tag{3.33}$$

Clearly, the two identical components can be replaced by one (with mixing probability given by the sum of the weights of the duplicate components) without disturbing the likelihood. Thus, we can legitimately regard the model-size as increasing during the relaxation process. Furthermore, we need not make any choice about how to group components: any grouping will yield the same sequence of likelihoods and extra components can be assigned as needed when a critical temperature is reached.

3.6 Cascading Model Selection

In our development to this point, we have tacitly assumed that the size of the eventual model is known. If we use REM-1, the model size is set at the outset and maintained throughout. If we use REM-2, the model-size grows during the relaxation, but is capped at the correct value. In practice, however, this knowledge is often not available *a priori*. In using a mixture model for clustering, for example, we may not know in advance the appropriate number of clusters. Instead, the model-size needs to be learnt along with the parameters of the appropriate model.

This is an example of the more general problem of model selection. We have already visited this problem twice in the course of this dissertation. Section 1.3 discussed the general theory and described a number of likelihood-penalty techniques that are used in practice, as well as related approaches such as cross-validation. Section 2.7.3 added a further technique, called the Cheeseman-Stutz criterion, which is suitable for latent variable models such as mixtures. In this section we will investigate the relationship between these techniques and REM.

3.6.1 A natural answer?

It is tempting to think that in certain situations, the phase transition structure of REM provides a natural answer to such problems, and, indeed, a number of authors have assumed this (see, for example, Rose (1998) or Weiss (1998)). Take the mixture of unit Gaussians that has been our running example in this chapter. Suppose we were to fit by relaxation a mixture with a very large number of components. Once the relaxation had run its course, we would find that only a small number of distinct component values existed in the final mixture. Furthermore, whether we had used REM-1 or REM-2 to find that mixture, it would always be the case that at unit temperature the equivalence between a mixture with duplicate components and a smaller one with all duplications removed would hold. Thus, we can safely assert that the relaxation procedure has found a solution with limited model-size. Is this the correct model-size?

Unfortunately, despite the suggestions to that effect that appear in the literature, it is not. This should be clear from the fact that ultimately, the technique by which the final mixture was found is not important. That mixture is simply a maximum — with luck, the global maximum — of the model likelihood. Choosing a number of components in the manner suggested is thus the same as choosing between different models solely on the basis of their unpenalized likelihoods. Such a choice is prone to over-fit for all of the reasons that were discussed in section 1.3. The estimate of the model-size will be biased upwards.

We can drive the point home by means of a simple example. Suppose that the data to be modeled have actually arisen from a single Gaussian distribution with zero mean and unit covariance matrix. We attempt to model this data with a mixture of Gaussians, each with unit covariance, fitting

the mixture by REM. As we have seen, at low values of the relaxation parameter, β, all of the mixture components coincide. However, once β reaches the inverse of the leading eigenvalue of the observed covariance matrix, more than one distinct mean will be observed. The eigenvalues of the observed covariance are asymptotically symmetrically distributed about 1 (the exact density is given by Anderson 1963). Thus, with a probability of approximately $1-2^{-p}$, where p is the dimensionality of the Gaussian, the leading eigenvalue will be greater than 1. In this case, the phase transition will occur with $\beta < 1$. If relaxation were to proceed to completion at $\beta = 1$, we would arrive at a solution with more than one component.

The situation is even more dire for other latent variable models. For example, if the covariances of the Gaussians are unknown (and perhaps unequal) the maximum likelihood solution given a sufficiently large number of components has each component concentrated around exactly one data point, giving rise to as many distinct components as data. Clearly, this is not a reasonable solution.

Another suggestion is as follows. The relaxation procedure is carried out using a large number of components, just as before. Now, however, a section of the data — a validation set — is held out and the (relaxation) likelihood of the optimal model at each temperature is evaluated on these data. After relaxation is complete, we select the model at which the validation likelihood was greatest.

This scheme is only meaningful in situations where the relaxation likelihood corresponds to an actual model. Even in such situations, though, it will tend to return the wrong answer; in this case the bias appears in the parameter estimates. Take the simple example of data from a single Gaussian. It is plausible that this scheme would correctly identify the optimal model-size as containing only one component. However, selecting this component will require choosing a solution at a non-unit temperature. Thus, the Gaussian will have a larger variance than appropriate.

The resolution would appear to be to use a model selection scheme (validation in this example) to choose the model-size, but then continue to relax the model of this size to unit temperature. We shall discuss a local version of this scheme in the next section.

3.6.2 Cascading model selection

Careful consideration of the nature of the relaxation likelihood has indicated that, despite the appealing natural limits that appear in the fixed-variance models commonly used in conjunction with deterministic annealing, to avoid bias the model-size must be chosen by a more traditional model selection technique. Nonetheless, the hierarchical "division" due to the phase transition structure that we saw in the case of the mixture model does still form an attractive basis for model selection. We shall see that it is indeed possible to exploit this structure. Through a progressive development we will arrive at an efficient method for choosing the correct model size, within the relaxation framework, that we call **cascading model selection**.

In what follows we shall consider the mixture model, with the selection of model-size being

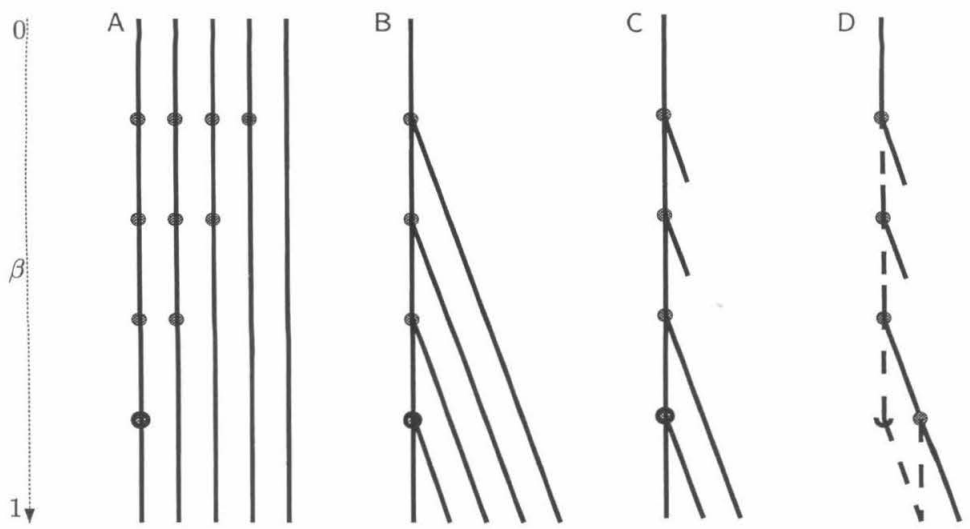


Figure 3.4: Schematic of model selection using REM

equivalent to choosing the correct number of components. The method is, however, quite general and can be applied with ease to any latent variable model for which an EM algorithm can be written.

The standard approach to model selection is as follows. Using some algorithm, which might just as well be REM, we obtain maximum likelihood fits for a variety of models with differing numbers of components. These models are then compared using one the methods discussed in sections 1.3 or 2.7.3. Many of these methods involve a comparison of the maximal log-likelihood values of the different models, reduced by a term that reflects the number of free parameters in the model. It is such penalized-likelihood methods that we shall consider first.

The various model selection schemes that we will discuss are shown schematically in figure 3.4. Panel A represents the basic procedure. The solid lines each represent the relaxation of a model, while the circles indicate the occurrence of phase transitions. The five models being fit are of different sizes, which is why they undergo different numbers of phase transitions. Roughly speaking, the total length of the lines in each panel represents the computational cost associated with each model selection strategy. The remaining panels will be described below.

If the optimization is carried out using REM-2 then the process of fitting the different size models can be made considerably more efficient. The relaxation process for models with M and with $M+1$ components is identical until the final phase transition of the larger model. Thus, there is no need to repeat the fitting process up to that point. As a result, we fit all of the models in a linear tree structure, shown in figure 3.4B, with a new branch emerging at each phase transition. (The schematic adopts the convention that the line emerging on the right of the circle has not undergone the phase transition, while the one that continues below has.) We note that this process is not

possible with either the conventional deterministic annealing algorithm or REM-1.

We can improve further on this scheme by allowing early pruning of some branches. This is facilitated by the following important result, which holds for models being fit by REM-2. Suppose we have an M component model in which one component is unstable in the sense of section 3.4.1, that is, if additional components are available it would undergo a phase transition. We compare the likelihoods of two models: $\mathcal{M}_1$ has only M components and therefore exhibits no phase transition, while $\mathcal{M}_2$ has a model-size of $M+1$ and thus has allowed the unstable component to "split". If the relaxation log-likelihood at some $\beta < 1$ of $\mathcal{M}_2$ exceeds that of $\mathcal{M}_1$ by Δ, then the final log-likelihood of $\mathcal{M}_2$ will exceed that of the smaller model by an amount larger than Δ. We offer an informal proof of this point.

Recall first that $\mathcal{M}_1$ is identical in likelihood to an $(M+1)$-component model $\mathcal{M}_{1*}$ in which the unstable component is duplicated, but both copies retain the same parameters. By assumption the relaxation log-likelihood of $\mathcal{M}_2$ exceeds that of $\mathcal{M}_{1*}$. Recall that this log-likelihood is obtained from the free-energy

$$F'_\beta(p,\theta) = \beta Q'(p,\theta) - \mathsf{KL}[p(\mathcal{Y}) || \mathsf{P}_\theta(\mathcal{Y})] \tag{3.28}$$

by setting $p(\mathcal{Y}) = \mathsf{P}_\theta(\mathcal{Y} \mid \mathcal{X})$. Now it must be the case that the Kullback-Leibler term for $\mathcal{M}_2$ is greater than that for $\mathcal{M}_{1*}$. If that were not true, the more complex model would be preferred even at $\beta = 0$, which we know not to be the case. Thus, it must also be true that the Q' term in the likelihood of $\mathcal{M}_2$ exceeds that of $\mathcal{M}_{1*}$ (and thus of $\mathcal{M}_1$).

How will the log-likelihoods of the two models change as relaxation progresses? Let $\ell_\beta(\theta^*)$ be the optimal relaxation log-likelihood, that is, the value of $F'_\beta(p,\theta)$ with $\theta = \theta^*$, the optimal parameters, and $p(\mathcal{Y}) = \mathsf{P}_{\theta^*}(\mathcal{Y} \mid \mathcal{X})$. The maximizing value of the model parameter vector, θ^*, is, of course, a function of the relaxation parameter β. Thus, we may differentiate the maximal log-likelihood with respect to β using the chain rule

$$\frac{d}{d\beta}\ell_\beta(\theta^*) = \frac{\partial}{\partial\beta}\ell_\beta(\theta^*) + \frac{\partial}{\partial\theta}\ell_\beta(\theta^*)\frac{d\theta^*}{d\beta} \tag{3.34}$$

But, since θ^* maximizes the log-likelihood, the gradient of $\ell_\beta(\theta)$ at θ^* for fixed β is 0. The partial with respect to β is obtained trivially from (3.28), and thus we find that

$$\frac{d}{d\beta}\ell_\beta(\theta^*) = Q'(\mathsf{P}_{\theta^*}(\mathcal{Y} \mid \mathcal{X}), \theta^*) \tag{3.35}$$

We have argued that the Q' term for $\mathcal{M}_2$ is greater than that for $\mathcal{M}_1$. Thus, we find that the optimal log-likelihood of the larger model is growing more rapidly than that of the smaller one (if both gradients are negative, then it is shrinking less rapidly). As a result, any difference in likelihoods at $\beta < 1$ can only grow as β increases.

Thus, it is possible to further streamline the model selection process. If, at any stage in the relaxation, the penalized relaxation log-likelihood of some model is exceeded by that of a larger model (that is, the difference in log-likelihoods is greater than the difference in penalties) we can immediately neglect the smaller model, effectively pruning that branch of the tree. This is indicated in figure 3.4C, where the first two models are pruned.

Finally, we arrive at the approach that we call **cascading model selection**. We assume that the penalized likelihood rises monotonically with model-size until the optimal value is reached. While this is not guaranteed to be the case, it is an intuitively appealing assumption and the experiments below suggest that, at least for simple mixture models, it is typically valid. Under these conditions, we need not even consider a model of size $M + 2$ until the model with M components has been rejected in favour of one with $M + 1$.

In our implementation of cascading model selection we think of a particular model size as being "current" at all times. This is indicated by the solid line in figure 3.4D. When a critical temperature is reached, the current model retains its size. However, we begin to track the optimum of a "shadow" model of larger size (and thus, which undergoes the phase transition). If the penalized likelihood of this shadow model exceeds that of the current one, we abandon the current model and make the shadow current. Sometimes, it will be the case that the shadow model reaches a critical temperature without having replaced the current model. If this happens, we simple maintain the shadow model's size and continue to relax; we do not introduce the larger model.

It might also be the case that the current model will encounter another critical temperature, even though it remains more likely than the shadow. In this case we need to introduce another shadow model, usually of the same model-size as the previous one, but resulting from a different phase transition. In the case of the mixture model, it is useful to think of a different component having "split". If, as relaxation progresses, we reach a point where either of these shadow models becomes more likely than the current one, we make that model current and abandon all the others.

The cascading model selection procedure is capable of find optima that the basic REM algorithm is not. To see why, consider the case described above where a second shadow model may be introduced. This shadow model is different from any that might be obtained by REM; to achieve it we have "disallowed" one phase transition but allowed another. If this model proves to have greater likelihood than the first shadow, and also to be preferred to the current model according to the penalized likelihoods, then we will arrive at a model with greater likelihood than that obtained by REM with the same number of components. Intuitively, the cascading model selection prevented us from "wasting" a component due to the phase transition at the higher temperature, instead reserving it for the more advantageous split. This point will be illustrated below.

Finally, we note that the core result of cascading model selection has been obtained only for a penalized likelihood style model selection procedure. However, to the extent that such methods

approximate techniques such as Bayesian model selection or cross-validation, we might believe that such techniques can be used in the same way. In particular, for mixture models the Cheeseman-Stutz criterion of section 2.7.3 often provides good results.

3.7 Experiments

As we first encountered the REM algorithm in section 3.3, we noted that, because the maximum of the free-energy does not, in fact, vary continuously with the relaxation parameter, the algorithm process cannot be guaranteed to find the global optimum of the likelihood. Instead, we appealed to an intuitively founded expectation that it would tend to find a good optimum. In this section we examine the results of numerical experiments to see if this is actually the case.

The experiments described here all involve the simple mixture of two-dimensional unit Gaussians model, which we have seen throughout this chapter. In all cases the relaxation is performed using the REM-2 algorithm. The basic outline of the experiments is as follows: we select a random mixture of unit Gaussians, generate data from it, and fit mixture models to these data using both the REM-2 and standard EM algorithms. We then compare the performance of the algorithms by computing the likelihoods of the resultant models. Any solution in which the likelihood of the fit model is greater than the likelihood of the true (that is, data-generating) model will be called "good."

The parameters of the generating mixture are all chosen randomly within pre-specified intervals. The number of components, M, is chosen from the discrete uniform distribution on the values 3, 4, 5 and 6. The mixing proportions are chosen by randomly partitioning the interval $(0, 1)$ as follows: $M - 1$ numbers in the interval $(0, 1)$ are chosen from a uniform distribution on the interval and then ordered, thereby inducing a partition into M subintervals; the lengths of these subintervals are taken to be the mixing probabilities. The means are generated from the two-dimensional uniform distribution on the rectangular region bounded by ± 5 in both dimensions. The covariances are all set to the identity matrix.

500 data points are generated randomly from this mixture distribution. Mixtures of the correct number of Gaussians are then fit both by REM-2 and by standard EM. For each data set, the standard EM algorithm is started 10 times, from 10 randomly selected initial conditions (see section 2.7.2). Both algorithms are iterated to the same convergence criterion, which is that the relative change in likelihood after a complete EM step should fall below 10^{-7}. The likelihoods of all of the models, including the generating one, are then evaluated. We call a fit model "poor" if its likelihood is less than that of the generating model on the given data.

This entire procedure is repeated for 200 different generating mixtures.

Figure 3.5 shows the number of "poor" optima achieved under the different algorithms. The 10 bars on the left show how the rate of success of the standard EM algorithm increases as a

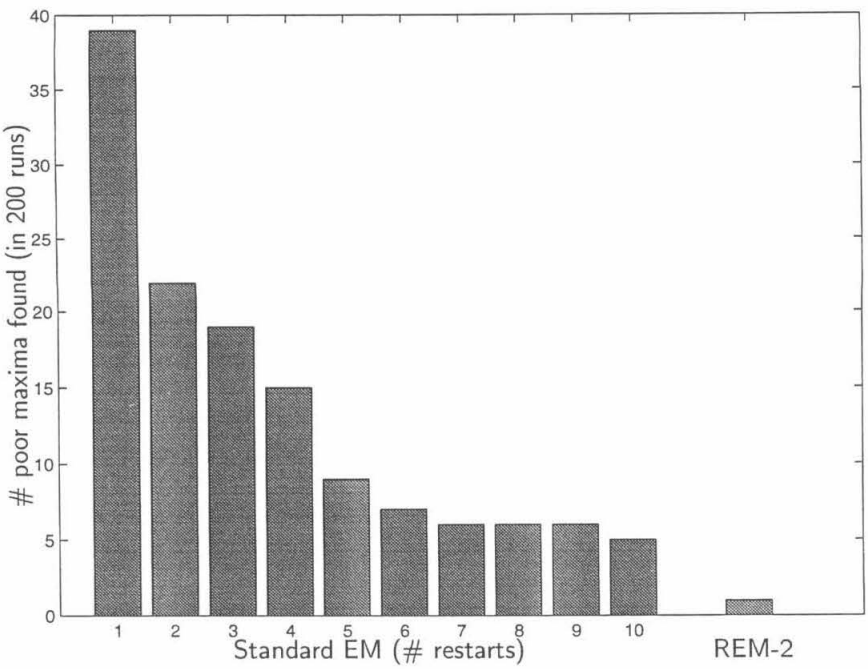


Figure 3.5: Frequency of poor maxima

progressively larger number of restarts are used. The likelihood used in the calculation of the bar labeled n is the largest of the likelihoods obtained from the first n restarts. The single bar on the right indicates that, for REM-2, only a single run achieved a poor optimum.

It is instructive to examine the single example in which REM-2 converged to a poor maximum. This is shown in figure 3.6. Panel A shows the model from which the data were generated. Panel B shows the optimum found by the REM-2 algorithm. Evidently, a phase transition that split the component in the middle-right was encountered before the phase transition that would correctly split the bottom-left component. In panel C we show the results of running REM-2 in conjunction with cascading model selection (using the BIC likelihood-penalty with no corrective constant). Whereas the standard REM-2 algorithm ran on a model with the correct number of components provided *a priori*, with cascading model selection this number could be determined from the data. Furthermore, it is evident that by incorporating on-line model selection, the early phase transition was rejected on the basis of the penalized likelihood , whereas the later, correct, one was subsequently accepted. It should be clear that without the cascading property this maximum could not have been found: had the different model sizes been compared after optimization (as is usual) then the model of size 5 would have been that of panel B. Thus, we observe that — as was suggested at the end of section 3.6 — besides the obvious benefits of automatic model size determination, the cascading model selection process can sometimes improve the optima found by REM.

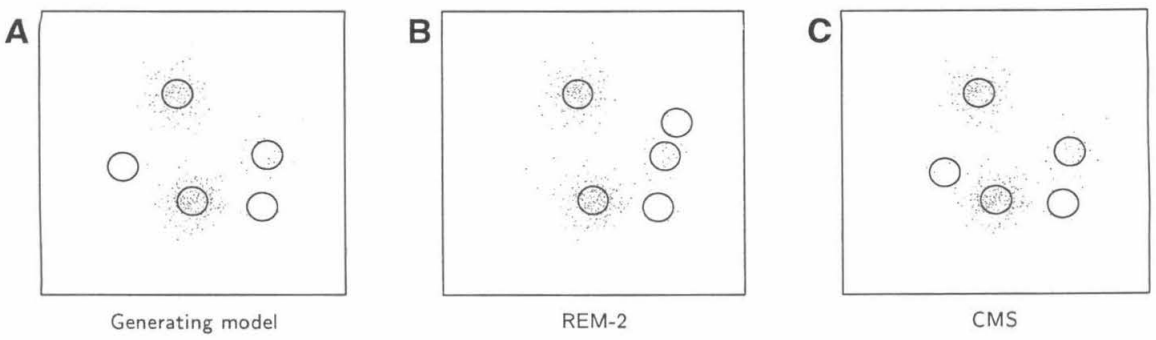


Figure 3.6: Cascading model selection can improve optima

A word of caution needs to appear here. The data shown in figure 3.5 suggest that, at least in this simple case, REM — perhaps in combination with cascading model selection — might well converge reliably to the global maximum of the likelihood. This is not actually the case. Closer inspection reveals that for 11 of the random mixtures at least one of the standard EM runs found a model with a likelihood more than 10^{-4} log-units larger than that found by REM-2. Furthermore, it is possible that even for the remaining mixtures the relaxation solution is not globally optimal, but that none of the standard EM iterations found the maximum either. Thus, REM does not always find the global optimum; indeed we cannot expect any algorithm of polynomial complexity to reliably do so. Nonetheless, figure 3.5 does suggest that it tends to find an optimum at least as good as the model that actually generated the given data with remarkable regularity.

Chapter 4 Sparse Hidden Markov Models

The **hidden Markov model** (HMM) is one of the most successful and widely used generative models in the field of statistical modeling. The statistical theory of HMMs has been driven in large part by the field of speech processing and is extremely well worked-out. Indeed, the Baum-Welch algorithm of the sixties is one of the earlier examples of an implementation of an EM algorithm, and much of the theory of EM was well understood in this context well before the publication of the general formulation. Nevertheless, advances in the theory of HMMs are still made. Recent examples include the factorial hidden Markov model Ghahramani and Jordan (1997).

In this chapter we review the generative model underlying the HMM, and discuss the applicable EM learning algorithm. We then examine a particular sub-class of the general model, the sparse HMM, in which the majority of outputs are zeros (or null). We then consider a "mixture" of these restricted models. This mixture-like compound model is a special case of the factorial HMM: we construct an EM algorithm with an imperfect E-step, of the form that was justified in section 1.8. This approach, though not exact, will come close to the true the maximum likelihood solution for certain classes of data.

4.1 The Generative Model

4.1.1 The Markov chain

The finite **Markov chain** (or Markov process) has been extensively studied in stochastic process theory. It consists of a series of N identically distributed discrete variables $\{y_i\}$, with the property that each is dependent only on the value of the preceding one. More precisely, the joint distribution over the variables factors as follows.

$$\mathsf{P}\left(y_1, y_2 \ldots\right) = \mathsf{P}\left(y_1\right) \prod_{i=2}^{N} \mathsf{P}\left(y_i \mid y_{i-1}\right) \tag{4.1}$$

As a result, y_i is conditionally independent of all of the variables $y_1 \ldots y_{i-2}$ given y_{i-1}.

The different values that the variables may take on are called the **states** of the process; in the models we discuss there is a finite number of such values and we take them to be the numbers $1 \ldots P$. The "state" terminology suggests a connection between a Markov process and a non-determinstic finite-state automaton. In fact, the sequence of states traversed by such an automaton in the absence of input (or given constant input) indeed forms a Markov sequence. We shall use the two sets of

terminology interchangeably, as is common in the field, referring, for instance, to the model as being in state p at step i when y_i takes the value p.

The joint distribution (4.1) is completely specified by the two discrete probability distributions, the **initial state probabilities** $\mathsf{P}(y_1)$ and the **state transition probabilities** $\mathsf{P}(y_i \mid y_{i-1})$ for $i > 1$. We can collect each of the transition probabilities into a $P \times P$ **transition matrix** T_+, so that $T_{+pq} = \mathsf{P}(y_i = p \mid y_{i-1} = q)$. The initial probabilities might be collected into a seperate vector T_0, however, in most cases it is more convenient to roll them into the transition matrix as follows. We introduce a new "random" variable y_0 which precedes (in the sense of the Markov conditioning criterion) the first actual random variable y_1. This variable assumes the value 0, which is not a possible outcome for any other variable, with probability one. In this model, the transition matrix is augmented to a $(P+1) \times (P+1)$ matrix T, with the first column containing the initial state probabilities; the first row being entirely zero to indicate that the system never makes a transition back into the state 0; and the remaining elements being the transition probabilities. For obvious reasons it will be convenient to number the rows and columns of T from 0, rather than 1. Once normalization requirements are accounted for, the augmented transition matrix T contains $P^2 - 1$ free parameters; $P - 1$ specify the initial probabilities and $P(P-1)$ specify the transition probabilities.

Using this notation, manipulations of the probability functions becomes quite straightforward. For example, if the marginal distribution of the variable y_{i-1} is given by the vector π_{i-1}, then the marginal distribution of y_i is given by $\mathsf{P}(y_i = p) = \sum_q \mathsf{P}(y_i = p \mid y_{i-1} = q)\,\mathsf{P}(y_{i-1} = q)$, which can be written more succintly as $\pi_i = T\pi_{i-1}$. As a result, the marginal distribution of the ith variable is

$$\pi_i = T^i \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \tag{4.2}$$

Given some basic regularity conditions on the transition matrix T, there exists a unique probability distribution over the states, represented by the vector π, which satisfies the condition

$$\mathsf{P}_T(y_i) = \pi \quad \Rightarrow \quad \mathsf{P}_T(y_{i+1}) = \pi \tag{4.3}$$

For obvious reasons, this is called the stationary distribution of the Markov process.

Clearly, π is a right eigenvector of the matrix T with eigenvalue 1. It can be shown, under some additional mild conditions on T (related to the ergodicity of the Markov process), that all other eigenvalues have absolute values strictly smaller than 1 (Seneta 1981; Karlin 1991). As a result, given any initial distribution on the states, after a sufficient number of steps the marginal

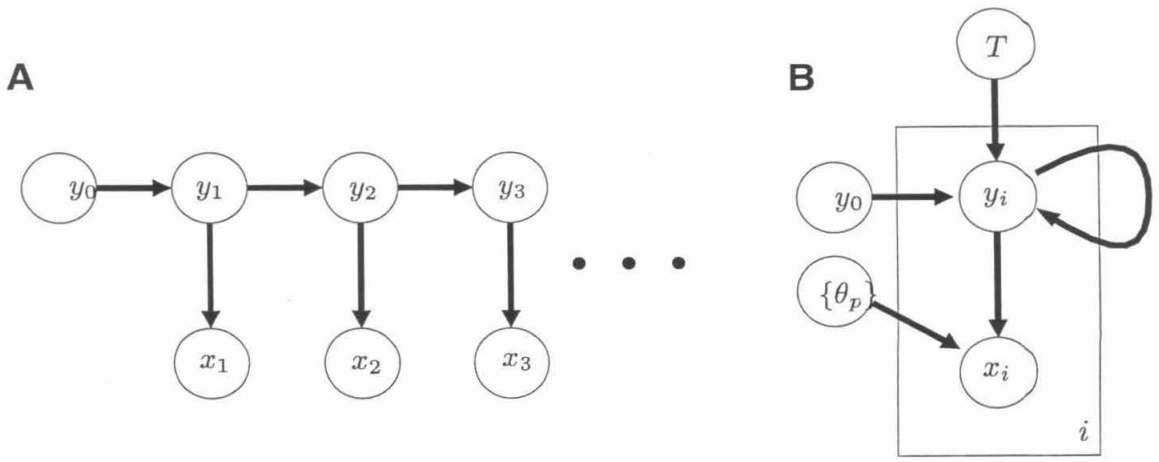


Figure 4.1: The hidden Markov model

distribution of the y_i will approach π. The stationary distribution is thus an attractor in the space of marginal distributions on the Markov variables. The magnitude of the largest non-unit eigenvalue sets the rate of decay of the non-stationary components, and thus the number of steps we need to wait in the typical case before the marginal state-distribution approaches the stationary one. This is called the **mixing time** of the (ergodic) chain.

4.1.2 The hidden Markov model

The **hidden Markov model** is a latent variable generative model derived from the basic Markov model described above. The structure of the model is drawn in graphical terms in figure 4.1. Panel A represents all of the variables of the model explicitly. The variables y_i form a Markov chain, but in this case they are not directly observed. Instead, we see output variables x_i which depend only on the corresponding state y_i; that is, each x_i is conditionally independent of all other variables, both observed and latent, given y_i. We adopt the convention of a deterministic initial state y_0 to compress all of the Markov parameters into a single matrix. There is no corresponding observable x_0.

The conditional distribution $\mathsf{P}(x_i \mid y_i)$ is stationary with respect to the instance variable i. Thus, associated with each state p (except 0) is an unchanging **output distribution** which plays a similar rôle to the component distributions of the mixture model. We will write θ_p for the parameters of this distribution and $\mathsf{P}_p(x)$ for the distribution (or density) function, just as in the case of the mixture model. Indeed, the connection between the two is quite deep. In figure 4.1B the same HMM, along with explicit parameter nodes, is shown in the more compact plate representation. It is clear that the structure is extremely similar to that of the mixture model; the only difference is the dependence of the latent variable between different instances. (As an aside, the plate notation is not well suited

for such models, since it does not make clear the essential Markov nature of the latent variable process, which is that the arrow linking the y_i nodes stretches only to the next plate.)

The parameters of model are the Markov probabilities contained in the matrix T along with all of the parameters θ_p of the output distributions. The likelihood of the parameters, with observations $\mathcal{X} = \{x_i\}$, is found by summing over all possible strings of Markov states $y_1 \ldots y_N$

$$\mathcal{L}_{\mathcal{X}}\left(T, \{\theta_p\}\right) = \sum_{y_1 \cdots y_N} \prod_i T_{y_i, y_{i-1}} \mathsf{P}_{y_i}\left(x_i\right) \tag{4.4}$$

An alternative, recursive, form for the calculation of this likelihood will appear below.

4.2 Learning: The Baum-Welch Algorithm

The commonly used learning algorithm for HMMs was developed in the course of classified work by Eric Baum and Lawrence Welch in the sixties. This algorithm turns out to be the standard EM algorithm applied to the generative model; however, its development pre-dated the publication of the original EM paper (Dempster *et al.* 1977) by at least a decade. The application is considerably more involved than the examples we have handled thus far. In particular, the E-step, in which parts of the conditional $\mathsf{P}_\theta\left(\mathcal{Y} \mid \mathcal{X}\right)$ are calculated, is sufficiently elaborate to have claimed a name of its own; it is called the forward–backward algorithm. Once this is completed, the M-step is more straightforward. The complete approach is commonly known as the Baum-Welch algorithm.

The joint data likelihood, based on observations, $\mathcal{X} = \{x_i\}$ and latent variable values $\mathcal{Y} = \{y_i\}$ is

$$\mathcal{L}_{\mathcal{X},\mathcal{Y}}\left(T, \{\theta_p\}\right) = \prod_{i=1}^{N} T_{y_i, y_{i-1}} \mathsf{P}_{y_i}\left(x_i\right) \tag{4.5}$$

leading to the log-likelihood

$$\ell_{\mathcal{X},\mathcal{Y}}\left(T, \{\theta_p\}\right) = \sum_i \log T_{y_i, y_{i-1}} + \sum_i \log \mathsf{P}_{y_i}\left(x_i\right) \tag{4.6}$$

As in the case of the mixture model, we introduce latent indicator variables in place of the discrete latent variables y_i. We define $z_{p,i}$ to take the value 1 if $y_i = p$ and 0 otherwise. We can then rewrite the log-likelihood as follows

$$\ell_{\mathcal{X},\mathcal{Z}}\left(T, \{\theta_p\}\right) = \sum_i \sum_{p,q} z_{p,i} z_{q,i-1} \log T_{pq} + \sum_i \sum_p z_{p,i} \log \mathsf{P}_p\left(x_i\right) \tag{4.7}$$

In the E-step for the nth iterarion, we take the expected value of this likelihood with respect to the conditional distribution determined by the parameter values on the $(n-1)$th step, $\mathsf{P}_{\theta^{n-1}}\left(\mathcal{Z} \mid \mathcal{X}\right)$.

This gives us

$$\begin{aligned} Q^n(T, \{\theta_p\}) &= \mathcal{E}_{\mathcal{Z}|\mathcal{X},\theta^{n-1}} \left[\ell_{\mathcal{X},\mathcal{Z}} \left(T, \{\theta_p\}\right)\right] \\ &= \sum_i \sum_{p,q} \mathcal{E}_{\mathcal{Z}|\mathcal{X},\theta^{n-1}} \left[z_{p,i} z_{q,i-1}\right] \log T_{pq} + \sum_i \sum_p \mathcal{E}_{\mathcal{Z}|\mathcal{X},\theta^{n-1}} \left[z_{p,i}\right] \log \mathsf{P}_p \left(x_i\right) \\ &= \sum_i \sum_{p,q} t^n_{pq,i} \log T_{pq} + \sum_i \sum_p s^n_{p,i} \log \mathsf{P}_p \left(x_i\right) \end{aligned} \tag{4.8}$$

where we have written $s^n_{p,i}$ for $\mathcal{E}_{\mathcal{Z}|\mathcal{X},\theta^{n-1}} \left[z_{p,i}\right]$ and $t^n_{pq,i}$ for $\mathcal{E}_{\mathcal{Z}|\mathcal{X},\theta^{n-1}} \left[z_{p,i} z_{q,i-1}\right]$. These quantities are analogous to the responsibilities of the mixture model, although that name is not used in this case. We shall call them the **state estimates** and **transition estimates** respectively. They are given by the probabilities

$$s^n_{p,i} = \mathsf{P}_{\theta^{n-1}} \left(z_{p,i} = 1 \mid x_1 \ldots x_N\right) \tag{4.9}$$

$$t^n_{pq,i} = \mathsf{P}_{\theta^{n-1}} \left(z_{p,i} = 1 \;\&\; z_{q,i-1} = 1 \mid x_1 \ldots x_N\right) \tag{4.10}$$

Unlike in the case of the mixture model, the conditioning on the observations does not reduce to conditioning only on x_i, due to the coupling of latent variables in this model. These probabilities need to be calculated by an iterative approach known as the **forward–backward algorithm**.

4.2.1 E-step: The forward–backward algorithm

The algorithm by which the state and transition estimates are found is a special case of a general inference algorithm on probabilistic graphical models (Jordan 1998). However, we have not developed the general theory of such models here. Therefore, we simply lay out the algorithm, and then show that it does indeed achieve the necessary estimates.

We are given a hidden Markov model with known parameters, T and $\{\theta_p\}$, and a set of observations $\{x_i\}$. We wish to calculate the marginal probabilities of (4.9) and (4.10). Introduce two quantities, each a joint probability distribution, whose values can be calculated recursively at each timestep. The first is the likelihood that the system emitted the observed values $x_1 \ldots x_i$ and was then in state p at the ith time-step.

$$F_{p,i} = \mathsf{P} \left(y_i = p, x_1 \ldots x_i\right) \tag{4.11}$$

$$= \mathsf{P}_p \left(x_i\right) \sum_q T_{pq} F_{q,i-1} \tag{4.12}$$

Note that the likelihood that the model generated the complete string of observations is then just

$$\mathcal{L}_{\mathcal{X}} \left(T, \{\theta_p\}\right) = \sum_p F_{p,N} \tag{4.13}$$

thus obtaining the promised recursive expression for this likelihood. We will need this value again below, and so reserve for it the symbol L.

The second recursive quantity we need is the likelihood that, starting from state p on step i the system generated the observed string $x_{i+1} \ldots x_N$.

$$\begin{aligned} B_{p,i} &= \mathsf{P}\left(x_{i+1} \ldots x_N \mid y_i = p\right) && (4.14) \\ &= \sum_q T_{qp} \mathsf{P}_q\left(x_{i+1}\right) B_{q,i+1} && (4.15) \end{aligned}$$

Note that due to the Markov nature of the latent variable chain, observations x_{i+1} and further are independent of all previous observations given the value of y_i and so $B_{p,i}$ is also equal to $\mathsf{P}\left(x_{i+1} \ldots x_N \mid y_i = p, x_1 \ldots x_i\right)$

Both recursions can be written more succinctly if we introduce a $(P+1) \times (P+1)$ diagonal matrix R_i (indexed, like T, from 0) with $R_{pp,i} = \mathsf{P}_p\left(x_i\right)$. We then obtain, with vector forms for both F and B

$$F_i = R_i T F_{i-1} \qquad \text{and} \qquad B_i = T^{\mathsf{T}} R_{i+1} B_{i+1} \tag{4.16}$$

Notice that one of these recursions runs forward over the observations, while the other runs backwards. Thus the name "forward–backward".

The estimates $s_{p,i}$ and $t_{pq,i}$ can be expressed in terms of F and B:

$$\begin{aligned} s_{p,i} &= \mathsf{P}\left(y_i = p \mid x_1 \ldots x_N\right) \\ &= \frac{\mathsf{P}\left(x_{i+1} \ldots x_N \mid y_i = p\right) \mathsf{P}\left(y_i = p, x_1 \ldots x_i\right)}{\mathsf{P}\left(x_1 \ldots x_N\right)} \\ &= F_{p,i} B_{p,i} / L && (4.17) \end{aligned}$$

and

$$\begin{aligned} t_{pq,i} &= \mathsf{P}\left(y_i = p, y_{i-1} = q \mid x_1 \ldots x_N\right) \\ &= \frac{\mathsf{P}\left(x_{i+1} \ldots x_N \mid y_i = p\right) \mathsf{P}\left(x_i \mid y_i = p\right) \mathsf{P}\left(y_i = p \mid y_{i-1} = q\right) \mathsf{P}\left(y_{i-1} = q, x_1 \ldots x_{i-1}\right)}{\mathsf{P}\left(x_1 \ldots x_N\right)} \\ &= B_{p,i} R_{pp,i} T_{pq} F_{q,i-1} / L && (4.18) \end{aligned}$$

where, in the second step of each of these results we have used the Markovian properties of the model to remove irrelevant conditioning variables.

The E-step of the Baum-Welch algorithm, then, is achieved by substituting into (4.17) and (4.18) the $(n-1)$th iteration parameter estimates, to obtain $s^n_{p,i}$ and $t^n_{pq,q}$.

4.2.2 M-step: Parameter re-estimation

The re-estimation of the Markov transition matrix is straightforward, and reminiscient of the re-estimation of the mixing probabilities of a mixture model. We optimize the expected log-likelihood of (4.8) with respect to T_{pq}, enforcing the constraint $\sum_p T_{pq} = 1$ with a Lagrange multiplier, to obtain

$$\left.\frac{\partial}{\partial T_{pq}}\right|_{T^n_{pq}} \left(\sum_i \sum_{p,q} t^n_{pq,i} \log T_{pq} - \lambda \sum_p T_{pq} \right) = \sum_i \frac{t^n_{pq,i}}{T^n_{pq}} - \lambda = 0 \tag{4.19}$$

From which we find that $T_{pq} \propto \sum_i t^n_{pq,i}$. The normalization constraint then gives us

$$T^n_{pq} = \frac{\sum_{i=1}^{N} t^n_{pq,i}}{\sum_{i=0}^{N-1} s^n_{q,i}} \tag{4.20}$$

where we use the fact that $\sum_p t^n_{pq,i} = s^n_{q,i-1}$ which follows from the marginalization of the joint distribution represented by $t_{pq,i}$

The remaining update rules, for the output distribution parameters $\{\theta_p\}$, depend on the form of the output distribution function. We can, however, make some headway. First, note that the θ_p are independent of each other, and so can each be optimized separately. Furthermore, only the second term in the expected log-likelihood (4.8) has any dependence on θ_p. As a result, we arrive at an update rule identical to that encountered in the case of the mixture model (2.15), with the responsibilities replaced by the state estimates $s^n_{p,i}$.

$$\theta^n_p = \operatorname*{argmax}_{\theta_p} \sum_i s^n_{p,i} \log \mathsf{P}_{\theta_p}(x_i) \tag{4.21}$$

As in the mixture case, we may interpret this as a weighted fit of the output distribution parameters to the observations x_i, with weights given by the estimates $s^n_{p,i}$.

4.3 Sparse HMMs

In this section, we introduce a special case of the HMM. This restricted model, the **sparse hidden Markov model** or SHMM, is one that may be encountered with some frequency in practical modeling situations; indeed we develop it here because it will be of use to us in a neural data analysis problem tackled in the following chapters. The restricted model itself will only be of limited interest from an algorithmic point of view: all of the standard HMM learning algorithms may be used and, though we will describe an adaptation of the standard Baum-Welch algorithm, the advantages thereby derived are merely in the realm of efficiency. However, the introduction of this model will allow us to speak meaningfully of a mixture of sparse HMMs, and derive an efficient learning algorithm for such a mixture.

The processes that we consider are sparse in the following sense. In each string of observations x_i, the majority yield a null value, which we represent by the symbol ø. This value tells us relatively little about the state of the underlying process; in effect, the process has no output at these observation times. Scattered within this string of øs are occasional non-null output values, but these are distributed sparsely. Nevertheless, they provide our only information about the state of the process.

We will examine hidden Markov models for such a process. Each model contains one or more states for which the output distribution produces the outcome ø with probability 1. We will refer to these as the null states. We will assume for the purposes of this discussion that the output distributions in the remaining states assign probability 0 to this outcome, although most of the results of this and the following sections can be carried through even if this were not the case. The sparsity of the process requires that the transition matrix be set up so that on the majority of time-steps the model is in a null state. On the whole, then, the transition probabilities from null states to states with full output distributions are relatively low, while transitions in the other direction are relatively likely.

How sparse is sparse? There is no precise answer to this question. All of the algorithms that we discuss can be equally well applied to models which spend little or no time in null states. However, it will be apparent that under that condition they would produce poor results. The transition between sparse and full, then, is a matter for empirical discovery within the framework of the application.

Learning in the SHMM may proceed by the standard Baum-Welch algorithm that was laid out in the case of the full HMM. However, it is possible to achieve some optimizations on the basis of the sparse output structure, which we will discuss here. Before we can do so, however, we need to recast the forward–backward algorithm slightly.

4.3.1 Another view of the forward–backward algorithm

The presentation in section 4.2.1 described the forward–backward algorithm in a notationally compact form ideal for exposition. In fact, as described, the algorithm is numerically unstable in implementations. This instability can be resolved by a small modification, which is the subject of this section. The same modification is important to adaptations of the algorithm to sparse HMMs.

The difficulty with the currently described algorithm is this. At each instance i, the conjunction of observations that appear in the likelihoods described by F_i and B_i is of a different size. For instance, F_1 describes the likelihood $\mathsf{P}(y_1, x_1)$, while F_N describes $\mathsf{P}(y_N, x_1 \ldots x_N)$. If the typical density at the observation point x_i is a, then while F_1 is of order a, F_N is of order a^N. Similarly, B_1 is of order a^{N-1}, while B_N is of order a^0. The product of the two terms is always of order a^N, and it is divided by the likelihood (also order a^N) to derive estimates $s_{p,i}$ and $t_{pq,i}$ of order 1. If the value a is considerably different from 1, the intermediate values in this calculation can become either very large or very small, and the computation may become numerically unstable.

We can resolve this problem by introducing an alternative group of recursive functions that remain of order 1 throughout. In fact, we need three functions

$$C_i = \mathsf{P}(x_i \mid x_1 \ldots x_{i-1}) \tag{4.22}$$

$$F_{p,i} = \mathsf{P}(y_i = p \mid x_1 \ldots x_i) \tag{4.23}$$

$$B_{p,i} = \frac{\mathsf{P}(x_{i+1} \ldots x_N \mid y_i = p)}{\mathsf{P}(x_{i+1} \ldots x_N \mid x_1 \ldots x_i)} \tag{4.24}$$

which are calculated recursively as follows.

$$C_i = \mathbf{1}^{\mathsf{T}} R_i T F_{i-1} \tag{4.25}$$

$$F_i = R_i T F_{i-1} / C_i \tag{4.26}$$

$$B_i = T^{\mathsf{T}} R_{i+1} B_{i+1} / C_{i+1} \tag{4.27}$$

where $\mathbf{1}$ is a vector of P ones, and is introduced to indicate a sum of the elements of the following vector-valued product.

Given these new functions, the state and transition estimates become

$$s_{p,i} = F_{p,i} B_{p,i} \qquad \text{and} \qquad t_{pq,i} = B_{p,i} R_{pp,i} T_{pq} F_{q,i-1} / C_i. \tag{4.28}$$

The normalization of the recursive terms F and B defined here is crucial to the following exposition of the forward–backward algorithm for SHMMs. Thus, all subsequent references to the algorithm, and the symbols F, B and C will refer to this recast version.

4.3.2 Forward–backward algorithm for sparse HMMs

By definition, the output sequences recorded from a sparse HMM tend to contain long stretches of null outputs. These segments leave the model in an identifiable configuration; that is, the value of F_i at the end, and B_i at the beginning of such a sequence is relatively independent of the measurements before and after such a segment.

Consider a long segment of null observations stretching from observation indices a to $a + l$. We assume that the values of the functions F_{a-1} and B_{a+l} are known, while we seek to calculate F_{a+l} and B_{a-1}.

Consider, first, the forward term. Let the notation $R_\emptyset$ stand for the value of the likelihood matrix R_i in cases where $x_i = \emptyset$. Recall that such matrices are diagonal, with $R_{pp,i} = \mathsf{P}_p(x_i)$. In this case, these elements are 1 for null states and 0 elsewhere. We then have

$$F_{a+l} \propto (R_\emptyset T)^{(l+1)} F_{a-1} \tag{4.29}$$

with the vector then normalized so that the sum of its elements is 1. Whatever the value of F_{a-1}, this expression will be dominated by the leading eigenvector of the matrix $R_\emptyset T$. We will write $F_\emptyset$ for the suitably normalized eigenvector — note that normalization here means that the sum of the elements, rather than the sum of the squares of the elements, is 1. In fact, $F_\emptyset$ is the stationary distribution of the Markov chain that is obtained by restricting the current estimate of the Markov model to only the null states, the transition matrix of which is given by renormalizing the columns of the matrix $R_\emptyset T R_\emptyset$. Thus the forward step after a sequence of null outputs is achieved by simply setting the value of the forward term to $F_\emptyset$.

Using a similar argument we can show that at the *beginning* of a long segment of nulls, the value of the backward term B_{a-1} will approach the leading eigenvector of the matrix $T^\mathsf{T} R_\emptyset$, suitably normalized. We write $\tilde{B}_\emptyset$ for the unnormalized eigenvector. Unlike the forward terms, B_i is not itself a probability distribution and thus we have no immediate way to normalize. However the products $F_i B_i = \mathsf{P}(y_i \mid x_1 \ldots x_N)$ are probabilities. Thus, knowing the value of F_{a-1} we can find the appropriate normalization for B_{a-1} (which is potentially different before each null segment).

The forward–backward steps across a sequence of nulls from a to $a+l$ is thus

$$F_{a+l} = F_\emptyset \tag{4.30}$$

$$B_{a-1} = \tilde{B}_\emptyset / F_{a-1}^\mathsf{T} \tilde{B}_\emptyset \tag{4.31}$$

The use of these forms limits the application of the full forward–backward algorithm to only those regions in which some non-null outputs are observed, often at a considerable computational savings.

4.4 Mixtures of Sparse HMMs

We consider the following model. We have M independent sparse hidden Markov models. Call the output of the mth model at time-step i, $x_{m,i}$[1]. We do not observe these variables directly, instead we make a single observation at each time-step, derived from these values according to the following

$$x_i = \begin{cases} \emptyset & \text{if all } x_{m,i} = \emptyset \\ x_{m^*,i} & \text{if only } x_{m^*,i} \neq \emptyset \\ \not{q} & \text{if multiple } x_{m,i} \neq \emptyset \end{cases} \tag{4.32}$$

[1] Variables in the ensuing development will often need to be identified by state, component model and observation number. We shall adopt two conventions to assist in correctly parsing all of these subscripts. 1. The order will always be (state, model, instance), but some indices might be omitted if unnecessary. 2. the letters p and q will be used to index state, m and l for model, and i for instance; n will be used in the superscript for EM iteration number as before.

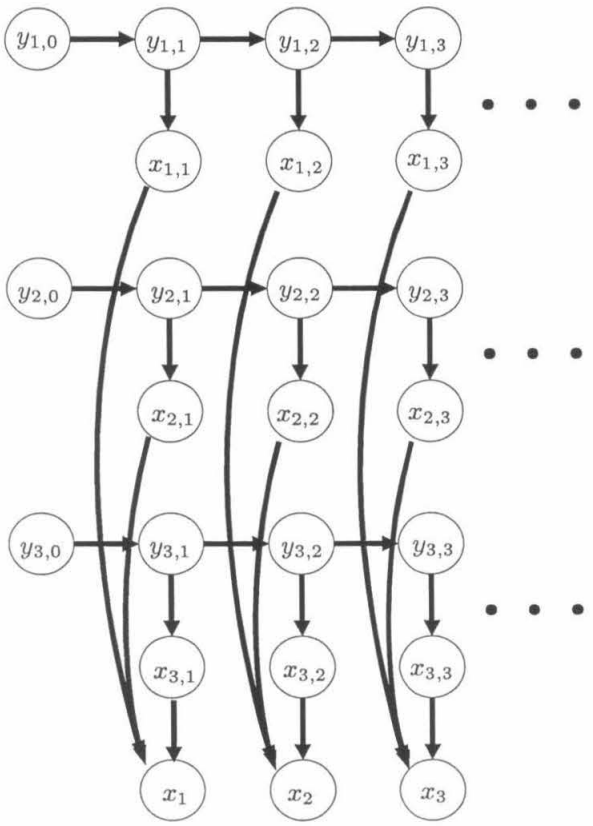


Figure 4.2: A mixture of sparse hidden Markov models

If more than one HMM has non-null output, we see only the fact that a collision occured, noted by the special output value ϕ. We obtain no information about which, nor even how many, of the HMMs had non-null outputs.

The model is illustrated in figure 4.2. The random variables in the model are the state variables $y_{m,i}$ and the corresponding outputs $x_{m,i}$. The observed value x_i is actually a deterministic function of the outputs, $x_{m,i}$, of each component sparse HMM.

4.4.1 Learning

Since the component SHMMs are presumed to be independent, the joint data likelihood, given observations $\mathcal{X} = \{x_i\}$, HMM outputs $\mathcal{X}_m = \{x_{m,i}\}$ and indicator variables $\mathcal{Z} = \{z_{m,i}\}$ is simply the product of the joint data likelihoods (4.5) for each of the component HMMs given observations $\{x_{m,i}\}$ and indicators $\{z_{m,i}\}$. In the log domain, this is

$$\ell_{\mathcal{X},\mathcal{X}_m,\mathcal{Z}}\left(\{T_m\},\{\theta_{p,m}\}\right) = \sum_m \sum_i \left(\sum_{p,q} z_{p,m,i} z_{q,m,i} \log T_{pq,m} + \sum_p z_{p,i} \log \mathsf{P}_{p,m}\left(x_{m,i}\right) \right) \tag{4.33}$$

The E-step involves calculation of the expected value of this expression with respect to the distribution $\mathsf{P}\left(z_{p,m,i}, x_{m,i} \mid x_i\right)$. Note that the expectation is taken not only with respect to the $z_{m,i}$ (as usual), but also with respect to the $x_{m,i}$, which are not directly observed in this case. The expected value is

$$\begin{aligned} Q^n(\{T_m\},\{\theta_{p,m}\}) &= \mathcal{E}_{\mathcal{Z},\mathcal{X}_m|\mathcal{X},\theta^{n-1}}\left[\ell_{\mathcal{X},\mathcal{X}_m,\mathcal{Z}}\left(\{T_m\},\{\theta_{p,m}\}\right)\right] \\ &= \sum_m \sum_i \sum_{p,q} \mathcal{E}_{\mathcal{Z},\mathcal{X}_m|\mathcal{X},\theta^{n-1}}\left[z_{p,m,i} z_{q,m,i-1}\right] \log T_{pq,m} \\ &\quad + \sum_m \sum_i \sum_p \mathcal{E}_{\mathcal{Z},\mathcal{X}_m|\mathcal{X},\theta^{n-1}}\left[z_{p,m,i} \log \mathsf{P}_{p,m}\left(x_{m,i}\right)\right] \\ &= \sum_m \sum_i \sum_{p,q} t^n_{pq,m,i} \log T_{pq,m} \\ &\quad + \sum_m \sum_i \sum_p s^n_{p,m,i} \mathcal{E}_{x_{m,i}|z_{p,m,i}=1,\mathcal{X},\theta^{n-1}}\left[\log \mathsf{P}_{p,m}\left(x_{m,i}\right)\right] \end{aligned} \tag{4.34}$$

Note the change in distribution that appears in the expectation of the final expression; we have used the fact that $z_{p,m,i}$ is an indicator variable as follows

$$\begin{aligned} &\mathcal{E}_{\mathcal{Z},\mathcal{X}_m|\mathcal{X},\theta^{n-1}}\left[z_{p,m,i} \log \mathsf{P}_{p,m}\left(x_{m,i}\right)\right] \\ &= \sum_{z_{p,m,i}} \int dx_{m,i}\ \mathsf{P}_{\theta^{n-1}}\left(z_{p,m,i}, x_{m,i} \mid \mathcal{X}\right)\ z_{p,m,i} \log \mathsf{P}_{p,m}\left(x_{m,i}\right) \\ &= \mathsf{P}_{\theta^{n-1}}\left(z_{p,m,i}=1 \mid \mathcal{X}\right) \int dx_{m,i}\ \mathsf{P}_{\theta^{n-1}}\left(x_{m,i} \mid z_{p,m,i}=1, \mathcal{X}\right) \log \mathsf{P}_{p,m}\left(x_{m,i}\right) \\ &\qquad + \mathsf{P}_{\theta^{n-1}}\left(z_{p,m,i}=0 \mid \mathcal{X}\right)\ 0 \\ &= s^n_{p,m,i} \mathcal{E}_{x_{m,i}|z_{p,m,i}=1,\mathcal{X},\theta^{n-1}}\left[\log \mathsf{P}_{p,m}\left(x_{m,i}\right)\right] \end{aligned} \tag{4.35}$$

What is this expected value? If no collision was observed then $x_{m,i}$ is completely determined by $z_{p,m,i}$ and x_i. If state p of model m is a null state, $x_{m,i} = \emptyset$; otherwise $x_{m,i} = x_i$. On the other hand, if a collision was observed then x_i tells us nothing about the value of $x_{m,i}$. It is still true that if the state (p, m) has no output, $x_{m,i} = \emptyset$; but now, if the state is non-null, $x_{i,m}$ is distributed according to $\mathsf{P}_{p,m}\left(x\right)$. Thus, for non-null states, we have

$$\mathcal{E}_{x_{m,i}|z_{p,m,i}=1,\mathcal{X},\theta^{n-1}}\left[\log \mathsf{P}_{p,m}\left(x_{m,i}\right)\right] = \begin{cases} \log \mathsf{P}_{p,m}\left(x_i\right) & \text{if } x_i \neq \emptyset \\ -\mathsf{H}[\mathsf{P}_{p,m}] & \text{if } x_i = \emptyset \end{cases} \tag{4.36}$$

where $\mathsf{H}[\cdot]$ indicates the entropy of the distribution.

4.4.2 Coupled forward–backward algorithm

We need to calculate the state and transition estimates that appear in (4.34). We do so by running the forward–backward algorithm separately on each component SHMM. Since direct observation of

the outputs of the component models is not possible, however, we must estimate those outputs using the observed output of the entire mixture, as well as the recursive terms, $F_{m,i-1}$ and $B_{m,i-1}$, from all of the components. This use of the values of the recursive terms from other component SHMMs leads to a coupling of the different instances of the forward–backward algorithm.

Despite this coupling, however, the separation of the estimation process into multiple component recursions constrains the E-step optimization to only those distributions which satisfy a factorization constraint of the form (for the F recursion)

$$.\mathsf{P}\left(\{y_{m,i}\} \mid x_1 \ldots x_i\right) = \prod_m \mathsf{P}\left(y_{m,i} \mid x_1 \ldots x_i\right) \tag{4.37}$$

as well as a second, similar, constraint due to the B recursion. Such imperfect E-steps were discussed briefly in section 1.8. At each time-step we calculate the full joint distribution of the $y_{m,i}$ (which contains P^M terms) but then store only the marginals (needing only $P \times M$ terms). Clearly, to calculate the state and transtion estimates we only need the marginals, and so from that point of view the restriction is reasonable. However, the $F_{m,i}$ are also used to estimate the distribution at the $(i+1)$th step. Use of the factorized distribution for the ith step, rather than the full joint distribution, leads to a mis-estimation of the joint distribution at the $(i+1)$th step. It is thus, that the constraint of (4.37) appears.

We will discuss the impact of this constraint on the EM process below. First, let us proceed with the exposition of the algorithm. The recursive terms are defined much as before.

$$C_i = \mathsf{P}\left(x_i \mid x_1 \ldots x_{i-1}\right) \tag{4.38}$$

$$F_{p,m,i} = \mathsf{P}\left(y_{m,i} = p \mid x_1 \ldots x_i\right) \tag{4.39}$$

$$B_{p,m,i} = \frac{\mathsf{P}\left(x_{i+1} \ldots x_N \mid y_{m,i} = p\right)}{\mathsf{P}\left(x_{i+1} \ldots x_N \mid x_1 \ldots x_i\right)} \tag{4.40}$$

However, in this case the x_i are not the direct outputs of the HMM, but are rather the overall observations from the mixture. Thus, the calculations become slightly more elaborate. We will obtain here expressions for only the forward terms C_i and $F_{p,m,i}$. The calculation of $B_{p,m,i}$ proceeds similarly.

We write $\tilde{F}_{p,m,i}$ for $\mathsf{P}\left(y_{m,i} = p \mid x_1 \ldots x_{i-1}\right)$, the probability of finding the mth model in state p on step i given the previous observations, but not the current one. This is, of course, based recursively on our estimate of the distribution of states $y_{m,i-1}$ given observations up to x_{i-1}. With our factorial assumption on the distribution of $y_{m,i-1}$ this is given by

$$\tilde{F}_{m,i} = T_m F_{m,i-1} \tag{4.41}$$

Also of interest will be the probability that model m is in a null state. We will write $\emptyset_{p,m} = 1$ if $\mathsf{P}_{p,m}(\emptyset) = 1$ and $\emptyset_{p,m} = 0$ otherwise. Using this indicator, we obtain $\tilde{F}_{\emptyset,m,i} = \sum_p \emptyset_{p,m} \tilde{F}_{p,m,i}$.

It will be useful to treat separately the three cases where x_i is 1. null, 2. non-null and non-collision, and 3. a collision.

1. $x_i = \emptyset$

   In this case C_i is the probability that every component is in a null state,

$$C_i = \prod_m \tilde{F}_{\emptyset,m,i} \tag{4.42}$$

   To calculate $F_{p,m,i}$ we need to find the distribution $\mathsf{P}(x_i = \emptyset, y_{m,i} = p \mid x_1 \ldots x_{i-1}) = \mathsf{P}(x_i = \emptyset \mid y_{m,i} = p, x_1 \ldots x_{i-1}) \tilde{F}_{p,m,i}$. This is clearly 0 if $\emptyset_{p,m} = 0$. If $\emptyset_{p,m} = 1$, then $\mathsf{P}(x_i = \emptyset \mid y_{m,i} = p, x_1 \ldots x_{i-1})$ is just the probability that all other components are in null states. Thus

$$\begin{aligned} F_{p,m,i} &= \frac{1}{C_i} \mathsf{P}(x_i = \emptyset, y_{m,i} = p \mid x_1 \ldots x_{i-1}) \\ &= \frac{1}{C_i} \emptyset_{p,m} \tilde{F}_{p,m,i} \prod_{l \neq m} \tilde{F}_{\emptyset,l,i} \\ &= \emptyset_{p,m} \frac{\tilde{F}_{p,m,i}}{\tilde{F}_{\emptyset,m,i}} \end{aligned} \tag{4.43}$$

2. $x_i \neq \emptyset, \not{c}$

   Here, C_i is the probability that one component outputs the observed value x_i, while all the other components are in null states.

$$C_i = \sum_m \sum_p \mathsf{P}_{p,m}(x_i) \tilde{F}_{p,m,i} \prod_{l \neq m} \tilde{F}_{\emptyset,l,i} \tag{4.44}$$

   $\mathsf{P}(x_i \mid y_{m,i} = p, x_1 \ldots x_{i-1})$ is straightforward if (p,m) is not null; being $\mathsf{P}_{p,m}(x_i)$ times the the probability that all other components are in null states. If, on the other hand, $\emptyset_{p,m} = 1$, then the conditional probability is given by the probability that exactly one of the remaining components outputs the value x_i.

$$F_{p,m,i} = \frac{1}{C_i} \tilde{F}_{p,m,i} \left((1 - \emptyset_{p,m}) \mathsf{P}_{p,m}(x_i) \prod_l \tilde{F}_{\emptyset,l,i} + \emptyset_{p,m} \sum_{l \neq m} \sum_p \mathsf{P}_{p,l}(x_i) \tilde{F}_{p,l,i} \prod_{k \neq l,m} \tilde{F}_{\emptyset,k,i} \right) \tag{4.45}$$

3. $x_i = \not{c}$

In this case, C_i is the probability that at least two components are in a non-null state

$$C_i = 1 - \left(\prod_m F_{\emptyset,m,i} \right) - \left(\sum_m (1 - F_{\emptyset,m,i}) \prod_{l \neq m} F_{\emptyset,m,i} \right) \tag{4.46}$$

The expression for $F_{p,m,i}$ is notationally cumbersome, so we will not write it explicitly. Instead, we note that $\mathsf{P}\left(x_i \mid y_{m,i} = p, x_1 \ldots x_{i-1}\right)$ is the probability that at least one other component is non-null if $\emptyset_{p,m} = 0$ and that at least two other components are non-null if $\emptyset_{p,m} = 1$. Both of these probabilities are found in a form similar to that of C_i, above.

Once the terms $F_{i,m}$ and $B_{i,m}$ have been calculated, the state and transition estimates are derived using (4.28) applied to each component in turn.

Consequences of the factorial approximation

To what extent does the factorial constraint of the coupled forward–backward algorithm affect the eventual parameter estimates? We may can make two separate arguments for robustness of the estimates to error.

First, it might be feared that, since the terms F and B are calculated recursively and since there is an error in each calculation, the estimated value and the true value would progressively diverge over time. This is not the case. Boyen and Koller (1999) have examined factorial approximations such as the present one in the context of general dynamic probabilistic networks. They argue that the approximation error does not grow over time because two forces oppose the growth. First, the incorporation of observed data tends to drive the approximated distribution towards the correct one. Second, the randomization due to the stochastic transition from the $(i-1)$th step to the ith tends to broaden both the correct distribution and the approximate one, which also has the effect of bringing them closer together. In other words, $T_m F_{m,i-1}$ may be closer to the true $\mathsf{P}\left(y_{m,i} \mid x_i \ldots x_{i-1}\right)$ than $F_{m,i-1}$ is to $\mathsf{P}\left(y_{m,i-1} \mid x_i \ldots x_{i-1}\right)$. Intuitively, we may think of each random transition contributing to a "forgetting" of the old, incorrect, distribution.

To these arguments we can add a third, peculiar to the current model. When the observation $x_i = \emptyset$, our forward and backward steps are correct. Recall from the discussion of the forward–backward algorithm for sparse HMMs that after a substantial stetch of null observations, F_i (B_i) is relatively independent of its value at the beginning (end) of the segment. Thus, in the mixture, whenever we enounter a stretch of null observations we tend to reset the forward–backward estimates to their correct values.

Second, even if the errors in the state and transition estimates are typically large, it is possible that their effect on parameter estimates derived through EM may be small. Constrained E-steps of the sort we perform here were discussed briefly in section 1.8. There it was pointed out that

generalized EM using a constrained optimization of the latent variable distribution will eventually yield the correct maximum-likelihood parameter estimates if and only if the conditional distribution at the optimum $\mathsf{P}_{\theta^*}(\mathcal{Y} \mid \mathcal{X})$ satisfies the constraint. In the present case, this will be true if, at the optimal parameter values, only one component is likely to be in a non-null state at each time-step where $x_i \neq \emptyset, \not{c}$. In other words, all observed data can be assigned with high likelihood to only one component. If, on the other hand, two different components claim equal responsibility for the point, then the factored distribution will assign a probability close to 0.25 that they were both in non-null states, whereas the correct joint probability would be 0 (if they were both in non-null states a collision would have been observed). Futhermore, provided that most data are well assigned in this way, the above arguments suggest that a small number of ambiguous points will not have a profound effect on the estimates associated with the others. Thus, in well clustered data, the approximation has little effect on the eventual estimates, even if, in intermediate steps of EM, it is inaccurate. Note that "well-clustered" here does not necessarily mean that the output distributions are well separated. Each data point must be assigned to a single component, either because only that component has an output distribution which assigns it high likelihood or because its temporal relationship to nearby points marks it as arising from a particular model.

4.4.3 Parameter re-estimation

The M-step requires optimization of the expected log-likelihood (4.34) with respect to the parameters, with the estimates $s^n_{p,m,i}$ and $t^n_{pq,m,i}$ fixed at the values derived from the E-step. The expression of (4.34) contains separate additive terms for each component model; as a result, it can be ompti-mized with respect to the parameters of each SHMM independently. The part that involves the mth model is

$$Q^n_m(T_m, \{\theta_{p,m}\}) = \sum_i \sum_{p,q} t^n_{pq,m,i} \log T_{pq,m} + \sum_i \sum_p s^n_{p,m,i} \mathcal{E}_{x_{m,i} | z_{p,m,i}=1, \mathcal{X}, \theta^{n-1}} [\log \mathsf{P}_{p,m}(x_{m,i})] \tag{4.47}$$

Optimization with respect to $T_{pq,m}$ can clearly proceed exactly as in the standard case, and so we obtain

$$T^n_{pq,m} = \frac{\sum_{i=1}^{N} t^n_{pq,m,i}}{\sum_{i=0}^{N-1} s^n_{q,m,i}} \tag{4.48}$$

Re-estimation of the output distribution parameters $\theta_{p,m}$ is almost the same as in the standard Baum–Welch algorithm. It is still the case that the different output distributions can be optimized independently. For states with null output distributions, of course, there are no parameters to fit.

For non-null distributions, we recall the result of (4.36) and find that

$$\theta^n_{p,m} = \underset{\theta_{p,m}}{\operatorname{argmax}} \left(\sum_{i:x_i \neq \emptyset, \not\sigma} s^n_{p,m,i} \log \mathsf{P}_{\theta_{p,m}}(x_i) - \sum_{i:x_i = \not\sigma} s^n_{p,m,i} \mathsf{H}[\mathsf{P}_{p,m}] \right) \tag{4.49}$$

(Note that if $x_i = \emptyset$ and (p, m) is not a null state, $s^n_{p,m,i}$ must be 0, and so we can ignore the corresponding terms). Thus, the parameters are fit to the observed non-null and non-collision data, weighted by the state estimates as usual, but with an additional entropy penalty on the likelihood which weighted by the sum of the state estimates for collision time-steps. In practice, if the number of collisions is small relative to the total number of non-null observations, we can often neglect this term.

Part II

Applications

Chapter 5 Spike Sorting

5.1 Introduction

In this chapter we take up the first and most extensive of our neural data-analytic applications of latent variable methods. Spike sorting allows scientists and technologists to efficiently and reliably monitor the signals emitted simultaneously by many different nerve cells within intact brains. To neuroscientists, interested in how the brain carries out it complex functions, such multi-neuron data is essential input to improved understanding. In addition, the ability to collect signals from large numbers of specific neurons brings biomedical engineers closer to the dream of prosthetic devices driven directly by neural output.

5.1.1 Extracellular recording: the source and nature of the signal

The action potential

Most neurons communicate with each other by means of short, local perturbations in the electrical potential across the cell membrane, called **action potentials**. The discovery of the mechanism that gives rise to the action potential was one of the seminal breakthroughs of early neurophysiology (Hodgkin and Huxley 1952), and the account made at that time of action potentials in the squid giant axon has proven to apply quite broadly. For the purposes of this discussion, we will not need a detailed account of the action potential. However, a qualitative understanding of some points will be important.

Protein complexes embedded in the membranes of neurons pump specific ions into or out of the cytoplasm so as to establish strong concentration gradients across the membrane. The membrane possesses a baseline permeability to some of these ions, and so the system equilibrates with an electrical potential opposing the chemical potential established by the ion pumps. This electrical potential, around -70 mV for most cells (the convention is that membrane potentials are measured inside the cell, with reference to the extracellular medium), is known as the **resting potential**. Cells at rest are said to be **polarized**. Two ions are important to the action potential. Sodium ions (Na^+) are concentrated outside the cell at rest, while potassium ions (K^+) are concentrated inside.

Besides the ion pumps, the membrane contains other proteins that serve as temporary **channels** to specific ions. These channel proteins have two or more metastable conformations. In one of these, the **open** conformation, the channel allows specific ions to pass through it. Thus, as the number of channels in the open state varies, the permeability of the membrane to specific ions changes. Two

types of channel, one permeable to Na^+ and the other to K^+, form the basic machinery of the action potential. Both channels are voltage-sensitive, that is, the probability of finding them in the open state depends on the electrical potential across the membrane. In particular, they are both more *likely* to open as the potential inside the cell increases.

The action potential is initiated when a patch of membrane becomes slightly depolarized. As the interior voltage increases, the voltage-sensitive sodium channels are faster to open that the potassium ones. Na^+ ions are driven into the cell through these open channels, further raising the interior potential and establishing a rapid positive-feedback loop. This feedback loop is terminated in two ways. First, once in the open state, the sodium channels begin to transition to a third, **inactivated** conformation. Here again the channel is impermeable to ions, but this configuration is different from the original, closed, one. In particular, the probability of transition back into the open state, while the membrane potential remains high, is now extremely low. The return transition, called **de-inactivation**, happens only at potentials near or below rest, when the protein switches directly to the closed state. Second, the potassium channels also open in response to the increased cellular potential. The diffusion gradient for K^+ is opposite to that for Na^+, and so K^+ ions leave the cell, restoring its polarization. In fact, the membrane potential falls below the resting level. As it falls, the potassium channels close (they have no inactivated state). Eventually, all of the voltage-sensitive channels are either inactivated or closed, returning the membrane to its baseline permeability and the resting potential.

The voltage-sensitive sodium channels are most highly concentrated on the cell body at the point where the axon emerges (the axon hillock). This is the first piece of cell membrane to undergo an action potential, usually initiated by the passive propagation of depolarizations caused by membrane channels in the dendrite that open due to synaptic input. This action potential depolarizes a nearby piece of membrane on the axon, thus launching it into an action potential too, which, in turn, depolarizes a further piece and so on. Thus, once initiated at the hillock, the action potential travels down the axon, eventually triggering the release of a neurotransmitter onto another cell.

As the membrane comes out of the action potential, a number of potassium channels are still open and many sodium channels remain inactivated. Thus, for a short period of time called the **absolute refractory period** it is impossible to induce a second action potential in the cell. Even after the potassium channels have all closed and enough sodium channels have de-inactivated to allow another action potential to begin, the threshold perturbation needed to seed the action potential will be higher than normal. This period is called the **relative refractory period**. Eventually the inactivation of the sodium channels drops to an equilibrium level and the cell returns to the rest state.

In many cases a cell will fire a group of action potentials spaced by little more than the absolute refractory period. Such a group is called a **burst** or, sometimes, a **complex spike**. In general,

such bursts are not driven entirely by synaptic input, but rather by the biophysics of the neuronal membrane. For example, extremely long time-constant voltage-sensitive calcium channels are found in some neurons. The first action potential in a burst causes some number of these to open, but they neither close nor inactivate rapidly. Ca^{++}, which is concentrated outside the cell by the ion pumps, flows in through these open channels. As a result, as soon as the first action potential is over and the potassium channels closed, the depolarizing calcium current can launch the next action potential. The cell is still in its relative refractory period, however, so many sodium channels are still inactivated. As a result, the currents that flow in this and subsequent action potentials may not be quite as strong as in the initial one.

In many, if not most, neurons, voltage-sensitive channels are to be found all over the cell body and dendritic surface. Recent work in pyramidal neurons has shown that the action potential propagates not only down the axon, but also from the axon hillock back into the dendrite (Stuart and Sakmann 1994; Stuart *et al.* 1997; Buzsaki and Kandel 1998). Further, the degree of penetration varies with the recent activity of the cell (Spruston *et al.* 1995; Svoboda *et al.* 1997). The later action potentials in a burst penetrate the dendrite to a much lesser degree than the first.

Extracellular recording

The mechanism of the action potential, as well as many other important neuronal phenomena, have been understood through measurements taken using an intracellular electrode, that is, one which penetrates the cell. Unfortunately it is difficult to record with such an electrode in an intact animal and all but impossible in many awake ones. Fortunately, if all that is needed is the timing of action potentials in the cells, it is possible to acquire this information with an extracellular electrode. The most common such electrode is a fine metal wire, insulated everywhere but at the tip, which is tapered to an extremely fine point of only a few microns diameter. The uninsulated tip acquires a layer of ions at its surface which form the second plate of an extremely thin capacitor. The resistive coupling of the electrode to the surrounding medium is generally weak; resistances in the hundreds of $M\Omega$ are not uncommon. However, the capacitive coupling is much stronger, with 1kHz impedances in the hundreds or thousands of $k\Omega$.

The electrical currents associated with the flow of ions through the membrane are transient. If the electrode tip is near the membrane surface during an action potential, these currents couple to the electrode, resulting in a transient change in the potential of the electrode measured relative to any stable external point. Thus, if we were to make a trace of the electrode potential over time, we would see **spikes**[1] in the trace corresponding to the action potentials in the cell near the tip. The

[1] In this chapter, "spikes" occur in the electrode voltage trace, while "action potentials" occur on the cell membrane. This sharp distinction is not entirely conventional, but it is useful, allowing us to speak, for example, of the "changing amplitude of a spike" without any implications about the maximal currents that flow across the cell membrane. The time of occurrence of the spike and action potential will be taken to be the same.

relationship between the intracellular trace of the action potential and the extracellularly recorded spike is complex. First, the extracellular probe records a integral current from many patches of membrane that may be in many different stages of the propagating action potential. Second, the tip geometry filters the measured spike; for an electrode with a smooth surface this filter is dominated by a single-pole high-pass component, but for porous electrode tips (plated with platinum black, for example) it is more complicated (Robinson 1968).

Many cells' membranes might lie close to the electrode tip so that spikes from many cells are recorded. Historically, the experimenter has manoeuvered the electrode so that the tip lies very close to one cell, and thus the spikes from this cell are far larger in amplitude than the spikes from other cells. A simple hardware device can then be used to record the times of these large spikes, and thus of the action potentials in a single cell. Even if the spike shape associated with the neuron varies, its amplitude remains greater than that of any other cell's spikes. This process is referred to as single-cell isolation. It is time-consuming and, in an awake animal, temporary. Movement of the tissue relative to the electrode eventually causes the experimenter to "lose" the cell.

Multineuron recording

One can only learn so much about the brain by monitoring one neuron at a time. As a result, there has been a great deal of recent interest in multineuron recording[2].

There is some reason to believe, based on the biophysics of neurons (the literature is extremely large, but see, for example, Softky and Koch 1993) as well as some direct experimental evidence (again a list of citations could be very long, so we choose a recent example: Usrey *et al.* 1998), that action potentials that occur simultaneously in a pair of neurons with a shared synaptic target are far more effective at causing the target to fire than are two non-coincident action potentials. It is possible, then, that coincident firing plays a significant role in the transmission of information within the nervous system. A number of experimenters have argued that indeed more, or different, information is available if the precise timing of action potentials across multiple cells is taken into account (e.g., Gray and Singer 1989). Furthermore, even if the exact relationship of firing times between cells is not functionally significant, this relationship can provide valuable (though indirect) clues to the micro-circuitry of the system (e.g., Alonso and Martinez 1998; Abeles *et al.* 1993).

It is possible to collect multineuron data by introducing many separate electrodes into the brain and isolating a single neuron with each one. Indeed many of the studies cited above were carried out in this way. This approach is, however, difficult to execute and difficult to scale. There are two approaches possible to obtaining many isolations. One can insert many individually positionable

[2]We shall take "multineuron recording" to mean that separate (or separated) spike trains from multiple cells are available. This situation is sometimes called "multiple simultaneous single-neuron recording" to distinguish it from the earlier use of the term "multineuron recording" which was applied to a single spike train representing all the action potentials in an unknown number of cells near the electrode tip. This earlier usage seems to be fading as technology advances, and the term "multineuron" is less cumbersome than "multiple simultaneous single-neuron".

electrodes and manoeuver each to isolate a cell, or one can insert a larger number of fixed electrodes and simply record from those that happen to provide a decent isolation. The former approach requires considerable time from the experimenter. Furthermore, since, at least in awake animals, isolations generally last for only a short time, as the experimenter isolates cells on more and more electrodes he risks losing the cells isolated at the outset. The latter of the two approaches will often lead to a more stable recording than can be obtained with manoeuverable electrodes, in part because the probes can be allowed to settle within the tissue over a long time. However, the yield of electrodes with single-cell spike trains can be extremely low.

5.1.2 Spike sorting

Spike sorting provides an alternative to physical isolation for multineuron recording. In this approach, the electrode is placed in the neuropil, with no effort being made to isolate a single cell. Instead, the spikes due to many cells are recorded and a data-analytic effort is made to sort them into groups according to their waveforms. Each such group is presumed to represent a single cell.

The attractions to this approach are clear. If repositionable electrodes are used, far less manoeuvering is needed in order to obtain clear spike information. If fixed electrodes are used, the yield of recordable cells from a given array is much increased. Beyond such issues of experimental efficiency, spike sorting approaches can provide data that is extremely difficult to obtain using one-cell-one-electrode approaches. All the cells detected on a single electrode lie within some few tens of microns of the tip, and thus of each other. Such cells are more likely to be functionally and anatomically related than well-separated neurons chosen at random.

Multiple-tip electrodes

Spike sorting can be made easier by use of a multi-tip electrode such as a stereotrode[3] (McNaughton *et al.* 1983) or tetrode (Recce and O'Keefe 1989). This is really a group of electrodes whose tips lie sufficiently close together that an action potential in a single cell generates a spike on more than one of the electrodes. Each electrode will have a different spatial relationship to the source cell, and so experience a slightly different spike waveform. Put together, these "multiple views" of the same action potential provide more information on which to base the sorting of the spikes.

An analogy may be drawn to stereophonic sound recording. Two instruments with similar timbre cannot be distinguished in a monophonic recording. With two microphones, the added spatial information allows us to hear the two different sources. This analogy can only be taken so far, however. In the stereophonic recording the scale of the separation between sources and microphones is very much greater than the scale of the sources and microphones themselves. This is not the

[3]Unfortunately, the term "stereotrode" has come to mean a two-wire electrode. We shall continue in this usage, even though a tetrode, with its four wires, is as much a stereotrode as its two-wire predecessor.

case in the neurophysiological recording. The tip size, the distance from the membrane and the segment of membrane that contributes to each recorded spike are all on the order of 10 microns. As a result, some of the simple sorting strategies suggested by the recorded music analogy are not actually workable.

5.2 Data Collection

The algorithms that appear in this chapter are expected to be of general applicability. They have been developed, however, with reference to data taken in two preparations: parietal cortex of macaque monkey[4] and locust lobula[5]. The methods of data collection are described here.

5.2.1 Monkey

Data have been collected from two adult rhesus monkeys (*Macaca mulatta*). A stainless steel head post, dental acrylic head cap, scleral search coil, and stainless steel recording chamber were surgically implanted in each monkey using standard techniques (Mountcastle *et al.* 1975; Judge *et al.* 1980). During recording, the monkeys sat in a primate chair (custom); the implanted head posts were secured to arms attached to the chairs, thereby immobilizing the animals' heads. Eye-positions were monitored in two dimensions by recording the level of *emf* induced in the scleral coil by two external magnetic fields that oscillated at non-reducible frequencies (Fuchs and Robinson 1966).

The recording chambers in each monkey were set over a craniotomy opened over the posterior parietal cortex. All electrodes were inserted in this area; in most cases they penetrated to the lateral intra-parietal area (LIP). During recording, the animals were awake and performing a "memory-saccade" task in which they remembered the location of a flash of light and then looked towards it on a cue. The details of the task will not be relevant to the present discussion.

In all cases a single tetrode was used for recording (Pezaris *et al.* 1997). The tetrodes were prepared from 13μm-diameter tungsten wire (California Fine Wire), insulated along its entire length. Four strands of wire were twisted together at approximately 1 turn/mm and heated so that the insulation fused over a length of some 10cm. One end of the fused region was cut with sharp scissors so that the tungsten conductor was exposed in all four strands. The impedance of the each conductor interface to physiological saline was measured to be between 0.4 and 0.7 MΩat 1kHz. At the other end the four strands remained separated and were individually stripped of their insulation with a chemical stripper and bonded with conductive paint to electrical connectors.

The tetrode was inserted into a construction of nested metal cannulae which provided mechanical support. The tip of the narrowest, innermost, cannula was sharpened and inserted through the dura

[4]Data collected in collaboration with J. S. Pezaris in Dr. R. A. Andersen's laboratory.
[5]Data collected in collaboration with M. Wehr and J. S. Pezaris in Dr G. Laurent's laboratory.

mater, with minimal penetration of the underlying neural tissue. The tetrode could then be advanced from within this cannula into the brain by a hydraulic microdrive (Frederick Haer Company). A series of tests in another animal revealed that the tetrodes tend to travel straight once inserted into the brain.

The electrical connector at the end of the tetrode was inserted into an amplifier head-stage (custom) with 100x gain. The animal, electrode and head-stage amplifier were all placed within an electromagnetically shielded room. Amplification was in differential mode, with the cannula assembly serving as the reference electrode. Four coaxial cables fed the signals from the head-stage amplifier to the main amplifier (custom) with adjustable gain. Besides enhancing it, the amplifiers also reversed the polarity of the signal. This resulted in the peak amplitude of each spike appearing positive, rather than negative as is the case at the electrode tip. We will maintain this convention throughout the chapter.

The amplified signals were filtered to prevent aliasing and digitized. The digitization rate at the A/D converters (Tucker Davis Technologies AD-2) varied between 12.8 and 20 kHz. The 9-pole Bessel low-pass anti-aliasing filters (Tucker Davis Technologies FT5-4) had corner frequencies of either 6.4 or 10kHz. The data were recorded to digital media and all subsequent operations performed off-line, although sometimes under simulated on-line conditions.

5.2.2 Locust

A difficulty common to almost all data sets used for the development of spike sorting techniques is ignorance of the ground truth. There is no independent way in which to establish the number of distinct cells whose spikes are present in the recording, nor to know which cell fired when. These data, collected from the lobula of the locust, were collected in an attempt to remedy at least one of these concerns. Recordings were carried out with a single tetrode as well as two sharp pipette, intracellular, electrodes. The intracellular electrodes provided incontrovertible information about the firing of up to two cells in the region. Often, one or both of these cells would invoke sizable spikes on the tetrode.

Experiments were carried out *in vivo* on adult female locusts (*Schistocerca americana*). Animals were restrained dorsal side up, the head was immobilized with beeswax, and a watertight beeswax cup was built around the head for saline superfusion. A window was opened in the cuticle of the head capsule between the eyes, and air sacs on the anterior surface of the brain carefully removed. For stability, the oesophagus was sectioned anterior to the brain, and the gut removed through a subsequently ligatured distal abdominal section. The brain was treated with protease (Sigma type, XIV), gently desheathed, and supported with a small metal platform. The head capsule was continuously superfused with oxygenated room-temperature physiological saline (in mM: 140 NaCl, 5 KCl, 5 $CaCl_2$, 4 $NaHCO_3$, 1 $MgCl_2$, 6.3 HEPES, pH 7.0).

Intracellular recordings were made using conventional sharp glass microelectrodes pulled with a horizontal puller (Sutter P-87), filled with 0.5 M KAc, for resistances of 100–300 MΩ. Intracellular recordings were done in bridge mode using an Axoclamp 2A amplifier (Axon Instruments) from the third optic lobe (lobula). Data were collected from 28 single neuron and 6 paired intracellular recordings, all with simultaneous tetrode recordings, from 7 animals. The tetrode was prepared as described above.

All signals were amplified, low-pass filtered at 10 kHz (8-pole analogue Bessel with gain, Brown-Lee Precision), digitized at 50 kHz with 16-bit resolution (Tucker Davis Technologies), and written to compact disc.

5.3 A Generative Model Schema for Extracellular Recording

The cornerstone of our approach to spike sorting will be the identification of an adequate generative model for the observed extracellular recording data. The model has to be powerful enough to account for most of the variability observed in the data, while being simple enough to allow tractable and robust inference. In fact, we will identify not one model, but a **model schema**, that is, a group of models of similar structure. The choice of a particular model from within this schema will be made on a case-by-case basis, using data-driven model selection procedures.

The recorded signal is dominated by the firing of nearby cells; in general the thermal noise in the electrode and noise in the amplification system can be neglected relative to the neural signal. For a 0.5 MΩelectrode at 300K (treated as a purely capacitive impedance) the root-mean-square amplitude of the thermal noise integrated over a 10kHz bandwidth is on the order of 5μV. As we will see (for example, see figure 5.2), this is generally smaller than the recorded amplitudes of neural signals.

We divide the cells into two groups — **foreground** and **background** — of which the second is much the larger. The division is somewhat arbitrary. Roughly, the foreground cells are those whose influence on the recorded signal is large enough that we expect to be able to recognize and sort spikes that arise from them, while the background cells are so distant that their spikes merge into an indistinguishable baseline. In practice, there will be cells whose spikes are occasionally, but not always, distinguishable. We treat these as foreground cells in the model, detecting those spikes that rise out of the background, but neglect the data thus obtained as unreliable.

Thus, we think of the recorded signal as the superposition of spikes from the foreground cells and a single, continuous background **noise process**, which is itself the superposition of all the spikes from the background cells, and other noise sources. Provided that the currents do not total to a sum that is beyond the ohmic limit of the intracellular medium, we expect each of these superpositions to be linear. Measurements made in the locust lobula show that at least in that preparation they

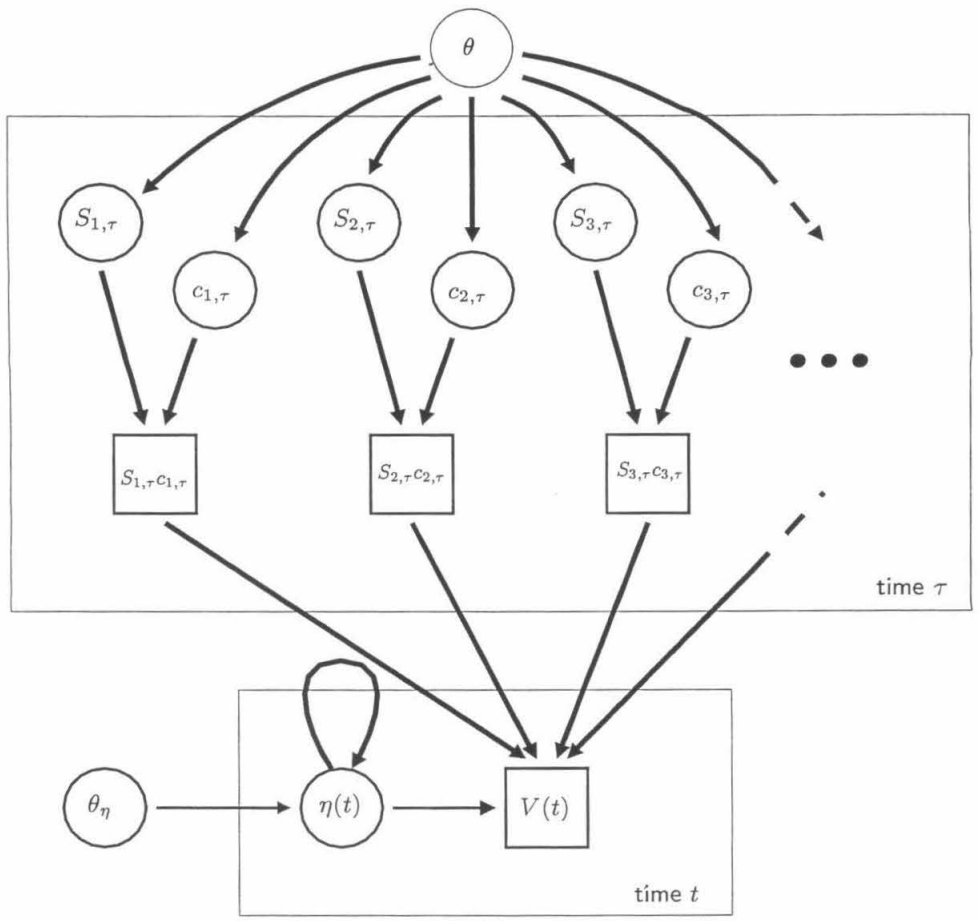


Figure 5.1: Spike sorting model schema

are indeed linear (Wehr *et al.* 1999), however we will take this fact on trust in other preparations.

The model is sketched in figure 5.1. We write $V(t)$ for the recorded potential, the only observed variable in the model. If a multichannel electrode, with tips whose listening spheres overlap (for instance, a tetrode) is used, this is a vector-valued function of time. If the multiple electrode tips are far enough apart that they cannot collect signals from the same cells (more than about 100 microns) we treat each as an independent process and model the recorded voltage traces one at a time. Our model can be written as

$$V(t) = \sum_{\tau} (c_{1,\tau} S_{1,\tau}(t-\tau) + c_{2,\tau} S_{2,\tau}(t-\tau) + \cdots) + \eta(t) \tag{5.1}$$

Here, $c_{m,\tau}$ is an indicator variable that takes the value 1 if the mth foreground cell fires at time τ and 0 otherwise. If cell m fires at τ it adds a deflection of shape $S_{m,\tau}(t-\tau)$ to the recorded potential. The functions $S_{m,\tau}$ have limited support, all of which is around 0. The effect of all the background

neural sources, along with any electrical noise which might be present, is gathered into a single term $\eta(t)$. For the multichannel electrode, both $\eta(\cdot)$ and $S_{m,\tau}(\cdot)$ are vector valued functions.

Note the subscript τ applied to the spike shape S_m. This allows for variability in the shape of the recorded action potential from a single foreground cell, over and above that due to the addition of the background noise. Such variability may arise due to changes in available membrane channels, or due to changes in the membrane surface that participates in each spike. The nature of this intrinsic variability will be discussed at greater length below. In any case, it is of a quite different character to that due to the background: it is potentially different for each cell, it need not be stationary over the course of the spike, and while we will argue below in favour of a Gaussian distribution for the background, this foreground variability is unlikely to be Gaussian in nature. The separation of the distribution of spike shapes from a single cell into these two parts is a critical feature of our approach, and one that was lacking in previous algorithms.

The random variables in our schema, as we have written it, are the background $\eta(t)$, the firing indicators $c_{m,\tau}$ and the spike shapes $S_{m,\tau}$. None of these are directly observed; however, we think of the foreground variables, $c_{m,\tau}$ and $S_{m,\tau}$ as the only latent variables in our model. We can treat $V(t)$ as a random variable, whose distribution conditioned on the latent variables subsumes the noise $\eta(t)$. The parameters of the model can be separated into two groups θ_η which governs the conditional $\mathsf{P}\left(V(t) \mid \{c_{m,\tau}, S_{m,\tau}\}\right)$ and, simply, θ governing the distribution of $S_{m,\tau}$ and $c_{m,\tau}$. Thus, we have factored the underlying distribution so:

$$\mathsf{P}\left(V(t)\right) = \mathsf{P}_{\theta_\eta}\left(V(t) - \textstyle\sum_{\tau,m} c_{m,\tau} S_{m,\tau}(t-\tau)\right) \mathsf{P}_\theta\left(\{c_{m,\tau}, S_{m,\tau}\}\right) \tag{5.2}$$

We have said nothing yet about the nature of the distributions in this factorization. This is why it is a schema and not a full blown model. We will argue that the background process is approximately zero-mean Gaussian, and the distribution of $V(t)$ conditioned on the latent variables will be normal in all of our instances of the schema. The distributions of the $c_{m,\tau}$ and $S_{m,\tau}$ will vary, and indeed, in applications will not always be the same for all foreground cells. Figure 5.1 is drawn as though all of the $c_{m,\tau}$ and $S_{m,\tau}$ were independent. This is merely for clarity in the diagram, we will consider below models for which this is not true.

Our eventual goal within each model is to infer the posterior distribution $\mathsf{P}\left(c_{m,\tau} \mid V(t)\right)$. In practice we will not carry out the marginalization over the parameters implied in that posterior; instead, we will approximate the marginal posterior by the posterior conditioned on estimated values of the parameters $\mathsf{P}\left(c_{m,\tau} \mid V(t), \hat{\theta}, \hat{\theta}_\eta\right)$. The rationale behind this approximation is explained in section section 1.2. In the next few sections we will address the problem of finding these estimates (that is, learning) within the various models that appear in our schema, as well as that of selecting an appropriate model from the schema. After this, we will turn to the question of efficient inference

of the foreground spike occurrence times.

5.4 Learning within the Schema

Separating foreground and background

The foreground and background cells in our model are distinguished entirely on the basis of the amplitudes of their spikes on the recording electrodes. It is therefore reasonable to identify the times of firing of the foreground cells using a simple amplitude threshold. We take the times at which the signal crosses the threshold (the details of which are discussed below) and extract a short segment of the signal, corresponding to the typical length of a spike waveform, around each one. These segments, which we shall refer to on occasion as **events**, contain the foreground spikes. The remaining stretches of signal are presumed to be generated by the background noise process.

This separation of foreground and background allows us to divide our learning procedure into two stages. We examine the stretches of background activity directly to estimate the parameters of the noise. Armed with this estimate, we learn the remaining parameters from the foreground events. This second stage is considerably more straightforward given an independent estimate of the background distribution. Earlier approaches, which did not differentiate between background noise and spike shape variability, did not enjoy this advantage. The choice of distribution and resulting parameter estimation for the noise will be explored in detail below.

Independent components analysis

We consider the problem of estimating the parameters θ which govern the distributions of the latent variables $c_{m,\tau}$ and $S_{m,\tau}$. On the surface, the model (5.2) is quite similar to the generative model which underlies statistical signal separation algorithms such as independent components analysis (ICA) (Jutten and Herault 1991; Comon 1994; Bell and Sejnowski 1995; MacKay 1999) or independent factor analysis (IFA) (Attias 1999). In these algorithms, signals from a group of independent non-Gaussian sources (in the spike sorting case these would be the different cells) are mixed linearly onto multiple channels of output. The output channels may then have noise, usually Gaussian, added. Learning algorithms in such models have been well studied.

Unfortunately, there are significant differences between our model and these ones. We shall note three here: two of these might be surmountable, but the third makes it very difficult to envisage such a solution in the current context.

1. ICA models generally involve exactly as many sources as output channels. If the number of cells is smaller than the number of channels this poses no problem; the algorithm would simply resolve some part of the noise as another "source", which could subsequently be discounted

using some heuristic. However, the number of cells may well be greater than the number of electrode tips that can be practically introduced. In hippocampal recordings, for example, more than 10 cells are often recorded on a single tetrode.

2. Most ICA models imply that the sources are mixed in an instantaneous manner (that is, the output at a point in time depends only on the source signals at that time). In the case of extracellular electrophysiological data, where the electrode tip properties result in filtering of the recorded signal, the mixing cannot be instantaneous. Recently, Attias and Schreiner (1998) have proposed a signal separation algorithm that resolves this difficulty.

3. The most severe difficulty is posed by the extended nature of the sources and recording surfaces. While it would seem sensible to regard each cell as a single source, the different electrode tips will, in fact, lie closest to different parts of the cell membrane, and thus record slightly different spike waveforms. As a result, one cannot treat an isolated foreground spike as a single waveform scaled linearly (or even filtered linearly) onto the multiple recorded channels. The spike waveform must itself be regarded as a fundamentally multichannel entity. This prevents the application of blind source separation techniques to spike sorting in many preparations, notably in neocortical recordings.

If we cannot use these well-established signal processing techniques, can we hope to solve the problem? In fact, ICA-like techniques fail to exploit the significant amount of prior knowledge available about the neural signal. Nowhere in the generative model for ICA, for example, is it acknowledged that a single source signal will always be a chain of approximately stereotypical pulses. It is this repetitive nature of the signal that we will exploit to solve the problem.

Before leaving this point, we make two additional observations. First, consider the following scheme for application of ICA. We regard each source as producing a train of delta-functions, with the spike waveform on each channel, however it is produced, appearing as the impulse response of a fictitious linear filter. The delta-function trains are convolved with their corresponding filters and summed (along with noise) to produce the recorded signal. The filtering and summing represent the mixing stage of a **dynamic components analysis** (DCA) model (Attias and Schreiner 1998). This treatment would seem to restore our faith in the applicability of an ICA-like algorithm. Even better, it would indeed incorporate our prior belief in the pulsatile nature of each source. The difficulty with this approach lies in the presence of spike waveform variability in the data. Since, in this scheme, the waveform information is treated as part of the mixing process rather than as a source signal, we would require a variable mixing process. Such variability cannot be handled within the DCA framework.

Second, it should be borne in mind that there may well be preparations in which ICA-like algorithms are applicable to spike sorting. For example, the form of ICA suggested in the preceding

paragraph might be successful in cases where there is little or no spike shape variability. Another example is provided by Brown *et al.* (1998) who have reported success in optical recordings of voltage-sensitive-die-treated *Tritonia* tissue. In this example, the recordings are sufficiently slowly sampled that the spread of signal across the membrane is effectively instantaneous (Brown, personal communication). As a result, the spike waveforms recorded on different photodetectors may indeed be linearly scaled versions of a single waveform. Furthermore, the optical nature of the recording ensures that the signal mixing at the detector is linear and instantaneous.

Clustering algorithms

Our approach to learning the waveform parameters is based on two observations. First, all the spikes recorded from a single cell are expected to be roughly similar. Indeed, we will specify the exact nature of the variability that we expect, by specifying the distribution of $S_{m,\tau}$ within the generative model schema. Second, the probability that two foreground cells will fire so close together in time that their spike waveforms overlap in the recorded signal is relatively low. As a result, most of the foreground events gathered by the application of our threshold represent only a single spike waveform. Thus we might expect to learn the shapes of the underlying waveforms (and the distributions of such shapes) by **clustering** these foreground events.

Consistent with our probabilistic viewpoint, we shall adopt a generative-model-based approach to clustering, as was outlined in chapter 2. To do this we need to transform the model of (5.2) into a suitable form.

Whereas (5.2) provides a model of the continuous waveform $V(t)$, we now desire a model that describes the set of extracted events, $\{V_i\}$. Each V_i is a vector of samples drawn from all of the channels of $V(t)$ around the time τ_i at which the ith event occurs. At all times τ other than the τ_i we assume that no foreground cell fired and so $c_{m,\tau} = 0$ for all m. We will employ the labels $c_{m,i}$ and $S_{m,i}$ for the latent variables at the times τ_i, in place of the more cumbersome forms such as c_{m,τ_i}.

The vectors V_i are taken to be conditionally independent, given the values of the latent variables $c_{m,i}$ and $S_{m,i}$. In other words, we assume that the separation between events is always greater than the correlation-time of the background noise process. The distribution of the ith vector is described by a **mixture** density, with one component for each possible value of the indicators $c_{m,i}, m = 1 \ldots M$. Let us consider these components one by one.

1. All $c_{m,i} = 0$. This implies that the threshold was reached by the background process alone without a foreground spike. In this case the density of the vector V_i is exactly that of the background noise, expressed as a vector density, rather than as a continuous process density.

We will introduce a new indicator variable $z_{\emptyset,i}$ to indicate this condition, and write

$$\mathsf{P}\left(V_i \mid z_{\emptyset,i}=1\right)=\mathsf{P}_{\theta_\eta}\left(V_i\right)=\mathsf{P}_{\emptyset}\left(V_i\right) \tag{5.3}$$

2. Only one of the $c_{m,i}=1$. Such events will make up the majority of those detected. We use indicators $z_{m,i}, m=1 \ldots M$ to represent each of these states (the $z_{m,i}$ are exactly the same as the corresponding $c_{m,i}$, though only in this condition). The density of the event vector is then

$$\mathsf{P}\left(V_i \mid z_{m,i}=1\right)=\int dS_{m,i}\, \mathsf{P}_{\theta_\eta}\left(V_i-S_{m,i}\right) \mathsf{P}_{\theta}\left(S_{m,i} \mid \{S_{n,j}, c_{n,j}: j<i\}, c_{m,i}=1\right) \tag{5.4}$$

Notice the conditioning of $S_{m,i}$ which depends only on the preceding latent variables to enforce causality. We will abbreviate this set of latent variables at all times earlier than τ_i by $\lambda_{<i}$ and write this density as $\mathsf{P}_m\left(V_i \mid \lambda_{<i}\right)$.

3. More than one $c_{m,i}=1$. In this case two foreground cells fired at close enough times that the threshold was only crossed once by the compound waveform. We expect such events to occur rarely *and will not explicitly model them as overlapped events at this stage.* Instead, we treat all such waveforms as "outliers", and model them by a single, uniform density (see section 2.7.1). We introduce a latent variable $z_{\not\emptyset,i}$ to indicate this condition. The corresponding density is simply

$$\mathsf{P}\left(V_i \mid z_{\not\emptyset,i}=1\right)=\begin{cases} \frac{1}{\|A\|} & \text{if} \quad V_i \in A \\ 0 & \text{if} \quad V_i \notin A \end{cases} \tag{5.5}$$

with A some region of the vector space of V_i and $\|A\|$ its volume. We will write this density as $\mathsf{P}_{\not\emptyset}\left(V_i\right)$.

The complete model for the ith vector is thus

$$\begin{aligned} \mathsf{P}\left(V_i\right) &= \mathsf{P}_\theta\left(z_{\emptyset,i}=1 \mid \lambda_{<i}\right) \mathsf{P}_{\emptyset}\left(V_i\right) \\ &+ \mathsf{P}_\theta\left(z_{\not\emptyset,i}=1 \mid \lambda_{<i}\right) \mathsf{P}_{\not\emptyset}\left(V_i\right) \\ &+ \sum_{m=1}^{M} \mathsf{P}_\theta\left(z_{m,i}=1 \mid \lambda_{<i}\right) \mathsf{P}_m\left(V_i \mid \lambda_{<i}\right) \end{aligned} \tag{5.6}$$

Once again, the distribution of the indicator variables is conditioned only on earlier latent variables so as to preserve causality in the model.

The latent indicator variables $z_{m,i}$, $m=\emptyset, \not\emptyset, 1 \ldots M$ are mutually exclusive: exactly one of them takes the value 1 for any i, while all of the rest are 0. As such, they closely resemble the mixture latent variables of chapter 2. In many of the models we will discuss, the indicators for each event will be drawn independently from a fixed distribution. In this case, the model is exactly a mixture

model. Even where this is not exactly true, however, we shall call this the **mixture form** of the generative model. Fitting such a model is what we will mean when we claim to be performing a parametric clustering of the spike events.

It is worthwhile to consider the impact of our choice not to model the overlapped spike events explicitly, but rather to sweep them into a single outlier distribution. Is it likely that this inaccuracy in the event model (5.6) will lead to estimates of the parameters that do not carry over to the true continuous signal model (5.2)? The mistreatment of overlaps poses two distinct dangers to accurate parameter estimation. The first is that some overlaps will be incorrectly interpreted as single spikes, and thus bias the estimate of the spike shape distribution of the misidentified cell. This possibility is slim. Overlaps need to be fortuitously exact to look anything like single spike waveforms. Most likely, they will fall quite far from any single cell cluster and be easily recognized as outliers. Furthermore, the use of a uniform outlier distribution minimizes the expected bias in estimates of the mean spike shapes of each cell (robustness to outliers in the fitting of mixture models is discussed in section 2.7.1). The second danger arises from the fact that the occurrence of an overlap "removes" an event which would otherwise contribute to the parameter estimation. For models in which the latent variables associated with each event are independent of all others (these are the true mixture models) this effect will be negligible, provided that the probability of overlap is small and independent of the latent variable values. However, for models in which the spike shape and probability of firing for each cell depend on its history, this can pose a real problem. We shall address it when we discuss such models.

For the sake of the reader familiar with previous spike sorting techniques it is worth emphasizing here a point that has appeared before, and will be addressed again in section 5.14. In the present approach to the problem, unlike in many (though not all) others, the clustering stage is a preliminary to the inference of spike arrival times. We use it as a device to learn the parameters θ that govern the distributions of $c_{m,\tau}$ and $S_{m,\tau}$. The actual inference of the variables $c_{m,\tau}$ is done within the more accurate superposition model (5.2), without the imposition of an artificial threshold, nor the rejection of overlapped spikes.

5.5 Event Detection

Our first step in the process of learning the model parameters is to identify the times at which foreground cells fired by comparing the recorded signal to a threshold amplitude.

A short segment of data recorded from the neocortex of a macaque monkey using a tetrode is shown in figure 5.2A (the four traces show the simultaneously recorded signals on the four wires). Large amplitude spikes are clearly superimposed on a lower amplitude background process. However, it is clear that the comparison of this raw signal to a fixed threshold will not achieve the separation

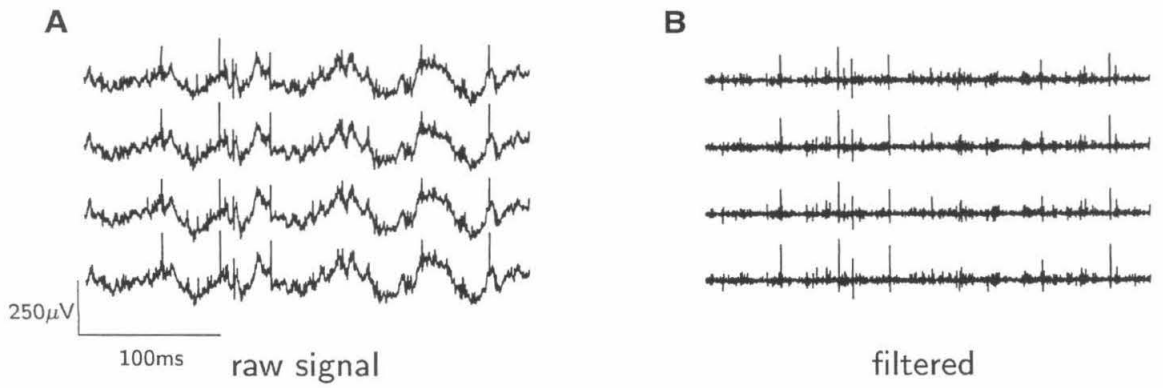


Figure 5.2: A sample extracellular recording.

we desire; the signal exhibits a low frequency baseline modulation with an amplitude comparable to that of the largest foreground spikes. This low-frequency field potential signal may be of considerable interest in itself, however the frequencies involved are too low to have an influence on the shapes of the relatively short spike waveforms and so it can safely be removed for the purposes of spike sorting. Figure 5.2B shows the same segment of data after it has been digitally high-pass filtered. The filter cutoff is chosen at the lowest frequency that can contribute to the foreground spike shapes, based on the length of those spikes. For neocortical recordings of the type shown in figure 5.2 the spike length is not longer than 2 milliseconds, implying a filter cutoff of at least 500Hz.

We wish to choose a threshold which allows us to identify the spikes that rise above the background process. To do this we need to know the statistics of the background, but, of course, we cannot measure these until we have separated background from foreground. We shall set the threshold in terms of the variance of the entire signal, foreground and background. In doing so, we assume that foreground spikes are rare enough that this measurement is dominated by the background. This may not always be true: if we record 4 foreground cells, all firing at about 50Hz, there would be a total of 200 spikes in one second of recording. As the large amplitude peak of each foreground spike can last up to half a millisecond, this would mean that one-tenth of the recording has large amplitude foreground contributions – enough to affect the background variance estimate. As a result, a certain degree of user intervention is useful in setting the threshold level. A typical choice of threshold is 3–5 times the root-mean-square value of the high-pass filtered signal.

Spike waveforms are generally biphasic pulses. The strongest currents during an action potential are associated with the influx of sodium that initiates the firing; as a result, the first phase is almost always the larger. The sodium current flows into the cell, away from the electrode tip. Thus, this first phase is negative on the electrode. Under the polarity convention adopted in this chapter (introduced in section 5.2) it will appear positive in our recordings. In order to reduce the probability

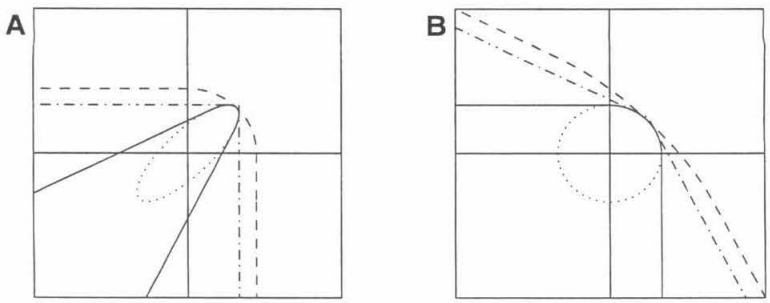


Figure 5.3: Event detection thresholds

of false triggers, and also to ensure that each spike causes only one threshold crossing, we apply the threshold in a one-sided manner, accepting only crossings where the recorded potential rises above the positive threshold value.

It is not obvious how to apply the threshold to multichannel data. We shall consider three schemes here, and it will be useful to compare them graphically. This is done for a hypothetical two-channel signal in figure 5.3. The axes in panel A represent the amplitude of the signal on the two channels: each sample of the signal is represented by a point in this plane. The thresholding schemes will be described by boundaries in the plane which separate regions where the signal is below the threshold from regions where it is above. The various lines in this panel, and the nature of panel B, will be described below.

The most commonly employed approach to multichannel data is to accept an event whenever any one channel rises above a scalar threshold. The acceptance boundary of such a threshold for the two-channel example is represented by the dash-dotted line in figure 5.3A. The signal has crossed this simple threshold if the point falls to the right of or above the line. We shall call this a **rectangular** threshold.

An alternative approach would be to threshold the total instantaneous power of the signal, that is, the sum of squares of the amplitudes on the various channels. Given the unidirectional nature of the spike peaks, we choose to half-wave rectify the signal before squaring. The resultant threshold, which we call **circular**, is shown by the dashed line.

The dotted ellipse in figure 5.3A shows a covariance contour for the background distribution, that is, a line drawn at a constant distance from 0 in the Mahanalobis metric defined by the distribution's covariance. The ellipse is drawn as though the background on the two channels is positively correlated. In fact, this is the overwhelmingly dominant case in experimental data. It is reasonable that electrode tips close enough to share spikes from the same foreground cells will also share background spikes.

A comparison between this elliptical noise contour and both of the threshold boundaries described

so far reveals the weakness in these approaches. Many points above and to the right of the ellipse are unlikely to arise purely from the background process, and yet are not detected as foreground events. A more sensible approach would seem to be to shape the boundary to match the contour of the second moment of the noise distribution. This is conceptually easiest in the noise-sphered space, which is obtained by an instantaneous linear transformation on the signal (if the noise covariance is Σ the sphering matrix is $\Sigma^{-1/2}$). This space is represented in figure 5.3B. The noise covariance matrix is now, by construction, spherical. The rectangular and circular thresholds are shown in the dot-dashed and dashed lines, as before. The solid line represents a threshold boundary constructed in the same way as the circular threshold, but now in the sphered space; the solid line in panel A shows the shape of this boundary in the original space. We refer to this as the **elliptical** threshold.

By construction, the elliptical threshold matches the covariance contour of the noise. If that noise is Gaussian distributed, this curve is also an iso-probability contour, so that the probability of the noise alone exceeding the threshold is independent of the direction (in the space of figure 5.3A) of the signal.

5.6 The Background Process

Once the times of the foreground events have been identified, we explore the statistics of the signal during the periods between these events, with the goal of characterizing the background process. In the first instance, we are interested in the distribution $\mathsf{P}_{\theta_\eta}(V_i)$ which expresses the background as a vector-output process. This distribution will be of critical importance in what follows: not only is it the distribution of the noise (5.3), it also makes a significant and common contribution to the distribution of spike waveforms recorded from each cell (5.4).

We estimate the distribution of the V_i directly, by sampling the background process at times when no foreground spike is present. The spikes extend for some time before and after the times of the threshold crossings; thus, we need to extract vectors away from these points so as not to overlap the foreground waveforms. For the data shown here, no samples were taken within 1.6ms of each crossing. The remaining signal is then broken up into segments whose length matches the duration of a foreground spike. Each such segment represents a single vector sample of the background process. We study the distribution of the ensemble of these vectors along the principal components.

Each of the columns of panels in figure 5.4 shows the density of the loadings of the noise vectors on a selection of the ensemble principal components, for an example macaque tetrode recording. In each column the upper and lower panels show the same data; the upper panel shows the density directly, while the lower panel shows the log density, thereby revealing the details of the tails of the distributions. The rank of the component on which the loadings are taken is indicated below the column. The dots represent the density histogram of the observed vectors. The continuous line

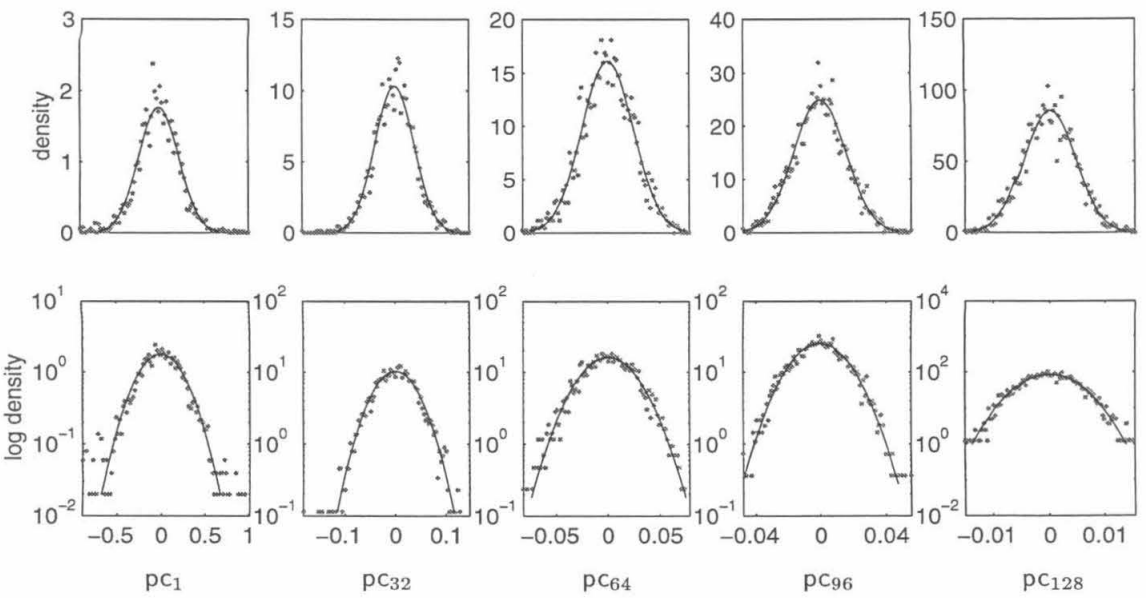


Figure 5.4: The distribution of background noise

represents a Gaussian density with the same variance as that of the observed loadings. It is clear that a Gaussian model for the background process is reasonable, although a slight excess in kurtosis is evident in the first components.

In the rest of this chapter we shall take the background to be Gaussian distributed. While figure 5.4 suggests that this is reasonably well supported by the data, it is not exactly true in all cases (Fee *et al.* 1996b). Our choice is driven by two observations. First, the Gaussian model considerably reduces the computational demands of the various approaches that we will discuss, and is quite important for efficient separation of overlapped spike waveforms. Second, we will introduce separate models for intrinsic spike variability that will be non-Gaussian. Thus, it is possible for some non-Gaussian background noise to be subsumed by these models. In situations where computational cost is no object, or where the data exhibit extreme departures from normality, an alternative distribution may be used for the background. Most of the generative models to be discussed will carry through with little modification. The largest cost will come in the final stages of spike-time inference, where the filtering scheme we adopt is critically dependent on Gaussian noise.

A zero-mean Gaussian density is entirely specified by its covariance matrix. Since the background process is stationary with respect to the duration of the spike waveform — that is, the statistics of the background are the same at each point along the spike — this covariance matrix may be constrained to have Töplitz (diagonally striped) structure. Thus, the only parameters of the noise distribution are given by the autocorrelation function of the background.

While the noise is almost certain be stationary on the time-scale of a single spike waveform,

it may well be appreciably non-stationary on time-scales of hundreds of milliseconds or more. In particular, as stimulus conditions change, the rate of firing of both foreground and background cells will change, quite probably in a correlated fashion. Thus, by sampling the background far from the locations of the foreground spikes we run the risk of measuring a background quite different from that which actually affects the distribution of event waveforms.

We can avoid this pitfall by biasing the sample of background vectors so that most are drawn close to, though not overlapping with, the foreground spikes. One simple procedure to ensure this it to sample a fixed offset from each foreground spike (after making sure that this would not result in an overlap with a different event). Another is to sample exactly in-between each pair of adjacent events (again making sure that the pair is far enough apart that this will not cause an overlap). Furthermore, in extended recording we can re-estimate the noise continuously, leading to an adaptive estimate that can track non-stationarities on the time-scale of seconds.

5.7 Foreground Events

Models within the mixture schema (5.6) describe a multivariate density for foreground events. In this section we shall examine the procedure by which a vector representation in constructed for each foreground spike. We proceed in two steps: in the first the vector elements are sampled directly from the voltage trace yielding relatively high-dimensional vectors; in the second we use a low-rank linear transform to reduce this dimensionality through a technique similar to principal components analysis.

5.7.1 Extraction and alignment

In the first stage, each element of the event vector will be a sample drawn from the recorded voltage trace near the time of the corresponding threshold crossing. The extracted samples will be separated by the Nyquist sampling period derived from the frequency content of the signal, which in turn is controlled by an analogue anti-aliasing filter. We order the samples forward in time, with all of the samples from the first channel appearing together, followed by the samples from the second channel if there is one, and so forth. In multichannel recordings, the corresponding samples on each channel will always be simultaneous.

A common approach to selecting the vector coordinates is to copy a fixed number of values from the digitized recording before and after the sample at which the threshold was crossed. This, however, does not ensure that the samples are taken at the same time relative to the underlying spike waveform. This jitter in sampling introduces artificial variability in the extracted set of vectors as illustrated in figure 5.5. Panel A shows one channel of a small number of recorded spike waveforms, all originating from a single cell. The samples extracted from the waveforms are shown by the dots;

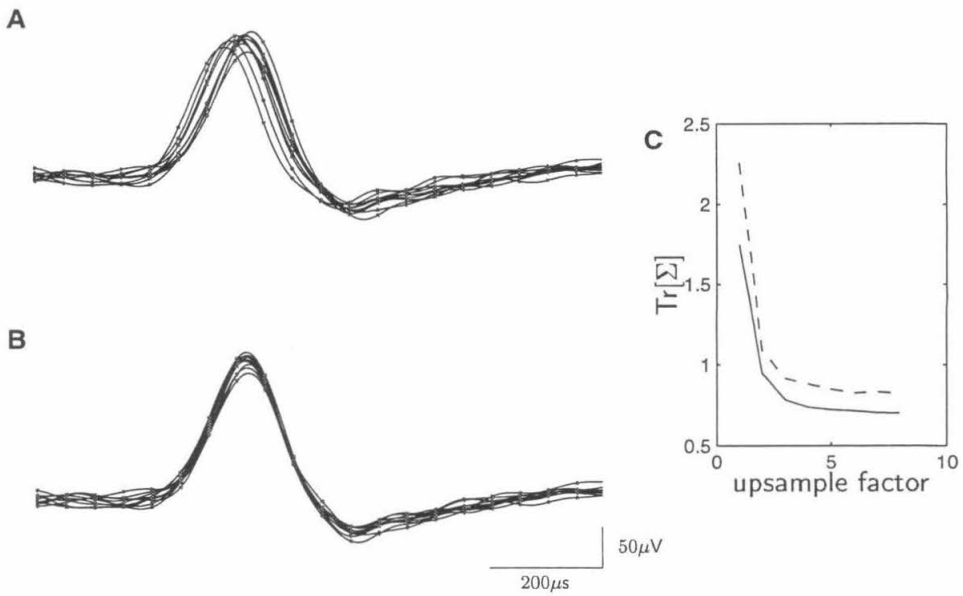


Figure 5.5: Alignment of spike waveforms.

the solid lines beneath show the Fourier reconstruction of the underlying signal, assuming the there was no power above half the sampling frequency. The variation in alignment of the underlying waveform is evident, and results in "noise" in the samples that can reach up to half of the spike amplitude. Furthermore, if the temporal jitter of the alignment is uniformly distributed within one sample interval, this apparent "noise" will also be almost uniform (its exact shape is set by the derivative of the underlying spike shape), making it difficult to model. Fortunately it can be mostly eliminated.

There are two sources of jitter. For the sake of argument, let us assume that the underlying spike waveform being measured has no intrinsic variability. In that case, there is a well defined time at which the waveform crosses the threshold, and we would like to align the samples in the event vector with this time. The first source of jitter is the background noise, the addition of which to the recorded spike waveform will result in that waveform crossing the threshold at a slightly different point from our reference time. The second source comes from the sampling of the waveform, which is unlikely to be aligned with the spike and thus the crossing-time will probably fall between two samples, rather than on one.

The jitter and its associated artifact can be reduced considerably by some amount of signal processing. The effect of the background on alignment can be reduced by choosing to align to a composite landmark, rather than a single sample level. We will use the "centre of mass" of the peak of the waveform, that is, the quantity $\tau_c = \int dt\ tS(t) / \int dt\ S(t)$ with the integrals limited to

the peak region of the spike waveform $S(t)$. This is estimated from sampled data S_n by a form similar to $\hat{\tau}_c = \sum t_n S_n / \sum S_n$, with the range of the sum limited to samples near the peak of the waveform. The sum over samples reduces the effect of the background on the alignment time. Temporal correlations in the background will interfere with this reduction, and so it is preferable to use the background-whitened signal (see section 5.6).

We can eliminate the sample-alignment jitter by resampling the waveform to align with the estimated centre of mass exactly, even if that estimate falls off the original sample grid. This resampling is achieved by interpolation, either with cubic splines, or "exactly" using Fourier techniques. The cubic spline interpolation is straightforward and will not be described here. The Fourier technique proceeds as follows. Conceptually, we find the discrete Fourier transform of the sampled waveform and treat the coefficients thus obtained as the coefficients of a finite Fourier series. Provided that the original signal was sampled at or above the Nyquist sampling frequency for its bandwidth, this series sums to the original, continuous signal (barring boundary effects). We draw new samples from this exact interpolant. The Fourier process described is equivalent to a kernel smoothing of the discrete sequence treated as a sum of delta-functions, where a sinc-function is used for the kernel. As might be expected from a sinc-function kernel, the interpolant will tend to ring near the boundaries of the interpolated segment; it is important, therefore, to use a segment sufficiently long that the region of interest does not fall critically close to a boundary.

The selection procedure for the samples to be used in calculation of the centre of mass has not yet been discussed. It proceeds as follows. First, the maximum sample within a short time after the detected threshold crossing is identified. In the region of this sample the waveform is "upsampled" by resampling from the interpolant at a higher rate. The region used extends sufficiently far on each side of the maximum to encompass the entire first peak of the spike waveform. Next the contiguous region of samples that encompassed the maximum and lies above a threshold value is identified. This threshold is chosen lower than the trigger threshold, so as to ensure that a large number of samples will fall above it. The threshold-based centre of mass calculation is preferred to use of a fixed number of samples around the maximum because it avoids the bias towards the centre of selected interval that is inherent in the latter approach.

The centre of mass is calculated by,

$$\hat{\tau}_c = \frac{\sum t_n (S_n - a)}{\sum (S_n - a)} \tag{5.7}$$

where the sums range over the contiguous samples S_n of the upsampled waveform that lie above the threshold a. The subtraction of the threshold from the sample values ensures that samples near the boundary of the selected region have little effect on the estimate, thereby protecting it from noise-driven variations in that boundary. A fixed number of samples, sufficient to encompass the

extent of the spike waveform, spaced by the Nyquist period and aligned with $\hat{\tau}_c$, are extracted from *each channel* of the recording and arranged into the event vector.

The results of this alignment procedure are shown in figure 5.5B. Clearly, the apparent noise has been reduced considerably. Given a group of waveforms known to originate from the same cell, we can measure the effect of the alignment procedure by calculating the trace of the covariance matrix of the spike waveforms after alignment. These values of are shown in figure 5.5C for a number of different algorithms. The dashed line represents alignment to the threshold crossing, while the solid line represents alignment to centre of mass. Furthermore, each reference point was extracted using varying degrees of upsampling (that is, interpolation). Two observations are clear: both techniques improve at about the same rate as finer upsampling is employed; and furthermore, the centre of mass reference point provides a constant benefit over the threshold crossing at all upsampling factors. The two different sources of jitter, along with the effectiveness of the proposed techniques in overcoming them, are evident.

5.7.2 Dimensionality reduction

The number of samples that goes into each vector might be quite large. For tetrode recordings in monkey neocortex, for example, a 10kHz signal bandwidth is suitable, spikes last over a millisecond in time, and so the vectors will contain more than 80 elements. Such large vectors lead to two difficulties. One is purely computational: calculations on lower-dimensional objects would be much faster. This is a particularly relevant concern for the case of on-line spike sorting. The second is perhaps more serious. As the dimensionality of the modeled space grows so does the number of parameters, and so the quantity of data needed to obtain good estimates can become very large. With insufficient data, the danger of over-fitting is considerable.

Fortunately, it is possible to reduce the dimensionality of the space efficiently and without any loss of useful information. In this discussion we will only consider linear dimensionality-reducing transforms. That is, we will seek a rectangular matrix, R, by which we can multiply the data vectors, V_i so as to obtain the lower-dimensional products $x_i = RV_i$. The x_i must retain as far as possible those features of the data set V_i which are essential to clustering.

Hand-picked features

Perhaps the most commonly adopted approach is to derive from each waveform a small group of features which might *a priori* be expected to carry much of the relevant information. For a multi-channel electrode, the most natural such features are the peak potentials attained on each recording surface. For tetrodes, then, each x_i becomes a point in $\mathbb{R}^4$. Figure 5.6 shows the events extracted from one tetrode recording, projected into this basis. The 4-dimensional space is represented by the 6 possible 2-dimensional axial projections. Thus, in the topmost panel the peak value on channel 2

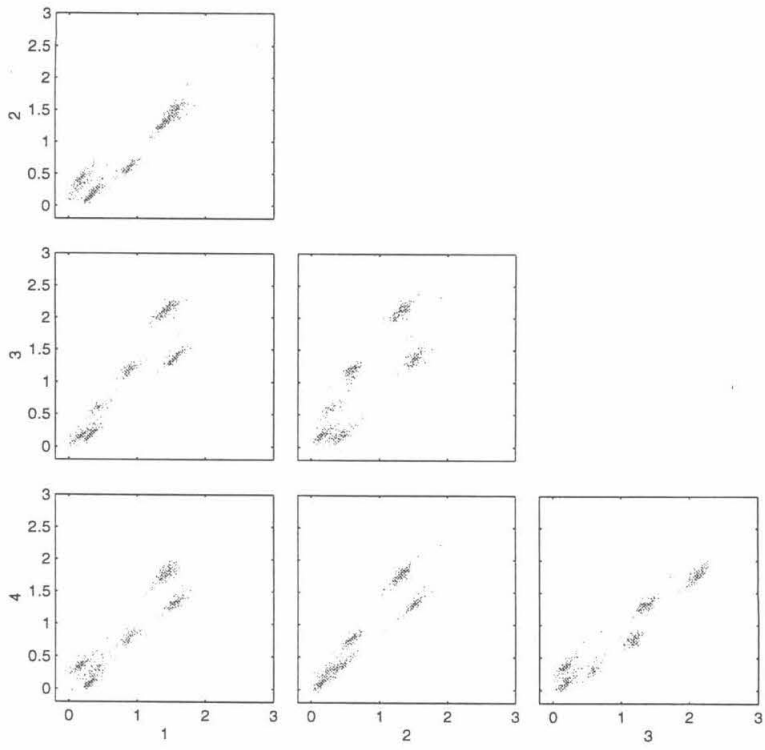


Figure 5.6: Events represented by peak voltage on four channels.

is plotted against the peak value on channel 1; in the panel immediately below, the peak on channel 3 is plotted against the peak on channel 1; to the right of this panel, the peak on channel 3 is plotted against that on channel 2 and so forth. A similar representation will be used many times in the following pages. While in figure 5.6 the numbers that appear below and to the left of the panels represent channels numbers, in many of the later diagrams they will indicate arbitrary basis vectors in the space of the events V_i.

Six distinct clusters are visible to the observed in the data of figure 5.6. However, the three closest to the origin, containing relatively low-amplitude spikes, are somewhat difficult to distinguish. Nevertheless, in this case, fitting a mixture model in this restricted subspace is likely to be quite effective.

In many cases we can reasonably define the "peak" on a given channel to be the value of a particular sample in the suitably aligned event waveform. In this case, the feature subspace can be obtained by a linear projection with a matrix R that contains mostly 0s, with a single 1 per row selecting the appropriate sample. This was the definition used to generate figure 5.6. Some other features in general use (such as the peak-to-trough amplitude) may also be extracted by linear projections. However, others, such as the width of the waveform peak, can not.

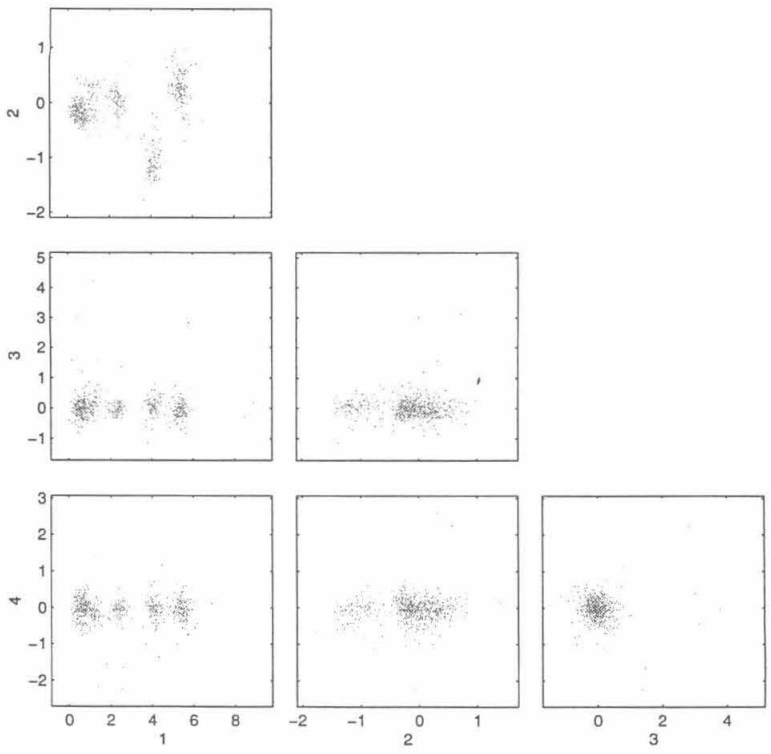


Figure 5.7: Events represented in the principal component subspace.

The attraction of linear dimensionality reduction is not simply a matter of algorithmic simplicity. A key feature of the model schema of section 5.3 is the single, consistent model for the contribution of the background process to the variability in the recorded waveforms. This simple fact remains true under any linear transformation of the space, indeed, the background model remains Gaussian to the extent described in section 5.6. Under a non-linear transformation such as spike-width, not only do we lose the Gaussian representation for the background contribution, but the contribution to the variability of this feature will be different for different underlying waveforms. This would violate the mixture schema of (5.6), making the task of statistical modeling far more difficult.

Principal components analysis

A linear approach, commonly used in situations such as this, is known as principal components analysis (PCA). PCA selects a subspace spanned by a small number of eigenvectors of the observed (total) covariance matrix

$$\Sigma_T = \frac{1}{N} \sum (V_i - \bar{V})(V_i - \bar{V})^T \tag{5.8}$$

The eigenvectors chosen are those with the largest associated eigenvalues. The resultant projection has the property that, among all the linear projections of the same rank, it retains the greatest

amount of the original data variance. We expect the PCA projection to be useful because clustering is likely to be easiest in those directions in which the data are well spread out. However, it may not be the optimal projection.

Figure 5.7 shows the projection into the first four principal components (in order) of the same data set as was shown in figure 5.6. In this case, our expectation that PCA will improve the separation of the clusters is belied. Where six different groups could be made out in figure 5.6, only four can be clearly resolved here. Furthermore, the clusters are separated in only the first two dimensions. This experience is not uncommon when handling tetrode data.

The optimal linear projection

It is well known that we can obtain the optimal linear projection *a posteriori*, that is, given knowledge about which cell each spike originated from. The procedure, known as linear discriminant analysis (LDA), selects the linear projection in which the separability of the clusters is maximized, that is, the ratio of the average distance between the clusters to the average spread of the data within each cluster is greatest.

We introduce two new covariance or scatter matrices, the between-class scatter Σ_B and the within-class scatter Σ_W. Let us identify the vectors that fall in the mth class by $V_{m,i}$, and write the mean of all such vectors as $\bar{V}_m$, with $\bar{V}$ being the overall mean as before. The number of vectors in the mth class will be written N_m, and the fraction of the total that this number represents, π_m (these fractions being equivalent to the mixing probabilities of a mixture model). The two new scatter matrices are defined thus

$$\Sigma_B = \sum_m \pi_m (\bar{V}_m - \bar{V})(\bar{V}_m - \bar{V})^T \tag{5.9}$$

$$\Sigma_W = \sum_m \pi_m \frac{1}{N_m} \sum_i (V_{m,i} - \bar{V}_m)(V_{m,i} - \bar{V}_m)^T \tag{5.10}$$

The symmetrized ratio we wish to see maximized in the projected space is $\Sigma_W^{-1/2} \Sigma_B \Sigma_W^{-1/2}$. Just as in PCA, we find the eigendecomposition of the corresponding matrix in the higher dimensional space and then project onto the space formed by the leading few eigenvectors.

It would appear that we can obtain little advantage from the discriminant approach, as the scatter matrices given by (5.9) and (5.10) cannot be calculated without access to the very information that we seek. However, it is possible to view the LDA procedure in a different light. Consider a transformation of the vectors $V_{i,m}$ by the matrix $\Sigma_W^{-1/2}$ to obtain new vectors $\tilde{V}_{i,m}$. Direct substitution into (5.10) reveals that in this transformed space, the within-class scatter, $\tilde{\Sigma}_W$, is the identity matrix. We shall refer to this as the **class-whitened** space. To now perform LDA, we need only maximize the between-class scatter $\tilde{\Sigma}_B$. It is straightforward to see that the subspace thus identified

is exactly the same as would be obtained by discriminant analysis in the original space. Indeed, this whiten-and-diagonalize algorithm is a common implementation for LDA (see, for example, Ripley (1996)). We can go one step further if we note that the total covariance in the class-whitened space is simply $\tilde{\Sigma}_T = \tilde{\Sigma}_B + \tilde{\Sigma}_W = \tilde{\Sigma}_B + I$. Thus the overall scatter matrix is diagonalized in the same basis as the between-class scatter matrix. LDA is equivalent to PCA in the class-whitened space.

The key point of this analysis is the simple relationship $\Sigma_T = \Sigma_B + \Sigma_W$. This implies that we need only one of the classification-dependent scatter matrices in order to find the optimal discriminant subspace, the other can be derived from the overall variance of the data. We do not know either of these matrices, but we do have an (under)estimate of the average within-class scatter Σ_W, provided by the direct measurement of the background. Thus, we can find a basis quite similar to the optimal LDA basis by taking the principal components in the **noise-whitened** vector space. An example of this procedure will appear in figure 5.8.

Robust principal component analysis

Inevitably, some events within the ensemble will fall far from any clusters. These are mostly the events that contain overlapped spikes as described in section 5.4. Since the data covariance matrix weights points by the square of their distance from the mean, principal components calculated from the entire data set are particularly sensitive to the number and location of these outliers. It is important, therefore, to obtain the components in a manner that is robust to outliers.

We will adopt an approach to robustness similar to that discussed in the context of the clustering algorithms in section 5.4. We can view the PCA procedure as fitting a multivariate Gaussian distribution to the data and then selecting a projection on the basis of the fit distribution. This relationship between PCA and Gaussian modeling has been explored quite extensively in the recent past (Tipping and Bishop 1997; Roweis 1998). Following the argument made during the discussion of the impact of outliers on clustering, we replace the single Gaussian by a mixture of a Gaussian and a uniform density (the limits of the uniform density being set by the maximum extent of the data). Recall from the discussion of section 5.4, that the introduction of the uniform component will not, on average, bias the estimates of the eigenvectors of the covariance of the Gaussian component. It is these eigenvectors which represent the principal component basis.

Figure 5.8 shows the subspace obtained when this robust PCA is applied in the noise-whitened space. The six clusters are now very much in evidence, and comparison with figure 5.6 suggests that they are better separated. Figure 5.9 shows the data set projected into the first four dimensions of the optimal linear discriminant space, calculated *a posteriori* from a mixture fit to these data. Clearly, for this recording, the noise-whitened robust PCA technique has identified a subspace remarkably close to the optimal one.

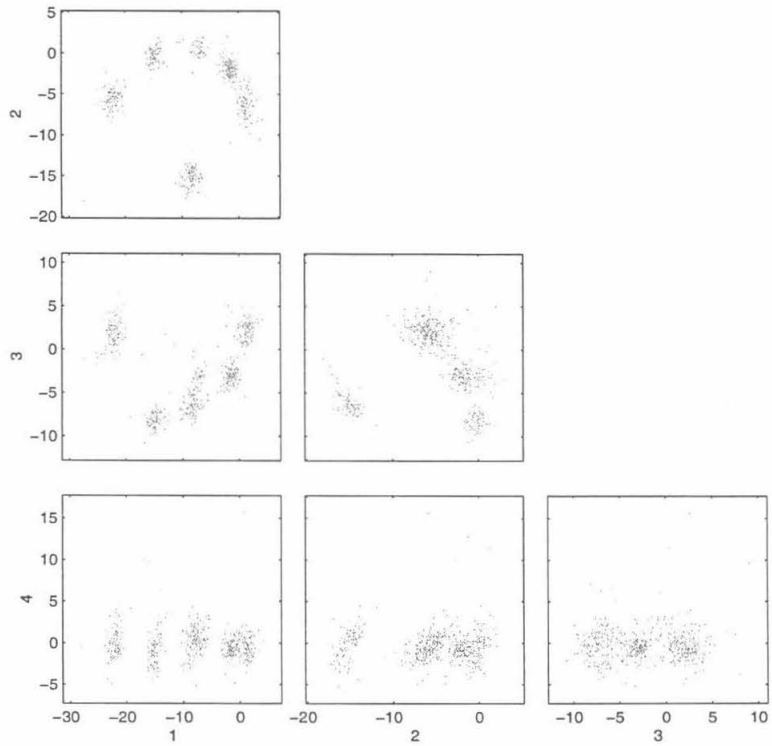


Figure 5.8: Events represented in the noise-whitened robust PCA subspace.

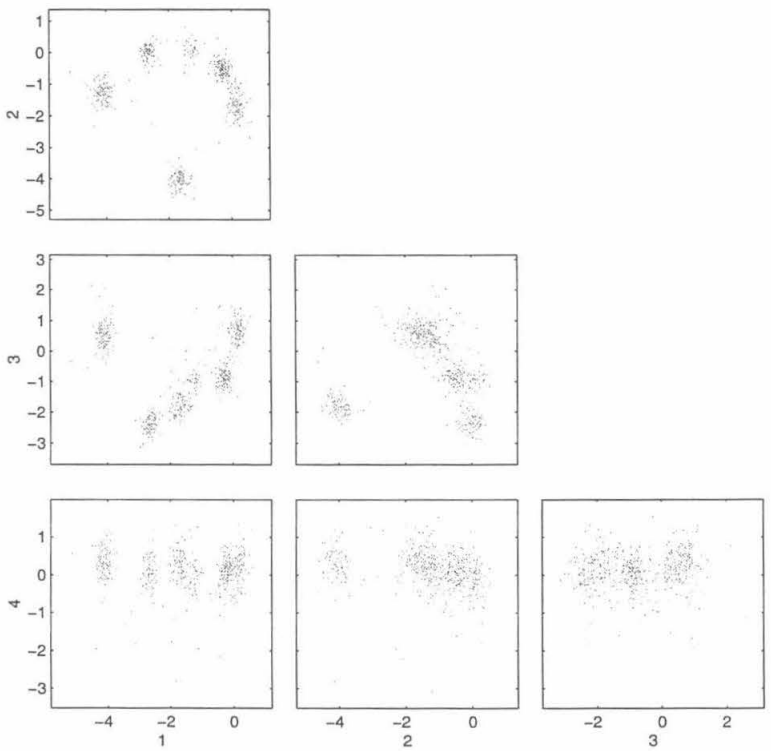


Figure 5.9: Events represented in the optimal linear discriminant space.

Outlier rejection

Dimensionality reduction carries with it the danger of reintroducing outliers into the main body of the ensemble. The danger arises in the case of outliers which fall outside the principal distribution along the directions which are to be suppressed, but whose projections onto the preserved space are not easily distinguished from those of normal spikes. Such outliers may bias the estimation of waveform parameters. Fortunately, they can be eliminated by removing from the ensemble spikes which exceed a data-set threshold in the suppressed directions. If the robust principal components analysis is used, they may be identified as points for which the uniform outlier component takes significant responsibility.

5.8 The Simple Mixture Model

5.8.1 The model

Once the ensemble of vectors has been extracted, we proceed to fit a model drawn from the schema (5.6), with the observations V_i replaced by the processed, lower dimensionality vectors, x_i. Initially, we shall examine the simplest possible such model.

We begin with two assumptions. First, each measured event vector is taken to be independent of all the others. This implies both that the set of indicators $\{z_{\emptyset,i}, z_{\not\sigma,i}, z_{m,i}\}$ are independent for different i (clearly, for any given i, they cannot be independent as only one can take the value 1) and also that the spike shape measured depends only on which cell fired, not on the previous waveforms emitted by that, or any other, cell. This assumption, allows us to drop the conditioning on the past latent variables (which was written "$\mid \lambda_{<i}$" in (5.6)). We write π_r for $\mathsf{P}_\theta\left(z_{r,i}=1\right)$ for $r = \emptyset, \not\sigma, 1 \ldots M$.

Second, the intrinsic variability in the spike shape is taken to be negligible, so that all of the observed variation is due to the addition of random background noise. In this case, each of the spike waveform densities $\mathsf{P}_m\left(x_i\right)$ is a Gaussian, whose mean is the spike shape associated with the mth cell and whose covariance is that of the background process. For noise-whitened data, this is the identity matrix.

Combining these assumptions with the mixture model schema (5.6), and restricting to the reduced-dimensionality space of the x_i, we obtain the basic model

$$\mathsf{P}\left(x_i\right) = \pi_\emptyset \left|2\pi I\right|^{-1/2} e^{-\frac{1}{2}\|x_i\|^2} + \sum_{m=1}^{M} \pi_m \left|2\pi I\right|^{-1/2} e^{-\frac{1}{2}\|x_i-\mu_m\|^2} + \pi_{\not\sigma}\mathsf{P}_{\not\sigma}\left(x_i\right) \tag{5.11}$$

where $\mathsf{P}_{\not\sigma}\left(x_i\right)$ is the uniform density given in (5.5).

5.8.2 Parameter estimation

Such a model is easy to fit. We employ the well-known Expectation–Maximization (EM) algorithm (Dempster *et al.* 1977; see chapters 1 and 2 of this dissertation) to find the maximum-likelihood parameter values. Other techniques, such as gradient-ascent or Fisher scoring may also be used for optimization. EM, however, offers some advantages.

1. EM is, perhaps, the most flexible of the various hill-climbing techniques, being easily extended to the more complex models to be discussed below. As a result, it provides a uniform approach to the fitting of the various models within the schema. Further, it is easily adapted to the situation in which different generative distributions are used for different cells, which will be discussed in section 5.11.

2. Incremental variants of EM are provably correct (Neal and Hinton 1998). While such proofs are derived in the case of static parameter values, they can give us confidence that similar variants will be well-behaved in the case of slowly drifting parameters, allowing us to track such drift.

3. The EM algorithm is very closely linked to the maximum-entropy deterministic annealing clustering technique (Rose *et al.* 1990). Indeed, the deterministic annealing approach can be extended to any latent variable model where EM is used by the Relaxation EM (REM) algorithm of chapter 3 (see also Ueda and Nakano (1998)). This technique provides a initial-condition-independent optimum, relatively immune to local maxima.

The EM iterations for simple mixture models such as this were derived in section 2.4. The current model has some additional constraints which further simplify the fitting procedure.

The background component distribution in (5.11) is fixed; only the mixing parameter $\pi_\emptyset$ needs to be learnt. The uniform outlier distribution has parameters that describe the region of support, A, in (5.5). We take this region to be rectangular in the transformed space of x_i (in fact, the shape is unimportant) and so it is specified by two opposite vertices. Provided the component is initialized with at least some responsibility for each of the data points, it is straightforward to see that the maximum likelihood solution will be such that A is the minimal region that contains all of the points. Furthermore, this value will ensure that in subsequent EM steps the component continues to have non-zero responsibility for each point and therefore maintains this parameter value. In practice, then, we can set the parameter directly from the data and update only the mixing component π_q.

The remaining components form a mixture of Gaussians. EM update rules for this model are given in section 2.6. We omit, of course, the update of the covariances as they are known in advance.

The update rules for parameter estimates at the nth step are thus

$$\begin{aligned} r^n_{m,i} &= \frac{\pi^{n-1}_m \mathsf{P}_{\theta^{n-1}_m}(x_i)}{\sum_l \pi^{n-1}_l \mathsf{P}_{\theta^{n-1}_l}(x_i)} \quad ; \quad m = \emptyset, \not{c}, 1 \ldots M \\ \pi^n_m &= \frac{\sum_i r^n_{m,i}}{|\mathcal{X}|} \quad ; \quad m = \emptyset, \not{c}, 1 \ldots M \\ \mu^n_m &= \frac{\sum_i r^n_{m,i} x_i}{\sum_i r^n_{m,i}} \quad ; \quad m = 1 \ldots M \end{aligned} \tag{5.12}$$

They are iterated until convergence.

It is guaranteed that this procedure will converge to a local maximum of the model likelihood. However, the identity of that maximum is crucially dependent on the initial parameter values used to seed the optimization. EM shares this dependence with other hill-climbing approaches, whether first or second order. We can avoid it by using a Relaxation Expectation–Maximization (REM) technique as described in chapter 3. In this simple case REM yields an algorithm very similar to the simple deterministic annealing example treated by Rose *et al.* (1990). The differences are primarily in the presence of the mixing probabilities and the single non-Gaussian component.

The REM update rules differ only in the update of the responsibilities, which become, for a relaxation parameter β,

$$r^n_{m,i} = \frac{\pi^{n-1}_m (\mathsf{P}_{\theta^{n-1}_m}(x_i))^\beta}{\sum_l \pi^{n-1}_l (\mathsf{P}_{\theta^{n-1}_l}(x_i))^\beta} \tag{5.13}$$

(we have given the E-step according to the REM-2 algorithm; see section 3.5). The parameter β is increased gradually from near 0 to 1, with the EM iterations being run to convergence at each value of β. An extensive discussion of the properties of this algorithm is given in chapter 3

The number of cells

In the absence of simultaneous high-power microscopy, we generally do not know how many foreground cells are to be expected in an extracellular recording. As a result, this quantity must be estimated from the data along with the parameters of the spike waveform distributions. In the mixture model framework this is equivalent to determining the correct number of components.

As was pointed out in section 2.7.3, this is essentially a model selection problem. We have already examined at some length in sections 1.3 and 2.7.3 techniques appropriate to carrying out this selection. The use of the REM algorithm for learning makes available a particularly efficient and effective framework within which to apply these techniques, which we have called cascading model selection. This was discussed in section 3.6.

For the most part these techniques, described in part I of this dissertation, can be applied without modification. Two components of the mixture, the noise model $\mathsf{P}_\emptyset(\cdot)$ and the overlap model $\mathsf{P}_{\not{c}}(\cdot)$ are always assumed to be present; thus, the model selection chooses between models with three or

more components.

5.9 Spike Shape Variability

The simple mixture model assumes that the action potential currents in each foreground cell are the same each time the cell fires, so that the only variability in the foreground spike waveform is due to the superposition of background spikes. In fact, this is rarely true.

Biophysically, one can imagine many reasons why the currents flowing across the somatic membrane might be variable. The concentrations of ions inside or outside the cell may vary. Ligand gated channels (for example, calcium-dependent potassium channels) may open on the membrane. A varying fraction, not large enough to prevent an action potential, of the sodium channels may be inactivated. Many of these conditions well depend on the recent activity of the cell, and this dependence will be examined more closely later. For the present, we will simply treat it as random variation.

5.9.1 Ratio methods

Some authors have argued (Rebrik *et al.* 1998; Zhang *et al.* 1997; Rinberg *et al.* 1999) that although the underlying action potential shape changes under these conditions, the *ratios* of the spike waveforms on the different channels should remain almost constant (disturbed only by the additive background noise). These ratios may be between maximal spike amplitudes, or between the magnitudes of the Fourier coefficients in various frequency bands. Such arguments are based on the same model as the ICA-based algorithms described earlier. The spikes recorded on the different channels are taken to be due to currents at a single point source which have been filtered differently by the extracellular medium through which they passed and by the electrode tip. If the source waveform (in the Fourier domain) is $S(\omega)$ the recorded signal on the nth channel will be $R_n(\omega) = F_n(\omega)S(\omega)$ where F_n is some linear filter. As the source changes, then, the spike shapes also change; but by taking the ratio of the recorded spike shapes $R_n(\omega)/R_m(\omega) = F_n(\omega)/F_m(\omega)$ we divide out the source signal and obtain a stable measure.

Once again, the arguments advanced against the applicability of ICA-models in, at least, neocortical tissue, apply here. The most severe is the fact that the simple model of one-source-multiple-detectors does not hold in preparations where the action potential travels over significant sections of cell membrane. In neocortical and hippocampal pyramidal cells, for example, action potentials are known to propagate over the dendrite (Stuart and Sakmann 1994; Stuart *et al.* 1997) and different electrode tips will record spikes due to different parts of the membrane (Buzsaki and Kandel 1998). In discussions of spike variability a further difficulty presents itself. The spread of the action potential across the membrane is known to be variable, depending on the recent firing activity of the cell

(Spruston *et al.* 1995; Svoboda *et al.* 1997). Thus, not only are the sources recorded by the different electrode tips spatially distinct, but these sources can vary in a distinct manner. As a result, there is reason to expect ratio methods to be inadequate in such preparations.

5.9.2 Models of the variability

Unable to remove the intrinsic variability in the waveforms, we seek to model it. In this section will we discuss models in which the underlying spike shapes are independent and identically distributed. Following this treatment, in section 5.10, we will discuss models which capture the dependence of the spike shape on the recent firing history of the cell.

Unconstrained Gaussians

One approach, attractive for its mathematical simplicity, is to model the underlying spike shape variability as Gaussian. If this model were correct, each observed spike waveform from a given cell would be the sum of two Gaussian random variates, and thus, would itself be Gaussian distributed. We have no independent data source from which to establish an appropriate covariance matrix for the intrinsic variability, and so the covariance must be learned along with the mean spike waveform. The measured background covariance can only provide a lower bound.

The general EM iterations for the arbitrary Gaussian mixture are as in (5.12), with the addition of a re-estimation rule for the mth covariance matrix

$$\Sigma_m^n = \frac{\sum_i r_{m,i}^n (x_i - \mu_m^n)(x_i - \mu_m^n)^T}{\sum_i r_{m,i}^n} \tag{5.14}$$

If the background covariance has been whitened, we can enforce the lower bound set by the background by diagonalizing the Σ_m^n obtained in this way, resetting any eigenvalues less than unity to 1, and then rotating back into the original space. If V is the matrix of eigenvectors of Σ_m^n, and the binary operator $\max(\cdot, \cdot)$ is taken to act element by element

$$\Sigma_m^n \leftarrow V \max(V^\mathsf{T} \Sigma_m^n V, I) V^\mathsf{T} \tag{5.15}$$

In the case of the background process, the superposed nature of the signal led us to expect it to be approximately Gaussian. In contrast, we have no reason to believe that the intrinsic variability should give rise to a Gaussian process, and so the validity of this model will rest entirely on the experimental evidence. In practice, cell waveform distributions in the macaque data set seemed to be well approximated in this fashion only if they did not fire bursts of closely spaced action potentials. The case of the bursting cells will be discussed more thoroughly below.

One issue introduced by the use of unconstrained Gaussians is the multiplicity of parameters. In

a D dimensional space, each component of the simple Gaussian model contributes only D parameters *to the model*. In contrast, the unconstrained Gaussian contributes $D(D+1)/2+D$ parameters. As the number of parameters increase the dangers of over-fitting and of being trapped in local maxima increase. The REM algorithm can alleviate the second of these to some extent, however strategies to reduce the complexity of the model are useful. On approach is to constrain the number of non-unit eigenvalues (in the background-whitened space) in each model. This leads (in the unwhitened space) to a mixture model, analogous to the mixture of factor analyzers model of Ghahramani and Hinton (1996). We will not explore this any further here, turning instead to a non-Gaussian generalization.

Hierarchical Gaussian mixtures

As was pointed out above, there is no *a priori* reason to expect the intrinsic variability to be Gaussian distributed. While such a model may provide a successful approximation in certain examples, it is insufficient to account for all of the observed data. Therefore, we will now investigate a non-parametric alternative.

The mixture model, which we have taken as the basic statistical model underlying probabilistic cluster analysis, has another rôle in the statistical literature. A mixture of relatively simple components (such as Gaussians) is often used to approximate a more complicated density, about which little is known *a priori*. Such an approach is called "non-parametric" because there is no explicit generative model of the density. It is not suggested that the data are in fact generated by any sort of mixture process. Rather, the mixture model is being used as an extremely flexible substrate for density approximation. (Compare the use of radial basis function networks in the function approximation literature).

Our alternative, then, is to fit an **hierarchical mixture model** in which the generative distribution for each cell is itself a mixture. We shall employ a mixture of Gaussians, each with a covariance matrix equal to that of the measured background noise. In a sense, this approximation may be viewed as identifying a small handful of "canonical" spike shapes, which span the range of possibilities. The generative process selects one of these shapes and then adds background noise to produce the observed spike waveform. In fact, the intrinsic waveform of the spike (before addition of the background) is not discrete in this fashion. This problem is mitigated by the fact that the Gaussian density provides significant probability mass in the region in between the selected points. We may think of the model as "tiling" the true density with a small set of identically shaped ellipses, the shape being set by the background covariance.

Let us write down the density that results from such a model. Suppose there are M clusters, with mixing proportions π_m. Each cluster is modeled by a mixture of P Gaussians, with mixing proportions $\rho_{m,p}$, means $\mu_{m,p}$ and unit covariances (we assume that we have whitened the background

process). The parameter set for the model is $\theta = \{\pi_m\} \cup \{\rho_{m,p}\} \cup \{\mu_{m,p}\}$. We have,

$$\mathsf{P}_\theta(\mathcal{X}) = \sum_i \sum_m \pi_m \sum_p \rho_{m,p} (2\pi)^{-d/2} e^{-\frac{1}{2}\|x_i - \mu_{m,p}\|^2} \tag{5.16}$$

If we distribute the factor π_m into the sum over p and write $\psi_{m,p} = \pi_m \rho_{m,p}$ it becomes clear that this density is identical to that derived from a mixture of $R = M \times P$ Gaussians. Indeed, any hierarchical mixture in which the total number of Gaussians is R, even if there are unequal numbers of components used to describe each cell, will yield the same form of the density.

This poses a serious problem from the point of view of model selection. Conventional model selection procedures may indicate the correct density from among a group of candidates. But, how are we to decide which components belong to which cell? Probabilistically, any such assignment would be equally valid, including the "flat" option in which every component represents a single cell. In short, from a probabilistic point of view, there is no such thing as a hierarchical mixture!

We may choose to exploit additional information in order to group the Gaussians.

One approach is as follows. Begin by fitting a mixture of a large number of Gaussians (all with unit variance) to the data. The actual number is not of great importance, provided it is significantly larger than the number of cells expected. It may be chosen arbitrarily, or by a model selection method. Then, form a graph, with one node for each Gaussian. An edge between two Gaussians is included if the densities exhibit a significant degree of overlap, that is, if the distance between their means is smaller than some chosen threshold. Each of the connected subgraphs that results is taken to represent a single cell. Such an approach would be similar in spirit, although different in detail, to that proposed by Fee *et al.* (1996a) (a detailed discussion of the relationship to their method is outlined in section section 5.14).

Alternatively, the additional information might be encoded as a prior on the parameters within a group. For example, we might expect that the means of the components that describe a single cell will lie close together, and will themselves be drawn from a Gaussian density of small variance.

In both these approaches, one or more control parameters must be chosen arbitrarily: either the overlap threshold for the graph formation, or the form and extent of the prior. In many cases, these parameters may be chosen anywhere within a fairly broad range of values, with identical results. However, it is in the case when the waveforms from two or more cells are very similar, and where the model selection procedure is thus most important, that the results become most sensitive to the choice of parameters.

In section 5.10.2 we will introduce a third approach to the resolution of the ambiguity in the hierarchical mixture likelihood, suitable for modeling variability intrinsic to bursts of action potentials. There, a dynamic model is proposed, in which the components representing a single cell are tied together by a learnt Markov transition structure. In that view, components belong to the same

cell provided that the timing of spikes that fall within them is consistent with a simple burst model.

5.10 Dynamic Models

In the models discussed thus far each spike waveform is generated independently of all others. We turn now to models in which the latent variables are dependent on each other.

5.10.1 Refractory period

One simple feature of the firing process has not yet been accounted for in any of our models. This is the occurrence of the refractory period, a short period after each action potential during which the cell that fired will not fire again. As it stands, the mixture model has no representation of the time of any event. We will discuss shortly a model in which time is explicitly represented. For the moment, though, it is possible to account for the refractory period by a simple modification to the basic mixture model. The method presented in the following may be applied to any of the various mixture models we discussed above; for simplicity we shall develop it in the case of the simple Gaussian mixture of section 5.8.

The joint data log-likelihood for such a model was given in section 2.6

$$\ell_{\mathcal{X},\mathcal{Z}}(\theta) = \sum_i \sum_m z_{m,i} \left(\log \pi_m - \frac{1}{2} \log |2\pi\Sigma_m| - \frac{1}{2}(x_i - \mu_m)^T \Sigma_m^{-1} (x_i - \mu_m) \right) \tag{2.17}$$

In the refractory case this expression remains valid for most data and parameter values; the exception is provided by sequences of $z_{m,i}$ that violate the refractory constraint by assigning to the same cell events that fall within a refractory period of each other, for which the log-likelihood diverges to $-\infty$. In taking the expected value of the log-likelihood, however, the probability of such a sequence is 0, and so we can discount this possibility. The expected log-likelihood under the distribution $\mathsf{P}_{\theta^{n-1}}(\mathcal{Z} \mid \mathcal{X})$ retains the general mixture form of (2.8)

$$\begin{aligned} Q^n(\theta) &= \sum_i \sum_m \mathcal{E}_{z_{m,i}|x_i,\theta^{n-1}} [z_{m,i}] \log \pi_m \mathsf{P}_{\theta_m}(x_i) \\ &= \sum_i \sum_m s_{m,i}^n \left(\log \pi_m - \frac{1}{2} \log |2\pi\Sigma_m| - \frac{1}{2}(x_i - \mu_m)^T \Sigma_m^{-1} (x_i - \mu_m) \right) \end{aligned} \tag{5.17}$$

except that, as we will see below, the expected values of the $z_{m,i}$ are different from before. To remind ourselves of this difference we use the notation $s_{m,i}^n$ for these new responsibilities, reserving the symbols $r_{m,i}^n$ for the responsibilities in the non-refractory case.

To obtain the new responsibilities, consider first the simple case where only two spikes have been observed and the second appears within a refractory period of the first. We have a joint distribution

over $z_{m,1}$ and $z_{m',2}$ with

$$\mathsf{P}\left(z_{m,1}, z_{m',2}\right) = \begin{cases} 0 & \text{if } m = m' \\ r^n_{m,1} r^n_{m',2}/Z & \text{otherwise} \end{cases} \tag{5.18}$$

where $Z = \sum_m \sum_{m' \neq m} r^n_{m,1} r^n_{m',2}$ is an appropriate normalizing constant. The expected values we seek are then just the marginals of this joint distribution, for example,

$$s^n_{m,1} = \sum_{m' \neq m} r^n_{m,1} r^n_{m',2}/Z = r^n_{m,1}(1 - r^n_{m,2})/Z \tag{5.19}$$

where we have used the fact that $\sum r^n_{m',i} = 1$.

This result easily generalizes to the case of many spikes

$$s^n_{m,i} = \frac{r^n_{m,i}}{Z_i} \prod_{i,j \text{ refractory}} (1 - r^n_{m,j}) \tag{5.20}$$

where Z_i is the appropriate normalizer and the product is taken over all spikes that are fall within one refractory period (before or after) the i spike.

The M-step is still a weighted Gaussian estimation as before, the weights now being the new responsibilities $s^n_{m,i}$.

5.10.2 Sparse hidden Markov models

Bursts

The intrinsic variability of spike waveforms is not entirely random for all cells. Many pyramidal cells, both in neocortex and in the hippocampus, sometimes fire action potentials in bursts. Action potentials within a burst are closely spaced (as little as 1ms apart), and the cell does not have enough time to recover from one before the next begins. Thus, the membrane currents associated with later action potentials are likely to be smaller, and a smaller portion of the dendritic membrane will participate in such spikes. As a result, the spike waveforms recorded later in the burst may be quite different from those associated with isolated action potentials.

In this section we will construct a statistical model to describe the change in action potential during a burst. At first glance, one might think that a sufficient model would have the expected spike waveform depend on the immediately preceding interval. In fact, the situation is considerably more complex than this. For example, the third spike in a regular burst will usually be smaller than the second, even though the preceding interval is the same. At the same time, it is true that after a longer interval the cell has had more time to recover and so the spike waveform is closer to the normal case.

Faced with the complexity of the mechanisms underlying the change in spike waveform during a burst, we will *not* attempt a biophysical model. Instead, we will use a simple statistical model that will capture the variation empirically.

A statistical model

The statistical model that we consider is a constrained version of the Hidden Markov Model (HMM). Each cell is modeled by a single HMM, which is independent of all of the others. In practice, it is often useful to use HMMs to model only a subset of the cells in a recording — those that exhibit bursts — and use Gaussians or other static distributions to describe the others.

The output symbols of the underlying Markov model are either complete spike waveforms represented as vectors (the events of the previous discussion) or a zero vector. The vast majority of symbols in any string generated from the Markov model will, in fact, be zero and so these models are sparse in the sense of chapter 4. The observed vector is the sum of the Markov model output and a random vector drawn from the background process. Thus one may think of the output distributions of the states of the HMM as Gaussians, centred either on zero or on a mean waveform which is to be learned. The output density is thus identical to that of the hierarchical Gaussian mixture model discussed in section 5.9.2. The difference is that events are not chosen from this density independently. This change in the model provides another approach to breaking the ambiguity inherent in the hierarchical model.

A Markov model describes a discrete time process. We choose to discretize time in fairly large steps, usually 0.5ms. The measured output symbol for any given time-bin is a spike waveform if the identified time (that is, the peak or centre of mass) of some event falls within that bin. Otherwise, the output symbol is taken to be 0.

The transition matrix of the Markov model is constrained so as to embody the structure expected from a bursting cell. This constrained structure is sketched in the left-hand part of figure 5.10. Each of the grey circles in this figure represents a state of the HMM. The left column of states all have zero output symbol and represent the cell in a non-firing state. States in the right column represent firing events in the cell and have non-zero output distributions. These distributions are indicated on the stylized event feature plot to the right. Each state is associated with a Gaussian output distribution indicated by an elliptical boundary. Together, these distributions "tile" one of the elongated clusters in the data set.

Each heavy arrow in the HMM diagram represents an allowed transition: where there is no arrow the transition probability is set to 0 and remains at this value throughout the learning process. The states are arranged in a "ladder" with states lower down the ladder corresponding to greater recent firing (and therefore greater inactivation of channels). The upper left-hand state is the "ground" state, in which the cell will be found after a long period of inactivity. Only two transitions are

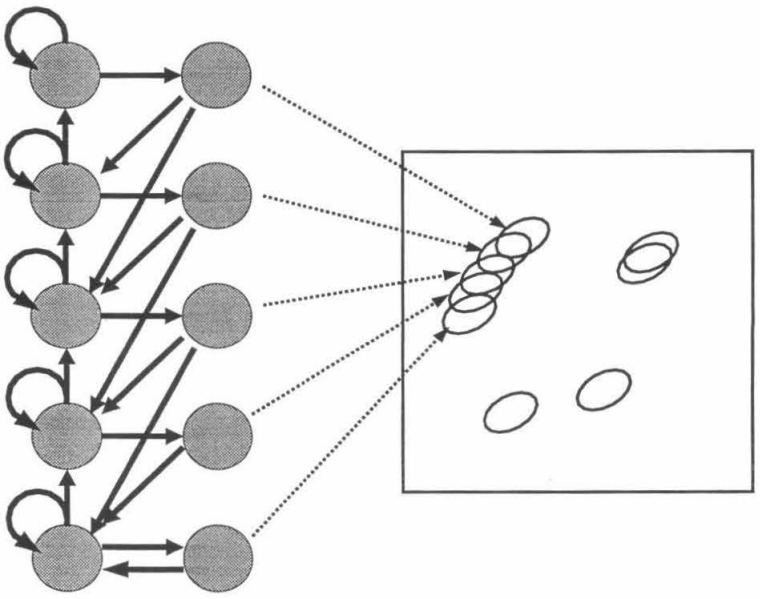


Figure 5.10: The HMM transition structure

possible from this state: the cell either fires an action potential, making the transition to the state on the right, or else remains in the same state. Once in the firing state, the cell makes a transition to a non-firing state below the ground state, thus preserving the memory of the recent firing. From this state, the cell can fire again, with a different output distribution, in which case it subsequently moves further down the ladder of states; it can remain in the same state; or it can make a transition up the ladder. This basic pattern is repeated for each of the rungs of the ladder.

Some features of this structure are worth pointing out. The only way for the cell to transition down the ladder is to fire. Once it fires it must enter a non-firing state and so cannot spike in successive time-bins; for 0.5ms bins this effectively enforces a short refractory period. If the cell finds itself some distance down the ladder, but does not subsequently fire for a number of time-steps, it will relax back to the ground state with an exponential decay profile.

Learning with HMMs

A learning algorithm for mixtures of sparse HMMs was discussed in section 4.4. Sparse HMMs were defined in that section to produce two types of output: either a null symbol, $\emptyset$, or a numerical value. When considering mixtures of sparse HMMs we introduced a third type of output, the symbol $\not\emptyset$, which was detected when two or more of the component HMMs emitted non-null outputs in the same time-step.

In the current application an output is defined for each 0.5ms time-bin as follows. If no event

has its peak (or centre of mass) within the bin the observation is taken to be ø. In most cases, if an event does peak within the bin, the observation is the reduced vector representation of that event. However, if the event has been classed as an outlier, then the symbol ¢ is observed. Outlier events are identified in three ways during our procedure. First, the waveform may exhibit a double peak or other heuristically excluded property during event extraction. Second, the event may fall outside the principal subspace during dimensionality reduction. Finally, it may be assigned with high probability to the outlier mixture component. This last poses a problem, since we cannot know before fitting is complete which events are to be classified in this way; but we also cannot fit the mixture of HMMs accurately without knowing which observations are collisions. In practice, this circularity is resolved by dynamically marking as a collision any event that is assigned to the outlier cluster with a probability that exceeds some set threshold on a given iteration.

Given these definitions, the learning algorithms of section 4.4 can be employed to optimize the mixture parameters.

5.11 Mixed Models

There is no reason to expect that all of the foreground cells present in a particular recording will exhibit the same type or degree of variability. A single site may yield some cells that tend to fire in bursts of action potentials; some that fire isolated, but stochastically variable spikes; and some that exhibit no detectable intrinsic variability at all. Thus, it is often useful to be able to combine the three types of waveform model we have discussed in this chapter — the fixed covariance Gaussian of the simple mixture model; the mixture of Gaussians of the hierarchical mixture model; and the sparse hidden Markov model — in a single overall mixture.

The framework in which to do so is provided by the mixture of sparse hidden Markov models discussed above, and at greater length in section 4.4. In particular, we observe that both the single, fixed covariance Gaussian and the mixture of fixed covariance Gaussians may both be expressed as special cases of the sparse HMM, with transition matrices constrained differently from the "ladder" of figure 5.10.

The simple fixed-covariance Gaussian model is equivalent to a two-state HMM. One state (say, the first) has null output, the other has an output distribution given by the Gaussian model. To reproduce the basic model exactly, the columns of the transition matrix must be identical. The augmented transition matrix (including the initial state probabilities; see section 4.1.1) is of the form

$$T_m = \begin{pmatrix} 0 & 0 & 0 \\ 1-\rho_m & 1-\rho_m & 1-\rho_m \\ \rho_m & \rho_m & \rho_m \end{pmatrix} \tag{5.21}$$

Here ρ_m represents the firing probability per time-step associated with the mth model of the overall mixture. It is related to the mixing probability π_m as follows. Suppose the total number of events in the training data (with collisions counted twice) is N and the total number of HMM time-steps is T. Given the stationarity assumption of the mixture, we expect there to be $\pi_m N$ spikes from the mth cell in this data, and so the probability of a spike per time-step is $\rho_m = \pi_m N/T$.

The transition matrix given in (5.21), allows for the cell to fire in adjacent time-bins with probability ρ^2. In fact, it is convenient to exploit the HMM transition structure to enforce a refractory period without requiring the scheme of section 5.10.1. In section 5.10.2 we achieved this by requiring that the model return to a null state after firing. For 0.5ms time-steps, this enforced a short, but reasonable refractory period. Thus, we alter the transition matrix to

$$T_m = \begin{pmatrix} 0 & 0 & 0 \\ 1-\rho_m & 1-\rho_m & 1 \\ \rho_m & \rho_m & 0 \end{pmatrix} \tag{5.22}$$

The value of the firing probability ρ_m must now be corrected. The new relationship is $\rho_m = \pi_m N/(T - \pi_m N)$.

The mixture of Gaussians model for a single cell is implemented similarly. For a P component mixture the HMM now contains $P+1$ states, one with null output (again, we take this to be first) and the others with output distributions corresponding to the components of the mixture. If the mixing probabilities of the cell model are $\pi_{p,m}$ and the overall mixing probability of this cell model within the hierarchical mixture is π_m we define densities by $\rho_{p,m} = \pi_{p,m}\pi_m N/(T - \pi_{p,m}\pi_m N)$. We write $\rho_m = \sum_p \rho_{p,m}$. Then the augmented transition matrix, corrected to enforce a refractory period, is given by

$$T_m = \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 \\ 1-\rho_m & 1-\rho_m & 1 & \cdots & 1 \\ \rho_{1,m} & \rho_{1,m} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \rho_{P,m} & \rho_{P,m} & 0 & \cdots & 0 \end{pmatrix} \tag{5.23}$$

Having converted each non-Markov model into a sparse hidden Markov model whose transition matrix embodies the appropriate structure, we can then proceed to learn the parameters using the algorithm described in section 4.4. In general, learning in such a model is more computationally expensive than in the basic mixture models. Thus, if no cells in a given data set appear to fire in bursts, so that the ladder-structure HMMs will not be needed, it is preferable to use the mixture model directly, possibly with the refractory modification of section 5.10.1. However, once the parameters are learned, the corresponding SHMMs can be constructed by the procedure given in this

section. These SHMMs can then be used for on-line spike recognition, as described in section 5.13.

5.12 On-line Learning

In many applications of spike sorting, recognition must be carried out in close to real time. In scientific experiments, for example, feedback in the form of sensory stimulus changes or even neural stimulation might need to be triggered within milliseconds of a particular pattern of action potentials being recorded. In neural prosthetic applications, neural activity needs to be transformed into a "motor" action on a similar time scale.

For the most part, such demands constrain the inference, or spike recognition, stage of sorting (to be discussed below) rather than the learning. We may collect an initial segment of data without the real time demands, train on these data off-line and then perform on-line inference.

However, it is useful to update the parameter estimates as more data are collected. For one thing, these updates will refine the estimates, yielding progressively more reliable data. As a result, it might be possible reduce the length of the initial training segment, leading to a smaller training down-time prior to on-line recognition.

More important, though, is the fact that in almost all recording situations, the parameters are likely to drift over time. Such drift generally occurs due to minute changes in the relative positions of the cells and electrodes, thus changing the recorded spike waveforms. Even without such physical displacement, however, the statistics of spiking of the different cells, which enter into the models in the form of mixing parameters or transition probabilities in the HMM, may change. For example, cells may switch between more or less bursty modes of firing in association with varying levels of drowsiness (or anesthesia) in the subject.

In this section we discuss techniques for on-line parameter adaptation. Similar techniques will allow both refinement of the estimates as new data come in, as well as tracking of slow drift in the parameters. We discuss these techniques as though the parameters are to be updated each time a new spike is observed. In practice this level of immediacy is unnecessary, and it is more efficient to collect spikes for a short period (say 1s) and apply the updates in a batch form.

5.12.1 Incremental EM

We showed in section 1.8 (following Neal and Hinton 1998) that the free energy interpretation of EM can be used to justify some variants on the basic algorithm. One of these is an incremental version in which the parameters are updated one data point at a time. This approach is valid in cases where both the observations x_i and the latent variables y_i are independent and drawn from fixed distributions, and so the conditional distribution $\mathsf{P}_\theta(\mathcal{Y} \mid \mathcal{X})$ factorizes over the y_i. Of the models we have discussed here, this is true only of the mixtures.

The iterations for the incremental EM algorithm, in the notation of section 1.8, are as follows.

IE-step: Choose some i. Maximize $F_i(p_i, \theta^{n-1})$ and leave the remaining $p_j, j \neq i$ unchanged.

$$\begin{aligned} p_i^n(y_i) &= \mathsf{P}_{n-1}(y_i \mid x_i) \\ p_j^n(y_j) &= p_j^{n-1}(y_j) \end{aligned} \tag{5.24}$$

M-step: Maximize F with respect to θ holding p constant.

For a mixture model, the probability distribution $p_i^n(y_i)$ is simply the set of responsibilities $r_{i,m}^n$, $m = 1 \ldots M$ and the M-step involves maximizing the weighted log-likelihood $\sum_i r_{i,m}^n \mathsf{P}_{\theta_m}(x_i)$ for each component.

The on-line version of this algorithm is different only in that there is no choice of i. The data are simply handled, one by one, as they arrive from an unlimited stream. The M-step update only involves, of course, the data collected to this point. We shall assume that the initial parameter values chosen are very close to the true values, being the result of training on a separate, off-line, data set. This assumption means that even though data are not revisited, the responsibilities assigned to them remain reasonably valid. An alternative approach is outlined in the next section.

Fortunately, for Gaussian mixtures (and indeed many other mixture models) it is not necessary to store all of the past responsibilities and observations in order to update the parameters in the M-step. We derive the M-step update rule for a general mixture of unconstrained Gaussians; the result for the various constrained Gaussian models used for spike sorting will follow immediately.

The usual M-step updates for a Gaussian mixture, given N data points, are

$$\pi_m^n = \frac{\sum_{i=1}^N r_{m,i}^n}{N} \tag{5.25}$$

$$\mu_m^n = \frac{\sum_{i=1}^N r_{m,i}^n x_i}{\sum_{i=1}^N r_{m,i}^n} \tag{5.26}$$

$$\Sigma_m^n = \frac{\sum_{i=1}^N r_{m,i}^n (x_i - \mu_m^n)(x_i - \mu_m^n)^T}{\sum_{i=1}^N r_{m,i}^n} \tag{5.27}$$

The $(N+1)$th data point, x_* arrives, triggering the $(n+1)$th update of the parameters. We calculate the responsibilities, r_{m*} of each of the components for this point in the usual fashion. According to the incremental EM algorithm, then, the new estimate for π_m is

$$\pi_m^{n+1} = \frac{1}{N+1} \sum_{i=1}^{N+1} r_{m,i}^{n+1} = \frac{1}{N+1} \left(\sum_{i=1}^{N} r_{m,i}^n + r_{m*} \right) = \frac{N}{N+1} \pi_m^n + \frac{1}{N+1} r_{m*} \tag{5.28}$$

where we have used the fact that $r_{m,i}^{n+1} = r_{m,i}^n$ for all $i < N+1$. Similarly, we find that (writing

$R_m^n = \sum_{i=1}^N r_{m,i}^n = N\pi_m^n$)

$$\mu_m^{n+1} = \frac{1}{R_m^{n+1}} \sum_{i=1}^{N+1} r_{m,i}^{n+1} x_i = \frac{1}{R_m^{n+1}} \left(\sum_{i=1}^{N} r_{m,i}^n x_i + r_{m,*} x_* \right) = \frac{R_m^n}{R_m^{n+1}} \mu_m^n + \frac{1}{R_m^{n+1}} r_{m*} x_* \tag{5.29}$$

Finally, the corresponding result for Σ_m^{n+1} follows by rewriting (5.27) as

$$\Sigma_m^n = \frac{\sum_{i=1}^N r_{m,i}^n x_i x_i^T}{\sum_{i=1}^N r_{m,i}^n} - \mu_m^n \mu_m^{n\,T} \tag{5.30}$$

from which we find that

$$\Sigma_m^{n+1} = \frac{R_m^n}{R_m^{n+1}} \left(\Sigma_m^n + \mu_m^n \mu_m^{n\,T} \right) + \frac{1}{R_m^{n+1}} r_{m*}\, x_*\, x_*^T - \mu_m^{n+1} \mu_m^{n+1\,T} \tag{5.31}$$

5.12.2 Parameter adaptation

When the update algorithms described above are used in an on-line fashion (without revisiting any data), the impact of each succeeding point on the parameter estimates grows progressively smaller. If the parameters are varying slowly, this is an unfortunate state of affairs, since information about the new values will be incorporated at an ever decreasing pace. Indeed, even if the parameters are stable, but the initial estimate of the model was far from the true value, this state of affairs is not too promising. The reason (stated here in terms of the incremental EM algorithm for mixtures, although it applies equally to the HMM) is that the responsibilities that were calculated for the first few data points become increasingly inaccurate as the model is optimized. While the effect of these early values on the estimate is diluted by ever more incoming data, leading to the correct result in the limit, convergence would be more rapid if we had a mechanism to "forget" them. (Note that the incremental EM algorithm as described by Neal and Hinton (1998) avoids this problem by revisiting all the data with some probability).

Notice that each of the update rules derived in the previous section (5.28), (5.29), (5.31) has the form of a weighted sum of old information and new. The form of amnesia we seek can be achieved by the simple measure of adjusting the weights in this sum to favour the new data.

One approach is suggested by Nowlan (1991). In this view, the optimal parameter values are maintained by a group of sufficient statistics; for the mixture of Gaussians, these statistics are $R_m^n = \sum_i r_{m,i}^n$, $S_m^n = \sum_i r_{m,i}^n x_i$ and $SS_m^n = \sum_i r_{m,i}^n x_i x_i^T$. Knowing the values of these statistics at any iteration n we can calculate the parameter values $\pi_m^n = R_m^n / \sum_m R_m^n$, $\mu_m^n = S_m^n / R_m^n$ and $\Sigma_m^n = SS_m^n / R_m^n - \mu_m^n \mu_m^{n\,T}$. The update rules derived in the previous section can then be easily expressed in terms of these sufficient statistics

$$R_m^{n+1} = R_m^n + r_{m*}\,; \qquad S_m^{n+1} = S_m^n + r_{m*}\, x_*\,; \qquad SS_m^{n+1} = SS_m^n + r_{m*}\, x_*\, x_*^T \tag{5.32}$$

The proposal made by Nowlan (1991) introduces a factor $\gamma < 1$ to regulate the decay of older information. The sufficient statistic update rules are replaced with these:

$$R_m^{n+1} = \gamma R_m^n + r_{m*} \; ; \qquad S_m^{n+1} = \gamma S_m^n + r_{m*}\, x_* \; ; \qquad SS_m^{n+1} = \gamma SS_m^n + r_{m*}\, x_*\, x_*^T \tag{5.33}$$

We can thus derive the parameter update rules under this approach. If we write N_e^n for $\gamma \sum_m R_m^n$ we obtain,

$$\pi_m^{n+1} = \frac{R_m^{n+1}}{\sum_m R_m^{n+1}} = \frac{\gamma R_m^n + r_{m*}}{\sum_m (\gamma R_m^n + r_{m*})} = \frac{N_e^n \pi_m^n + r_{m*}}{N_e^n + 1} \tag{5.34}$$

and

$$\mu_m^{n+1} = \frac{S_m^{n+1}}{R_m^{n+1}} = \frac{\gamma R_m^n \mu_m^n + r_{m*}\, x_*}{R_m^{n+1}} = \frac{N_e^n \pi_m \mu_m^n + r_{m*}\, x_*}{R_m^{n+1}} \tag{5.35}$$

with a similar result for the covariance update. Comparison with (5.28) and (5.29) suggests that the term N_e^n plays the rôle of an effective number of data. Note that $N_e^{n+1} = \gamma N_e^n + 1$. Thus if $N_e^n = (1-\gamma)^{-1}$ then $N_e^{n+1} = N_e^n$ and otherwise $N_e^{n+1} > N_e^n$. The effective number of data climbs until it reaches the value $(1-\gamma)^{-1}$ and then remains constant. Thus we may think of this approach as limiting the effective number of data used.

Such an approach is seen to be reasonable in situations where the parameters change at a rate linked to the number of data measured (or in the case where such adaptation is needed to speed on-line convergence given poor initial parameter values). In the spike sorting example, however, we expect the parameter variation to occur at a rate constant in time, even if the overall spike rate varies. We would like the effective number N_e^n to be dependent on the recent firing rate of the cells being recorded.

The formulation in terms of an effective number of data makes this easy. We replace the term N_e^n in the above by a firing-rate dependent term that varies in time $N_e(t)$. The dependency on firing rate might set $N_e(t)$ to the number of spikes recorded within a window. It should be borne in mind that this approach is different to simply using only the last $N_e(t)$ data points to estimate the parameter values. The estimates are based on all previous data; however, the estimate derived from these data is weighted as though it was derived from only $N_e(t)$ points.

5.12.3 Limited look-ahead forward–backward

The scheme described in the previous section is appropriate for on-line adaptation of the parameters of mixture models, whether of the simple Gaussian type, or more elaborate. What about the dynamic hidden Markov model, proposed in section 5.10.2? At first glance, the situation appears impossible. Recall that to perform even a single E-step of the learning algorithm requires a traversal through all of the data by the forward–backward algorithm. It would seem, then, that we cannot even begin to learn the parameters of the model until all of the data have been collected.

Of course, this is not exactly true. If the parameters were stationary we would expect that parameter estimates derived from a moderately long sequence of data would be reasonable, and affected only marginally by the incorporation of additional observations. The critical point is that the influence of later observations on earlier state and transition estimates is diminished by mixing in the Markov chain. Thus, although in principal the backward pass of the inference algorithm should begin at the very end of the data set, if it is instead begun earlier, only the immediately preceding state estimates (those within one mixing time) will be substantially incorrect. This feature is exploited by Boyen and Koller (1999) in the context of general dynamic probabilistic networks. For the sparse hidden Markov model the situation is further improved, because, as was argued in section 4.3.2, long stretches of null observations tend to "reset" the model. "Long," in this context, refers to the mixing time of the null-state restricted Markov chain; in the spike sorting context this is the time taken for a cell to reset after a burst and thus may well be on the order of 20ms.

The incremental approach to learning the HMM thus involves re-running the backward pass of the forward–backward algorithm only as far back as the last segment of moderate silence. To be conservative, one might discount state estimates in the M-step until they become "protected" by a stretch of nulls, although in practice this rarely makes any difference. In any case, if one realigns the notion of the "current" time to the last estimate that can be trusted, we may think of this procedure as taking into account a short sequence of data in the future. Thus the name **limited look-ahead forward–backward algorithm**.

As new state information becomes available it is combined with the earlier information by a procedure analogous to (5.29) and (5.31), with the state estimates $s^n_{p,m,i}$ replacing the responsibilities. The update of the transition matrix is similar in spirit to (5.28), but differs slightly. We write $t_{pq,m,*}$ for the new transition estimate and $S^n_{q,m} = \sum_{i=0}^{N-1} s^n_{q,m,i}$ to obtain

$$T^{n+1}_{pq,m} = \frac{\sum_{i=1}^{N+1} t^{n+1}_{pq,m,i}}{\sum_{i=0}^{N} s^{n+1}_{q,m,i}} = \frac{\sum_{i=1}^{N} t^{n}_{pq,m,i} + t_{pq,m*}}{S^{n+1}_{q,m}} = \frac{S^n_{q,m}}{S^{n+1}_{q,m}} T^n_{pq,m} + \frac{1}{S^{n+1}_{q,m}} t_{pq,m*}. \tag{5.36}$$

For non-stationary parameters we can implement adaptive rules by weighting the updates by an effective data size just as in (5.33) and following. In this case, since a new estimate is generated at every time-step whether a spike occurred or not, we do not need to worry about varying the effective number of data, and we simply choose a fixed value of the decay constant γ.

5.13 Spike Time Detection

Given the model structure and parameters, the third and final stage of the spike sorting process is the inference of the firing times. To perform this inference accurately, and in particular to resolve overlapped spikes, we will return to the full superposition model (5.2), using the distributions for

the firing indicators $c_{m,\tau}$ and waveforms $S_{m,\tau}$ derived from the learnt mixture model. Many, if not most, previous spike sorting approaches have not made this distinction: inference is performed on extracted events using a cluster assignment model and is not actively distinguished from the learning of the model. Such an approach leaves three issues unresolved. First, the threshold-based event detection heuristic of section 5.5 can be improved upon once the true spike shapes have been determined. Second, if all events are to be clustered, the sorting process must occur off-line, ruling out experiments in which rapid feedback about the cells' responses is needed. Third, the clustering procedure has discarded the superposed events, or else collected them into an unresolved overlap cluster, rather than resolving them into their constituent spike forms.

The correct solution to the inference problem involves a search through all possible combinations of spike arrival times, and is computationally prohibitive. Lewicki (1994) suggests that with optimized programming techniques, and suitable, but severe approximations, it is possible to complete *this* search in close to real time on a computer workstation. We shall not review his implementation here; the interested reader is referred to the cited paper. Instead, we discuss an alternative set of approximations that lead to a straightforward, single-pass, greedy algorithm. This approach is particularly well-suited to parallel implementation on arrays of digital signal processors (DSPs).

We shall derive the procedure in the context of the sparse hidden Markov models of section 5.10.2, where the output distribution of each component is either null or a Gaussian of fixed covariance (set by the background). As was seen in section 5.11, other cell models that we have considered can also be expressed in this form, and so the detection method we discuss will apply equally well to the simple Gaussian model of section 5.8 or to the hierarchical Gaussian mixture of section 5.9.2. It will not, however, apply to the unconstrained Gaussian model of section 5.9.2 without considerable modification.

The basic structure of the scheme is as follows. At each time-step we begin by estimating the prior probability distribution over the states of each SHMM, based on our estimates of the states at the preceding time-step. Using these probabilities, and the data recorded around the given point in time, we obtain the occupancy likelihoods for each of the firing states of each of the models, along with the likelihood that no spike was observed. We accept the event associated with the largest likelihood. If this optimal likelihood is for no spike, then we re-derive the posterior state distribution for each model as though a null symbol was observed. If, on the other hand, the optimal likelihood is due to one of the firing states, we assume that the appropriate model is, in fact, to be found in that state. The corresponding mean spike waveform is subtracted from the recorded data; and again the likelihoods of the remaining models having fired, or of there having been no second spike are calculated. This is repeated until no more spikes remain to be accounted for at this time-step. The initial state probabilities for the next step are then inferred by transitions from the posterior estimates of the states at the current time.

This is a recursive procedure similar to the forward step of the coupled forward–backward algorithm. We will examine in detail a single step of the procedure in analogy to the treatment of section 4.4.2.

We assume that at the $(i-1)$th time-step, the current state probability estimates are given by $E_{p,m,i-1}$[6]. Since the Markov transitions are taken to be independent, these are propagated forward to provide initial estimates of the probabilities at the ith step by the relation

$$\tilde{E}_{p,m,i} = T_m E_{p,m,i-1} \tag{5.37}$$

We need to assess the probability of a spike being present on this time-step. However, we are no longer dealing with pre-extracted and aligned spike waveforms and so the spike, if any, may have occurred at any point within the time interval under study. We can measure the probability by the maximal output of a simple matched filter. Suppose that the pth component of the mth model has a non-null output distribution, with mean waveform (transformed into the time domain from whatever subspace was used to fit) given by $S_{p,m}(t)$. We assume that the background has been whitened, so that the covariance of this output distribution, and all the others, is I. The joint log-likelihood of a spike having been generated from this particular component (that is, that the state variable $y_{m,i} = p$) at a particular time τ, under the observed trace $V(t)$, is

$$\begin{aligned} &\log \mathsf{P}\left(V(t) \mid y_{m,i} = p, \tau\right) \\ &\quad \propto \quad -\frac{1}{2}\int dt\ \left(V(t) - S_{p,m}(t-\tau)\right)^2 \\ &\quad = \quad \int dt\ V(t) S_{p,m}(t-\tau) - \frac{1}{2}\int dt\ V(t)^2 - \frac{1}{2}\int dt\ S_{p,m}(t-\tau)^2 \end{aligned} \tag{5.38}$$

while the likelihood that there was no spike is simply

$$\log \mathsf{P}\left(V(t) \mid \emptyset\right) \propto -\frac{1}{2}\int dt\ V(t)^2 \tag{5.39}$$

The spike time τ will be assumed to lie within the short interval under consideration for this time-step. The integrals over t extend through all time; although we will soon drop the integral of $V(t)^2$, and the others can be limited to the support of $S_{p,m}(t-\tau)$. Note that the final term in (5.38) is, in fact, independent of the spike time τ; we will therefore write $\alpha_{p,m} = \int dt\ S_{p,m}(t)^2$ for the total power in the waveform associated with the distribution (p, m).

We can combine these expressions with our prior expectations of each state given by $\tilde{E}_{p,m,i}$, and drop the common term that depends only on $V(t)$ to obtain the following weighted matched-filter

[6] We adopt the same conventions for subscripts as we did in section 4.4, so that p refers to the state, m to the model and $i-1$ to the time-step.

outputs:

$$\mathcal{F}_{p,m,i}(\tau) = \int dt\, V(t) S_{p,m}(t-\tau) - \frac{1}{2}\alpha_{p,m} + \log \tilde{E}_{p,m,i}/\delta \tag{5.40}$$

$$\mathcal{F}_{\emptyset,i}(\tau) = \log \sum_{\emptyset_{p,m}=1} \tilde{E}_{p,m,i}/\delta \tag{5.41}$$

where δ is the length of the time-step. The first of these is calculated only for non-null states, while the sum in the second is over all null states. Up to a shared constant term, these two expressions indicate the posterior probabilities of a spike having occurred at time τ from component (p,m) (5.40) and of no spike having occurred (5.41), respectively. The first of these may be seen to be result of a matched filter with impulse response $S_{p,m}(-\tau)$ being applied to the data.

It is here that we make our greedy step. We select the single largest probability from among the values (5.40) and (5.41), over all times τ within the time-step window (in fact, if this maximum lies at the boundary of the interval we extend the search to the closest peak in the filter value). If this is $\mathcal{F}_{\emptyset,i}$ we assume no spike occurred in the interval. In this case the new state estimates are given by

$$E_{p,m,i} = \emptyset_{p,m} \frac{\tilde{E}_{p,m,i}}{\sum_p \emptyset_{p,m} \tilde{E}_{p,m,i}} \tag{5.42}$$

in agreement with (4.43).

If, however, the maximum is achieved by one of the filter outputs, say $\mathcal{F}_{p^*,m^*,i}(\tau^*)$, we assume that the corresponding spike really did occur. In this case we set $E_{p^*,m^*,i}$ to 1 and all other state probabilities for the m^*th model to 0. We then subtract from the data stream the waveform $S_{p,m}(t-\tau^*)$ and recalculate the filter outputs to see if perhaps another spike occurred as well. In practice, since the filters are linear, we can actually subtract the appropriate filtered version of the waveform directly from the filter output. The procedure is then repeated, with the m^*th model discounted. We continue to subtract and repeat until no further spikes are detected.

The procedure described here yields reasonable results in many cases. In the context of non-trivial HMM transition matrices, however, it can be improved upon by the use of the standard Viterbi decoding algorithm of HMM theory, adapted in a manner similar to the coupled forward–backward algorithm discussed in section 4.4. In particular, we note that the forward pass of the decoding does not need to be run to completion before the backward pass (in which the most probable states are identified) can begin. Instead, the optimal sequence can be determined each time a block of nulls of sufficient length is encountered (see section 4.3.2).

5.14 Comparison with Previous Work

Spike sorting is by no means a new problem. Extracellular recording has been a routine electrophysiological method for decades, and single units have been isolated from voltage traces for many years. Nonetheless, it is only quite recently, as multiple electrode recording has become more widespread and as fast computers have become easily available, that interest in fully automatic spike sorting has arisen, and a full statistical analysis of the problem has not, to date, been carried out.

In this section, we review some previous approaches, both manual and automatic, used or proposed for spike sorting. The discussion of prior art has been postponed to this late stage because it is now, armed with the full statistical analysis of the problem, that it will be possible to properly understand the techniques proposed and their shortcomings, if any. We shall find that most approaches to be discussed will address only a subset of the issues brought out in our treatment.

This review of earlier work does not purport to be exhaustive. As might be expected of a subject so fundamental to experimental neuroscience, hundreds of papers have been published on spike sorting. The few that are mentioned below have been selected on two bases: first, they are the best examples of the different common classes of algorithm; and second, in many cases they have been quite influential in the creation of the current work. In some cases, mention of earlier work has already been made in the course of the development above, in which case only a note to that effect will appear here.

5.14.1 Window discriminators

The most basic tool for the detection of spikes in extra-cellular recording is a simple threshold device known as a Schmidt trigger. In the last few decades a slightly more sophisticated version of this venerable tool has come into use, known as the **window discriminator**, and it is this that we shall describe here. The discriminator is usually a hardware device — although the same functionality can easily be implemented on a computer — designed to identify spikes from a single cell. The amplified signal from the electrode is compared to a manually-fixed threshold applied to either the signal voltage or to its derivative. Each time the threshold is triggered, the subsequent waveform is displayed on an oscilloscope (or computer) screen. Observing these waveforms, the user sets a number of time-voltage windows that bracket the waveforms that he wishes to identify as foreground spikes. Any triggered waveform that passes through all of these windows is accepted as a spike, and the time of occurrence is logged.

These devices have typically been used in conjunction with manual isolation of a single spike, so that all that needs to be done with the windows is to distinguish this single waveform from the background. However, software versions of the same device may allow multiple sets of windows to bracket spikes of different shapes (or more than one hardware discriminator may be used on the

same signal), and in some cases spikes from more than one cell can be reasonably detected in this manner.

We can view this procedure as a special case of the manual clustering approach to be described below. The trigger simultaneously extracts and aligns the waveforms. As can be seen from figure 5.5C, as long as the threshold crossing is detected in the analogue signal (that is, there is no, or else only extremely fast, sampling involved) this procedure yields reasonably well-aligned spikes; alignment to a centre of mass is, however, very slightly better. The time-positions of the windows relative to the threshold crossing select the dimensions of the waveform space used to cluster, and the voltage-extents of the windows set the cluster boundaries within this space. Thus, the clustering is constrained to occur within an axis-aligned subspace and the cluster boundaries are constrained to be rectangular. One advantage to this scheme over many standard clustering packages is that it allows the user to select the appropriate dimensions from among all of the axial directions. Another advantage (in terms of manual clustering) is that the high-dimensional space of waveforms is compactly visualized on a two-dimensional screen. Nonetheless, the restrictions on subspace dimensions and on cluster shape can be quite restrictive.

5.14.2 Manual clustering

The advent of multi-wire electrodes, and the availability of commercial software, has popularized the use of clustering approaches to spike sorting. The basic framework of these approaches is as follows. Event waveforms are extracted using a fairly basic threshold trigger. In general, no attempt is made to resample or to realign the event. These waveforms are then grouped into clusters, sometimes by an *ad hoc* clustering algorithm, but often by having the operator draw out the cluster boundaries in various two-dimensional projections. There is no separate spike-detection phase; membership of the clusters, along with the recorded time of threshold crossing, fully specifies the estimated spike identity and time. Examples of procedures of this sort have been described by Abeles and Goldstein (1974), Gray *et al.* (1995), Rebrik *et al.* (1998) and many others.

In general, the clustering is carried out in a subspace of reduced dimension. Above, we pointed out that window discriminators can be viewed as selecting a subset of event coordinates for clustering. Other techniques that have been employed are those that were described in section 5.7.2; hand-picked features, often derived from the spike waveform in a non-linear fashion, are common (see, for example, products from DataWave Technologies), while PCA has also been used (Abeles and Goldstein 1974; Gray *et al.* 1995). In section 5.9.1 we also discussed some proposals to reduce dimensionality in such a way as to suppress spike-shape variability.

Frequently, the cluster shapes are constrained to be rectangular; we pointed out above that this is implicit in the window discrimination approach to clustering, while in many explicit clustering packages it appears to be imposed as a matter of programming convenience. Other computer pack-

ages allow elliptical (for example, the latest product from DataWave Technologies) or more general polygonal (such as the program xclust, written by M. Wilson) boundaries.

In detail, these techniques can certainly be improved in the light of the analysis that has appeared here. Event alignment, discussed in section 5.7.1, would reduce the apparent cluster noise; projection into the noise-whitened robust principal component space, discussed in section 5.7.2, would improve separation. On the issue of the quality of the resultant clustering, however, we expect that the human eye is a sufficiently sophisticated pattern recognition engine to yield fairly accurate results, provided that it is assisted by a proper presentation of the data. One of the advantages to this approach is that it obviates the need to find explicit general models of the spike-shape variability. The operator can, instead, assess the pattern of variability on a cell-by-cell basis. (Of course, clustering packages which restrict the cluster boundaries to be rectangular can hamper this flexibility.)

The difficulties in such methods fall into four groups. First, if the cluster assignments provide the final estimates of spike identity there is no way to resolve overlapped waveforms. Second, the lack of a probabilistic underpinning reduces the degree to which the quality of the solution can be assessed. With probabilistic methods the likelihood of the optimal fit can provide some indication of whether the data have been reasonably modeled or not. Furthermore, a probabilistic technique leads to "soft" or "fuzzy" clusters, which, in turn, lend themselves to the assessment of the degree of confidence with which any given assignment can be made. Both of these features are lacking the "hard" clustering schemes that are commonly used. The third set of issues arises from the fact of human intervention. Spike assignments generated in this fashion may be not be reproducible across different experimenters. Further, the need for considerable experimenter input limits the degree to which the method can be scaled. As we acquire the technology to record from hundreds of electrodes at once, the need for an operator to examine waveforms from each one becomes a prohibitive obstacle. Finally, clustering schemes such as these cannot operate on-line in real time. Thus, they are inappropriate for experiments in which immediate feedback is needed, nor can they be used in neural prosthetic applications.

5.14.3 Automatic techniques

Gaussian models

Lewicki (1994) provides an analysis of the problem that is closest in spirit to that provided here. The model described is based on a single spike waveform per cell, with added spherical Gaussian noise. While the algorithms are derived from an explicitly Bayesian point of view, the resulting steps are similar to those that we describe in section 5.8. Many of the details, however, are different. Thus, Lewicki treats the alignment of the waveform within the sampled event as a latent variable and re-estimates its value on each fitting iteration, while we attempt to eliminate the variation in alignment

by the technique described in section 5.7.1. His model contains no explicit outlier component, and *instead* low occupancy models need to be inspected and possibly rejected by the operator.

A significant difference lies in his approach to the model selection problem. Rather than the cascading model selection procedure that we have proposed, which might be viewed as a form of divisive clustering, he initially fits a mixture with more components than expected and then fuses adjacent clusters together based on the calculation of an approximate Bayes factor.

The most significant shortcoming in Lewicki's proposal is the lack of more sophisticated models for the spike distribution from a single cell. We described in section 5.9 the reasons that we might expect a single Gaussian to be an inadequate model. Similar concerns led Fee *et al.* (1996a) (see below) to abandon the explicitly probabilistic approach. The methods described in this dissertation demonstrate that more powerful models capable of modeling the intrinsic variability in the spike waveforms, can, indeed, be implemented within the probabilistic point of view, thereby gaining all of the advantages implied by that approach.

Agglomerative clustering

In response to Lewicki (1994), Fee *et al.* (1996a) argue, as we did in section 5.9, that in many cases the distribution of waveforms from a single cell does not appear to be Gaussian. They therefore propose an agglomerative clustering scheme which is *ad hoc* in the sense of not being probabilistically founded. The scheme is as follows.

Events are extracted and aligned to a centre of mass calculated in a manner similar, though not identical, to (5.7). The resultant vectors are first partitioned into small clusters by a "recursive bisection" algorithm somewhat similar to divisive k-means. These clusters are then agglomerated into larger groups. Two clusters are grouped together if they exhibit a large "boundary interaction"; that is, roughly, if the density of points in the region of the boundary between them exceeds some threshold.

This may be viewed as an *ad hoc* version of the hierarchical mixture model described in section 5.9.2. The hierarchical mixture provides all the advantages, described above, of the "soft" probabilistic approach. Furthermore, the agglomeration procedure proposed in section 5.9.2 is more satisfying in that it requires explicit overlap of the components. This is made possible by the use of a mixture model, in which the component densities are able to overlap, rather than k-means clustering in which the clusters are compelled to be disjoint.

ART networks

Another proposal that has appeared in the literature is the use of a generic neural network classifier. Oghalai *et al.* (1994) suggest the application of an ART-2 network (the acronym ART comes from the adaptive resonance theory of Carpenter and Grossberg 1987a, 1987b, 1990). This is a neural

network architecture designed for unsupervised clustering problems, and as such appears to be a likely candidate. Closer inspection, however, reveals some weaknesses. In particular, ART implies an odd distance metric in which clusters whose centers have smaller L_1 norms are favoured. Furthermore, as each incoming vector is classified, the center is updated by taking the point-by-point minimum of the old center and the new point. Neither of these details seems to match the noise characteristics we have seen. ART is also a sequential clustering scheme, in which the order in which the data are presented is important. Moore (1989) has argued that it is particularly sensitive to noise in the data. Overall it cannot be thought of as any better than any of the *ad hoc* clustering schemes discussed in section 2.1.

5.14.4 Spike time detection

Some authors have made the same distinction between clustering and spike time detection that we have. In general, they have been motivated by a desire to correctly identify overlapped spikes within the recording, although these techniques may often bring with them the additional benefits that we described in section 5.13.

Lewicki (1994) proposes that the space of all possible waveform overlaps can be searched by the introduction of some approximations and the use of efficient programming techniques. It should be noted that in making this claim, he is addressing detection in the context of a Gaussian clustering model that yields a single mean waveform for each cell. For the more complex distributions, involving multiple components for each cell, the computational difficulty is further increased. Nonetheless, in situations where adequate computational power is available, this is an attractive approach. However, the greedy approximation made in section 5.13 is expected to exhibit slightly improved scaling.

Roberts and Hartline (1975) (see also Roberts 1979) propose an "optimal" linear filtering algorithm, similar to the standard Wiener matched-filter. Expressed in the frequency domain, the filter used to detect the mth spike shape is given by the transform of the associated waveform divided by the sum of the power in the other waveforms and the noise. This filter has the property of responding minimally to the other waveforms (and to noise), while maintaining its output in response to the target waveform at a fixed level. In essence, the filters transform the data to a basis in which the different spike shapes are orthogonal; in this basis overlaps are easily identified.

In the context of the tetrode recordings described here, this approach has not proven to be very successful. The problem seems to be that spike shapes from different cells are spectrally similar enough that the attempted orthogonalization is impossible. The matched filtering technique described in section 5.13 differs from this one in that no effort is made to orthogonalize the targets. Instead, the interaction between the filters is handled explicitly by subtracting the waveform with the largest response from the data and re-filtering. While slower, this approach yields more reliable results.

It should be noted that Gozani and Miller (1994) report success with this technique. Their recordings were made with multiple hook electrodes arranged along a nerve bundle. Spike waveforms might have differed in their propagation velocity along this nerve, a feature which would have facilitated orthogonalization. For cortical tetrode data, or other data recorded within neuropil with a multi-tip electrode, differences in propagation velocity are quite unlikely to be detected.

Chapter 6 Doubly Stochastic Poisson Models

6.1 Introduction

In this chapter we turn from the study of models of spike waveforms, to models of the arrival times of the action potentials invoked in response to an experimental stimulus. The work described here was carried out jointly with J. Linden. The methods that will be discussed have been applied to data[1] collected from the lateral intraparietal area in two macaques during fixation and saccade tasks involving visual and auditory targets. A detailed discussion of this application is presented by Linden (1999).

6.1.1 Point processes

In chapter 5 we examined a variety of statistical models that described the spike waveforms recorded by extracellular electrodes. While the shape of the waveform provided us with information about the identity of the neuron in which the associated action potential occurred, it is not actually used by the nervous system to transmit information between neurons. Instead, from the point of view of the neuron, the action potential is an all-or-nothing pulse: any information that needs to be relayed between cells is carried in the occurrence and timing of the pulses alone.

Statistically, we may view a train of action potentials or spikes[2] from a single neuron as the outcome of a **stochastic point process**. The theory of such processes has been studied extensively in the statistics literature (Cox and Lewis 1966; Cox and Isham 1980; Snyder and Miller 1991). The outcome of a point process may be represented in one of two ways: either as a sequence of N event times $\{\tau_i : i = 1 \ldots N\}$ or as a sequence of T counts $\{x_t : t = 1 \ldots T\}$. The count x_t indicates the number of events that fall within the small interval $[t\delta, (t+1)\delta)$; thus $\sum_t x_t = N$ and $0 \leq t_i < T\delta$. We will always take the intervals to be of the same length, given by the **bin width**, δ. In this chapter we will be concerned solely with the counting representation. It will frequently be useful to collect the counts x_t into the vector, $\mathbf{x}$.

A prominent distribution, that plays a rôle in point-process theory quite similar to that of the Gaussian in continuous random variable theory, is the **Poisson process**. In particular, this is the maximum entropy distribution for a given density of events. Under the Poisson distribution for a counting process each of the counting random variables is independent. A single parameter, ρ_t, the

[1] The data were collected by J. Linden and Dr. A. Grunewald, in Dr. R. A. Andersen's laboratory.
[2] For the purposes of this chapter we need not distinguish between the two.

mean or **rate** of the process, characterizes the distribution of the variable x_t

$$\mathsf{P}_{\rho_t}(x_t) = \frac{e^{-\rho_t}\rho_t^{x_t}}{x_t!} \tag{6.1}$$

Thus the probability of the count vector $\mathbf{x}$, given a rate vector $\boldsymbol{\rho}$ is

$$\mathsf{P}_{\boldsymbol{\rho}}(\mathbf{x}) = \prod_{t=1}^{T} \frac{e^{-\rho_t}\rho_t^{x_t}}{x_t!} \tag{6.2}$$

If ρ_t is the same for each interval the Poisson process is called **homogeneous**. In this chapter we will be primarily concerned with **inhomogeneous** processes.

6.1.2 Spike response variability

Many neurophysiological experiments are conducted as follows. A stimulus is presented to an animal subject and the times of action potentials in one or more neurons in the subject's brain are recorded. The stimulus may well elicit some trained behaviour from the animal: action potentials are recorded for the entire duration of experimental interest around both the stimulus presentation and behavioural event, if any. The same stimulus (and, presumably, behaviour) is then repeated over many different experimental trials, often randomly interleaved with other, similar, stimuli. On each repetition, the times of the action potentials that arise in the same neurons are noted. The result is a database of stimulus-response pairs for each cell.

The neurons of interest in a given experiment usually alter their patterns of firing during the trial, in a manner linked to the presentation of the stimulus or to the execution of the behaviour (or both). Such neurons appear to be related to the processing of either the stimulus or the behavioural response. However, very rarely does a neuron respond to multiple trials in an exactly repeatable manner; this is particularly true of cells in the cerebral cortex of mammals, such as those to be modeled here. This variability in the response of a neuron is what leads us to treat the pattern of spikes as the output of a stochastic process.

Spike trains observed in response to the same stimulus have often been modeled as independently drawn from a single inhomogeneous Poisson process (Perkel *et al.* 1967). In detail such a model must be wrong. Both the refractory period and the presence of bursts violate the independence assumption of the Poisson counting process. However, in situations where the counting intervals are sufficiently large, it has been thought to be a reasonable approximation.

Poisson processes, including those with inhomogeneous rate, have the property that the distribution of counts retains the form (6.1) whatever the choice of the counting interval. In particular, we might select the interval $[0, T)$, to obtain the total spike count during a trial. Provided the original process is Poisson, this count will still be distributed according to (6.1). That distribution has the

property that its variance is equal to its mean.

In practice, the variance in spike count from across repeated, experimentally identical, trials is often larger than can be accounted for by the simple Poisson model (Tolhurst *et al.* 1981; Dean 1981; Tolhurst *et al.* 1983; Vogels *et al.* 1989; Softky and Koch 1993; Gershon *et al.* 1998; Shadlen and Newsome 1998). This same result is apparent in the data to be modeled here (Linden 1999), where the ratio between variance and mean (known as the **Fano factor**) appears to be closer to 1.5 than to 1. One possible source of this additional variance across trials might be slow changes in the overall excitability of neurons or of the cortical area. A number of recent reports have provided direct or indirect evidence for this idea (Brody 1998; Oram *et al.* 1998; also see Tomko and Crapper 1974; Rose *et al.* 1990; Tolhurst *et al.* 1981; Arieli *et al.* 1996). Such slow variation in neuronal excitability might result in an apparently stochastic scaling of the underlying inhomogeneous Poisson rate. This hypothesis will form the basis of the model to be discussed here.

6.2 The Generative Model

The generative model for a spike train $\mathbf{x}$, output by a given cell in response to given experimental conditions, is as follows. The cell-stimulus pair is taken to specify a non-negative **intensity profile**, $\boldsymbol{\lambda}$, that describes the time-course of the cell's response to the stimulus. This profile is scaled by a latent variable, s, which is drawn from a gamma distribution with unit mean, and which is meant to represent the excitability of the neuron on a given trial. The action potential times are then generated by an inhomogeneous Poisson process with rate vector $\boldsymbol{\rho} = s\boldsymbol{\lambda}$.

This model is known in the point process literature as an inhomogeneous Polya process (see Snyder and Miller 1991). It is a special case of the **doubly stochastic Poisson process**: "doubly stochastic" because the Poisson rate is itself a random variable (Cox 1955; Snyder and Miller 1991). Clearly, any such process is a latent variable model. Other examples of doubly stochastic Poisson processes have also been used to model neural spike data by other investigators; for example, some authors have taken the rate to be a piecewise constant function generated from a Markov chain (Radons *et al.* 1994; Abeles *et al.* 1993; Seidemann *et al.* 1996; Gat *et al.* 1997). The present choice is, in part, appealing for its simplicity and relative tractability. As can be seen from the applications discussed by Linden (1999), it can produce useful results.

The standard form of the gamma density (for the scale s) depends on two parameters α and β. It is given by

$$\mathsf{P}_{\alpha,\beta}(s) = \frac{1}{\Gamma(\alpha)\beta^{\alpha}} s^{\alpha-1} e^{-s/\beta} \tag{6.3}$$

It may be easily verified that the mean of this distribution is $\alpha\beta$. Thus, our requirement that the distribution have unit mean constrains the parameters such that $\beta = 1/\alpha$, and we obtain instead

the single parameter density

$$\mathsf{P}_\alpha(s) = \frac{\alpha^\alpha}{\Gamma(\alpha)} s^{\alpha-1} e^{-s\alpha} \tag{6.4}$$

We will refer to the parameter α as the **stability**, since as it grows the variability in spike count drops.

Combining this with the expression for the inhomogeneous Poisson process probability (6.2), we obtain the joint density of a spike train $\mathbf{x}$ being observed along with a scale factor s.

$$\mathsf{P}_{\boldsymbol{\lambda},\alpha}(\mathbf{x}, s) = \left(\prod_{t=1}^{T} \frac{e^{-s\lambda_t}(s\lambda_t)^{x_t}}{x_t!}\right)\left(\frac{\alpha^\alpha}{\Gamma(\alpha)} s^{\alpha-1} e^{-s\alpha}\right) \tag{6.5}$$

The scale, s, is not directly observable, making this a latent variable model. While we may approach learning in this model by the EM algorithm that we have used before, in this case it proves to be useful to obtain a closed form for the marginal distribution function of $\mathbf{x}$, by integrating the joint density of (6.5) with respect to s. The resultant marginal is

$$\mathsf{P}_{\boldsymbol{\lambda},\alpha}(\mathbf{x}) = \left(\prod_{t=1}^{T} \frac{\lambda_t^{x_t}}{x_t!}\right)\left(\frac{\Gamma(X+\alpha)}{\Gamma(\alpha)}\right)\alpha^\alpha(\Lambda+\alpha)^{-(X+\alpha)} \tag{6.6}$$

Here, Λ and X are the sums of the elements in the corresponding vectors: $\Lambda = \sum_{t=1}^{T} \lambda_t$ and $X = \sum_{t=1}^{T} x_t$.

We assume that a set of spike trains, $\mathcal{X} = \{\mathbf{x}_1 \ldots \mathbf{x}_N\}$, collected from the same cell under identical trial conditions, is obtained by drawing each one independently from this distribution. We use the subscript n to identify the spike train and write X_n for the corresponding total spike count. Thus, we obtain the log-likelihood of the parameters $\boldsymbol{\lambda}$ and α under the set of observations $\mathcal{X}$,

$$\ell_{\mathcal{X}}(\boldsymbol{\lambda}, \alpha) = \log Z + \sum_{n=1}^{N}\left(\sum_{t=1}^{T} x_{nt}\log\lambda_t + \log\left(\frac{\Gamma(X_n+\alpha)}{\Gamma(\alpha)}\right) + \alpha\log\alpha - (X_n+\alpha)\log(\Lambda+\alpha)\right) \tag{6.7}$$

where the normalizing constant Z absorbs terms independent of the parameters.

As it stands, this model has a large number of independent degrees of freedom in its parameters. In particular, for small counting intervals and reasonable experimental durations, the vector $\boldsymbol{\lambda}$ may have hundreds of elements. It is impractical to expect reasonable parameter estimates from the small amounts of data that can usually be collected. Therefore, we impose a prior density on the parameters. The prior introduces inter-dependencies between the elements of $\boldsymbol{\lambda}$, reducing the effective number of degrees of freedom.

The stability parameter, α is taken to be independent of the intensity function and is distributed according to the density $e^{-1/\alpha}$. As a result, small values of α are subject to a slight penalty. In practice, this prior is vague enough to have little effect on the parameter estimates and is included

only for completeness.

The prior distribution of the intensity function is a stationary Gaussian process with zero mean and covariance matrix C. The stationarity indicates that we have no prior belief about the course of the intensity function during the experiment. In mathematical terms, it requires that the matrix C be Töplitz (that is, diagonally striped).

The resultant log posterior can be written:

$$\log \mathsf{P}\left(\boldsymbol{\lambda}, \alpha \mid \mathbf{x}_1, \ldots, \mathbf{x}_N\right) = \log Z - \frac{1}{2}\boldsymbol{\lambda}^{\mathsf{T}}\mathsf{C}^{-1}\boldsymbol{\lambda} - \frac{1}{\alpha} + \sum_{n=1}^{N}\left(\mathbf{x}_n^{\mathsf{T}}\log\boldsymbol{\lambda} - (X_n+\alpha)\log(\Lambda+\alpha) + \alpha\log\alpha + \log\left(\frac{\Gamma(X_n+\alpha)}{\Gamma(\alpha)}\right)\right) \tag{6.8}$$

where Z has now absorbed, in addition, the normalization term of the Gaussian.

The reduction in degrees of freedom is achieved by choice of a suitable prior. We select a matrix *which* is based on an auto-covariance function that is Gaussian[3] in shape: that is, the covariance between two elements of the intensity vector λ_s and λ_t under the prior is of the form

$$\mathsf{C}_{st} = \exp\left(-\frac{(s-t)^2}{2\Delta^2}\right) \tag{6.9}$$

The quantity Δ, which is chosen *a priori*, reflects the expected time-scale of changes in the intensity function, expressed in terms of the counting interval length δ. Thus, this choice of prior covariance expresses a belief in the smoothness of the underlying intensity function.

If Δ is fairly large, the matrix C will be ill-conditioned. As such, the inverse that appears in (6.8) creates a numerical instability. This can be resolved by diagonalizing the covariance matrix. Recall that the eigenvectors of any Töplitz matrix are the basis vectors of the discrete Fourier transform (DFT), and so C is diagonalized by the DFT matrix $\mathsf{F}^*_{st} = \frac{1}{\sqrt{T}}\exp(-2\pi i(s-1)(t-1)/T)$. Rather than use this complex form, it will be convenient to introduce a real transform matrix which separates the real and imaginary parts. Such a matrix is given by

$$\hat{\mathsf{F}}_{st} = \frac{1}{\sqrt{T}} \times \begin{cases} 1 & \text{if} \quad s = 1 \\ \cos(2\pi\frac{s}{2}\frac{(t-1)}{T}) & \text{if} \quad s > 1 \text{ and is even} \\ \sin(2\pi\frac{(s-1)}{2}\frac{(t-1)}{T}) & \text{if} \quad s > 1 \text{ and is odd} \end{cases} \tag{6.10}$$

We have assumed that T, the total number of counting intervals, is even.

Thus, the matrix $\hat{\mathsf{F}}\mathsf{C}\hat{\mathsf{F}}^{\mathsf{T}}$ is diagonal, representing the independence of the Fourier components of a stationary process. The ill-conditioning now reveals itself in the presence of one or more diagonal elements that are very close to zero. Thus, in the frequency domain, the ill-conditioning of C is

[3]It is important to distinguish between the Gaussian *distribution* of the prior and the Gaussian *shape* of the auto-covariance. One does not imply the other.

easy to interpret; it reflects the fact that in certain frequencies very little power is expected under the prior. In effect, the prior imposes a band-limitation on the intensity function. The particular choice of Gaussian auto-covariance function, for example, leads to a half-Gaussian shaped fall-off in expected power as frequency increases from 0, with the highest frequencies effectively excluded. It is important to realize, however, that the imposition of this prior is not equivalent to simply filtering the intensity function by the expected frequency profile.

We now restrict the transform matrix to a rectangular form F in which rows corresponding to the eigenvalues of C that fall below some low threshold have been eliminated. Thus the matrix FCF^T is also diagonal, but is of order less than T and is well-conditioned. We will also apply this restricted transform to the intensity function. In doing so, we force the power of the intensity function to zero at those frequencies at which the expected power is vanishingly small.

We proceed to rewrite the posterior (6.8) in terms of this transformed intensity function. In practice, it proves to be useful to represent the intensity function by the transformed logarithm $\phi = \mathsf{F} \log \boldsymbol{\lambda}$ (where the logarithm is taken to apply element by element). The introduction of the logarithm enforces the requirement that the intensity be positive; this would otherwise be difficult to ensure when working in the frequency domain. The log-posterior now becomes

$$\begin{aligned} \log \mathsf{P}\left(\phi, \alpha \mid \mathbf{x}_1, \ldots, \mathbf{x}_N\right) = \log Z - \frac{1}{2} e^{\phi^\mathsf{T} \mathsf{F}} \mathsf{R} e^{\mathsf{F}^\mathsf{T} \phi} - \frac{1}{\alpha} + \langle \mathbf{x} \rangle^\mathsf{T} \mathsf{F}^\mathsf{T} \phi \\ -(\langle \mathbf{x} \rangle^\mathsf{T} \mathbf{1} + N\alpha) \log(e^{\phi^\mathsf{T} \mathsf{F}} \mathbf{1} + \alpha) + N\alpha \log \alpha + \sum_{n=1}^{N} \log \left(\frac{\Gamma(X_n + \alpha)}{\Gamma(\alpha)} \right) \end{aligned} \tag{6.11}$$

where $\langle \mathbf{x} \rangle$ represents the sum of the different observations, $\mathbf{1}$ is a vector of T ones introduced to indicate summation of elements, and $\mathsf{R} = \mathsf{F}^\mathsf{T}(\mathsf{FCF}^\mathsf{T})^{-1}\mathsf{F}$. Exponentiation of a vector term is taken to apply element by element.

6.3 Optimization

We have presented a latent variable model for spike generation. In principle, we might employ the EM algorithm to find the maximum-likelihood — or, given the prior, maximum *a posteriori* — parameter estimates, as we have done with the other latent variable models discussed in this dissertation. Inspection of the joint probability (6.5), however, suggests that this may not be as easy as in our earlier examples. The latent variable, s, will enter into the joint log-likelihood in the logarithm. Thus, calculation of the expected value of this likelihood requires not only the first one or two moments of the latent variable posterior, as in our previous examples, but also the expectation of $\log s$.

To avoid this, we optimize the marginalized posterior (6.11) directly by numerical gradient-based methods. Conceptually, this may be thought of as a simple gradient ascent algorithm, although, in

practice, better results are obtained by use of a quasi-second order method (see, for example, Press *et al.* 1993). Such optimizations can be efficiently executed using numerical methods software such as the MATLAB package.

6.4 Goodness of Fit

While the basic structure of the statistical model described in this chapter has been chosen to embody our beliefs about the origin of neuronal variability, the exact densities used (that is, the gamma and Poisson) have by and large been selected arbitrarily. Both are high entropy distributions, which is appropriate in situations where little constraining knowledge is available, but it must be admitted that, to a significant extent, the choice has been driven by mathematical expediency. In some details, we must expect the model to be incorrect. As was already pointed out, both the refractory period and the tendency of some cells to fire in bursts, violate the independence of counts assumption inherent in the Poisson process. Similarly, we have no guarantee that the scaling will be gamma distributed, nor even that the variability due to excitability can be expressed entirely as multiplicative scaling (on this last point see Linden 1999).

In this section we will investigate through Monte-Carlo means the degree to which the model is appropriate to describe a given set of spike trains recorded in mammalian cortex. These data were collected by J. Linden and A. Grunewald from area LIP of 2 macaque monkeys. For data collection procedures and further information the reader is referred to Linden (1999).

In general, such goodness of fit testing is a difficult problem. We have encountered the issue of model selection repeatedly in this dissertation, where the best of a group of competing models needs to be selected. In this case, though, there is no clear alternative. Based solely on the single model and the available data, we would like to decide whether or not the model is acceptable; that is, whether it is plausible that the data are indeed distributed in the manner specified. The general framework for making such decisions falls within the Neyman-Pearson significance testing literature that is fundamental to traditional developments of statistical theory (see, for example, Hoel *et al.* 1971). Many specific tests have been developed for particular simple distributions (some examples may be found in Zar 1998). For one dimensional data a general technique, known as the Kolmogorov-Smirnov test, is available to assess the validity of an arbitrary distribution (see, for example, Press *et al.* 1993). This can be extended into a small number of dimensions (Fasano and Franceschini 1987), but for more complicated models, describing higher dimensional data, as in the current instance, such straightforward techniques are not available.

Instead, we approach the problem by a novel Monte-Carlo technique, asking whether the obtained likelihood of the best fit model for the observed data matches corresponding values obtained for simulated data known to be generated from the distribution. The steps of the procedure are as

follows.

- Given a set of observed spike trains $\mathcal{X}^o = \{\mathbf{x}_1^o \ldots \mathbf{x}_N^o\}$, find the MAP parameter estimates $\boldsymbol{\lambda}^o$ and α^o.
- Calculate the likelihood on the observed data

$$\ell^o = \ell_{\mathcal{X}^o}(\boldsymbol{\lambda}^o, \alpha^o) \tag{6.12}$$

- Repeat for $s = 1 \ldots S$:
  - Generate a set of simulated spike trains from the optimized model

$$\mathcal{X}^s = \{\mathbf{x}_1^s \ldots \mathbf{x}_N^s\} \sim \text{ iid } \mathsf{P}_{\boldsymbol{\lambda}^o, \alpha^o}(\mathbf{x}) \tag{6.13}$$

  - Re-fit the model to the simulated data $\mathcal{X}^s$ to obtain new MAP estimates $\boldsymbol{\lambda}^s, \alpha^s$.
  - Obtain the optimal likelihood on the simulated data

$$\ell^s = \ell_{\mathcal{X}^s}(\boldsymbol{\lambda}^s, \alpha^s) \tag{6.14}$$

- Find the rank of the observed likelihood within the set of simulated likelihoods

$$r^o = |\{s : \ell^s < \ell^o\}| \tag{6.15}$$

If this procedure is repeated a number of times — each time starting with a different set of observed spike trains, perhaps derived from a different cell — and if the model represents the correct family of distributions, we would expect the resultant ranks to be uniformly distributed between 0 and S.

Two points about the process might require elucidation. First, the simulated data are generated using the MAP parameter values so that the likelihoods measured in the simulations are drawn from the same region of the parameter space as the true likelihoods. Likelihoods under simulated data taken in an an entirely different parameter regime might be quite different. Second, the likelihoods under the simulated data need to be evaluated at the re-fit parameter values so as to avoid a bias due to over-fitting. If this were not done, we would expect the observed likelihoods ℓ^o to be larger than the simulated values, as the parameters would be perfectly tailored to the observed data alone.

In principle, we may now test for uniformity of the ranks by a Kolmogorov-Smirnov or other, more specialized, hypothesis test. In practice it is obvious from inspection that, in this case, the ranks are not uniformly distributed. Figure 6.1 shows the ranks obtained using different groups of

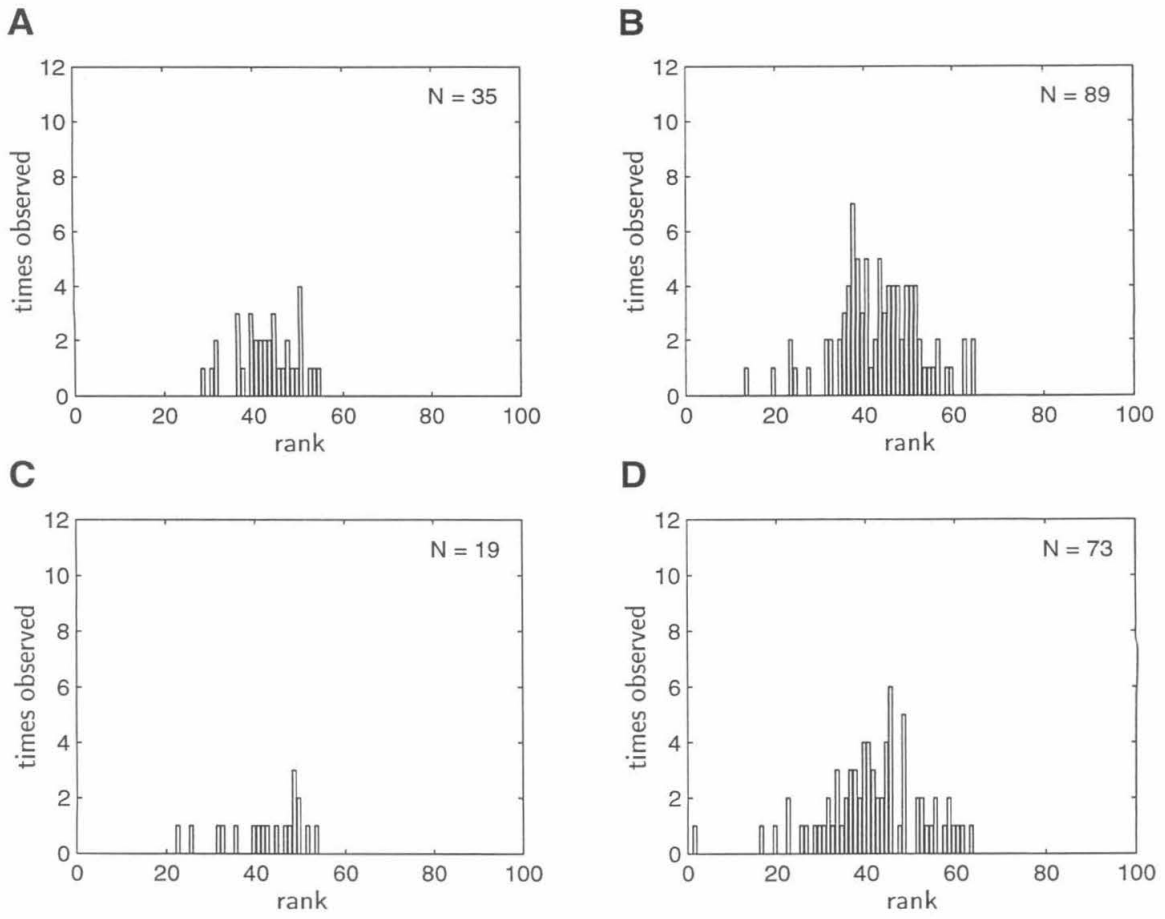


Figure 6.1: Distributions of likelihood ranks

cells under different stimulus conditions. Each panel represents a set of spike trains collected under identical experimental conditions. Only spike trains from cells that appeared to be responsive under the specific conditions were used (the number of these is given by the quoted value of N in each panel), and a single set was taken from each such cell. In each case, the number of simulations, S, was 100.

It is clear from the distributions in figure 6.1 that the ranks are far from uniformly distributed. This suggests that the model we have developed in this chapter is not, in fact, an accurate description of the recorded data. However, had the model been entirely off base, we might have expected the simulated data to almost always have yielded higher best-fit likelihoods than the real observations. For example, if the smoothing invoked by the prior were too severe then the derived intensity function would be greatly inaccurate for the real data, leading to much lower probabilities. Clearly, this is not the case either; almost half the time ℓ^s is smaller than ℓ^o. Thus, we conclude that while the model

is not correct, it is reasonably capable of describing the data. In particular, it would be difficult to *tell, simply by looking* at the optimal likelihood, whether a given set of spike trains were genuine neural data or simply simulations.

A further point of interest in figure 6.1 is that the distributions of ranks obtained for the four different experimental conditions — and frequently, from different cells — are extremely similar. We might take this as evidence that the statistics of the spike trains from these different cells and under these different experimental conditions are actually the same. Thus, while our current model is inadequate, we might hope that by some refinement we can, in fact, find an appropriate model.

6.5 Clustering Spike Trains

It is often a matter of scientific interest to ask whether the cells within a given area of the brain fall into clusters based on the time-courses of their responses to a given stimulus. If such clusters are apparent, they may indicate the presence of distinct sub-populations of neurons that play different rôles in the neural computation.

A common difficulty encountered when attempting to apply traditional clustering techniques such as the k-means algorithm or its variants, to spike trains, is the problem of finding a suitable metric. Such algorithms require a notion of distance between two spike trains, but how is such a distance to be defined? One approach has been to smooth the spike trains, by binning or by convolving with a Gaussian kernel, and then to sample each such smoothed spike train to obtain a vector representation (see, for example, Richmond and Optican 1987; Optican and Richmond 1987; McClurkin *et al.* 1991). These vectors are then treated as though they were embedded in the standard Euclidean inner-product space. There is, however, no *a priori* reason to expect such a distance to be an appropriate metric for spike train clustering. This point is discussed at some length by Victor and Purpura (1997), who propose an alternative metric, though also on an *ad hoc* basis.

Fortunately, we can avoid this problem. In chapter 2 we saw that, in many cases, the generative modeling approach to clustering is to be preferred. In particular, this is true if we are interested in identifying the process from which the observed data arose, rather than simply grouping the data themselves. The appropriate generative model in such situations is the mixture model given by the weighted sum of M component distributions:

$$\mathsf{P}_{\boldsymbol{\theta}}(\mathbf{x}) = \sum_{m=1}^{M} \pi_m \mathsf{P}_{\boldsymbol{\theta}_m}(\mathbf{x}) \tag{6.16}$$

The parameters of the mixture decompose into independent and disjoint sets $\boldsymbol{\theta} = (\boldsymbol{\theta}_1 \ldots \boldsymbol{\theta}_M, \pi_1 \ldots \pi_M)$, where the parameters $\boldsymbol{\theta}_m$ describe the mth component or cluster. Learning algorithms for such mix-

tures were discussed at length in chapters 2 and 3.

Such an approach effectively sidesteps the issue of identifying a suitable metric within the space of spike trains. The clusters are no longer described within the observation space; instead, they are described by the parameters $\boldsymbol{\theta}_m$ which live in a different space altogether. We no longer need to compute the separation between two spike trains: we need only find the "distance" between a spike train and the cluster parameters. A natural candidate for such a distance is obvious: the probability of the spike train under the cluster model. Thus, the probabilistic treatment espoused throughout this dissertation allows us to rigourously arrive at a unique clustering solution from only a few explicitly stated assumptions about the distributions of spike trains.

To this point, we have regarded each spike train $\mathbf{x}_n$ as a separate observation; now, we will instead treat all of the spike trains collected from the same cell under the same experimental conditions as a single outcome of the generative model. For the ith cell-experiment pair we can collect the N_i individual count vectors into a matrix X_i, in which each count vector appears as a column. Careful inspection of the probability (6.7) reveals that, in fact, we are only interested in the marginal sums of this matrix. Thus, we compute and store the following sufficient statistics: the sum of the count vectors $\mathsf{X}_i\mathbf{1}$, the vector of total spike counts $\mathsf{X}_i^\mathsf{T}\mathbf{1}$, and the total of all the elements $\mathbf{1}^\mathsf{T}\mathsf{X}_i\mathbf{1}$. In these expressions the vector $\mathbf{1}$ should be taken to contain either T or N_i ones as appropriate.

We can then write the form of the mth component probability distribution, written in terms of the Fourier domain intensity ϕ_m and the stability α_m,

$$\mathsf{P}_m\left(\mathsf{X}_i\right) \propto e^{\phi_m^\mathsf{T}\mathsf{F}\mathsf{X}_i\mathbf{1}}\alpha_m^{N_i\alpha_m}\left(e^{\phi_m^\mathsf{T}\mathsf{F}}\mathbf{1}+\alpha_m\right)^{-\left(\mathbf{1}^\mathsf{T}\mathsf{X}_i\mathbf{1}+N_i\alpha_m\right)}\exp\left[\mathbf{1}^\mathsf{T}\log\left(\frac{\Gamma(\mathsf{X}_i^\mathsf{T}\mathbf{1}+\alpha_m\mathbf{1})}{\Gamma(\alpha_m)}\right)\right] \tag{6.17}$$

In the final factor, the gamma function and the logarithm should be taken to apply element by element. We have left out a factor given by the product of the factorials of each of the elements in X_i. This factor is identical across all of the component distributions and thus has no impact on any of the optimization algorithms and need never be computed.

We then fit a mixture model for the entire ensemble of recordings taken across multiple cells $\mathcal{X} = \{\mathsf{X}_i\}$, given by $\mathsf{P}_{\boldsymbol{\theta}}\left(\mathcal{X}\right) = \prod_i \sum_m \mathsf{P}_m\left(\mathsf{X}_i\right)$. In doing so, we assume that a "cluster" of spike trains are such that they may have arisen from exactly the same intensity function, although with possibly different scalings. The "extent" of the cluster is defined by the model, as well as by the learned value of the stability parameter.

For the single component model, the introduction of the prior was important to achieve regularized estimation. In the mixture, this regularization is, if anything, more important as the complexity of the model has increased. We choose the prior on the parameter set $\{\phi_m\} \cup \{\alpha_m\}$ to factor over the different components; that is, the intensity function and stability for one component are *a priori* independent of those of any other component distribution. For any one component we choose the

priors on ϕ_m and α_m to be exactly as before. The covariance matrix C is taken to be common to all of the clusters. The mixing parameters π_m are subject to a uniform prior: this does not affect the results of the estimation and will be not be written explicitly.

The basic EM algorithm suitable for learning in such models was described in section 2.4. We recall that the E-step involves computation of responsibilities according to (2.9)

$$r_{m,i} = \frac{\pi_m \mathsf{P}_m(x_i)}{\sum_l \pi_l \mathsf{P}_l(x_i)} \tag{6.18}$$

where, the component distributions are given by (6.17). The M-step update of the mixing probabilities is common to all mixture models (2.12)

$$\pi_m \leftarrow \frac{\sum_i r_{m,i}}{|\mathcal{X}|} \tag{6.19}$$

The update of the component parameters in the maximum likelihood context of chapter 2 was given by (2.15)

$$\theta_m \leftarrow \operatorname*{argmax}_{\theta_m} \sum_i r_{m,i} \log \mathsf{P}_{\theta_m}(\mathsf{X}_i) \tag{6.20}$$

where θ_m stands for the parameters of the mth component. In the present example, however, we have a non-trivial prior distribution on the component parameters. Given our assumption that the prior factorizes over the different models, we can correct (6.20) by the addition of the log-prior for the mth model to the right hand side. The updated parameters of the mth component are thus obtained by optimizing the expression

$$\begin{aligned} Q(\phi_m, \alpha_m) = \log Z - \frac{1}{2} e^{\phi_m^\mathsf{T} \mathsf{F}} \mathsf{R} e^{\mathsf{F}^\mathsf{T} \phi_m} - \frac{1}{\alpha_m} \\ + \sum_i r_{m,i} \quad \Big[\phi_m^\mathsf{T} \mathsf{F} \mathsf{X}_i \mathbf{1} - (\mathbf{1}^\mathsf{T} \mathsf{X}_i \mathbf{1} + N\alpha_m) \log(e^{\phi_m^\mathsf{T} \mathsf{F}} \mathbf{1} + \alpha_m) \\ + N\alpha_m \log \alpha_m + \mathbf{1}^\mathsf{T} \log \left(\frac{\Gamma(\mathsf{X}_i^\mathsf{T} \mathbf{1} + \alpha_m)}{\Gamma(\alpha_m)} \right) \Big] \end{aligned} \tag{6.21}$$

As before, this optimization must be performed numerically, and thus, the computational cost of the M-step is considerably greater than that of the E-step. It is useful to recall the Generalized EM (GEM) algorithm, mentioned briefly in section 1.8, in which the M-step is only partially completed; that is, the free energy is increased by the update of the parameters, but not necessarily maximized. This generalization shares the guaranteed convergence with the standard EM algorithm, but is more efficient. In the present case, this partial completion is equivalent to executing only a limited number of steps of the numerical optimization at each M-step.

The GEM algorithm described above was run on a subset of the data described previously, that was collected from different cells under the same experimental conditions. The results are shown

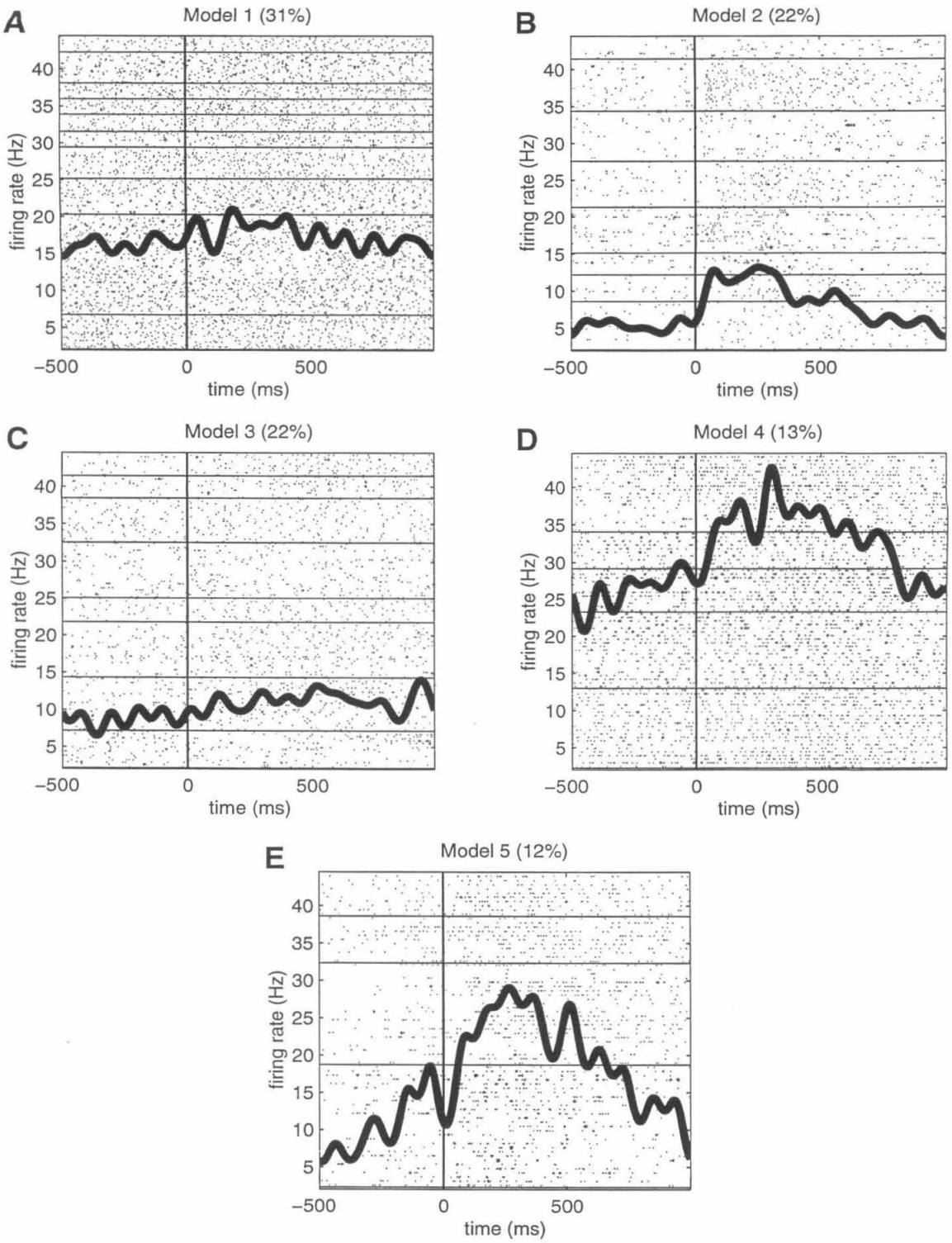


Figure 6.2: Clusters of spike trains

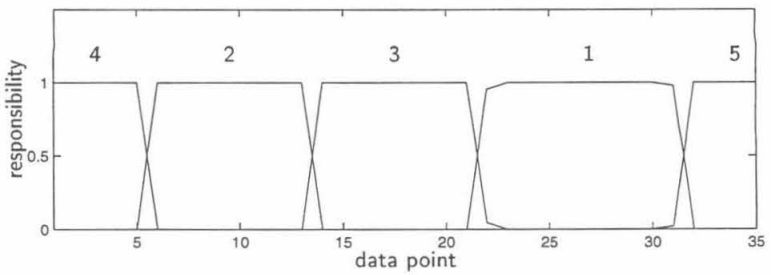


Figure 6.3: Responsibilities of the different models.

in figure 6.2. The size of the model was determined by the BIC penalized likelihood procedure (see section 1.3), which yielded a mixture of five components. The intensity function learned for each of these components is shown by the heavy black line in each panel of the figure. The mixing probabilities are indicated by the percentage figures above each panel. Cells have been assigned to the most likely cluster (that is, the one with the largest responsibility for the data from the cell), and the corresponding spike trains then shown in the background of the appropriate panel. The representation is similar to the conventional spike raster diagram: each row of dots represents a single trial; the presence of a dot time indicates that at least one spike was counted in a 5ms window around that time; the size of the dot indicates the number of spikes. The horizontal black lines separate spike trains from different cells.

Do the spike trains classified in figure 6.2 really fall into five distinct clusters? The fact that BIC model selection rejected the option of more components in the mixture suggests that this may well be the case. As a further reassurance we can examine the posterior assignment probabilities, or responsibilities (6.18), under the maximum likelihood solution. These values indicate the surety with which each data point is assigned to each cluster. If the components tended to share the responsibility for each spike train it would suggest that the clusters were not well separated. The responsibilities of each of the five component models are shown in figure 6.3. Each line shows the assignment probabilities of one model, indicated by the number above the line, for all of data; the data have been reordered to group spike trains assigned to the same cluster together. In all cases, only one model has high responsibility, very close to 1. This suggests that the clusters shown in figure 6.2 really are well separated.

6.6 Summary

In this chapter we have introduced a latent variable model to describe spike trains generated by a neuron under constant experimental conditions. The model is designed to capture certain recent observations about the statistics of neural responses: in particular, the fact that the variability in

cortical spike trains is often greater than that predicted by the Poisson process assumption, and that in many cases this greater variability might result from changes in the overall excitability of the neuron or cortical area. Although the EM algorithm involves a difficult E-step, it proves to be possible to fit the model by direct numerical optimization.

Using a Monte-Carlo goodness of fit procedure, we saw that the model does not describe the statistics of spiking exactly. However, the maximal likelihood values for the best-fit model under real neural data are quite similar to the values under simulated data generated from the model itself. Thus, we conclude that model is a reasonable, but not exact description.

The statistical model provides a rigorous foundation on which to base two analyses of neural data. First, maximum *a posteriori* optimization of the model with a suitable prior imposed on the parameters, leads to a smoothed estimate of the underlying spike-rate intensity. This technique provides a solid statistical basis for the smoothing, as well as correctly accounting for biases that might be introduced by any variable excitability. Second, by use of a mixture of such models, we are able to identify clusters of cells whose spike trains in response to the same stimuli are similar. *Ad hoc* methods for clustering spike trains suffer from the serious difficulty of the absence of a natural metric. In contrast, the probabilistic procedure avoids the issue of a distance measure entirely, and leads to a natural clustering algorithm.

Bibliography

Abeles, M., H. Bergman, E. Margalit, and E. Vaadia (1993). Spatiotemporal firing patterns in the frontal- cortex of behaving monkeys. *Journal of Neurophysiology 70*(4), 1629–1638.

Abeles, M. and M. H. Goldstein (1974). Multispike train analysis. *Proceedings of the IEEE 65*(5), 762–773.

Akaike, H. (1974). A new look at statistical model identification. *IEEE Transactions on Automatic Control 19*, 716–723.

Alonso, J.-M. and L. M. Martinez (1998). Functional connectivity between simple cells and complex cells in cat striate cortex. *Nature Neuroscience 1*(5), 395–403.

Anderson, T. W. (1963). Asymptotic theory for principal component analysis. *Annals of Mathematical Statistics 34*, 122–148.

Arieli, A., A. Sterkin, A. Grinvald, and A. Aertsen (1996). Dynamics of ongoing activity: explanation of large variability in evoked cortical responses. *Science 273*, 1868–1871.

Attias, H. (1999). Independent factor analysis. *Neural Computation 11*(4), 803–851.

Attias, H. and C. E. Schreiner (1998). Blind source separation and deconvolution: The dynamic component analysis algorithm. *Neural Computation 10*, 1373–1424.

Backer, E. (1978). *Cluster Analysis by Optimal Decomposition of Induced Fuzzy Sets.* Delft, The Netherlands: Delft University Press.

Banfield, J. D. and A. E. Raftery (1993). Model-based Gaussian and non-Gaussian clustering. *Biometrics 49*, 803–821.

Bell, A. and T. Sejnowski (1995). An information maximization approach to blind separation and blind deconvolution. *Neural Computation 7*(6), 1129–1159.

Bertsimas, D. and J. Tsitsiklis (1993). Simulated annealing. *Statistical Science 8*, 10–15.

Bezdek, J. C. (1981). *Pattern Recognition with Fuzzy Objective Function Algorithms.* New York, NY: Plenum Press.

Boyen, X. and D. Koller (1999). Approximate learning of dynamic models. In M. S. Kearns, S. A. Solla, and D. A. Cohn (Eds.), *Advances in Neural Information Processing Systems*, Volume 11. MIT Press.

Breiman, L. (1996). Stacked regressions. *Machine Learning 24*(1), 49–64.

Brody, C. D. (1998). Slow covariations in neuronal resting potentials can lead to artifactually fast cross-correlations in their spike trains. *Journal of Neurophysiology 80*, 3345–3351.

Brown, G. D., S. Yamada, H. Luebben, and T. J. Sejnowski (1998). Spike sorting and artifact rejection by independent component analysis of optical recordings from Tritonia. *Society for Neurocience Abstracts 24*, 655.3.

Buhmann, J. and H. Kuhnel (1993). Vector quantization with complexity costs. *IEEE Transactions on Information Theory 39*(4), 1133–1145.

Buzsaki, G. and A. Kandel (1998). Somadendritic backpropagation of action potentials in cortical pyramidal cells of the awake rat. *Journal of Neurophysiology 79*(3), 1587–1591.

Carpenter, G. A. and S. Grossberg (1987a). ART 2: Stable self-organization of stable category recognition codes for analog input patterns. *Applied Optics 26*, 4919–4930.

Carpenter, G. A. and S. Grossberg (1987b). A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision, Graphics and Image Processing 37*, 54–115.

Carpenter, G. A. and S. Grossberg (1990). ART 3: Hierarchical search using chemical transmitters in self-organizing pattern recognition architectures. *Neural Networks 3*, 129–152.

Cheeseman, P. and J. Stutz (1996). Bayesian classification (AutoClass): Theory and results. In U. Fayyad, G. Piatesky-Shapiro, P. Smyth, and R. Uthurasamy (Eds.), *Advances in Knowledge Discovery and Data Mining*, pp. 153–180. Menlo Park, CA: AAAI Press.

Comon, P. (1994). Independent component analysis, a new concept. *Signal Processing 36*(3), 287–314.

Cover, T. M. and J. A. Thomas (1991). *Elements of Information Theory*. New York: Wiley.

Cox, D. R. (1955). Some statistical methods connected with series of events. *Journal of the Royal Statistical Society Series B 17*, 129–164.

Cox, D. R. and V. Isham (1980). *Point Processes*. Monographs on Applied Probability and Statistics. Chapman and Hall.

Cox, D. R. and P. A. W. Lewis (1966). *The Statistical Analysis of Series of Events*. Monographs on Applied Probability and Statistics. Chapman and Hall.

Dayan, P., G. E. Hinton, R. M. Neal, and R. S. Zemel (1995). The Helmholtz machine. *Neural Computation 7*(5), 889–904.

Dean, A. F. (1981). The variability of discharge of simple cells in the cat striate cortex. *Experimental Brain Research 44*, 437–440.

Dempster, A. P., N. M. Laird, and D. B. Rubin (1977). Maximum likelihood from incomplete data via the EM algorithm (with discussion). *Journal of the Royal Statistical Society B 39*, 1–38.

Duda, R. O. and P. E. Hart (1973). *Pattern Classification and Scene Analysis*. John Wiley & Sons.

Durbin, R., R. Szeliski, and A. Yuille (1989). An analysis of the elastic net approach to the traveling salesman problem. *Neural Computation 1*(3), 348–358.

Durbin, R. and D. Willshaw (1987). An analog approach to the traveling salesman problem using an elastic net method. *Nature 326*(6114), 689–691.

Fasano, G. and A. Franceschini (1987). A multidimensional version of the Kolmogorov-Smirnov test. *Monthly Notices of the Royal Astronomical Society 225*, 155–170.

Fee, M. S., P. P. Mitra, and D. Kleinfeld (1996a). Automatic sorting of multiple-unit neuronal signals in the presence of anisotropic and non-Gaussian variability. *Journal of Neuroscience Methods 69*(2), 175–188.

Fee, M. S., P. P. Mitra, and D. Kleinfeld (1996b). Variability of extracellular spike waveforms of cortical neurons. *Journal of Neurophysiology 76*(3), 3823–3833.

Forgy, E. W. (1965). Cluster analysis of multivariate data: Efficiency vs interpretability of classifications. *Biometrics 21*, 768–769.

Fuchs, A. F. and D. A. Robinson (1966). A method for measuring horizontal and vertical eye movement chronically in the monkey. *Journal of Applied Physiology 21*, 1068–1070.

Gat, I., N. Tishby, and M. Abeles (1997). Hidden Markov modelling of simultaneously recorded cells in the associative cortex of behaving monkeys. *Network: Computation in Neural Systems 8*(3), 297–322.

Geiger, D., D. Heckerman, and C. Meek (1998). Asymptotic model selection for directed networks with hidden variables. See Jordan (1998), pp. 461–477.

Gelfand, A. and A. Smith (1990). Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association 85*(410), 398–409.

Geman, S. and D. Geman (1984). Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis 6*, 721–741.

Gershon, E. D., M. C. Wiener, P. E. Latham, and B. J. Richmond (1998). Coding strategies in monkey V1 and inferior temporal cortices. *Journal of Neurophysiology 79*, 1135–1144.

Ghahramani, Z. and G. E. Hinton (1996). The EM algorithm for mixtures of factor analyzers. Technical Report CRG-TR-96-1, University of Toronto.

Ghahramani, Z. and M. I. Jordan (1997). Factorial hidden Markov models. *Machine Learning 29*, 245–273.

Gozani, S. N. and J. P. Miller (1994). Optimal discrimination and classification of neuronal action-potential waveforms from multiunit, multichannel recordings using software-based linear filters. *IEEE Transactions on Biomedical Engineering 41*(4), 358–372.

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

Gray, C. M. and W. Singer (1989). Stimulus-specific neuronal oscillations in orientation columns of cat visual-cortex. *Proceedings of the National Academy of Sciences of the United States of America 86*(5), 1698–1702.

Hall, D. J. and G. B. Ball (1965). ISODATA: A novel method of data analysis and pattern classification. Technical report, Stanford Research Institute, Menlo Park, CA.

Hartigan, J. A. and M. A. Wong (1979). Algorithm AS136. A K-means clustering algorithm. *Applied Statistics 28*, 100–108.

Haughton, D. M. A. (1988). On the choice of a model to fit data from an exponential family. *The Annals of Statistics 16*, 342–355.

Hertz, J. A., A. Krogh, and R. G. Palmer (1991). *Introduction to the Theory of Neural Computation.* Santa Fe Institute Studies in the Sciences of Complexity. Redwood City, CA: Addison–Wesley.

Hinton, G. E., P. Dayan, B. J. Frey, and R. M. Neal (1995). The "wake-sleep" algorithm for unsupervised neural networks. *Science 268*(5214), 1158–1161.

Hodgkin, A. L. and A. F. Huxley (1952). A quantitative description of membrane current and its application to conduction and excitation in nerve. *Journal of Physiology – London 117*, 500–544.

Hoel, P. G., S. C. Port, and C. J. Stone (1971). *Introduction to Statistical Theory.* Houghton Mifflin.

Jain, A. K. and R. C. Dubes (1988). *Algorithms for Clustering Data.* Englewood Cliffs, N.J.: Prentice Hall.

Jordan, M. I. (Ed.) (1998). *Learning in Graphical Models.* Kluwer Academic Press.

Judge, S. J., B. J. Richmond, and F. C. Shu (1980). Implantation of magnetic search coils for measurement of eye position: an improved method. *Vision Research 20*, 535–537.

Jutten, C. and J. Herault (1991). Blind separation of sources, Part I: An adaptive algorithm based on neuromimetic architecture. *Signal Processing 24*(1), 1–10.

Karlin, S. (1991). *A Second Course in Stochastic Processes.* London: Academic Press.

Kass, R. E., L. Tierney, and J. B. Kadane (1990). The validity of posterior asymptotic expansions based on Laplace's method. In S. Geisser, J. S. Hodges, S. J. Press, and A. Zellner (Eds.), *Bayesian and Likelihood Methods in Statistics and Econometrics.* New York: North-Holland.

Kirkpatrick, S., C. Gelatt, and M. Vecchi (1983). Optimization by simulated annealing. *Science 220*(4598), 671–680.

Kittel, C. and H. Kroemer (1980). *Thermal Physics* (2nd ed.). New York: W. H. Freeman and Company.

Kloppenburg, M. and P. Tavan (1997). Deterministic annealing for density estimation by multivariate normal mixtures. *Physical Review E 55*(3), 2089–2092.

Lauritzen, S. L. (1996). *Graphical Models.* Oxford: Oxford University Press.

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

Linden, J. (1999). *Responses to auditory stimuli in macaque lateral intraparietal area.* Ph. D. thesis, California Institute of Technology, Pasadena, California.

MacKay, D. J. C. (1992). Bayesian interpolation. *Neural Computation 4*(3), 415–447.

MacKay, D. J. C. (1999). Maximum likelihood and covariant algorithms for independent component analysis. Unpublished manuscript available from `ftp://wol.ra.phy.cam.ac.uk/pub/mackay/ica.ps.gz`.

Mathews, J. and R. L. Walker (1970). *Mathematical Methods of Physics* (2nd ed.). New York: W. A. Benjamin.

McClurkin, J. W., T. J. Gawne, L. M. Optican, and B. J. Richmond (1991). Lateral geniculate neurons in behaving primates. II. Encoding of visual information in the temporal shape of the response. *Journal of Neurophysiology 66*(3), 794–808.

McLachlan, G. J. and T. Krishnan (1996). *The EM Algorithm and Extensions.* Wiley Series in Probability and Statistics. John Wiley & Sons.

McNaughton, B. L., J. O'Keefe, and C. A. Barnes (1983). The stereotrode - a new technique for simultaneous isolation of several single units in the central nervous-system from multiple unit records. *Journal of Neuroscience Methods 8*(4), 391–397.

McQueen, J. B. (1967). Some methods of classification and analysis of multivariate observations. In *Proceedings of Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pp. 281–297.

Miller, D., A. Rao, K. Rose, and A. Gersho (1996). A global optimization technique for statistical classifier design. *IEEE Transactions on Signal Processing 44*(12), 3108–3122.

Moore, B. (1989). ART 1 and pattern clustering. In D. Touretzky, G. E. Hinton, and T. Sejnowski (Eds.), *Proceedings of the 1988 Connectionist Models Summer School*, San Mateo, CA, pp. 174–185. Morgan Kaufmann.

Mountcastle, V. B., J. C. Lynch, A. Georgeopoulos, H. Sakata, and C. Acuna (1975). Posterior parietal association cortex of the monkey: command functions for operations within extrapersonal space. *Journal of Neurophysiology 38*(2), 871–908.

Murata, N., S. Yoshizawa, and S. Amari (1991). A criterion for determining the number of parameters in an artificial neural network model. In T. Kohonen, K. Mäkisara, O. Simula, and J. Kangas (Eds.), *Artificial Neural Networks. Proceeding of ICANN-91*, Volume I, Amsterdam, pp. 9–14. North Holland.

Murata, N., S. Yoshizawa, and S. Amari (1993). Learning curves, model selection and complexity of neural networks. In S. J. Hanson, J. D. Cowan, and C. L. Giles (Eds.), *Advances in Neural Information Processing Systems*, Volume 5, San Mateo, pp. 607–614. Morgan Kaufmann.

Murata, N., S. Yoshizawa, and S. Amari (1994). Network information criterion – determining the number of hidden units for artificial neural network models. *IEEE Transactions on Neural Networks 5*, 865–872.

Neal, R. M. (1991). Bayesian mixture modeling by Monte-Carlo simulation. Technical Report CRG-TR-91-2, Department of Computer Science, University of Toronto.

Neal, R. M. (1993). Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG-TR-93-1, Department of Computer Science, University of Toronto.

Neal, R. M. (1996). *Bayesian Learning for Neural Networks*, Volume 118 of *Lecture Notes in Statistics*. New York: Springer Verlag.

Neal, R. M. and G. E. Hinton (1998). A view of the EM algorithm that justifies incremental, sparse, and other variants. See Jordan (1998), pp. 355–370.

Nowlan, S. J. (1991). *Soft Competetive Adaptation: Neural Network Learning Algorithms Based on Fitting Statistical Mixtures*. Ph. D. thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh.

Oghalai, J. S., W. N. Street, and W. S. Rhode (1994). A neural network-based spike discriminator. *Journal of Neuroscience Methods 54*, 9–22.

Optican, L. M. and B. J. Richmond (1987). Temporal encoding of two-dimensional patterns by single units in primate inferior temporal cortex. III. Information theoretic analysis. *Journal of Neurophysiology 57*(1), 162–178.

Oram, M. W., R. Lestienne, M. C. Wiener, and B. J. Richmond (1998). Accurately predicting precisely replicating spike patterns in neural responses of monkey striate cortex and LGN. *Society for Neurocience Abstracts 24*, 1258.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* San Mateo, CA: Morgan Kaufmann.

Perkel, D. H., G. L. Gerstein, and G. P. Moore (1967). Neuronal spike trains and stochastic point processes I. The single spike train. *Biophysical Journal 7*, 391–418.

Pezaris, J. S., M. Sahani, and R. A. Andersen (1997). Tetrodes for monkeys. In J. M. Bower (Ed.), *Computational Neuroscience: Trends in Research, 1997.* Plenum.

Press, W. H., S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery (1993). *Numerical Recipes in C: The Art of Scientific Computing* (2nd ed.). Cambridge: Cambridge University Press.

Radons, G., J. D. Becker, B. Dülfer, and J. Krüger (1994). Analysis, classification, and coding of multielectrode spike trains with hidden Markov models. *Biological Cybernetics 71*, 359–373.

Rao, A., D. Miller, K. Rose, and A. Gersho (1997). Mixture of experts regression modeling by deterministic annealing. *IEEE Transactions on Signal Processing 45*(11), 2811–2820.

Rao, A., D. Miller, K. Rose, and A. Gersho (1999). A deterministic annealing approach for parsimonious design of piecewise regression models. *IEEE Transactions on Pattern Analysis and Machine Intelligence 21*(2), 159–173.

Rebrik, S., S. Tzonev, and K. D. Miller (1998). Analysis of tetrode recordings in cat visual system. In J. M. Bower (Ed.), *Computational Neuroscience: Trends in Research, 1998.* Plenum.

Recce, M. L. and J. O'Keefe (1989). The tetrode: An improved technique for multi-unit extracellular recording. *Society for Neuroscience Abstracts 15*(2), 1250.

Redner, R. A. and H. F. Walker (1984). Mixture densities, maximum likelihood, and the EM algorithm. *SIAM Review 26*, 195–239.

Richmond, B. J. and L. M. Optican (1987). Temporal encoding of two-dimensional patterns by single units in primate inferior temporal cortex. II. Quantification of response waveform. *Journal of Neurophysiology 57*(1), 147–161.

Rinberg, D., H. Davidowitz, and N. Tishby (1999). Multi-electrode spike sorting by clustering transfer functions. In M. S. Kearns, S. A. Solla, and D. A. Cohn (Eds.), *Advances in Neural Information Processing Systems*, Volume 11. MIT Press.

Ripley, B. D. (1996). *Pattern recognition and Neural networks.* Cambridge: Cambridge University Press.

Roberts, W. M. (1979). Optimal recognition of neuronal waveforms. *Biological Cybernetics 35*, 73–80.

Roberts, W. M. and D. K. Hartline (1975). Separation of multi-unit nerve impulse trains by a multi-channel linear filter algorithm. *Brain Research 94*, 141–149.

Robinson, D. A. (1968). The electrical properties of metal microelectrodes. *Proceedings of the IEEE 56*(6), 1065–1071.

Rose, K. (1998). Deterministic annealing for clustering, compression, classification, regression, and related optimization problems. *Proceedings of the IEEE 86*(11), 2210–2239.

Rose, K., E. Gurewitz, and G. Fox (1993). Constrained clustering as an optimization method. *IEEE Transactions on Pattern Analysis and Machine Intelligence 15*(8), 785–794.

Rose, K., E. Gurewitz, and G. C. Fox (1990). Statistical mechanics and phase transitions in clustering. *Physical Review Letters 65*(8), 945–948.

Roweis, S. (1998). EM algorithms for PCA and SPCA. In M. I. Jordan, M. J. Kearns, and S. A. Solla (Eds.), *Advances in Neural Information Processing Systems*, Volume 10. The MIT Press.

Schwartz, G. (1978). Estimating the dimension of a model. *The Annals of Statistics 6*, 461–464.

Scott, A. J. and M. J. Symons (1971). Clustering methods based on likelihood ratio criteria. *Biometrics 27*, 387–397.

Seidemann, E., I. Meilijson, M. Abeles, H. Bergman, and E. Vaadia (1996). Simultaneously recorded single units in the frontal cortex go through sequences of discrete and stable states in monkeys performing a delayed localization task. *Journal of Neuroscience 16*(2), 752–768.

Seneta, E. (1981). *Non-Negative Matrices and Markov Chains* (2nd ed.). Springer Series in Statistics. New York: Springer.

Shadlen, M. N. and W. T. Newsome (1998). The variable discharge of cortical neurons: implications for connectivity, computation, and information coding. *Journal of Neuroscience 18*(10), 3870–3896.

Simic, P. D. (1990). Statistical mechanics as the underlying theory of 'elastic' and 'neural' optimisations. *Network: Computation in Neural Systems 1*(1), 89–103.

Smith, A. and G. Roberts (1993). Bayesian computation via the Gibbs sampler and related Markov-chain Monte-Carlo methods. *Journal of the Royal Statistical Society Series B-Methodological 55*(1), 3–23.

Snyder, D. L. and M. I. Miller (1991). *Random Point Processes in Time and Space* (2nd ed.). New York: Springer Verlag.

Softky, W. R. and C. Koch (1993). The highly irregular firing of cortical-cells is inconsistent with temporal intergration of random EPSPs. *Journal of Neuroscience 13*(1), 334–350.

Spiegelhalter, D. J., A. Thomas, and N. G. Best (1996). Computation on Bayesian graphical models. In J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith (Eds.), *Bayesian statistics 5*, pp. 407–425. Oxford: Clarendon Press.

Spruston, N., Y. Schiller, G. Stuart, and B. Sakmann (1995). Activity-dependent action-potential invasion and calcium influx into hippocampal CA1 dendrites. *Science 268*(5208), 297–300.

Stone, M. (1974). Cross-validatory choice and assessment of statistical predictions (with discussion). *Journal of the Royal Statistical Society series B 36*, 111–147.

Stuart, G. and B. Sakmann (1994). Active propagation of somatic action-potentials into neocortical pyramidal cell dendrites. *Nature 367*(6458), 69–72.

Stuart, G., J. Schiller, and B. Sakmann (1997). Action potential initiation and propagation in rat neocortical pyramidal neurons. *Journal of Physiology – London 505*(3), 617–632.

Svoboda, K., W. Denk, D. Kleinfeld, and D. Tank (1997). In vivo dendritic calcium dynamics in neocortical pyramidal neurons. *Nature 385*(6612), 161–165.

Thomas, A. (1994). BUGS: a statistical modelling package. *RTA/BCS Modular Languages Newsletter 2*, 36–38.

Tierney, L. and J. B. Kadane (1986). Accurate approximations for posterior moments and marginal densities. *Journal of the American Statistical Association 81*, 82–86.

Tipping, M. E. and C. M. Bishop (1997). Probabilistic principal component analysis. Technical Report NCRG/97/010, Neural Computing Research Group, Aston University, Birmingham B4 7ET U. K. Available from
`http://neural-server.aston.ac.uk/Papers/postscript/NCRG_97_010.ps.Z`.

Tolhurst, D. J., J. A. Movshon, and A. F. Dean (1983). The statistical reliability of signals in single neurons in cat and monkey visual cortex. *Vision Research 23*(8), 775–785.

Tolhurst, D. J., J. A. Movshon, and I. D. Thompson (1981). The dependence of response amplitude and variance of cat visual cortical neurones on stimulus contrast. *Experimental Brain Research 41*, 414–419.

Tomko, G. J. and D. R. Crapper (1974). Neuronal variability: non-stationary responses to identical visual stimuli. *Brain Research 79*, 405–418.

Ueda, N. and R. Nakano (1998). Deterministic annealing EM algorithm. *Neural Networks 11*(2), 271–282.

Usrey, W. M., J. B. Reppas, and R. C. Reid (1998). Paired-spike interactions and synaptic efficacy of retinal inputs to the thalamus. *Nature 395*(6700), 384–387.

Victor, J. D. and K. P. Purpura (1997). Metric-space analysis of spike trains: Theory, algorithms and application. *Network: Computation in Neural Systems 8*, 127–164.

Vogels, R., W. Spileers, and G. A. Orban (1989). The response variability of striate cortical neurons in the behaving monkey. *Experimental Brain Research 77*, 432–436.

Wehr, M., J. S. Pezaris, and M. Sahani (1999). Simultaneous paired intracellular and tetrode recordings for evaluating the performance of spike sorting algorithms. *Neurocomputing (in press)*.

Weiss, Y. (1998). Phase transitions and the perceptual organization of video sequences. In M. I. Jordan, M. J. Kearns, and S. S. Solla (Eds.), *Advances in Neural Information Processing Systems*, Volume 10, Cambridge, MA. MIT Press.

Wolpert, D. (1992). Stacked generalization. *Neural Networks 5*(2), 241–259.

Wolpert, D. H. (1996). The lack of a priori distinctions between learning algorithms. *Neural Computation 8*(7), 1341–1390.

Xu, L. and M. I. Jordan (1996). On convergence properties of the EM algorithm for Gaussian mixtures. *Neural Computation 8*(1), 129–151.

Yuille, A. L. (1990). Generalized deformable models, statistical physics, and matching problems. *Neural Computation 2*(1), 1–24.

Yuille, A. L., P. Stolorz, and J. Utans (1994). Statistical physics, mixtures of distributions, and the EM algorithm. *Neural Computation 6*(2), 334–340.

Zar, J. H. (1998). *Biostatistical Analysis* (4th ed.). Prentice Hall.

Zhang, K., T. J. Sejnowski, and B. L. McNaughton (1997). Automatic separation of spike parameter clusters from tetrode recordings. *Society for Neurocience Abstracts 23*(1), 504.