Chapter 4

Variational Bayesian Mixtures of Factor Analysers

4.1 Introduction

This chapter is concerned with learning good representations of high dimensional data, with the goal being to perform well in density estimation and pattern classification tasks. The work described here builds on work in Ghahramani and Beal (2000), which first introduced the variational method for Bayesian learning of a mixtures of factor analysers model, resulting in a tractable means of integrating over all the parameters in order to avoid overfitting.

In the following subsections we introduce factor analysis (FA), and the mixtures of factor analysers (MFA) model which can be thought of as a mixture of reduced-parameter Gaussians. In section 4.2 we explain why an exact Bayesian treatment of MFAs is intractable, and present a variational Bayesian algorithm for learning. We show how to learn distributions over the parameters of the MFA model, how to optimise its hyperparameters, and how to automatically determine the dimensionality of each analyser using automatic relevance determination (ARD) methods. In section 4.3 we propose heuristics for efficiently exploring the (one-dimensional) space of the number of components in the mixture, and in section 4.5 we present synthetic experiments showing that the model can simultaneously learn the number of analysers and their intrinsic dimensionalities. In section 4.6 we apply the VBMFA to the real-world task of classifying digits, and show improved performance over a BIC-penalised maximum likelihood approach. In section 4.7 we examine the tightness of the VB lower bound using importance sampling estimates of the exact marginal likelihood, using as importance distributions the posteriors from the VB optimisation. We also investigate the effectiveness of using heavy-tailed and mixture distributions in this procedure. We then conclude in section 4.8 with a brief outlook on recent research progress in this area.

4.1.1 Dimensionality reduction using factor analysis

Factor analysis is a method for modelling correlations in multidimensional data, by expressing the correlations in a lower-dimensional, oriented subspace. Let the data set be $\mathbf{y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$. The model assumes that each p-dimensional data vector $\mathbf{y}_i$ was generated by first linearly transforming a $k < p$ dimensional vector of unobserved independent zero-mean unit-variance Gaussian sources (factors), $\mathbf{x}_i = [\mathbf{x}_{i1}, \ldots, \mathbf{x}_{ik}]$, translating by a fixed amount $\boldsymbol{\mu}$ in the data space, followed by adding p-dimensional zero-mean Gaussian noise, $\mathbf{n}_i$, with diagonal covariance matrix Ψ (whose entries are sometimes referred to as the *uniquenesses*). Expressed mathematically, we have

$$\mathbf{y}_i = \Lambda \mathbf{x}_i + \boldsymbol{\mu} + \mathbf{n}_i \tag{4.1}$$

$$\mathbf{x}_i \sim \mathrm{N}(\mathbf{0}, \mathrm{I}), \qquad \mathbf{n}_i \sim \mathrm{N}(\mathbf{0}, \Psi) \; , \tag{4.2}$$

where $\Lambda \; (p \times k)$ is the linear transformation known as the *factor loading* matrix, and $\boldsymbol{\mu}$ is the mean of the analyser. Integrating out $\mathbf{x}_i$ and $\mathbf{n}_i$, it is simple to show that the marginal density of $\mathbf{y}_i$ is Gaussian about the displacement $\boldsymbol{\mu}$,

$$p(\mathbf{y}_i \,|\, \Lambda, \boldsymbol{\mu}, \Psi) = \int d\mathbf{x}_i \; p(\mathbf{x}_i) p(\mathbf{y}_i \,|\, \mathbf{x}_i, \Lambda, \boldsymbol{\mu}, \Psi) = \mathrm{N}(\mathbf{y}_i \,|\, \boldsymbol{\mu}, \Lambda\Lambda^\top + \Psi) \; , \tag{4.3}$$

and the probability of an i.i.d. data set $\mathbf{y} = \{\mathbf{y}_i\}_{i=1}^n$ is given by

$$p(\mathbf{y} \,|\, \Lambda, \boldsymbol{\mu}, \Psi) = \prod_{i=1}^{n} p(\mathbf{y}_i \,|\, \Lambda, \boldsymbol{\mu}, \Psi) \; . \tag{4.4}$$

Given a data set $\mathbf{y}$ having covariance matrix Σ^* and mean $\boldsymbol{\mu}^*$, factor analysis finds the Λ, $\boldsymbol{\mu}$ and Ψ that optimally fit Σ^* in the maximum likelihood sense. Since $k < p$, a factor analyser can be seen as a reduced parameterisation of a full-covariance Gaussian. The (diagonal) entries of the Ψ matrix concentrate on fitting the axis-aligned (sensor) noise in the data, leaving the factor loadings in Λ to model the remaining (assumed-interesting) covariance structure.

The effect of the mean term $\boldsymbol{\mu}$ can be assimilated into the factor loading matrix by augmenting the vector of factors with a constant bias dimension of 1, and adding a corresponding column $\boldsymbol{\mu}$ to the matrix Λ. With these modifications, learning the Λ matrix incorporates learning the mean; in the equations of this chapter we keep the parameters separate, although the implementations consider the combined quantity.

Dimensionality of the latent space, k

A central problem in factor analysis is deciding on the dimensionality of the latent space. If too low a value of k is chosen, then the model has to discard some of the covariance in the data as noise, and if k is given too high a value this causes the model to fit spurious correlations in the data. Later we describe a Bayesian technique to determine this value automatically, but here we first give an understanding for an upper bound on the required value for k, by comparing the number of degrees of freedom in the covariance specification of the data set and the degrees of freedom that the FA parameterisation has in its parameters. We need to distinguish between the number of parameters and the degrees of freedom, which is really a measure of how many independent directions in parameter space there are that affect the generative probability of the data. The number of degrees of freedom in a factor analyser with latent space dimensionality k cannot exceed the number of degrees of freedom of a full covariance matrix, $\frac{1}{2}p(p+1)$, nor can it exceed the degrees of freedom offered by the parameterisation of the analyser, which is given by $d(k)$,

$$d(k) = kp + p - \frac{1}{2}k(k-1) \; . \tag{4.5}$$

The first two terms on the right hand side are the degrees of freedom in the Λ and Ψ matrices respectively, and the last term is the degrees of freedom in a $(k \times k)$ orthonormal matrix. This last term needs to be subtracted because it represents a redundancy in the factor analysis parameterisation, namely that an arbitrary rotation or reflection of the latent vector space leaves the covariance model of the data unchanged:

$$\text{under } \Lambda \to \Lambda U, \qquad \Lambda\Lambda^\top + \Psi \to \Lambda U(\Lambda U)^\top + \Psi \tag{4.6}$$

$$= \Lambda U U^\top \Lambda^\top + \Psi \tag{4.7}$$

$$= \Lambda\Lambda^\top + \Psi \; . \tag{4.8}$$

That is to say we must subtract the degrees of freedom from degeneracies in Λ associated with arbitrary arrangements of the (a priori identical) hidden factors $\{\mathbf{x}_{ij}\}_{j=1}^{k}$. Since a p-dimensional covariance matrix contains $p(p+1)/2$ pieces of information, in order to be able to perfectly capture the covariance structure of the data the number of degrees of freedom in the analyser (4.5) would have to exceed this. This inequality is a simple quadratic problem, for $k \leq p$

$$kp + p - \frac{1}{2}k(k-1) \geq \frac{1}{2}p(p+1) \tag{4.9}$$

whose solution is given by

$$k_{\text{max}} = \left\lceil p + \frac{1}{2}\left[1 - \sqrt{1+8p}\right] \right\rceil \; . \tag{4.10}$$

We might be tempted to conclude that we only need k_{max} factors to model an arbitrary covariance in p dimensions. However this neglects the constraint that all the diagonal elements of Ψ have to be positive. We conjecture that because of this constraint the number of factors needed to model a full covariance matrix is $p - 1$. This implies that for high dimensional data, if we want to be able to model a full covariance structure, we cannot expect to be able to reduce the number of parameters by that much at all using factor analysis. Fortunately, for many real data sets we have good reason to believe that, at least locally, the data lies on a low dimensional manifold which we can capture with only a few factors. The fact that this is a good approximation only locally, when the manifold may be globally non-linear, is the motivation for mixture models, discussed next.

4.1.2 Mixture models for manifold learning

It is often the case that apparently high dimensional data in fact lies, to a good approximation, on a low dimensional manifold. For example, consider the data set consisting of many different images of the same digit, given in terms of the pixel intensities. This data has as many dimensions as there are pixels in each image. To explain this data we could first specify a mean digit image, which is a point in this high dimensional space representing a set of pixel intensities, and then specify a small number of transformations away from that digit that would cover small variations in style or perhaps intensity. In factor analysis, each factor dictates the amount of each linear transformation on the pixel intensities. However, with factor analysis we are restricted to linear transformations, and so any one analyser can only explain well a small region of the manifold in which it is locally linear, even though the manifold is globally non-linear.

One way to overcome this is to use mixture models to tile the data manifold. A mixture of factor analysers models the density for a data point $\mathbf{y}_i$ as a weighted average of factor analyser densities

$$p(\mathbf{y}_i \,|\, \boldsymbol{\pi}, \Lambda, \boldsymbol{\mu}, \Psi) = \sum_{s_i=1}^{S} p(s_i \,|\, \boldsymbol{\pi}) p(\mathbf{y}_i \,|\, s_i, \Lambda, \boldsymbol{\mu}, \Psi) \; . \tag{4.11}$$

Here, S is the number of mixture components in the model, $\boldsymbol{\pi}$ is the vector of mixing proportions, s_i is a discrete indicator variable for the mixture component chosen to model data point i, $\Lambda = \{\Lambda^s\}_{s=1}^{S}$ is a set of factor loadings with Λ^s being the factor loading matrix for analyser s, and $\boldsymbol{\mu} = \{\boldsymbol{\mu}^s\}_{s=1}^{S}$ is the set of analyser means. The last term in the above probability is just the single analyser density, given in equation (4.3). The directed acyclic graph for this model is depicted in figure 4.1, which uses the *plate* notation to denote repetitions over a data set of size n. Note that there are different indicator variables s_i and latent space variables $\mathbf{x}_i$ for each plate.

By exploiting the factor analysis parameterisation of covariance matrices, a mixture of factor analysers can be used to fit a mixture of Gaussians to correlated high dimensional data without requiring $O(p^2)$ parameters, or undesirable compromises such as axis-aligned covariance matrices. In an MFA each Gaussian cluster has intrinsic dimensionality k, or k_s if the dimensions are allowed to vary across mixture components. Consequently, the mixture of factor analysers

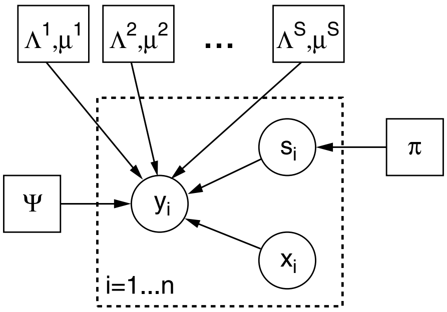


Figure 4.1: Generative model for Maximum Likelihood MFA. Circles denote random variables, solid rectangles parameters, and the dashed rectangle the plate (repetitions) over the data.

simultaneously addresses the problems of clustering and local dimensionality reduction. When Ψ is a multiple of the identity the model becomes a mixture of probabilistic PCAs (pPCA). Tractable maximum likelihood (ML) procedures for fitting MFA and pPCA models can be derived from the Expectation Maximisation algorithm, see for example Ghahramani and Hinton (1996b); Tipping and Bishop (1999). Factor analysis and its relationship to PCA and mixture models is reviewed in Roweis and Ghahramani (1999).

4.2 Bayesian Mixture of Factor Analysers

The maximum likelihood approach to fitting an MFA has several drawbacks. The EM algorithm can easily get caught in local maxima, and often many restarts are required before a good maximum is reached. Technically speaking the log likelihoods in equations (4.3) and (4.11) are not bounded from above, unless constraints are placed on the variances of the components of the mixture. In practice this means that the covariance matrix $\Lambda^s \Lambda^{s\top} + \Psi$ can become singular if a particular factor analyser models fewer points than the degrees of freedom in its covariance matrix. Most importantly, the maximum likelihood approach for fitting MFA models has the severe drawback that it fails to take into account model complexity. For example the likelihood can be increased by adding more analyser components to the mixture, up to the extreme where each component models a single data point, and it can be further increased by supplying more factors in each of the analysers.

A Bayesian approach overcomes these problems by treating the parameters of the model as unknown quantities and averaging over the ensemble of models they produce. Defining $\boldsymbol{\theta} = (\Lambda, \boldsymbol{\mu}, \boldsymbol{\pi}, \Psi)$, we write the probability of the data averaged over a prior for parameters:

$$
\begin{aligned}
p(\mathbf{y}) &= \int d\boldsymbol{\theta}\; p(\boldsymbol{\theta}) p(\mathbf{y} \,|\, \boldsymbol{\theta}) && (4.12)\\
&= \int d\boldsymbol{\theta}\; p(\boldsymbol{\theta}) \prod_{i=1}^{n} p(\mathbf{y}_i \,|\, \boldsymbol{\theta}) && (4.13)\\
&= \int d\boldsymbol{\pi}\; p(\boldsymbol{\pi}) \int d\Lambda\; p(\Lambda) \int d\boldsymbol{\mu}\; p(\boldsymbol{\mu}) \cdot \\
&\qquad \prod_{i=1}^{n} \left[\sum_{s_i=1}^{S} p(s_i \,|\, \boldsymbol{\pi}) \int d\mathbf{x}_i\; p(\mathbf{x}_i) p(\mathbf{y}_i \,|\, s_i, \mathbf{x}_i, \Lambda, \boldsymbol{\mu}, \Psi) \right] . && (4.14)
\end{aligned}
$$

Equation (4.14) is the marginal likelihood of a dataset (called the marginal probability of the data set by some researchers to avoid confusion with the likelihood of the *parameters*). By integrating out all those parameters whose number increase as the model complexity grows, we effectively penalise models with more degrees of freedom, since they can a priori model a larger range of data sets. By model complexity, we mean the number of components and the dimensionality of each component. Integrating out the parameters naturally embodies the principle of Occam's razor (MacKay, 1992; Jefferys and Berger, 1992). As a result no parameters are ever *fit* to the data, but rather their posterior *distributions* are inferred and used to make predictions about new data. For this chapter, we have chosen not to integrate over Ψ, although this could also be done (see, for example, chapter 5). Since the number of degrees of freedom in Ψ does not grow with the number of analysers or their dimensions, we treat it as a hyperparameter and optimise it, even though this might result in some small degree of overfitting.

4.2.1 Parameter priors for MFA

While arbitrary choices can be made for the priors in (4.14), choosing priors that are conjugate to the likelihood terms greatly simplifies inference and interpretability. Therefore we choose a symmetric Dirichlet prior for the mixing proportion $\boldsymbol{\pi}$, with strength α^*,

$$
p(\boldsymbol{\pi} \,|\, \alpha^* \mathbf{m}^*) = \mathrm{Dir}(\boldsymbol{\pi} \,|\, \alpha^* \mathbf{m}^*) , \qquad \text{such that} \quad \mathbf{m}^* = \left[\frac{1}{S}, \ldots, \frac{1}{S} \right] . \tag{4.15}
$$

In this way the prior has a single hyperparameter, its strength α^*, regardless of the dimensionality of $\boldsymbol{\pi}$. This hyperparameter is a measure of how we expect the mixing proportions to deviate from being equal. One could imagine schemes in which we have non-symmetric prior mixing proportion; an example could be making the hyperparameter in the Dirichlet prior an exponentially decaying vector with a single decay rate hyperparameter, which induces a natural ordering in the mixture components and so removes some identifiability problems. Nevertheless for our purposes a symmetric prior suffices, and expresses the notion that each component has equal a priori chance of being used to generate each data point.

For the entries of the factor loading matrices, $\{\Lambda^s\}_{s=1}^S$, we choose a hierarchical prior in order to perform automatic relevance determination (ARD). Each column of each factor loading matrix has a Gaussian prior with mean zero and a different precision parameter (drawn from a gamma distribution with fixed hyperparameters, see equation (4.18) below):

$$p(\Lambda \,|\, \boldsymbol{\nu}) = \prod_{s=1}^{S} p(\Lambda^s \,|\, \boldsymbol{\nu}^s) = \prod_{s=1}^{S} \prod_{j=1}^{k_s} p(\Lambda^s_{\cdot j} \,|\, \nu^s_j) = \prod_{s=1}^{S} \prod_{j=1}^{k_s} \mathrm{N}(\Lambda^s_{\cdot j} \,|\, \mathbf{0}, \mathrm{I}/\nu^s_j) \,, \tag{4.16}$$

where $\Lambda^s_{\cdot j}$ denotes the vector of entries in the jth column of the sth analyser in the mixture, and ν^s_j is the same scalar precision for each entry in the corresponding column. The role of these precision hyperparameters is explained in section 4.2.2. Note that because the spherical Gaussian prior is separable into each of its p dimensions, the prior can equivalently be thought of as a Gaussian with axis-aligned elliptical covariance on each row of each analyser:

$$p(\Lambda \,|\, \boldsymbol{\nu}) = \prod_{s=1}^{S} \prod_{q=1}^{p} p(\Lambda^s_{q\cdot} \,|\, \boldsymbol{\nu}^s) = \prod_{s=1}^{S} \prod_{q=1}^{p} \mathrm{N}(\Lambda^s_{q\cdot} \,|\, \mathbf{0}, \mathrm{diag}\,(\boldsymbol{\nu}^s)^{-1}) \,, \tag{4.17}$$

where here $\Lambda^s_{q\cdot}$ is used to denote the qth row of the sth analyser. It will turn out to be simpler to have the prior in this form conceptually for learning, since the likelihood terms for Λ factor across its rows.

Since the number of hyperparameters in $\boldsymbol{\nu} = \{\{\nu^s_j\}_{j=1}^{k_s}\}_{s=1}^S$ increases with the number of analysers and also with the dimensionality of each analyser, we place a hyperprior on every element of each $\boldsymbol{\nu}^s$ precision vector, as follows:

$$p(\boldsymbol{\nu} \,|\, a^*, b^*) = \prod_{s=1}^{S} p(\boldsymbol{\nu}^s \,|\, a^*, b^*) = \prod_{s=1}^{S} \prod_{j=1}^{k_s} p(\nu^s_j \,|\, a^*, b^*) = \prod_{s=1}^{S} \prod_{j=1}^{k_s} \mathrm{Ga}(\nu^s_j \,|\, a^*, b^*) \,, \tag{4.18}$$

where a^* and b^* are shape and inverse-scale hyperhyperparameters for a gamma distribution (see appendix A for a definition and properties of the gamma distribution). Note that the same hyperprior is used for every element in $\boldsymbol{\nu}$. As a point of interest, combining the priors for Λ and $\boldsymbol{\nu}$, and integrating out $\boldsymbol{\nu}$, we find that the marginal prior over each Λ^s is Student-t distributed. We will not need to make use of this result right here, but will return to it in section 4.7.1.

Lastly, the means of each analyser in the mixture need to be integrated out. A Gaussian prior with mean $\boldsymbol{\mu}^*$ and axis-aligned precision $\mathrm{diag}\,(\boldsymbol{\nu}^*)$ is placed on each mean $\boldsymbol{\mu}^s$. Note that these hyperparameters hold $2p$ degrees of freedom, which is not a function of the size of the model. The prior is the same for every analyser:

$$p(\boldsymbol{\mu} \,|\, \boldsymbol{\mu}^*, \boldsymbol{\nu}^*) = \prod_{s=1}^{S} p(\boldsymbol{\mu}^s \,|\, \boldsymbol{\mu}^*, \boldsymbol{\nu}^*) = \prod_{s=1}^{S} \mathrm{N}(\boldsymbol{\mu}^s \,|\, \boldsymbol{\mu}^*, \mathrm{diag}\,(\boldsymbol{\nu}^*)^{-1}) \tag{4.19}$$

Note that this prior has a different precision for each dimension of the output, whereas the prior over the entries in the factor loading matrix uses the same precision on each row, and is different only for each column of each analyser.

If we are to use the implementational convenience of augmenting the latent space with a constant bias dimension, and adding a further column to each factor loading matrix to represent its mean, then the prior over all the entries in the augmented factor loading matrix no longer factorises over rows (4.17) or columns (4.18), but has to be expressed as a product of terms over every entry of the matrix. This point will be made clearer when we derive the posterior distribution over the augmented factor loading matrix.

We use Θ to denote the set of hyperparameters of the model:

$$\Theta = (\alpha^* \mathbf{m}^*, a^*, b^*, \boldsymbol{\mu}^*, \boldsymbol{\nu}^*, \Psi) \ . \tag{4.20}$$

The directed acyclic graph for the generative model for this Bayesian MFA is shown graphically in figure 4.2. Contrasting with the ML graphical model in figure 4.1, we can see that all the model parameters (with the exception of the sensor noise Ψ) have been replaced with uncertain variables, denoted with circles, and now have hyperparameters governing their prior distributions. The generative model for the data remains the same, with the plate over the data denoting i.i.d. instances of the hidden factors $\mathbf{x}_i$, each of which gives rise to an output $\mathbf{y}_i$. We keep the graphical model concise by also using a plate over the S analysers, which clearly shows the role of the hyperpriors.

As an aside, we do not place a prior on the number of components, S. We instead place a symmetric Dirichlet prior over the mixing proportions. Technically, we should include a (square boxed) node S, as the parent of both the plate over analysers and the hyperparameter $\alpha\mathbf{m}$. We have also not placed priors over the number of factors of each analyser, $\{k_s\}_{s=1}^{S}$; this is intentional as there exists an explicit penalty for using more dimensions — the extra entries in factor loading matrix Λ^s need to be explained under a hyperprior distribution (4.16) which is governed by a new hyperparameter $\boldsymbol{\nu}^s$, which itself has to be explained under the hyperhyperprior $p(\boldsymbol{\nu}^s \,|\, a, b)$ of equation (4.18).

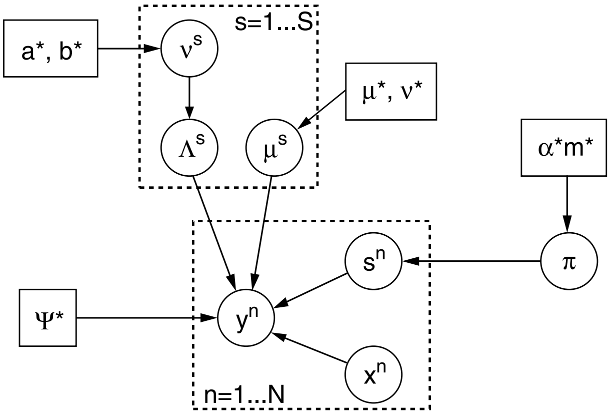


Figure 4.2: A Bayesian formulation for MFA. Here the plate notation is used to denote repetitions over data n and over the S analysers in the generative model. Note that all the parameters in the ML formulation, except Ψ, have now become uncertain random variables in the Bayesian model (circled nodes in the graph), and are governed by hyperparameters (square boxes). The number of hyperparameters in the model is constant and is not a function of the number of analysers or their dimensionalities.

4.2.2 Inferring dimensionality using ARD

Each factor analyser s in the MFA models its local data as a linear projection of k_s-dimensional spherical Gaussian noise into the p-dimensional space. If a maximum dimensionality k_{max} is set, then there exist $k_{\text{max}} \times \cdots \times k_{\text{max}} = (k_{max})^S$ possible subspace configurations amongst the S analysers. Thus determining the optimal configuration is exponentially intractable if a discrete search is employed over analyser dimensionalities. Automatic relevance determination (ARD) solves this discrete search problem with the use of continuous variables that allow a *soft blend* of dimensionalities. Each factor analyser's dimensionality is set to k_{max} and we use priors that discourage large factor loadings. The width of each prior is controlled by a hyperparameter (explained below), and the result of learning with this method is that only those hidden factor dimensions that are required remain active after learning — the remaining dimensions are effectively 'switched off'. This general method was proposed by MacKay and Neal (see MacKay, 1996, for example), and was used in Bishop (1999) for Bayesian PCA, and is closely related to the method given in Neal (1998a) for determining the relevance of inputs to a neural network.

Considering for the moment a single factor analyser. The ARD scheme uses a Gaussian prior with a zero mean for the entries of the factor loading matrix, as shown in (4.16), given again here:

$$p(\Lambda^s \mid \boldsymbol{\nu}^s) = \prod_{j=1}^{k_{\text{max}}} p(\Lambda^s_{\cdot j} \mid \nu^s_j) = \prod_{j=1}^{k_{\text{max}}} \mathrm{N}(\Lambda^s_{\cdot j} \mid \mathbf{0}, \mathrm{I}/\nu^s_j) \,, \tag{4.21}$$

where $\boldsymbol{\nu}^s = \{\nu^s_1, \ldots, \nu^s_{k_{\text{max}}}\}$ are the precisions on the columns of Λ^s, which themselves are denoted by $\{\Lambda_{\cdot 1}, \ldots, \Lambda_{\cdot k_{\text{max}}}\}$. This zero-mean prior couples within-column entries in Λ^s, favouring lower magnitude values.

If we apply this prior to each analyser in the mixture, each column of each factor loading matrix is then governed by a separate ν_l^s parameter. If one of these precisions $\boldsymbol{\nu}_l^s \to \infty$ then the outgoing weights (column l entries in Λ^s) for the lth factor in the sth analyser will have to be very close to zero in order to maintain a high likelihood under this prior, and this in turn leads the analyser to ignore this factor, and thus allows the model to reduce the intrinsic dimensionality of $\mathbf{x}$ in the *locale of that analyser* if the data does not warrant this added dimension. We have not yet explained how some of these precisions come to tend to infinity; this will be made clearer in the derivations of the learning rules in section 4.2.5.

The fully Bayesian application requires that we integrate out all parameters that scale with the number of analyser components and their dimensions; for this reason we use the conjugate prior for a precision variable, a gamma distribution with shape a^* and inverse scale b^*, to integrate over the ARD hyperparameters. Since we are integrating over the hyperparameters, it now makes sense to consider removing a redundant factor loading when the *posterior distribution* over the hyperparameter $\boldsymbol{\nu}_l^s$ has most of its mass near infinity. In practice we take the mean of this posterior to be indicative of its position, and perform removal when it becomes very large. This reduces the coding cost of the parameters, and as a redundant factor is not used to model the data, this must increase the marginal likelihood $p(\mathbf{y})$. We can be harsher still, and prematurely remove those factors which have $\boldsymbol{\nu}_l^s$ escaping to infinity, provided the resulting marginal likelihood is better (we do not implement this scheme in our experiments).

4.2.3 Variational Bayesian derivation

Now that we have priors over the parameters of our model, we can set about computing the marginal likelihood of data. But unfortunately, computing the marginal likelihood in equation (4.14) is intractable because integrating over the parameters of the model induces correlations in the posterior distributions between the hidden variables in all the n plates. As mentioned in section 1.3, there are several methods that are used to approximate such integrals, for example MCMC sampling techniques, the Laplace approximation, and the asymptotic BIC criterion.

For MFA and similar models, MCMC methods for Bayesian approaches have only recently been applied by Fokoué and Titterington (2003), with searches over model complexity in terms of both the number of components and their dimensionalities carried out by reversible jump techniques (Green, 1995). In related models, Laplace and asymptotic approximations have been used to approximate Bayesian integration in mixtures of Gaussians (Roberts et al., 1998). Here our focus is on analytically tractable approximations based on lower bounding the marginal likelihood.

We begin with the log marginal likelihood of the data and first construct a lower bound using a variational distribution over the parameters $\{\boldsymbol{\pi}, \boldsymbol{\nu}, \Lambda, \boldsymbol{\mu}\}$, and then perform a similar lower bounding using a variational distribution for the hidden variables $\{s_i, \mathbf{x}_i\}_{i=1}^n$. As a point of nomenclature, just as we have been using the same notation $p(\cdot)$ for every prior distribution, even though they may be Gaussian, gamma, Dirichlet etc., in what follows we also use the same $q(\cdot)$ to denote different variational distributions for different parameters. The form of $q(\cdot)$ will be clear from its arguments.

Combining (4.14) with the priors discussed above including the hierarchical prior on Λ, we obtain the log marginal likelihood of the data, denoted $\mathcal{L}$,

$$\mathcal{L} \equiv \ln p(\mathbf{y}) = \ln \left(\int d\boldsymbol{\pi}\; p(\boldsymbol{\pi} \,|\, \alpha^* \mathbf{m}^*) \int d\boldsymbol{\nu}\; p(\boldsymbol{\nu} \,|\, a^*, b^*) \int d\Lambda\; p(\Lambda \,|\, \boldsymbol{\nu}) \int d\boldsymbol{\mu}\; p(\boldsymbol{\mu} \,|\, \boldsymbol{\mu}^*, \boldsymbol{\nu}^*) \cdot \right.$$
$$\left. \prod_{i=1}^n \left[\sum_{s_i=1}^S p(s_i \,|\, \boldsymbol{\pi}) \int d\mathbf{x}_i\; p(\mathbf{x}_i) p(\mathbf{y}_i \,|\, s_i, \mathbf{x}_i, \Lambda, \boldsymbol{\mu}, \Psi) \right] \right) . \tag{4.22}$$

The marginal likelihood $\mathcal{L}$ is in fact a function of the hyperparameters $(\alpha^* \mathbf{m}^*, a^*, b^*, \boldsymbol{\mu}^*, \boldsymbol{\nu}^*)$, and the sensor noise Ψ; this dependence is left implicit in this derivation. We introduce an arbitrary distribution $q(\boldsymbol{\pi}, \boldsymbol{\nu}, \Lambda, \boldsymbol{\mu})$ to lower bound (4.22), followed by a second set of distributions $\{q(s_i, \mathbf{x}_i)\}_{i=1}^n$ to further lower bound the bound,

$$\begin{aligned}
\mathcal{L} \geq & \int d\boldsymbol{\pi}\, d\boldsymbol{\nu}\, d\Lambda\, d\boldsymbol{\mu}\; q(\boldsymbol{\pi}, \boldsymbol{\nu}, \Lambda, \boldsymbol{\mu}) \left(\ln \frac{p(\boldsymbol{\pi} \,|\, \alpha^* \mathbf{m}^*) p(\boldsymbol{\nu} \,|\, a^*, b^*) p(\Lambda \,|\, \boldsymbol{\nu}) p(\boldsymbol{\mu} \,|\, \boldsymbol{\mu}^*, \boldsymbol{\nu}^*)}{q(\boldsymbol{\pi}, \boldsymbol{\nu}, \Lambda, \boldsymbol{\mu})} \right. \\
& \left. + \sum_{i=1}^n \ln \left[\sum_{s_i=1}^S p(s_i \,|\, \boldsymbol{\pi}) \int d\mathbf{x}_i\; p(\mathbf{x}_i) p(\mathbf{y}_i \,|\, s_i, \mathbf{x}_i, \Lambda, \boldsymbol{\mu}, \Psi) \right] \right) \qquad (4.23) \\
\geq & \int d\boldsymbol{\pi}\, d\boldsymbol{\nu}\, d\Lambda\, d\boldsymbol{\mu}\; q(\boldsymbol{\pi}, \boldsymbol{\nu}, \Lambda, \boldsymbol{\mu}) \left(\ln \frac{p(\boldsymbol{\pi} \,|\, \alpha^* \mathbf{m}^*) p(\boldsymbol{\nu} \,|\, a^*, b^*) p(\Lambda \,|\, \boldsymbol{\nu}) p(\boldsymbol{\mu} \,|\, \boldsymbol{\mu}^*, \boldsymbol{\nu}^*)}{q(\boldsymbol{\pi}, \boldsymbol{\nu}, \Lambda, \boldsymbol{\mu})} \right. \\
& \left. + \sum_{i=1}^n \left[\sum_{s_i=1}^S \int d\mathbf{x}_i\; q(s_i, \mathbf{x}_i) \left(\ln \frac{p(s_i \,|\, \boldsymbol{\pi}) p(\mathbf{x}_i)}{q(s_i, \mathbf{x}_i)} + \ln p(\mathbf{y}_i \,|\, s_i, \mathbf{x}_i, \Lambda, \boldsymbol{\mu}, \Psi) \right) \right] \right) .
\end{aligned}$$
$$\tag{4.24}$$

In the first inequality, the term on the second line is simply the log likelihood of $\mathbf{y}_i$ for a fixed setting of parameters, which is then further lower bounded in the second inequality using a set of distributions over the hidden variables $\{q(s_i, \mathbf{x}_i)\}_{i=1}^n$. These distributions are *independent* of the settings of the parameters $\boldsymbol{\pi}, \boldsymbol{\nu}, \Lambda$, and $\boldsymbol{\mu}$, and they correspond to the standard variational approximation of the factorisation between the parameters and the hidden variables:

$$p(\boldsymbol{\pi}, \boldsymbol{\nu}, \Lambda, \boldsymbol{\mu}, \{s_i, \mathbf{x}_i\}_{i=1}^n \,|\, \mathbf{y}) \approx q(\boldsymbol{\pi}, \boldsymbol{\nu}, \Lambda, \boldsymbol{\mu}) \prod_{i=1}^n q(s_i, \mathbf{x}_i) . \tag{4.25}$$

The distribution of hidden variables factorises across the plates because both the generative model is i.i.d. *and* we have made the approximation that the parameters and hidden variables are independent (see proof of theorem 2.1 in section 2.3.1). Here we use a further variational approximation amongst the parameters, which can be explained by equating the functional derivatives of equation (4.24) with respect to $q(\boldsymbol{\pi}, \boldsymbol{\nu}, \Lambda, \boldsymbol{\mu})$ to zero. One finds that

$$\begin{aligned} q(\boldsymbol{\pi}, \boldsymbol{\nu}, \Lambda, \boldsymbol{\mu}) &\propto p(\boldsymbol{\pi} \,|\, \alpha^* \mathbf{m}^*) p(\boldsymbol{\nu} \,|\, a^*, b^*) p(\Lambda \,|\, \boldsymbol{\nu}) p(\boldsymbol{\mu} \,|\, \boldsymbol{\mu}^*, \boldsymbol{\nu}^*) \cdot \\ &\qquad \exp \left[\sum_{i=1}^{n} \sum_{s_i=1}^{S} \langle \ln p(s_i \,|\, \boldsymbol{\pi}) p(\mathbf{y}_i \,|\, s_i, \mathbf{x}_i, \Lambda, \boldsymbol{\mu}, \Psi) \rangle_{q(s_i, \mathbf{x}_i)} \right] \qquad (4.26) \\ &= q(\boldsymbol{\pi}) q(\boldsymbol{\nu}, \Lambda, \boldsymbol{\mu}) \qquad (4.27) \\ &\approx q(\boldsymbol{\pi}) q(\boldsymbol{\nu}) q(\Lambda, \boldsymbol{\mu}) \; . \qquad (4.28) \end{aligned}$$

In the second line, the approximate posterior factorises exactly into a contribution from the mixing proportions and the remaining parameters. Unfortunately it is not easy to take expectations with respect to the joint distribution over Λ and its parent parameter $\boldsymbol{\nu}$, and therefore we make the second variational approximation in the last line, equation (4.28). The very last term $q(\Lambda, \boldsymbol{\mu})$ turns out to be jointly Gaussian, and so is of tractable form.

We should note that except for the initial factorisation between the hidden variables and the parameters, the factorisation $q(\boldsymbol{\nu}, \Lambda, \boldsymbol{\mu}) \approx q(\boldsymbol{\nu}) q(\Lambda, \boldsymbol{\mu})$ is the only other approximating factorisation we make; all other factorisations fall out naturally from the conditional independencies in the model. Note that the complete-data likelihood for mixtures of factor analysers is in the exponential family, even after the inclusion of the precision parameters $\boldsymbol{\nu}$. We could therefore apply the results of section 2.4, but this would entail finding expectations over gamma-Gaussian distributions jointly over $\boldsymbol{\nu}$ and Λ. Although it is possible to take these expectations, for convenience we choose a separable variational posterior on $\boldsymbol{\nu}$ and Λ.

From this point on we assimilate each analyser's mean position $\boldsymbol{\mu}^s$ into its factor loading matrix, in order to keep the presentation concise. The derivations use $\tilde{\Lambda}$ to denote the concatenated result $[\Lambda \; \boldsymbol{\mu}]$. Therefore the prior over the entire factor loadings $\tilde{\Lambda}$ is now a function of the precision parameters $\{\boldsymbol{\nu}^s\}_{s=1}^{S}$ (which themselves have hyperparameters a, b) and the hyperparameters $\boldsymbol{\mu}^*, \boldsymbol{\nu}^*$. Also, the variational posterior $q(\Lambda, \boldsymbol{\mu})$ becomes $q(\tilde{\Lambda})$.

Substituting the factorised approximations (4.25) and (4.28) into the lower bound (4.24) results in the following lower bound for the marginal likelihood,

$$\begin{aligned}
\mathcal{L} \geq& \int d\boldsymbol{\pi}\; q(\boldsymbol{\pi}) \ln \frac{p(\boldsymbol{\pi} \,|\, \alpha^*, \mathbf{m}^*)}{q(\boldsymbol{\pi})} \\
&+ \sum_{s=1}^{S} \int d\boldsymbol{\nu}^s\; q(\boldsymbol{\nu}^s) \left[\ln \frac{p(\boldsymbol{\nu}^s \,|\, a^*, b^*)}{q(\boldsymbol{\nu}^s)} + \int d\tilde{\Lambda}^s\; q(\tilde{\Lambda}^s) \ln \frac{p(\tilde{\Lambda}^s \,|\, \boldsymbol{\nu}^s, \boldsymbol{\mu}^*, \boldsymbol{\nu}^*)}{q(\tilde{\Lambda}^s)} \right] \\
&+ \sum_{i=1}^{n} \sum_{s_i=1}^{S} q(s_i) \left[\int d\boldsymbol{\pi}\; q(\boldsymbol{\pi}) \ln \frac{p(s_i \,|\, \boldsymbol{\pi})}{q(s_i)} + \int d\mathbf{x}_i\; q(\mathbf{x}_i \,|\, s_i) \ln \frac{p(\mathbf{x}_i)}{q(\mathbf{x}_i \,|\, s_i)} \right. \\
&\qquad \left. + \int d\tilde{\Lambda}\; q(\tilde{\Lambda}) \int d\mathbf{x}_i\; q(\mathbf{x}_i \,|\, s_i) \ln p(\mathbf{y}_i \,|\, s_i, \mathbf{x}_i, \tilde{\Lambda}, \Psi) \right] && (4.29) \\
\equiv& \mathcal{F}(q(\boldsymbol{\pi}), \{q(\boldsymbol{\nu}^s), q(\tilde{\Lambda}^s), \{q(s_i), q(\mathbf{x}_i \,|\, s_i)\}_{i=1}^{n}\}_{s=1}^{S}, \alpha^*\mathbf{m}^*, a^*, b^*, \boldsymbol{\mu}^*, \boldsymbol{\nu}^*, \Psi, \mathbf{y}) && (4.30) \\
=& \mathcal{F}(q(\boldsymbol{\theta}), q(\mathbf{s}, \mathbf{x}), \Theta) \;. && (4.31)
\end{aligned}$$

Thus the lower bound is a functional of the variational posterior distributions over the parameters, collectively denoted $q(\boldsymbol{\theta})$, a functional of the variational posterior distribution over the hidden variables of every data point, collectively denoted $q(\mathbf{s}, \mathbf{x})$, and also a function of the set of hyperparameters in the model Θ, as given in (4.20). In the last line above, we have dropped $\mathbf{y}$ as an argument for the lower bound since it is fixed. The full variational posterior is

$$p(\boldsymbol{\pi}, \boldsymbol{\nu}, \Lambda, \boldsymbol{\mu}, \mathbf{s}, \mathbf{x} \,|\, \mathbf{y}) \approx q(\boldsymbol{\pi}) \prod_{s=1}^{S} q(\boldsymbol{\nu}^s) q(\tilde{\Lambda}^s) \cdot \prod_{i=1}^{n} \prod_{s_i=1}^{S} q(s_i) q(\mathbf{x}_i \,|\, s_i) \;. \qquad (4.32)$$

Note that if we had not made the factorisation $q(\boldsymbol{\nu}, \Lambda, \boldsymbol{\mu}) \approx q(\boldsymbol{\nu}) q(\Lambda, \boldsymbol{\mu})$, then the last term in $\mathcal{F}$ would have required averages not over $q(\tilde{\Lambda})$, but also over the combined $q(\boldsymbol{\nu}, \tilde{\Lambda})$, which would have become fairly cumbersome, although not intractable.

Decomposition of $\mathcal{F}$

The goal of learning is then to maximise $\mathcal{F}$, thus increasing the lower bound on $\mathcal{L}$, the exact marginal likelihood. Note that there is an interesting trade-off at play here. The last term in equation (4.29) is the log likelihood of the data set averaged over the uncertainty we have in the hidden variables and parameters. We can increase this term by altering Ψ and the variational posterior distributions $q(\boldsymbol{\theta})$ and $q(\mathbf{s}, \mathbf{x})$ so as to maximise this contribution. However the first three lines of (4.29) contain terms that are negative Kullback-Leibler (KL) divergences between the approximate posteriors over the parameters and the priors we hold on them. So to increase the lower bound on the marginal likelihood (which does not necessarily imply that the marginal likelihood itself increases, since the bound is not tight), we should also consider moving our approximate posteriors towards the priors, thus decreasing the respective KL divergences. In this manner $\mathcal{F}$ elegantly incorporates the trade-off between modelling the data and remaining consistent with our prior beliefs. Indeed if there were no contributions from the data (i.e. the last term in equation (4.29) were zero) then the optimal approximate posteriors would default to the prior distributions.

At this stage it is worth noting that, with the exception of the first term in equation (4.29), $\mathcal{F}$ can be broken down into contributions from each component of the mixture (indexed by s). This fact that will be useful later when we wish to compare how well each component of the mixture is modelling its respective data.

4.2.4 Optimising the lower bound

To optimise the lower bound we simply take functional derivatives with respect to each of the $q(\cdot)$ distributions and equate these to zero to find the distributions that extremise $\mathcal{F}$ (see chapter 2). Synchronous updating of the variational posteriors is not guaranteed to increase $\mathcal{F}$ but consecutive updating of dependent distributions is. The result is that each update is guaranteed to monotonically and maximally increase $\mathcal{F}$.

The update for the variational posterior over mixing proportions $\boldsymbol{\pi}$:

$$\frac{\partial \mathcal{F}}{\partial q(\boldsymbol{\pi})} = \ln p(\boldsymbol{\pi} \,|\, \alpha^* \mathbf{m}^*) + \sum_{i=1}^{n} \sum_{s_i=1}^{S} q(s_i) \ln p(s_i \,|\, \boldsymbol{\pi}) - \ln q(\boldsymbol{\pi}) + c \tag{4.33}$$

$$= \ln \left[\prod_{s=1}^{S} \pi_s^{\alpha^* m_s^* - 1} \cdot \prod_{i=1}^{n} \prod_{s_i=1}^{S} \pi_{s_i}^{q(s_i)} \right] - \ln q(\boldsymbol{\pi}) + c \tag{4.34}$$

$$= \ln \left[\prod_{s=1}^{S} \pi_s^{\alpha^* m_s^* + \sum_{i=1}^{n} q(s_i) - 1} \right] - \ln q(\boldsymbol{\pi}) + c \tag{4.35}$$

$$\Longrightarrow \quad q(\boldsymbol{\pi}) = \mathrm{Dir}(\boldsymbol{\pi} \,|\, \alpha \mathbf{m}) \;, \tag{4.36}$$

where each element of the variational parameter $\alpha \mathbf{m}$ is given by:

$$\alpha m_s = \alpha^* \mathbf{m}_s^* + \sum_{i=1}^{n} q(s_i) \;, \tag{4.37}$$

which gives $\alpha = \alpha^* + n$. Thus the strength of our posterior belief in the mean $\mathbf{m}$ increases with the amount of data in a very simple fashion. For this update we have taken $m_s^* = 1/S$ from (4.15), and used $\sum_{s=1}^{S} m_s = 1$.

The variational posterior in the precision parameter for the l^{th} column of the s^{th} factor loading matrix Λ^s,

$$\frac{\partial \mathcal{F}}{\partial q(\nu_l^s)} = \ln p(\nu_l^s \,|\, a^*, b^*) + \int d\Lambda^s \; q(\Lambda^s) \ln p(\Lambda_l^s \,|\, \nu_l^s) - \ln q(\nu_l^s) + c \tag{4.38}$$

$$= (a^* - 1) \ln \nu_l^s - b^* \nu_l^s + \frac{1}{2} \sum_{q=1}^{p} \left[\ln \nu_l^s - \nu_l^s \left\langle {\Lambda_{ql}^s}^2 \right\rangle_{q(\Lambda^s)} \right] - \ln q(\nu_l^s) + c \,, \tag{4.39}$$

which implies that the precision is Gamma distributed:

$$q(\nu_l^s) = \mathrm{Ga}(\nu_l^s \mid \, a^* + \frac{p}{2}, \; b^* + \frac{1}{2} \sum_{q=1}^{p} \left\langle {\Lambda_{ql}^s}^2 \right\rangle_{q(\Lambda^s)}) = \mathrm{Ga}(\nu_l^s \,|\, a, b_l^s) \,, \tag{4.40}$$

Note that these updates constitute the key steps for the ARD mechanisms in place over the columns of the factor loading matrices.

The variational posterior over the centres and factor loadings of each analyser is obtained by taking functional derivatives with respect to $q(\tilde{\Lambda})$:

$$\frac{\partial \mathcal{F}}{\partial q(\tilde{\Lambda}^s)} = \int d\boldsymbol{\nu}^s \; q(\boldsymbol{\nu}^s) \ln p(\tilde{\Lambda}^s \,|\, \boldsymbol{\nu}^s, \boldsymbol{\mu}^*, \boldsymbol{\nu}^*)$$

$$+ \sum_{i=1}^{n} q(s_i) \int d\mathbf{x}_i \; q(\mathbf{x}_i \,|\, s_i) \ln p(\mathbf{y}_i \,|\, s_i, \mathbf{x}_i, \tilde{\Lambda}^{s_i}, \Psi) - \ln q(\tilde{\Lambda}^s) + c \tag{4.41}$$

$$\begin{aligned} &= \frac{1}{2} \int d\boldsymbol{\nu}^s \; q(\boldsymbol{\nu}^s) \sum_{q=1}^{p} \sum_{l=1}^{k} [\ln \nu_l^s - \nu_l^s {\Lambda_{ql}^s}^2] \\ &+ \frac{1}{2} \sum_{q=1}^{p} \left[\ln \nu_q^* - \nu_q^* \left(\mu_q^s - \mu_q^* \right)^2 \right] \qquad - \ln q(\Lambda^s, \boldsymbol{\mu}^s) + c \\ &- \frac{1}{2} \sum_{i=1}^{n} q(s_i) \mathrm{tr} \left[\Psi^{-1} \left\langle \left(\mathbf{y}_i - \begin{bmatrix} \Lambda^s & \boldsymbol{\mu}^s \end{bmatrix} \begin{bmatrix} \mathbf{x}_i \\ 1 \end{bmatrix} \right) \left(\mathbf{y}_i - \begin{bmatrix} \Lambda^s & \boldsymbol{\mu}^s \end{bmatrix} \begin{bmatrix} \mathbf{x}_i \\ 1 \end{bmatrix} \right)^{\top} \right\rangle_{q(\mathbf{x}_i \,|\, s_i)} \right] \end{aligned} \tag{4.42}$$

where were have moved from the $\tilde{\Lambda}$ notation to using both Λ and $\boldsymbol{\mu}$ separately to express the different prior form separately. In (4.42), there are two summations over the rows of the factor loading matrix, and a trace term, which can also be written as a sum over rows. Therefore the posterior factorises over the rows of $\tilde{\Lambda}^s$,

$$q(\tilde{\Lambda}^s) = \prod_{q=1}^{p} q(\tilde{\Lambda}_{q\cdot}^s) = \prod_{q=1}^{p} \mathrm{N}(\tilde{\Lambda}_{q\cdot}^s \,|\, \overline{\tilde{\Lambda}}_{q\cdot}^s, \tilde{\Gamma}_q^s) \,, \tag{4.43}$$

where $\tilde{\Lambda}^s_{q\cdot}$ denotes the column vector corresponding to the qth row of $\tilde{\Lambda}^s$, which has $k_s + 1$ dimensions. To clarify the notation, this vector then has mean $\overline{\tilde{\Lambda}}^s_{q\cdot}$, and covariance matrix $\tilde{\Gamma}^s_q$. These variational posterior parameters are given by:

$$\tilde{\Gamma}^s_q = \begin{bmatrix} \Sigma^{q,s\ -1}_{\Lambda\Lambda} & \Sigma^{q,s\ -1}_{\Lambda\mu} \\ \Sigma^{q,s-1}_{\mu\Lambda} & \Sigma^{q,s-1}_{\mu\mu} \end{bmatrix}^{-1} \qquad \text{of size } (k_s+1)\times(k_s+1) \tag{4.44}$$

$$\overline{\tilde{\Lambda}}^s_{q\cdot} = \begin{bmatrix} \overline{\Lambda}^s_{q\cdot} \\ \overline{\mu}^s_q \end{bmatrix} \qquad \text{of size } (k_s+1)\times 1 \tag{4.45}$$

with

$$\Sigma^{q,s\ -1}_{\Lambda\Lambda} = \operatorname{diag}\left\langle \boldsymbol{\nu}^s \right\rangle_{q(\boldsymbol{\nu}^s)} + \Psi^{-1}_{qq} \sum_{i=1}^n q(s_i) \left\langle \mathbf{x}_i \mathbf{x}_i{}^\top \right\rangle_{q(\mathbf{x}_i \,|\, s_i)} \tag{4.46}$$

$$\Sigma^{q,s-1}_{\mu\mu} = \nu^*_q + \Psi^{-1}_{qq} \sum_{i=1}^n q(s_i) \tag{4.47}$$

$$\Sigma^{q,s\ -1}_{\Lambda\mu} = \Psi^{-1}_{qq} \sum_{i=1}^n q(s_i) \left\langle \mathbf{x}_i \right\rangle_{q(\mathbf{x}_i \,|\, s_i)} = \Sigma^{q,s\ -1\ \top}_{\mu\Lambda} \tag{4.48}$$

$$\overline{\Lambda}^s_{q\cdot} = \left[\tilde{\Gamma}^s_q\right]_{\Lambda\Lambda} \left(\Psi^{-1}_{qq} \sum_{i=1}^n q(s_i) \mathbf{y}_{i,q} \left\langle \mathbf{x}_i \right\rangle_{q(\mathbf{x}_i \,|\, s_i)} \right) \tag{4.49}$$

$$\overline{\mu}^s_q = \left[\tilde{\Gamma}^s_q\right]_{\mu\mu} \left(\Psi^{-1}_{qq} \sum_{i=1}^n q(s_i) \mathbf{y}_{i,q} + \nu^*_q \mu^*_q \right) \ . \tag{4.50}$$

This somewhat complicated posterior is the result of maintaining a tractable joint over the centres and factor loadings of each analyser. Note that the optimal distribution for each $\tilde{\Lambda}^s$ matrix as a whole now has block diagonal covariance structure: even though each $\tilde{\Lambda}^s$ is a $(p \times (k_s + 1))$ matrix, its covariance only has $O(p(k_s + 1)^2)$ parameters — a direct consequence of the likelihood factorising over the output dimensions.

The variational posterior for the hidden factors $\mathbf{x}_i$, conditioned on the indicator variable s_i, is given by taking functional derivatives with respect to $q(\mathbf{x}_i \,|\, s_i)$:

$$\begin{aligned} \frac{\partial \mathcal{F}}{\partial q(\mathbf{x}_i \,|\, s_i)} &= q(s_i) \ln p(\mathbf{x}_i) + \int d\tilde{\Lambda}^{s_i}\ q(\tilde{\Lambda}^{s_i}) q(s_i) \ln p(\mathbf{y}_i \,|\, s_i, \mathbf{x}_i, \tilde{\Lambda}^{s_i}, \Psi) \\ &\qquad - q(s_i) \ln q(\mathbf{x}_i \,|\, s_i) + c & (4.51) \\ &= q(s_i) \left[-\frac{1}{2} \mathbf{x}_i{}^\top I \, \mathbf{x}_i - \frac{1}{2} \mathrm{tr} \left[\Psi^{-1} \left\langle \left(\mathbf{y}_i - \tilde{\Lambda}^{s_i} \begin{bmatrix} \mathbf{x}_i \\ 1 \end{bmatrix} \right) \left(\mathbf{y}_i - \tilde{\Lambda}^{s_i} \begin{bmatrix} \mathbf{x}_i \\ 1 \end{bmatrix} \right)^\top \right\rangle_{q(\Lambda^{s_i})} \right] \right. \\ &\qquad \left. - \ln q(\mathbf{x}_i \,|\, s_i) \right] + c & (4.52) \end{aligned}$$

which, regardless of the value of $q(s_i)$, produces the Gaussian posterior in $\mathbf{x}_i$ for each setting of s_i:

$$q(\mathbf{x}_i \,|\, s) = \mathrm{N}(\mathbf{x}_i \,|\, \overline{\mathbf{x}}_i^s, \Sigma^s) \tag{4.53}$$

$$\text{with} \quad [\Sigma^s]^{-1} = \mathrm{I} + \left\langle \Lambda^{s\top} \Psi^{-1} \Lambda^s \right\rangle_{q(\tilde{\Lambda}^s)} \tag{4.54}$$

$$\overline{\mathbf{x}}_i^s = \Sigma^s \left\langle \Lambda^{s\top} \Psi^{-1} (\mathbf{y}_i - \boldsymbol{\mu}^s) \right\rangle_{q(\tilde{\Lambda}^s)} \tag{4.55}$$

Note that the covariance Σ^s of the hidden state is the same for every data point, and is not a function of the posterior responsibility $q(s_i)$, as in ordinary factor analysis — only the *mean* of the posterior over $\mathbf{x}_i$ is a function of the data $\mathbf{y}_i$. Note also that the $\overline{\mathbf{x}}_i^s$ depend indirectly on the $q(s_i)$ through (4.49), which is the update for the factor loadings and centre position of analyser s.

The variational posterior for the set of indicator variables $\mathbf{s} = \{s_i\}_{i=1}^n$ is given by

$$\begin{aligned}\frac{\partial \mathcal{F}}{\partial q(s_i)} = &\int d\boldsymbol{\pi} \; q(\boldsymbol{\pi}) \ln p(s_i \,|\, \boldsymbol{\pi}) - \int d\mathbf{x}_i \; q(\mathbf{x}_i \,|\, s_i) \ln q(\mathbf{x}_i \,|\, s_i) \\ &+ \int d\tilde{\Lambda}^{s_i} \; q(\tilde{\Lambda}^{s_i}) \int d\mathbf{x}_i \; q(\mathbf{x}_i \,|\, s_i) \ln p(\mathbf{y}_i \,|\, s_i, \mathbf{x}_i, \tilde{\Lambda}^{s_i}, \Psi) - \ln q(s_i) + c\end{aligned} \tag{4.56}$$

which, utilising a result of Dirichlet distributions given in appendix A, yields

$$\begin{aligned} q(s_i) = \frac{1}{\mathcal{Z}_i} \exp \Bigg[&\psi(\alpha m_{s_i}) - \psi(\alpha) + \frac{1}{2} \ln |\Sigma^{s_i}| \\ &- \frac{1}{2} \mathrm{tr} \left[\Psi^{-1} \left\langle \left(\mathbf{y}_i - \tilde{\Lambda}^{s_i} \begin{bmatrix} \mathbf{x}_i \\ 1 \end{bmatrix} \right) \left(\mathbf{y}_i - \tilde{\Lambda}^{s_i} \begin{bmatrix} \mathbf{x}_i \\ 1 \end{bmatrix} \right)^{\top} \right\rangle_{q(\tilde{\Lambda}^{s_i}) q(\mathbf{x}_i \,|\, s_i)} \right] \Bigg] , \end{aligned} \tag{4.57}$$

where $\mathcal{Z}_i$ is a normalisation constant for each data point, such that $\sum_{s_i=1}^S q(s_i) = 1$, and $\psi(\cdot)$ is the digamma function.

By examining the dependencies of each variational posterior's update rules on the other distributions, it becomes clear that certain update orderings are more efficient than others in increasing $\mathcal{F}$. For example, the $q(\mathbf{x}_i \,|\, s_i)$, $q(\tilde{\Lambda})$ and $q(s_i)$ distributions are highly coupled and it therefore might make sense to perform these updates several times before updating $q(\boldsymbol{\pi})$ or $q(\boldsymbol{\nu})$.

4.2.5 Optimising the hyperparameters

The hyperparameters for a Bayesian MFA are $\Theta = (\alpha^* \mathbf{m}^*, a^*, b^*, \boldsymbol{\mu}^*, \boldsymbol{\nu}^*, \Psi)$.

Beginning with Ψ, we simply take derivatives of $\mathcal{F}$ with respect to Ψ^{-1}, leading to:

$$\begin{aligned}\frac{\partial \mathcal{F}}{\partial \Psi^{-1}} = -\frac{1}{2}\sum_{i=1}^{n}\sum_{s_i=1}^{S} q(s_i) \int d\tilde{\Lambda}^{s_i}\, q(\tilde{\Lambda}^{s_i}) \int d\mathbf{x}_i\, q(\mathbf{x}_i \,|\, s_i) \cdot \\ \frac{\partial}{\partial \Psi^{-1}} \left[\left(\mathbf{y}_i - \tilde{\Lambda}^{s_i} \begin{bmatrix} \mathbf{x}_i \\ 1 \end{bmatrix} \right)^{\top} \Psi^{-1} \left(\mathbf{y}_i - \tilde{\Lambda}^{s_i} \begin{bmatrix} \mathbf{x}_i \\ 1 \end{bmatrix} \right) + \ln |\Psi| \right]\end{aligned} \tag{4.58}$$

$$\implies \Psi^{-1} = \mathrm{diag}\left[\frac{1}{N} \sum_{i=1}^{n} \left\langle \left(\mathbf{y}_i - \tilde{\Lambda}^{s} \begin{bmatrix} \mathbf{x}_i \\ 1 \end{bmatrix} \right) \left(\mathbf{y}_i - \tilde{\Lambda}^{s} \begin{bmatrix} \mathbf{x}_i \\ 1 \end{bmatrix} \right)^{\top} \right\rangle_{q(\tilde{\Lambda}^s) q(s_i) q(\mathbf{x}_i \,|\, s_i)} \right] \tag{4.59}$$

where here we use diag as the operator which sets off-diagonal terms to zero.

By writing $\mathcal{F}$ as a function of a^* and b^* only, we can differentiate with respect to these hyperparameters to yield the fixed point equations:

$$\begin{aligned}\mathcal{F}(a^*, b^*) &= \sum_{s=1}^{S} \int d\boldsymbol{\nu}^s\, q(\boldsymbol{\nu}^s) \ln p(\boldsymbol{\nu}^s \,|\, a^*, b^*) + c & (4.60)\\ &= \sum_{s=1}^{S}\sum_{l=1}^{k} \int d\boldsymbol{\nu}_l^s\, q(\boldsymbol{\nu}_l^s) \left[a^* \ln b^* - \ln \Gamma(a^*) + (a^* - 1) \ln \nu_l^s - b^* \nu_l^s \right] + c\,, & (4.61)\end{aligned}$$

$$\frac{\partial \mathcal{F}}{\partial a^*} = 0 \quad \implies \quad \psi(a^*) = \ln(b^*) + \frac{1}{Sk} \sum_{s=1}^{S}\sum_{l=1}^{k} \langle \ln \nu_l^s \rangle_{q(\nu_l^s)} \tag{4.62}$$

$$\frac{\partial \mathcal{F}}{\partial b^*} = 0 \quad \implies \quad b^{*-1} = \frac{1}{a^* Sk} \sum_{s=1}^{S}\sum_{l=1}^{k} \langle \nu_l^s \rangle_{q(\nu_l^s)} \ . \tag{4.63}$$

Solving for the fixed point amounts to setting the prior distribution's first moment and first logarithmic moment to the respective averages of those quantities over the factor loading matrices. The expectations for the gamma random variables are given in appendix A.

Similarly, by writing $\mathcal{F}$ as a function of α^* and $\mathbf{m}^*$ only, we obtain

$$\begin{aligned}\mathcal{F}(\alpha^*, \mathbf{m}^*) &= \int d\boldsymbol{\pi}\, q(\boldsymbol{\pi}) \ln p(\boldsymbol{\pi} \,|\, \alpha^* \mathbf{m}^*) & (4.64)\\ &= \int d\boldsymbol{\pi}\, q(\boldsymbol{\pi}) \left[\ln \Gamma(\alpha^*) - \sum_{s=1}^{S} \left[\ln \Gamma(\alpha^* m_s^*) - (\alpha^* m_s^* - 1) \ln \pi_s \right] \right] \ . & (4.65)\end{aligned}$$

Bearing in mind that $q(\boldsymbol{\pi})$ is Dirichlet with parameter $\alpha\mathbf{m}$, and that we have a scaled prior $m_s^* = 1/S$ as given in (4.15), we can express the lower bound as a function of α^* only:

$$\mathcal{F}(\alpha^*) = \ln\Gamma(\alpha^*) - S\ln\Gamma\left(\frac{\alpha^*}{S}\right) + \left(\frac{\alpha^*}{S} - 1\right)\sum_{s=1}^{S}\left[\psi(\alpha m_s) - \psi(\alpha)\right] \tag{4.66}$$

Taking derivatives of this quantity with respect to α^* and setting to zero, we obtain:

$$\psi(\alpha^*) - \psi(\frac{\alpha^*}{S}) = \frac{1}{S}\sum_{s=1}^{S}\left[\psi(\alpha) - \psi(\alpha m_s)\right]. \tag{4.67}$$

The second derivative with respect to α^* of (4.66) is negative for $\alpha^* > 0$, which implies the solution of (4.67) is a maximum. This maximum can be found using gradient following techniques such as Newton-Raphson. The update for $\mathbf{m}^*$ is not required, since we assume that the prior over the mixing proportions is symmetric.

The update for the prior over the centres $\{\boldsymbol{\mu}^s\}_{S=1}^{S}$ of each of the factor analysers is given by considering terms in $\mathcal{F}$ that are functions of $\boldsymbol{\mu}^*$ and $\boldsymbol{\nu}^*$:

$$\mathcal{F}(\boldsymbol{\mu}^*, \boldsymbol{\nu}^*) = \int d\boldsymbol{\mu}\; q(\boldsymbol{\mu}) \ln p(\boldsymbol{\mu} \,|\, \boldsymbol{\mu}^*, \boldsymbol{\nu}^*) \tag{4.68}$$

$$= \frac{1}{2}\sum_{s=1}^{S}\int d\boldsymbol{\mu}^s\; q(\boldsymbol{\mu}^s)\left[\ln\left|\mathrm{diag}\left(\boldsymbol{\nu}^*\right)\right| - (\boldsymbol{\mu}^s - \boldsymbol{\mu}^*)^\top \mathrm{diag}\left(\boldsymbol{\nu}^*\right)(\boldsymbol{\mu}^s - \boldsymbol{\mu}^*)\right]. \tag{4.69}$$

Taking derivatives with respect to $\boldsymbol{\mu}^*$ first, and then $\boldsymbol{\nu}^*$, equating each to zero yields the updates

$$\boldsymbol{\mu}^* = \frac{1}{S}\sum_{s=1}^{S}\langle\boldsymbol{\mu}^s\rangle_{q(\boldsymbol{\mu}^s)} \tag{4.70}$$

$$\boldsymbol{\nu}^* = [\nu_1^*, \ldots, \nu_p^*], \text{ with } \boldsymbol{\nu}_q^* = \frac{1}{S}\sum_{s=1}^{S}\left\langle(\mu_q^s - \mu_q^*)(\mu_q^s - \mu_q^*)\right\rangle_{q(\boldsymbol{\mu}^s)}, \tag{4.71}$$

where the update for $\boldsymbol{\nu}^*$ uses the already updated $\boldsymbol{\mu}^*$.

4.3 Model exploration: birth and death

We already have an ARD mechanism in place to discover the local dimensionality for each analyser in the mixture, as part of the inference procedure over the precisions $\boldsymbol{\nu}$. However we have not yet addressed the problem of inferring the number of analysers.

The advantage of the Bayesian framework is that different model structures can be compared without having to rely on heuristic penalty or cost functions to compare their complexities;

ideally different model structures m and m' should be compared using the difference of log marginal likelihoods $\mathcal{L}(m)$ and $\mathcal{L}(m')$. In this work we use $\mathcal{F}(m)$ and $\mathcal{F}(m')$ as guides to the intractable log marginal likelihoods.

This has advantages over unpenalised maximum likelihood methods where, for example, in the split and merge algorithm described in Ueda et al. (2000) changes to model complexity are limited to simultaneous split and merge operations such that the number of components in the mixture remain the same. Whilst this approach is unable to explore differing sizes of models, it is successful in avoiding some local maxima in the optimisation process. For example, a Gaussian component straddled between two distinct clusters of data is an ideal candidate for a split operation — unfortunately their method requires that this split be accompanied with a merging of two other components elsewhere to keep the number of components fixed.

In our Bayesian model, though, we are allowed to propose any changes to the number of components in the mixture. We look at the simple cases of incremental and decremental changes to the total number, S, since we do not expect wild changes to the model structure to be an efficient method for exploring the space. This is achieved through *birth* and *death* ‘moves’, where a component is removed from or introduced into the mixture model. This modified model is then trained further as described in section 4.2.4 until a measure of convergence is reached (see below), at which point the proposal is accepted or rejected based on the change in $\mathcal{F}$. Another proposal is then made and the procedure repeated, up to a point when no further proposals are accepted. In this model (although not in a general application) component death occurs naturally as a by-product of the optimisation; the following sections explain the death mechanism, and address some interesting aspects of the birth process, which we have more control over.

Our method is similar to that of Reversible Jump Markov chain Monte Carlo (RJMCMC) (Green, 1995) applied to mixture models, where birth and death moves can also be used to navigate amongst different sized models (Richardson and Green, 1997). By sampling in the full space of model parameters for all structures, RJMCMC methods converge to the exact posterior distribution over structures. However, in order to ensure reversibility of the Markov chain, complicated Metropolis-Hastings acceptance functions need to be derived and evaluated for each proposal from one parameter subspace to another. Moreover, the method suffers from the usual problems of MCMC methods, namely difficulty in assessing convergence and long simulation run time. The variational Bayesian method attempts to estimate the posterior distribution directly, not by obtaining samples of parameters and structures, but by attempting to directly integrate over the parameters using a lower bound arrived at deterministically. Moreover, we can obtain a surrogate for the posterior distribution over model structures, $p(m \,|\, \mathbf{y})$, which is not represented as some large set of samples, but is obtained using a quantity proportional to $p(m)\exp\{\mathcal{F}(m)\}$, where $\mathcal{F}(m)$ is the optimal (highest) lower bound achieved for a model m of particular structure.

4.3.1 Heuristics for component death

There are two routes for a component death occurring in this model, the first is by natural causes and the second through intervention. Each is explained in turn below.

When optimising $\mathcal{F}$, occasionally one finds that for some mixture component s': $\sum_{i=1}^{n} q(s'_i) = 0$ (to machine precision), even though the component still has non-zero prior probability of being used in the mixture, $p(s'_i) = \int d\boldsymbol{\pi} p(\boldsymbol{\pi}) p(s'_i \,|\, \boldsymbol{\pi})$. This is equivalent to saying that it has no responsibility for any of the data, and as a result its parameter posterior distributions have defaulted exactly to the priors. For example, the mean location of the centre of the analyser component is at the centre of the prior distribution (this can be deduced from examining (4.50) for the case of $q(s'_i) = 0 \; \forall \; i$), and the factor loadings have mean zero and high precisions $\boldsymbol{\nu}^{s'}$, referring to (4.40). If the mean of the prior over analyser centres is not located near data (see next removal method below), then this component is effectively redundant (it cannot even model data with the uniquenesses matrix Ψ, say), and can be removed from the model. How does the removal of this component affect the lower bound on the marginal likelihood, $\mathcal{F}$? Since the posterior responsibility of the component is zero it does not contribute to the last term of (4.29), which sums over the data, n. Also, since its variational posteriors over the parameters are all in accord with the priors, then the KL divergence terms in (4.29) are all zero, *except* for the very first term which is the negative KL divergence between the variational posterior and prior distribution over the mixing proportions $\boldsymbol{\pi}$. Whilst the removal of the component leaves all other terms in $\mathcal{F}$ unchanged, not having this ‘barren’ dimension s' to integrate over should increase this term.

It seems counter-intuitive that the mean of the prior over factor analyser centres might be far from data, as suggested in the previous paragraph, given that the hyperparameters of the prior are updated to reflect the position of the analysers. However, there are cases in which the distribution of data is ‘hollow’ (see, for example, the spiral data set of section 4.5.3), and in this case redundant components are very easily identified with zero responsibilities, and removed. If the redundant components default to a position which is close to data, their posterior responsibilities may not fall to exactly zero, being able to still use the covariance given in Ψ to model the data. In this case a more aggressive pruning procedure is required, where we examine the change in $\mathcal{F}$ that occurs after removing a component we suspect is becoming, or has become, redundant. We gain by not having to code its parameters, but we may lose if the data in its locale are being uniquely modelled by it, in which case $\mathcal{F}$ may drop. If $\mathcal{F}$ should drop, there is the option of continuing the optimisation to see if $\mathcal{F}$ eventually improves (see next section on birth processes), and rejecting the removal operation if it does not. We do not implement this ‘testing’ method in our experiments, and rely solely on the first method and remove components once their total posterior responsibilities fall below a reasonable level (in practice less than one data point’s worth).

This mechanism for (automatic) removal of components is useful as it allows the data to dictate how many mixture components are required. However we should note that if the data is not distributed as a mixture of Gaussian components, the size of the data set will affect the returned number of components. Thus the number of components should not be taken to mean the number of 'clusters'.

4.3.2 Heuristics for component birth

Component birth does not happen spontaneously during learning, so we have to introduce a heuristic. Even though changes in model structure may be proposed at any point during learning, it makes sense only to do so when learning has plateaued, so as to exploit (in terms of $\mathcal{F}$) the current structure to the full. We define an *epoch* as that period of learning beginning with a proposal of a model alteration, up to the point of convergence of the variational learning rules.

One possible heuristic for deciding at which point to end an epoch can be constructed by looking at the rate of change of the lower bound with iterations of variational EM. If $\Delta\mathcal{F} = \mathcal{F}^{(t)} - \mathcal{F}^{(t-1)}$ falls below a critical value then we can assume that we have plateaued. However it is not easy to define such simple thresholds in a manner that scales appropriately with both model complexity and amount of data. An alternative (implemented in the experiments) is to examine the rate of change of the posterior class-conditional responsibilities, as given in the $q(s_i)$ matrix $(n \times S)$. A suitable function of this sort can be such that it does not depend directly on the data size, dimensionality, or current model complexity. In this work we consider the end of an epoch to be when the *rate of change of responsibility* for each analyser, averaged over all data, falls below a tolerance — this has the intuitive interpretation that the components are no longer 'in flux' and are modelling their data as best they can in that configuration. We shall call this quantity the *agitation*:

$$agitation(s)^{(t)} \equiv \frac{\sum_{i=1}^{n} \left| q(s_i)^{(t)} - q(s_i)^{(t-1)} \right|}{\sum_{i=1}^{n} q(s_i)^{(t)}} , \tag{4.72}$$

where (t) denotes the iteration number of VBEM. We can see that the agitation of each analyser does not directly scale with number of analysers, data points, or dimensionality of the data. Thus a fixed tolerance for this quantity can be chosen that is applicable throughout the optimisation process. We should note that this measure is one of many possible, such as using squared norms etc.

A sensible way to introduce a component into the model is to create that component in the image of an existing component, which we shall call the *parent*. Simply reproducing the exact parameters of the parent does not suffice as the symmetry of the resulting pair needs to be broken for them to model the data differently.

One possible approach would be to remove the parent component, s', and *replace* it with two components, the 'children', with their means displaced symmetrically about the parent's mean, by a vector sampled from the parent's distribution, its covariance ellipsoid given by $\Lambda^{s'}\Lambda^{s'\top} + \Psi$. We call this a *spatial* split. This appeals to the notion that one might expect areas of data that are currently being modelled by one elongated Gaussian to be modelled better by two, displaced most probably along the major axis of variance of that data. However this approach is hard to fine tune so that it scales well with the data dimensionality, p. For example, if the displacement is slightly too large then it becomes very likely in high dimensions that both children model the data poorly and die naturally as a result. If it is too small then the components will diverge very slowly.

Again appealing to the class-conditional responsibilities for the data, we can define a procedure for splitting components that is not directly a function of the dimensionality, or any length scale of the local data. The approach taken in this work uses a partition of the parent's posterior responsibilities for each of the data, $q(s_i = s')$, along a direction $\mathbf{d}^{s'}$ sampled from the parent's covariance ellipsoid. Those data having a positive dot product with the sampled direction donate their responsibilities to one child s^a, and vice-versa for the other child s^b. Mathematically, we sample a direction $\mathbf{d}$ and define an allocation indicator variable for each data point,

$$\mathbf{d} \sim \mathrm{N}(\mathbf{d} \,|\, \langle \boldsymbol{\mu}^{s'} \rangle_{q(\boldsymbol{\mu}^{s'})}, \langle \Lambda^{s'} \Lambda^{s'\top} \rangle_{q(\Lambda^{s'})} + \Psi) \tag{4.73}$$

$$r_i = \begin{cases} 1 & \text{if} \quad (\mathbf{y}_i - \boldsymbol{\mu}^{s'})^\top \mathbf{d} \geq 0 \\ 0 & \text{if} \quad (\mathbf{y}_i - \boldsymbol{\mu}^{s'})^\top \mathbf{d} < 0 \end{cases} \qquad \text{for } i = 1, \dots, n\,. \tag{4.74}$$

We then set the posterior probabilities in $q(s_i)$ to reflect these assignments, introducing a *hardness* parameter α_h, ranging from .5 to 1:

$$q(s_i^a) = q(s_i') \left[\alpha_h r_i + (1 - \alpha_h)(1 - r_i) \right] \tag{4.75}$$

$$q(s_i^b) = q(s_i') \left[(1 - \alpha_h) r_i + \alpha_h (1 - r_i) \right] \tag{4.76}$$

When $\alpha_h = 1$, all the responsibility is transferred to the assigned child, and when $\alpha_h = .5$ the responsibility is shared equally. In the experiments in this chapter we use $\alpha_h = 1$.

The advantage of this approach is that the birth is made in *responsibility* space rather than data-space, and is therefore dimension-insensitive. The optimisation then continues, with the s' analyser removed and the s^a and s^b analysers in its place. The first variational updates should be for $q(\Lambda^{s^a})$ and $q(\Lambda^{s^b})$ since these immediately reflect the change (note that the update for $q(\mathbf{x}_i)$ is not a function of the responsibilities — see equation (4.53)).

The mechanism that chooses which component is to be the parent of a pair-birth operation must allow the space of models to be explored fully. A simple method would be to pick the component at random amongst those present. This has an advantage over a deterministic method, in that the latter could preclude some components from ever being considered. Interestingly though, there is information in $\mathcal{F}$ that can be used to guide the choice of component to split: with the exception of the first term in equation (4.29), the remaining terms can be decomposed into component-specific contributions, $\mathcal{F}_s$. An ordering for parent choice can be defined using $\mathcal{F}_s$, with the result that is it possible to concentrate attempted births on those components that are not currently modelling their data well. This mirrors the approach taken in Ueda et al. (2000), where the criterion was the (KL) discrepancy between each analyser's local density model and the empirical density of the data.

If, at the end of an epoch, we reject the proposed birth so returning to the original configuration, we may either attempt to split the same component again, but with a new randomly sampled direction, or move on to the next 'best' component in the ordering. We use the following function to define $\mathcal{F}_s$, from which the ordering is recalculated after every successful epoch:

$$\begin{aligned}
\mathcal{F}_s &= \mathcal{F}(\{Q\}, \alpha^*\mathbf{m}^*, a^*, b^*, \boldsymbol{\mu}^*, \boldsymbol{\nu}^*, \Psi \,|\, Y) \\
&= \int d\boldsymbol{\nu}^s \, q(\boldsymbol{\nu}^s) \left[\ln \frac{p(\boldsymbol{\nu}^s \,|\, a^*, b^*)}{q(\boldsymbol{\nu}^s)} + \int d\tilde{\Lambda}^s \, q(\tilde{\Lambda}^s) \frac{p(\tilde{\Lambda}^s \,|\, \boldsymbol{\nu}^s, \boldsymbol{\mu}^*, \boldsymbol{\nu}^*)}{q(\tilde{\Lambda}^s)} \right] \\
&\quad + \frac{1}{\sum_{i=1}^n q(s_i)} \sum_{i=1}^n q(s_i) \Bigg[\int d\boldsymbol{\pi} \, q(\boldsymbol{\pi}) \ln \frac{p(s_i \,|\, \boldsymbol{\pi})}{q(s_i)} + \int d\mathbf{x}_i \, q(\mathbf{x}_i \,|\, s_i) \ln \frac{p(\mathbf{x}_i)}{q(\mathbf{x}_i \,|\, s_i)} \\
&\quad + \int d\tilde{\Lambda}^s \, q(\tilde{\Lambda}^s) \int d\mathbf{x}_i \, q(\mathbf{x}_i \,|\, s_i) \ln p(\mathbf{y}_i \,|\, s_i, \mathbf{x}_i, \tilde{\Lambda}^s, \Psi) \Bigg]
\end{aligned} \tag{4.77}$$

This has the intuitive interpretation as being the likelihood of the data (weighted by its data responsibilities) under analyser s, normalised by its overall responsibility, with the relevant (KL) penalty terms as in $\mathcal{F}$. Those components with lower $\mathcal{F}_s$ are preferentially split. The optimisation completes when all existing mixture components have been considered as parents, with no accepted epochs.

Toward the end of an optimisation, the remaining required changes to model structure are mainly local in nature and it becomes computationally wasteful to update the parameters of all the components of the mixture model at each iteration of the variational optimisation. For this reason only those components whose responsibilities are in flux (to some threshold) are updated. This partial optimisation approach still guarantees an increase in $\mathcal{F}$, as we simply perform updates that guarantee to increase parts of the $\mathcal{F}$ term in 4.29.

It should be noted that no matter which heuristics are used for birth and death, ultimately the results are always compared in terms of $\mathcal{F}$, the lower bound on the log marginal likelihood $\mathcal{L}$. Therefore different choices of heuristic can only affect the *efficiency* of the search over model structures and not the theoretical validity of the variational approximation. For example, although it is perfectly possible to start the model with many components and let them die, it is computationally more efficient and equally valid to start with one component and allow it to spawn more when necessary.

4.3.3 Heuristics for the optimisation endgame

In the previous subsection we proposed a heuristic for terminating the optimisation, namely that every component should be unsuccessfully split a number of times. However, working in the space of components seems very inefficient. Moreover, there are several pathological birth-death scenarios which raise problems when counting the number of times each component has been split; for example, the identities of nearby components can be switched during an epoch (parent splits into two children, first child usurps an existing other component and models its data, whilst that component switches to model the old parent's data, and the second child dies).

One possible solution (personal communication, Y. Teh) is based on a responsibility accumulation method. Whenever a component s is chosen for a split, we store its responsibility vector (of length n) for all the data points $q(\mathbf{s}) = [q(s_1)\ q(s_2)\ \dots\ q(s_n)]$, and proceed with the optimisation involving its two children. At the end of the epoch, if we have not increased $\mathcal{F}$, we add $q(\mathbf{s})$ to a running total of 'split data' responsibilities, $\mathbf{t} = (t_1, t_2, \dots, t_n)$. That is $\forall i :\ t_i \leftarrow \min(t_i + q(s_i), t_{\text{max}})$, where t_{max} is some saturation point. If by the end of the epoch we have managed to increase $\mathcal{F}$, then the accumulator $\mathbf{t}$ is reset to zero for every data point.

From this construction we can derive a stochastic procedure for choosing which component to split, using the softmax of the quantity $c(s) = \beta \sum_{i=1}^{n} (t_{\text{max}} - t_i) q(s_i)$. If $c(s)$ is large for some component s, then the data it is responsible for has not 'experienced' many birth attempts, and so it should be a strong candidate for a split. Here $\beta \geq 0$ is a temperature parameter to be set as we wish. As β tends to infinity the choice of component to split becomes deterministic, and is based on which has least responsibility overlap with already-split data. If β is very small (but non-zero) the splits become more random. Whatever setting of β, attempted splits will be automatically focused on those components with more data and unexplored regions of data space. Furthermore, a termination criterion is automatic: continue splitting components until every entry of the $\mathbf{t}$ vector has reached saturation — this corresponds to splitting every *data point* a certain number of times (in terms of its responsibility under the split parent), before we terminate the entire optimisation. This idea was conceived of only after the experiments were completed, and so has not been thoroughly investigated.

4.4 Handling the predictive density

In this section we set about trying to get a handle on the predictive density of VBMFA models using bounds on approximations (in section 4.7.1 we will show how to estimate the density using sampling methods). In order to perform density estimation or classification of a new test example, we need to have access to the predictive density

$$p(\mathbf{y}' \,|\, \mathbf{y}) = \frac{p(\mathbf{y}', \mathbf{y})}{p(\mathbf{y})} = \int d\boldsymbol{\theta} \; p(\boldsymbol{\theta} \,|\, \mathbf{y}) p(\mathbf{y}' \,|\, \boldsymbol{\theta}) \tag{4.78}$$

where $\mathbf{y}'$ is a set of test examples $\mathbf{y}' = \{\mathbf{y}'_1, \ldots, \mathbf{y}'_{n'}\}$, and $\mathbf{y}$ is the training data. This quantity is simply the probability of observing the test examples for a particular setting of the model parameters, averaged over the posterior distribution of the parameters given a training set. Unfortunately, the very intractability of the marginal likelihood in equation (4.14) means that the predictive density is also intractable to compute exactly.

A poor man's approximation uses the variational posterior distribution in place of the posterior distribution:

$$p(\mathbf{y}' \,|\, \mathbf{y}) \approx \int d\boldsymbol{\theta} \; q(\boldsymbol{\theta}) p(\mathbf{y}' \,|\, \boldsymbol{\theta}) \; . \tag{4.79}$$

However we might expect this to overestimate the density of $\mathbf{y}'$ in typical regions of space (in terms of where the training data lie), as the variational posterior tends to over-neglect areas of low posterior probability in parameter space. This is a result of the asymmetric KL divergence measure penalty in the optimisation process.

Substituting the form for MFAs given in (4.14) into (4.79)

$$p(\mathbf{y}' \,|\, \mathbf{y}) \approx \int d\boldsymbol{\pi} \int d\tilde{\Lambda} \; q(\boldsymbol{\pi}, \tilde{\Lambda}) \left[\prod_{i=1}^{n'} \sum_{s_i=1}^{S} p(s_i \,|\, \boldsymbol{\pi}) p(\mathbf{y}'_i \,|\, s_i, \tilde{\Lambda}, \Psi) \right] \; , \tag{4.80}$$

which is still intractable for the same reason that the marginal likelihoods of training set were so. We can lower bound the log of the predictive density using variational distributions over the hidden variables corresponding to each test case:

$$\ln p(\mathbf{y}' \,|\, \mathbf{y}) \approx \ln \int d\boldsymbol{\pi} \int d\tilde{\Lambda}\; q(\boldsymbol{\pi}, \tilde{\Lambda}) \left[\prod_{i=1}^{n'} \sum_{s_i=1}^{S} p(s_i \,|\, \boldsymbol{\pi}) p(\mathbf{y}'_i \,|\, s_i, \tilde{\Lambda}, \Psi) \right] \tag{4.81}$$

$$\geq \sum_{i=1}^{n'} \int d\boldsymbol{\pi} \int d\tilde{\Lambda}\; q(\boldsymbol{\pi}, \tilde{\Lambda}) \left[\ln \sum_{s_i=1}^{S} p(s_i \,|\, \boldsymbol{\pi}) p(\mathbf{y}'_i \,|\, s_i, \tilde{\Lambda}, \Psi) \right] \tag{4.82}$$

$$= \sum_{i=1}^{n'} \int d\boldsymbol{\pi}\; q(\boldsymbol{\pi}) \int d\tilde{\Lambda}\; q(\tilde{\Lambda}) \left[\ln \sum_{s_i=1}^{S} q(s_i) \frac{p(s_i \,|\, \boldsymbol{\pi}) p(\mathbf{y}'_i \,|\, s_i, \tilde{\Lambda}, \Psi)}{q(s_i)} \right] \tag{4.83}$$

$$\geq \sum_{i=1}^{n'} \int d\boldsymbol{\pi}\; q(\boldsymbol{\pi}) \int d\tilde{\Lambda}\; q(\tilde{\Lambda}) \sum_{s_i=1}^{S} q(s_i) \ln \frac{p(s_i \,|\, \boldsymbol{\pi}) p(\mathbf{y}'_i \,|\, s_i, \tilde{\Lambda}, \Psi)}{q(s_i)} \tag{4.84}$$

$$\geq \sum_{i=1}^{n'} \sum_{s_i=1}^{S} q(s_i) \Bigg[\int d\boldsymbol{\pi}\; q(\boldsymbol{\pi}) \ln \frac{p(s_i \,|\, \boldsymbol{\pi})}{q(s_i)} + \int d\mathbf{x}_i\; q(\mathbf{x}_i \,|\, s_i) \ln \frac{p(\mathbf{x}_i)}{q(\mathbf{x}_i \,|\, s_i)}$$

$$+ \int d\tilde{\Lambda}^{s_i}\; q(\tilde{\Lambda}^{s_i}) \int d\mathbf{x}_i\; q(\mathbf{x}_i \,|\, s_i) \ln p(\mathbf{y}'_i \,|\, s_i, \mathbf{x}_i, \tilde{\Lambda}^{s_i}, \Psi) \Bigg] . \tag{4.85}$$

The first inequality is a simple Jensen bound, the second is another which introduces a set of variational distributions $q(s_i)$, and the third a further set of distributions over the hidden variables $q(\mathbf{x}_i \,|\, s_i)$. Note that these distributions correspond to the *test* data, indexed from $i = 1, \ldots, n'$. This estimate of the predictive density is then very similar to the lower bound of the marginal likelihood of the training data (4.29), except that the training data $\mathbf{y}_i$ has been replaced with the test data $\mathbf{y}'_i$, and the KL penalty terms on the parameters have been removed. This carries the interpretation that the distribution over parameters of the model is decided upon and fixed (i.e. the variational posterior), and we simply need to explain the test data under this ensemble of models.

This lower bound on the approximation to the predictive density can be optimised in just *two updates* for each test point. First, infer the distribution $q(\mathbf{x}_i \,|\, s_i)$ for each test data point, using the analogous form of update (4.53). Then update the distribution $q(s_i)$ based on the resulting distributions over $q(\mathbf{x}_i \,|\, s_i)$ using the analogous form of update (4.57). Since the $q(\mathbf{x}_i \,|\, s_i)$ update was not a function of $q(s_i)$, we do not need to iterate the optimisation further to improve the bound.

4.5 Synthetic experiments

In this section we present three toy experiments on synthetic data which demonstrate certain features of a Bayesian mixture of factor analysers. The first experiment shows the ability of the algorithm's birth and death processes to find the number of clusters in a dataset. The second experiment shows more ambitiously how we can simultaneously recover the number of clusters and their dimensionalities, and how the complexity of the model depends on the amount of data support. The last synthetic experiment shows the ability of the model to fit a low dimensional manifold embedded in three-dimensional space.

4.5.1 Determining the number of components

In this toy example we tested the model on synthetic data generated from a mixture of 18 Gaussians with 50 points per cluster, as shown in figure 4.3(a). The algorithm was initialised with a single analyser component positioned at the mean of the data. Birth proposals were made using spatial splits (as described above). Also shown is the progress of the algorithm after 7, 14, 16 and 22 accepted epochs (figures 4.3(b)-4.3(e)). The variational algorithm has little difficulty finding the correct number of components and the birth heuristics are successful at avoiding local maxima.

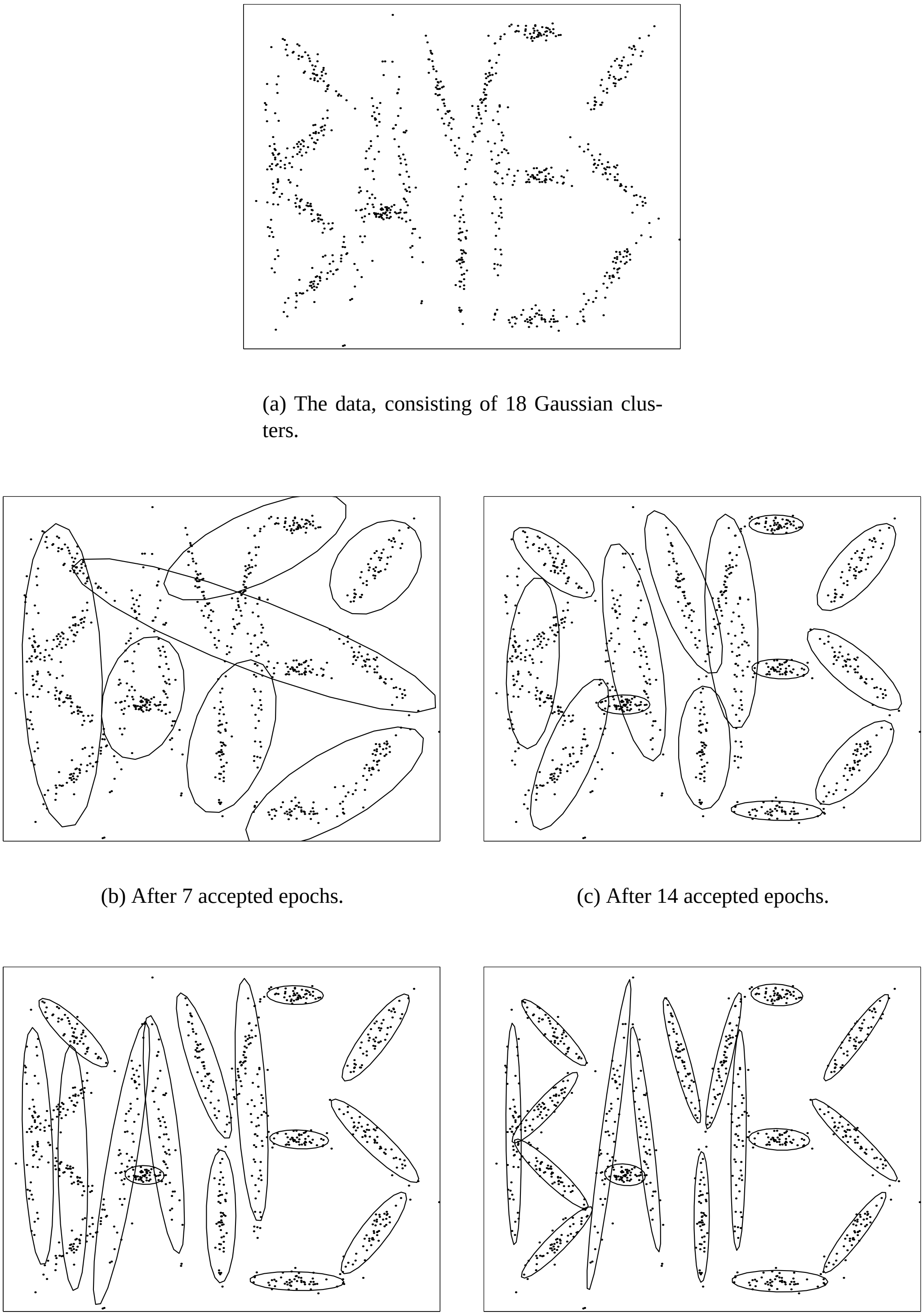
After finding the 18 Gaussians repeated splits are attempted and mostly rejected. Those epochs that are accepted always involve the birth of a component followed at some point by the death of another component, such that the number of components remain 18; the increase in $\mathcal{F}$ over these epochs is extremely small, usually due to the refinement of other components.

4.5.2 Embedded Gaussian clusters

In this experiment we examine the ability of the Bayesian mixture of factor analysers to automatically determine the local dimensionality of high dimensional data. We generated a synthetic data set consisting of 300 data points drawn from each of 6 Gaussian clusters with intrinsic dimensionalities (7 4 3 2 2 1), embedded at random orientations in a 10-dimensional space. The means of the Gaussians were drawn uniformly under $[0, 3]$ in each of the data dimensions, all Gaussian variances set to 1, and sensor noise of covariance $.01$ added in each dimension.

A Bayesian MFA was initialised with one mixture component centred about the data mean, and trained for a total of 200 iterations of variational EM with spatial split heuristics for the birth proposals. All the analysers were created with a maximum dimensionality of 7. The variational Bayesian approach correctly inferred both the number of Gaussians and their intrinsic dimensionalities, as shown in figure 4.4. The dimensionalities were determined by examining the posterior distributions over the precisions of each factor analyser's columns, and thresholding on the mean of each distribution.

We then varied the number of data points in each cluster and trained models on successively smaller data sets. Table 4.1 shows how the Bayesian MFA partitioned the data set. With large



(a) The data, consisting of 18 Gaussian clusters.

(b) After 7 accepted epochs.

(c) After 14 accepted epochs.

(d) After 16 accepted epochs.

(e) After 22 accepted epochs.

Figure 4.3: The original data, and the configuration of the mixture model at points during the optimisation process. Plotted are the 2 s.d. covariance ellipsoids for each analyser in the mixture. To be more precise, the centre of the ellipsoid is positioned at the mean of the variational posterior over the analyser's centre, and each covariance ellipsoid is the expected covariance under the variational posterior.

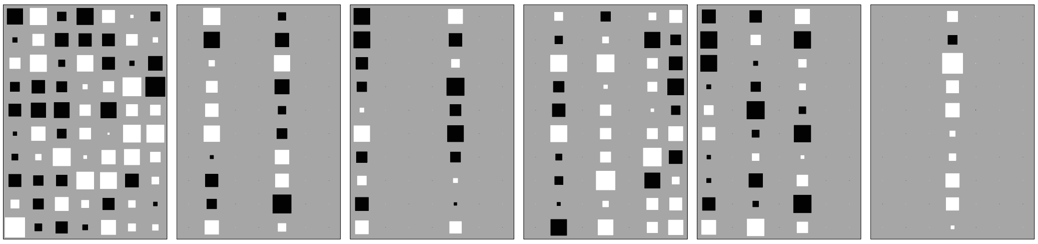


Figure 4.4: Learning the local intrinsic dimensionality. The maximum dimensionality of each analyser was set to 7. Shown are Hinton diagrams for the means of the factor loading matrices $\{\overline{\Lambda}^s\}_{s=1}^S$ for each of the 6 components, after training on the data set with 300 data points per cluster. Note that empty columns correspond to unused factors where the mass of $q(\nu_l^s)$ is at very high values, so the learnt dimensionalities are (7,2,2,4,3,1).

<table>
<tr><th rowspan="2">number of points per cluster</th><th colspan="6">intrinsic dimensionalities</th></tr>
<tr><th>1</th><th>7</th><th>4</th><th>3</th><th>2</th><th>2</th></tr>
<tr><td>8</td><td colspan="4">2</td><td colspan="2">1</td></tr>
<tr><td>8</td><td colspan="2">1</td><td colspan="4">2</td></tr>
<tr><td>16</td><td>1</td><td colspan="4">4</td><td>2</td></tr>
<tr><td>32</td><td>1</td><td>6</td><td>3</td><td>3</td><td>2</td><td>2</td></tr>
<tr><td>64</td><td>1</td><td>7</td><td>4</td><td>3</td><td>2</td><td>2</td></tr>
<tr><td>128</td><td>1</td><td>7</td><td>4</td><td>3</td><td>2</td><td>2</td></tr>
</table>

Table 4.1: The recovered number of analysers and their intrinsic dimensionalities. The numbers in the table are the dimensionalities of the analysers and the boxes represent analysers modelling data from more than one cluster. For a large number of data points per cluster (≥ 64), the Bayesian MFA recovers the generative model. As we decrease the amount of data, the model reduces the dimensionality of the analysers and begins to model data from different clusters with the same analyser. The two entries for 8 data points are two observed configurations that the model converged on.

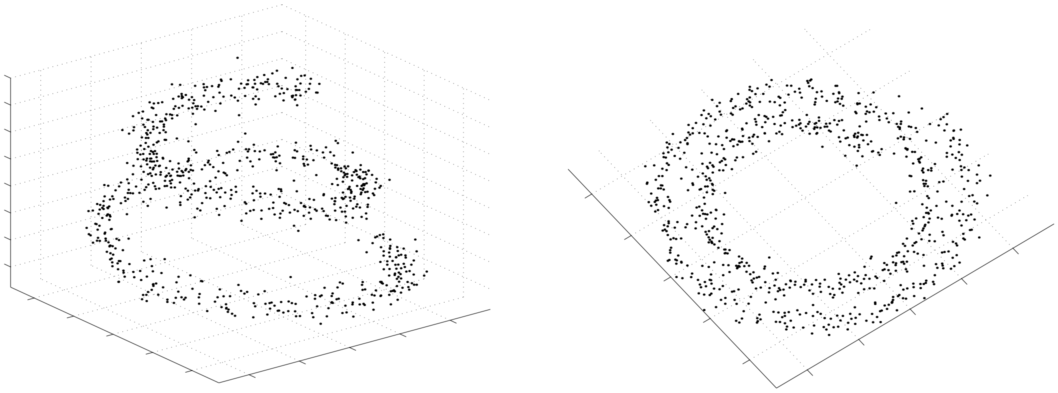
amounts of data the model agrees with the true model, both in the number of analysers and their dimensionalities. As the number of points per cluster is reduced there is insufficient evidence to support the full intrinsic dimensionality, and with even less data the number of analysers drop and they begin to model data from more than one cluster.

4.5.3 Spiral dataset

Here we present a simple synthetic example of how Bayesian MFA can learn locally linear models to tile a manifold for globally non-linear data. We used the dataset of 800 data points from a noisy shrinking spiral, as used in Ueda et al. (2000), given by

$$\mathbf{y}_i = [(13 - 0.5t_i)\cos t_i, \quad -(13 - 0.5t_i)\sin t_i, \quad t_i)] + \mathbf{w}_i \tag{4.86}$$

$$\text{where} \quad t_i \in [0, 4\pi]\,, \qquad \mathbf{w}_i \sim \mathrm{N}(\mathbf{0}, \mathrm{diag}\,([.5\ .5\ .5])) \tag{4.87}$$



(a) An elevated view of the spiral data set (see text for reference).

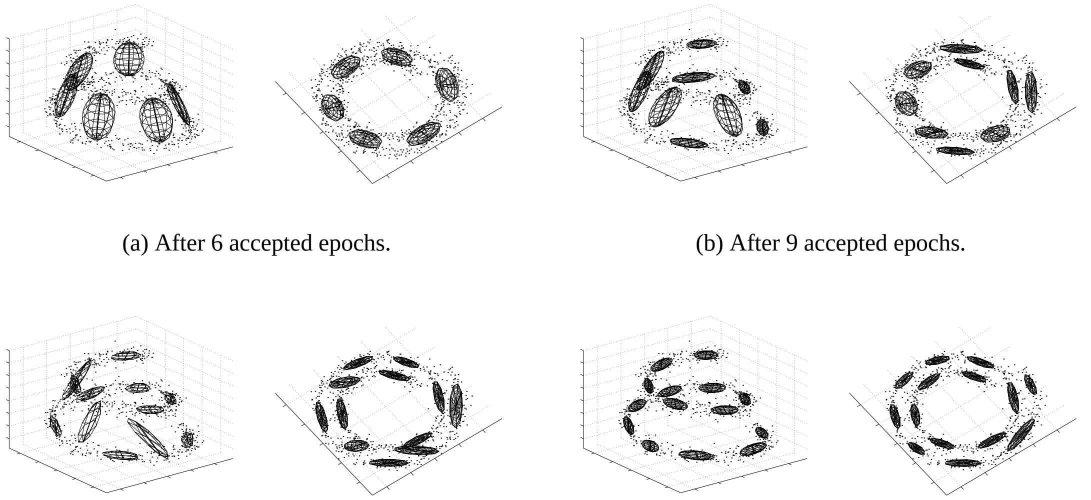
(b) The same data set viewed perpendicular to the third axis.

Figure 4.5: The spiral data set as used in Ueda et al. (2000). Note that the data lie on a 1-dimensional manifold embedded non-linearly in the 3-dimensional data space.

where the parameter t determines the point along the spiral in one dimension. The spiral is shown in figure 4.5, viewed from two angles. Note the spiral data set is really a 1-dimensional manifold embedded non-linearly in the 3-dimensional data space and corrupted by noise.

As before we initialised a variational Bayesian MFA model with a single analyser at the mean of the data, and imposed a maximum dimensionality of $k = 2$ for each analyser. For this experiment, as for the previous synthetic experiments, the spatial splitting heuristic was used. Again local maxima did not pose a problem and the algorithm always found between 12-14 Gaussians. This result was repeatable even when the algorithm was initialised with 200 randomly positioned analysers. The run starting from a single analyser took about 3-4 minutes on a 500MHz Alpha EV6 processor. Figure 4.6 shows the state of the algorithm after 6, 9, 12 and 17 accepted epochs.

Figure 4.7 shows the evolution of the lower bound used to approximate the marginal likelihood of the data. Thick and thin lines in the plot correspond to accepted and rejected epochs, respectively. There are several interesting aspects one should note. First, at the beginning of most of the epochs there is a drop in $\mathcal{F}$ corresponding to a component birth. This is because the model now has to code the parameters of the new analyser component, and initially the model is not fit well to the data. Second, most of the compute time is spent on accepted epochs, suggesting that our heuristics for choosing which components to split, and how to split them, are good. Referring back to figure 4.6, it turns out that it is often components that are straddling arms of the spiral that have low $\mathcal{F}_s$, as given by (4.77), and these are being correctly chosen for splitting ahead of other components modelling their local data better (for example, those aligned on the spiral). Third, after about 1300 iterations, most of the proposed changes to model structure are rejected, and those that are accepted give only a small increase in $\mathcal{F}$.



(a) After 6 accepted epochs.

(b) After 9 accepted epochs.

(c) After 12 accepted epochs.

(d) After 17 accepted epochs.

Figure 4.6: The evolution of the variational Bayesian MFA algorithm over several epochs. Shown are the 1 s.d. covariance ellipses for each analyser: these are the *expected* covariances, since the analysers have distributions over their factor loadings. After 17 accepted epochs the algorithm has converged to a solution with 14 components in the mixture. Local optima, where components are straddled across two arms of the spiral (see **(b)** for example), are successfully avoided by the algorithm.

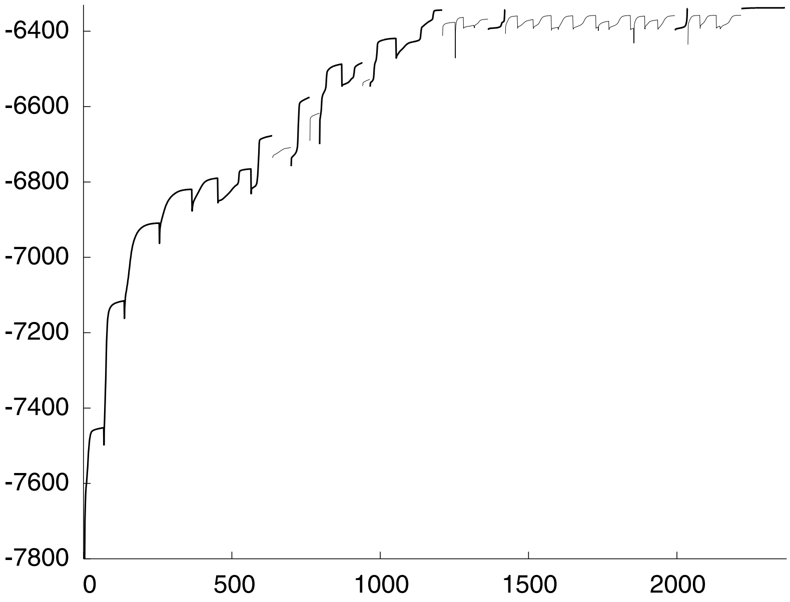


Figure 4.7: Evolution of the lower bound $\mathcal{F}$, as a function of iterations of variational Bayesian EM, for the spiral problem on a typical run. Drops in $\mathcal{F}$ constitute component births. The thick and thin lines represent whole epochs in which a change to model structure was proposed and then eventually accepted or rejected, respectively.

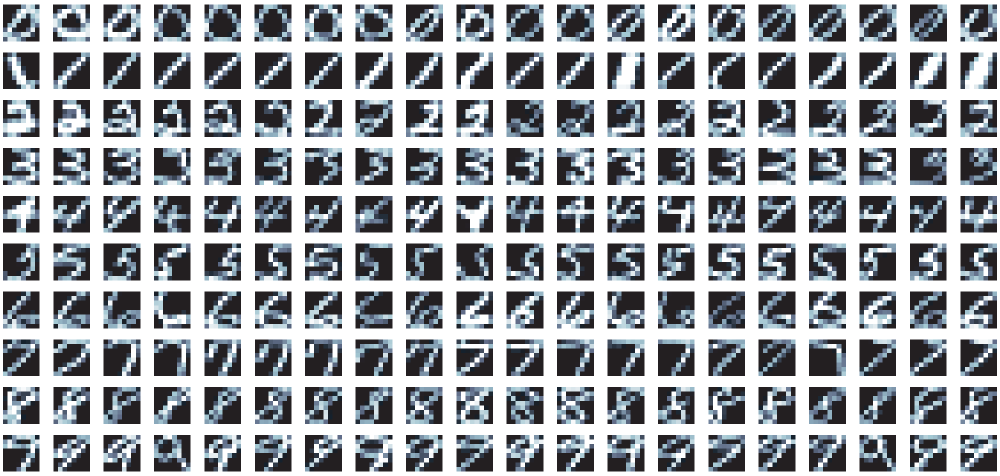


Figure 4.8: Some examples of the digits 0-9 in the training and test data sets. Each digit is 8×8 pixels with gray scale 0 to 255. This data set was normalised before passing to VBMFA for training.

4.6 Digit experiments

In this section we present results of using variational Bayesian MFA to learn both supervised and unsupervised models of images of 8×8 digits taken from the CEDAR database (Hull, 1994). This data set was collected from hand-written digits from postal codes, and are labelled with the classes 0 through to 9. Examples of these digits are given in figure 4.8. The entire data set was normalised before being passed to the VBMFA algorithm, by first subtracting the mean image from every example, and then rescaling each individual pixel to have variance 1 across all the examples. The data set was then partitioned into 700 training and 200 test examples for each digit. Based on density models learnt from the digits, we can build classifiers for a test data set. Histograms of the pixel intensities after this normalisation are quite non-Gaussian, and so factor analysis is perhaps not a good model for this data. Before normalising, we could have considered taking the logarithm or some other non-linear transformation of the intensities to improve the non-Gaussianity, but this was not done.

4.6.1 Fully-unsupervised learning

A *single* VBMFA model was trained on 700 examples of every digit 0-9, using birth proposals and death processes as explained in section 4.3. The maximum dimensionality for each analyser k_{max} was set to 6, and the number of components initialised to be 1. Responsibility-based splits were used for the birth proposals (section 4.3.2) as we would expect these to perform better than spatial-splits given the high dimensionality of the data (using the fraction of accepted splits as a criterion, this was indeed confirmed in preliminary experiments with high dimensional data sets). The choice of when to finish an epoch of learning was based on the rate of change of the

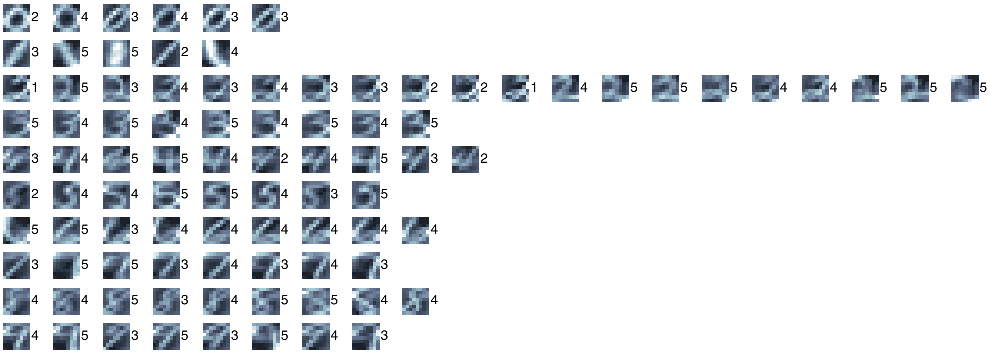


Figure 4.9: A typical model learnt by fully-unsupervised VBMFA using the birth and death processes. Each digit shown represents an analyser in the mixture, and the pixel intensities are the means of the posterior distribution over the centre of the analyser, $\langle \boldsymbol{\mu}^s \rangle_{q(\boldsymbol{\mu}^s)}$. These means can be thought of as *templates*. These intensities have been inversely-processed to show pixel intensities with the same scalings as the training data. The number to the right of each image is that analyser's dimensionality. In this experiment the maximum dimensionality of the latent space was set to $k_{\text{max}} = 6$. As can be seen from these numbers, the highest required dimensionality was 5. The within-row ordering indicates the creation order of the analysers during learning, and we have arranged the templates across different rows according to the 10 different digits in 4.8. This was done by performing a sort of higher-level clustering which the unsupervised algorithm cannot in fact do. Even though the algorithm itself was not given the labels of the data, we as experimenters can examine the posterior responsibilities of each analyser for every item in the training set (whose labels we have access to), and find the majority class for that analyser, and then assign that analyser to the row corresponding to the class label. This is purely a visual aid — in practice if the data is not labelled we have no choice but to call each mixture component in the model a separate class, and have the mean of each analyser as the class template.

component posterior responsibilities (section 4.3.2). The optimisation was terminated when no further changes to model structure managed to increase $\mathcal{F}$ (based on three unsuccessful splits for every component in the model).

Figure 4.9 shows the final model returned from the optimisation. In this figure, each row corresponds to a different digit, and each digit image in the row corresponds to the mean of the posterior over the centre position of each factor analyser component of the mixture. We refer to these as ‘templates’ because they represent the mean of clusters of similar examples of the same digit. The number to the right of each template is the dimensionality of the analyser, determined from examining the posterior over the precisions governing that factor loading matrix's columns $q(\boldsymbol{\nu}^s) = [q(\nu_1^s), \ldots, q(\nu_{k_{\text{max}}}^s)]$.

For some digits the VBMFA needs to use more templates than others. These templates represent distinctively different styles for the same digit. For example, some 1's are written slanting to the left and others to the right, or the digit 2 may or may not contain a loop. These different styles are in very different areas of the high dimensional data space; so each template explains all the

Classified

True	0	1	2	3	4	5	6	7	8	9
0	687	7	.	2	.	1	3	.	.	.
1	.	699	.	.	.	1	.	.	.	.
2	1	7	671	1	2	.	7	4	7	.
3	.	3	7	629	.	27	1	2	30	1
4	.	3	4	.	609	.	3	14	1	66
5	3	7	5	46	.	618	5	.	16	.
6	.	3	.	.	32	1	664	.	.	.
7	.	2	3	1	3	.	.	589	2	100
8	1	14	1	27	2	43	1	4	603	4
9	.	4	1	.	13	.	.	65	3	614

Training data

Classified

True	0	1	2	3	4	5	6	7	8	9
0	196	1	.	.	.	1	2	.	.	.
1	.	200	.	.	.	.	.	.	.	.
2	.	1	186	1	1	1	2	1	6	1
3	.	2	1	181	.	10	.	.	5	1
4	.	1	.	.	175	.	.	1	.	23
5	1	2	.	13	.	180	.	1	2	1
6	.	1	.	.	5	1	193	.	.	.
7	.	2	1	.	.	.	.	176	.	21
8	.	1	.	5	.	9	.	1	179	5
9	.	4	.	.	3	.	.	17	.	176

Test data

Figure 4.10: Confusion tables for digit classification on the training (700) and test (200) sets. The mixture of factor analysers with 92 components obtains 8.8% and 7.9% training and test classification errors respectively.

examples of that style that can be modelled with a linear transformation of the pixel intensities. The number of dimensions of each analyser component for each digit template corresponds very roughly to the number of degrees of freedom there are for that template, and the degree with which each template's factor analyser's linear transformation can extrapolate to the data between the different templates. By using a few linear operations on the pixel intensities of the template image, the analyser can mimic small amounts of shear, rotation, scaling, and translation, and so can capture the main trends in its local data.

When presented with a test example digit from 0-9, we can classify it by asking the model which analyser has the highest posterior responsibility for the test example (i.e. a hard assignment), and then finding which digit class that analyser is clustered into (see discussion above). The result of classifying the training and test data sets are shown in figure 4.10, in confusion matrix form. Each row corresponds to the true class labelling of the digit, and each column corresponds to the digit cluster that the example was assigned to, via the most-responsible analyser in the trained VBMFA model. We see that, for example, about $1/7$ of the training data 8's are misclassified as a variety of classes, and about $1/7$ of the training data 7's are misclassified as 9's (although the converse result is not as poor). These trends are also seen in the classifications of the test data.

The overall classification performance of the model was 91.2% and 92.1% for the training and test sets respectively. This can be compared to simple K-means (using an isotropic distance measure on the identically pre-processed data), with the number of clusters set to the same as inferred in the VBMFA optimisation. The result is that K-means achieves only 87.8% and 86.7% accuracy respectively, despite being initialised with part of the VB solution.

Computation time

The full optimisation for the VBMFA model trained on all 7000 64-dimensional digit examples took approximately 4 CPU days on a Pentium III 500 MHz laptop computer. We would expect the optimisation to take considerably less time if any of the following heuristics were employed. First, one could use partial VBEM updates for $\mathcal{F}$ to update the parameter distributions of only those components that are currently in flux; this corresponds to assuming that changing the modelling configuration of a few analysers in one part of the data space often does not affect the parameter distributions of the overwhelming majority of remaining analysers. In fact, partial updates can be derived that are guaranteed to increase $\mathcal{F}$, simply by placing constraints on the posterior responsibilities of the fixed analysers. Second, the time for each iteration of VBEM can be reduced significantly by removing factors that have been made extinct by the ARD priors; this can even be done prematurely if it increases $\mathcal{F}$. In the implementation used for these experiments all analysers always held factor loading matrix sizes of $(p \times k_{\text{max}})$, despite many of them having far fewer active factors.

4.6.2 Classification performance of BIC and VB models

In these experiments VBMFA was compared to a BIC-penalised maximum likelihood MFA model, in a digit classification task. Each algorithm learnt separate models for each of the digits 0-9, and attempted to classify a data set of test examples based on the predictive densities under each of the learnt digit models. For the VB model, computing the predictive density is intractable (see section 4.4) and so an approximation is required. The experiment was carried out for 7 different training data set sizes ranging from $(100, 200, \ldots 700)$, and repeated 10 times with different parameter initialisations and random subsets of the full 700 images for each digit.

The maximum dimensionality of any analyser component for BIC or VB was set to $k_{\text{max}} = 5$. This corresponds to the maximum dimensionality required by the fully-unsupervised VB model in the previous section's experiments. For the BIC MFA implementation there is no mechanism to prune the factors from the analysers, so all 5 dimensions in each BIC analyser are used all the time.

The same heuristics were used for model search in both types of model, as described in section 4.3. In order to compute a component split ordering, the ML method used the empirical KL divergence to measure the quality of each analyser's fit to its local data (see Ueda et al., 2000, for details). The criterion for ending any particular epoch was again based on the rate of change of component posterior responsibilities. The termination criterion for both algorithms was, as before, three unsuccessful splits of every mixture component in a row. For the ML model, a constraint had to be placed on the Ψ matrix, allowing a minimum variance of 10^{-5} in any direction in the normalised space in which the data has identity covariance. This constraint was

	% correct test classifications	
n	BIC MLMFA	VBMFA
100	88.8 ± .3	89.3 ± .5
200	90.6 ± .4	91.9 ± .3
300	91.1 ± .3	92.7 ± .2
400	91.6 ± .3	92.8 ± .2
500	92.2 ± .3	92.9 ± .2
600	93.0 ± .2	93.3 ± .1
700	93.2 ± .2	93.4 ± .2

Table 4.2: Test classification performance of BIC ML and VB mixture models with increasing data. The standard errors are derived from 10 repetitions of learning with randomly selected training subsets.

introduced to prevent the data likelihood from diverging as a result of the covariance collapsing to zero about any data points.

For the BIC-penalised likelihood, the approximation to the marginal likelihood is given by

$$\ln p(\mathbf{y}) \approx \ln p(\mathbf{y} \mid \boldsymbol{\theta}_{\mathrm{ML}}) - \frac{D}{2} \ln n \tag{4.88}$$

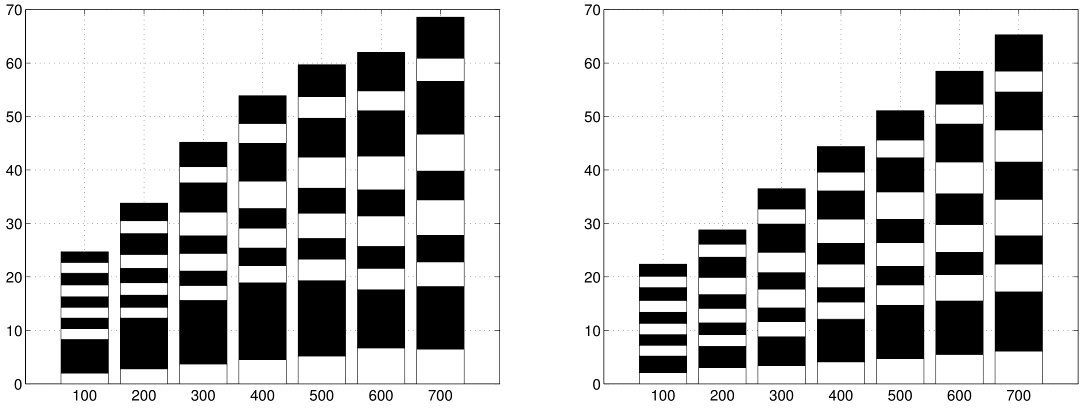
where n is the number of training data (which varied from 100 to 700), and D is the number of degrees of freedom in an MFA model with S analysers with dimensionalities $\{k_s\}_{s=1}^{S}$ (see $d(k)$ of equation (4.5)), which we approximate by

$$D = S - 1 + p + \sum_{s=1}^{S} \left[p + pk_s - \frac{1}{2} k_s(k_s - 1) \right] . \tag{4.89}$$

This quantity is derived from: $S - 1$ degrees of freedom in the prior mixture proportions $\boldsymbol{\pi}$, the number of parameters in the output noise covariance (constrained to be diagonal), p, and the degrees of freedom in the mean and factor loadings of each analyser component. Note that D is only an approximation to the number of degrees of freedom, as discussed in section 4.1.1.

The results of classification experiments for BIC ML and VB are given in table 4.2. VB consistently and significantly outperforms BIC, and in fact surpasses the 92.1% test error performance of the fully-unsupervised VB model on 700 training points. The latter comment is not surprising given that this algorithm receives labelled data. We should note that neither method comes close to state-of-the-art discriminative methods such as support vector machines and convolutional networks, for example *LeNet* (LeCun and Bengio, 1995). This may indicate limitations of the mixture of factor analysers as a generative model for digits.

Figure 4.11 displays the constituents of the mixture models for both BIC and VB for training set sizes $\{100,200,\ldots,700\}$. On average, BIC ML tends to use models with slightly more components than does VB, which does not coincide with the common observation that the BIC



(a) BIC.

(b) VB.

Figure 4.11: The average number of components used for each digit class by the **(a)** BIC and **(b)** VB models, as the size of the training set increases from 100 to 700 examples. As a visual aid, alternate digits are shaded black and white. The white bottom-most block in each column corresponds to the '0' digit and the black top-most block to the '9' digit. Note that BIC consistently returns a greater total number of components than VB (see text).

penalty over-penalises model complexity. Moreover, BIC produces models with a disproportionate number of components for the '1' digit. VB also does this, but not nearly to the same extent. There may be several reasons for these results, listed briefly below.

First, it may be that the criterion used for terminating the epoch is not operating in the same manner in the VB optimisation as in the ML case — if the ML criterion is ending epochs too early this could easily result in the ML model carrying over some of that epoch's un-plateaued optimisation into the next epoch, to artificially improve the penalised likelihood of the next more complicated model. An extreme case of this problem is the epoch-ending criterion that says "end this epoch just as soon as the penalised likelihood reaches what it was before we added the last component". In this case we are performing a purely exploratory search, as opposed to an exploitative search which plateaus before moving on. Second, the ML model may be concentrating analysers on single data points, despite our precision limit on the noise model. Third, there is no mechanism for component death in the ML MFA model, since in these experiments we did not intervene at any stage to test whether the removal of low responsibility components improved the penalised likelihood (see section 4.3.1). It would be interesting to include such tests, for both ML MFA and VB methods.

4.7 Combining VB approximations with Monte Carlo

In this and other chapters, we have assumed that the variational lower bound is a reliable guide to the log marginal likelihood, using it to infer hidden states, to learn distributions over parameters and especially in this chapter to guide a search amongst models of differing complexity. We have not yet addressed the question of how reliable the bounds are. For example, in section 2.3.2 we mentioned that by using $\mathcal{F}$ for model selection we are implicitly assuming that the KL divergences between the variational and exact posterior distributions over parameters and hidden variables are constant between models. It turns out that we can use the technique of importance sampling to obtain consistent estimators of several interesting quantities, including this KL divergence. In this technique the variational posterior can be used as an importance distribution from which to sample points, as it has been optimised to be representative of the exact posterior distribution.

This section builds on basic claims first presented in Ghahramani and Beal (2000). There it was noted that importance sampling can easily fail for poor choices of importance distributions (personal communication with D. MacKay, see also Miskin, 2000, chapter 4). We also present some extensions to simple importance sampling, including using mixture distributions from several runs of VBEM, and also using heavy-tailed distributions derived from the variational posteriors.

4.7.1 Importance sampling with the variational approximation

Section 4.4 furnishes us with an estimate of the predictive density. Unfortunately this does not even constitute a bound on the predictive density, but a bound on an *approximation* to it. However it is possible to approximate the integrals for such quantities by *sampling*. In this subsection we show how by importance sampling from the variational approximation we can obtain estimators of three important quantities: the exact predictive density, the exact log marginal likelihood $\mathcal{L}$, and the KL divergence between the variational posterior and the exact posterior.

The expectation ε of a function $f(\boldsymbol{\theta})$ under the posterior distribution $p(\boldsymbol{\theta} \mid \mathbf{y})$ can be written as

$$\varepsilon = \int d\boldsymbol{\theta} \; p(\boldsymbol{\theta} \mid \mathbf{y}) \, f(\boldsymbol{\theta}) \, . \tag{4.90}$$

Given that such integrals are usually analytically intractable, they can be approximated by the Monte Carlo average:

$$\hat{\varepsilon}(M) \simeq \frac{1}{M} \sum_{m=1}^{M} f(\boldsymbol{\theta}^{(m)}) \, , \qquad \boldsymbol{\theta}^{(m)} \sim p(\boldsymbol{\theta} \mid \mathbf{y}) \, . \tag{4.91}$$

where $\boldsymbol{\theta}^{(m)}$ are random draws from the posterior $p(\boldsymbol{\theta} \,|\, \mathbf{y})$. In the limit of large number of samples M, $\hat{\varepsilon}$ converges to ε:

$$\lim_{M\to\infty} \hat{\varepsilon}(M) = \varepsilon \; . \tag{4.92}$$

In many models it is not possible to sample directly from the posterior, and so a Markov Chain Monte Carlo approach is usually taken to help explore regions of high posterior probability. In most applications this involves designing tailored Metropolis-Hastings acceptance rules for moving about in the space whilst still maintaining detailed balance.

An alternative to finding samples using MCMC methods is to use *importance sampling*. In this method we express the integral as an expectation over an *importance distribution* $g(\boldsymbol{\theta})$:

$$\begin{aligned} \varepsilon &= \int d\boldsymbol{\theta} \; p(\boldsymbol{\theta} \,|\, \mathbf{y}) \, f(\boldsymbol{\theta}) & (4.93) \\ &= \int d\boldsymbol{\theta} \; g(\boldsymbol{\theta}) \, \frac{p(\boldsymbol{\theta} \,|\, \mathbf{y})}{g(\boldsymbol{\theta})} f(\boldsymbol{\theta}) & (4.94) \\ \hat{\varepsilon}(M) &\simeq \frac{1}{M} \sum_{m=1}^{M} \frac{p(\boldsymbol{\theta}^{(m)} \,|\, \mathbf{y})}{g(\boldsymbol{\theta}^{(m)})} f(\boldsymbol{\theta}^{(m)}) \; , \qquad \boldsymbol{\theta}^{(m)} \sim g(\boldsymbol{\theta}) \; , & (4.95) \end{aligned}$$

so that now the Monte Carlo estimate (4.95) is taken using samples drawn from $g(\boldsymbol{\theta})$. Weighting factors are required to account for each sample from $g(\boldsymbol{\theta})$ over- or under-representing the actual density we wish to take the expectation under. These are called the *importance weights*

$$\omega^{(m)} = \frac{1}{M} \frac{p(\boldsymbol{\theta} \,|\, \mathbf{y})}{g(\boldsymbol{\theta})} \; . \tag{4.96}$$

This discretisation of the integral then defines a weighted sum of densities:

$$\hat{\varepsilon}(M) = \sum_{m=1}^{M} \omega^{(m)} f(\boldsymbol{\theta}^{(m)}) \; . \tag{4.97}$$

Again, if $g(\boldsymbol{\theta})$ is non-zero wherever $p(\boldsymbol{\theta} \,|\, \mathbf{y})$ is non-zero, it can be shown that $\hat{\varepsilon}$ converges to ε in the limit of large M.

Having used the VBEM algorithm to find a lower bound on the marginal likelihood, we have at our disposal the resulting variational approximate posterior distribution $q(\boldsymbol{\theta})$. Whilst this distribution is not equal to the posterior, it should be a good candidate for an importance distribution because it contains valuable information about the shape and location of the exact posterior, as it was chosen to minimise the KL divergence between it and the exact posterior (setting aside local optima concerns). In addition it usually has a very simple form and so can be sampled from easily. We now describe several quantities that can be estimated with importance sampling using the variational posterior.

Exact predictive density

An asymptotically exact predictive distribution $p(\mathbf{y}' \,|\, \mathbf{y})$ is that given by a weighted average of the likelihood under a set of parameters drawn from the variational posterior $q(\boldsymbol{\theta})$,

$$\begin{aligned} p(\mathbf{y}' \,|\, \mathbf{y}) &= \int d\boldsymbol{\theta}\; p(\boldsymbol{\theta} \,|\, \mathbf{y})\, p(\mathbf{y}' \,|\, \boldsymbol{\theta}) && (4.98) \\ &= \int d\boldsymbol{\theta}\; q(\boldsymbol{\theta}) \frac{p(\boldsymbol{\theta} \,|\, \mathbf{y})}{q(\boldsymbol{\theta})} p(\mathbf{y}' \,|\, \boldsymbol{\theta}) \quad \Big/ \quad \int d\boldsymbol{\theta}\; q(\boldsymbol{\theta}) \frac{p(\boldsymbol{\theta} \,|\, \mathbf{y})}{q(\boldsymbol{\theta})} && (4.99) \\ &\simeq \frac{1}{M} \sum_{m=1}^{M} \frac{p(\boldsymbol{\theta}^{(m)} \,|\, \mathbf{y})}{q(\boldsymbol{\theta}^{(m)})} p(\mathbf{y}' \,|\, \boldsymbol{\theta}^{(m)}) \quad \Big/ \quad \frac{1}{M} \sum_{o=1}^{M} \frac{p(\boldsymbol{\theta}^{(o)} \,|\, \mathbf{y})}{q(\boldsymbol{\theta}^{(o)})} && (4.100) \\ &= \sum_{m=1}^{M} \omega^{(m)}\, p(\mathbf{y}' \,|\, \boldsymbol{\theta}^{(m)}) \;, && (4.101) \end{aligned}$$

where $\boldsymbol{\theta}^{(m)} \sim q(\boldsymbol{\theta})$ are samples from the variational posterior, and the ω_m are given by

$$\begin{aligned} \omega^{(m)} &= \frac{p(\boldsymbol{\theta}^{(m)} \,|\, \mathbf{y})}{q(\boldsymbol{\theta}^{(m)})} \quad \Big/ \quad \sum_{o=1}^{M} \frac{p(\boldsymbol{\theta}^{(o)} \,|\, \mathbf{y})}{q(\boldsymbol{\theta}^{(o)})} && (4.102) \\ &= \frac{p(\boldsymbol{\theta}^{(m)}, \mathbf{y})}{q(\boldsymbol{\theta}^{(m)})} \quad \Big/ \quad \sum_{o=1}^{M} \frac{p(\boldsymbol{\theta}^{(o)}, \mathbf{y})}{q(\boldsymbol{\theta}^{(o)})} && (4.103) \\ &= \frac{1}{Z_\omega} \frac{p(\boldsymbol{\theta}^{(m)}, \mathbf{y})}{q(\boldsymbol{\theta}^{(m)})} \;, && (4.104) \end{aligned}$$

and Z_ω is defined as

$$Z_\omega = \sum_{m=1}^{M} \frac{p(\boldsymbol{\theta}^{(m)}, \mathbf{y})}{q(\boldsymbol{\theta}^{(m)})} \;. \tag{4.105}$$

In the case of MFAs, each such sample $\boldsymbol{\theta}^{(m)}$ is an instance of a mixture of factor analysers with predictive density $p(\mathbf{y}' \,|\, \boldsymbol{\theta}^{(m)})$ as given by (4.11). Since the $\omega^{(m)}$ are normalised to sum to 1, the predictive density for MFAs given in (4.101) represents a *mixture* of mixture of factor analysers.

Note that the step from (4.102) to (4.103) is important because we cannot evaluate the exact posterior density $p(\boldsymbol{\theta}^{(m)} \,|\, \mathbf{y})$, but we can evaluate the *joint* density $p(\boldsymbol{\theta}^{(m)}, \mathbf{y}) = p(\boldsymbol{\theta}^{(m)})p(\mathbf{y} \,|\, \boldsymbol{\theta}^{(m)})$. Furthermore, note that Z_ω is a function of the weights, and so the estimator in equation (4.101) is really a *ratio* of Monte Carlo estimates. This means that the estimate for $p(\mathbf{y}' \,|\, \mathbf{y})$ is no longer guaranteed to be unbiased. It is however a *consistent* estimator (provided the variances of the numerator and denominator are converging) meaning that as the number of samples tends to infinity its expectation will tend to the exact predictive density.

Exact marginal likelihood

The exact marginal likelihood can be written as

$$\ln p(\mathbf{y}) = \ln \left(\int d\boldsymbol{\theta} \; q(\boldsymbol{\theta}) \frac{p(\boldsymbol{\theta}, \mathbf{y})}{q(\boldsymbol{\theta})} \right) \tag{4.106}$$

$$= \ln \langle \omega \rangle_{q(\boldsymbol{\theta})} + \ln Z_{\omega} \tag{4.107}$$

where $\langle \cdot \rangle$ denotes averaging with respect to the distribution $q(\boldsymbol{\theta})$. This gives us an unbiased estimate of the marginal likelihood, but a biased estimate of the log marginal likelihood. Both estimators are consistent however.

KL divergence

This measure of the quality of the variational approximation can be derived by writing $\mathcal{F}$ in the two ways

$$\mathcal{F} = \int d\boldsymbol{\theta} \; q(\boldsymbol{\theta}) \ln \frac{p(\boldsymbol{\theta}, \mathbf{y})}{q(\boldsymbol{\theta})} \tag{4.108}$$

$$= \langle \ln \omega \rangle_{q(\boldsymbol{\theta})} + \ln Z_{\omega}, \quad \text{or} \tag{4.109}$$

$$\mathcal{F} = \int d\boldsymbol{\theta} \; q(\boldsymbol{\theta}) \ln \frac{p(\boldsymbol{\theta} \,|\, \mathbf{y})}{q(\boldsymbol{\theta})} + \ln p(\mathbf{y}) \tag{4.110}$$

$$= -\mathrm{KL}(q(\boldsymbol{\theta}) \| p(\boldsymbol{\theta} \,|\, \mathbf{y})) + \ln \langle \omega \rangle_{q(\boldsymbol{\theta})} + \ln Z_{\omega}. \tag{4.111}$$

By equating these two expressions we obtain a measure of the divergence between the approximating and exact parameter posteriors,

$$\mathrm{KL}(q(\boldsymbol{\theta}) \| p(\boldsymbol{\theta} \,|\, \mathbf{y})) = \ln \langle \omega \rangle_{q(\boldsymbol{\theta})} - \langle \ln \omega \rangle_{q(\boldsymbol{\theta})} \tag{4.112}$$

Note that this quantity is not a function of Z_{ω}, since it was absorbed into the difference of two logarithms. This means that we need not use normalised weights for this measure, and base the importance weights on $p(\boldsymbol{\theta}, \mathbf{y})$ rather than $p(\boldsymbol{\theta} \,|\, \mathbf{y})$, and the estimator is unbiased.

Three significant observations should be noted. First, the same importance weights can be used to estimate all three quantities. Second, while importance sampling can work very poorly in high dimensions for *ad hoc* proposal distributions, here the variational optimisation is used in a principled manner to provide a $q(\boldsymbol{\theta})$ that is a good approximation to $p(\boldsymbol{\theta} \,|\, \mathbf{y})$, and therefore hopefully a good proposal distribution. Third, this procedure can be applied to any variational approximation.

4.7.2 Example: Tightness of the lower bound for MFAs

In this subsection we use importance sampling to estimate the tightness of the lower bound in a digits learning problem. In the context of a mixture of factor analysers, $\boldsymbol{\theta} = (\boldsymbol{\pi}, \Lambda, \boldsymbol{\mu}) = \{\pi_s, \tilde{\Lambda}^s\}_{s=1}^S$, and we sample $\boldsymbol{\theta}^{(m)} \sim q(\boldsymbol{\theta}) = q(\boldsymbol{\pi})q(\tilde{\Lambda})$. Each such sample is an instance of a mixture of factor analysers with predictive density given by equation (4.11). Note that Ψ is treated as a hyperparameter so need not be sampled (although we could envisage doing so if we were integrating over Ψ). We weight these predictive densities by the importance weights $w^{(m)} = p(\boldsymbol{\theta}^{(m)}, \mathbf{y})/q(\boldsymbol{\theta}^{(m)})$, which are easy to evaluate. When sampling the parameters $\boldsymbol{\theta}$, one needs only to sample $\boldsymbol{\pi}$ vectors and $\tilde{\Lambda}$ matrices, as these are the only parameters that are required to replicate the generative model of mixture of factor analysers (in addition to the hyperparameter Ψ which has no distribution in our model). Thus the numerator in the importance weights are obtained by calculating

$$\begin{aligned} p(\boldsymbol{\theta}, \mathbf{y}) &= p(\boldsymbol{\pi}, \tilde{\Lambda}) p(\mathbf{y} \,|\, \boldsymbol{\pi}, \tilde{\Lambda}) && (4.113) \\ &= p(\boldsymbol{\pi} \,|\, \alpha^*, \mathbf{m}^*) \int d\boldsymbol{\nu}\; p(\tilde{\Lambda} \,|\, \boldsymbol{\nu}, \boldsymbol{\mu}^*, \boldsymbol{\nu}^*) p(\boldsymbol{\nu} \,|\, a^*, b^*) \prod_{i=1}^n p(\mathbf{y}_i \,|\, \boldsymbol{\pi}, \tilde{\Lambda}) && (4.114) \\ &= p(\boldsymbol{\pi} \,|\, \alpha^*, \mathbf{m}^*) p(\tilde{\Lambda} \,|\, a^*, b^*, \boldsymbol{\mu}^*, \boldsymbol{\nu}^*) \prod_{i=1}^n p(\mathbf{y}_i \,|\, \boldsymbol{\pi}, \tilde{\Lambda}) \,. && (4.115) \end{aligned}$$

On the second line we express the prior over the factor loading matrices as a hierarchical prior involving the precisions $\{\boldsymbol{\nu}^s\}_{s=1}^S$. It is not difficult to show that marginalising out the precision for a Gaussian variable yields a multivariate Student-t prior distribution for each row of each $\tilde{\Lambda}^s$, from which we can sample directly. Substituting in the density for an MFA given in (4.11) results in:

$$p(\boldsymbol{\theta}, \mathbf{y}) = p(\boldsymbol{\pi} \,|\, \alpha^*, \mathbf{m}^*) p(\tilde{\Lambda} \,|\, a^*, b^*, \boldsymbol{\mu}^*, \boldsymbol{\nu}^*) \prod_{i=1}^n \left[\sum_{s_i=1}^S p(s_i \,|\, \boldsymbol{\pi}) p(\mathbf{y}_i \,|\, s_i, \tilde{\Lambda}, \Psi) \right] . \qquad (4.116)$$

The importance weights are then obtained after evaluating the density under the variational distribution $q(\boldsymbol{\pi})q(\tilde{\Lambda})$, which is simple to calculate. Even though we require all the training data to generate the importance weights, once these are made, the importance weights $\{\omega^{(m)}\}_{m=1}^M$ and their locations $\{\boldsymbol{\pi}^{(m)}, \tilde{\Lambda}^{(m)}\}_{m=1}^M$ then capture all the information about the posterior distribution that we will need to make predictions, and so we can discard the training data.

A training data set consisting of 700 examples of each of the digits 0, 1, and 2 was used to train a VBMFA model in a fully-unsupervised fashion. After every successful epoch, the variational posterior distributions over the parameters $\tilde{\Lambda}$ and $\boldsymbol{\pi}$ were recorded. These were then used off-line to produce $M = 100$ importance samples from which a set of importance weights $\{\omega^{(m)}\}_{m=1}^M$ were calculated. Using results of the previous section, these weights were used to estimate the following quantities: the log marginal likelihood, the KL divergence between the variational posterior $q(\boldsymbol{\pi})q(\tilde{\Lambda})$ and the exact posterior $p(\boldsymbol{\pi}, \tilde{\Lambda} \,|\, \mathbf{y})$, and the KL divergence between the full variational posterior over all hidden variables and parameters and the exact full posterior. The latter quantity is simply the difference between the estimate of the log marginal likelihood and the lower bound $\mathcal{F}$ used in the optimisation (see equation (4.29)).

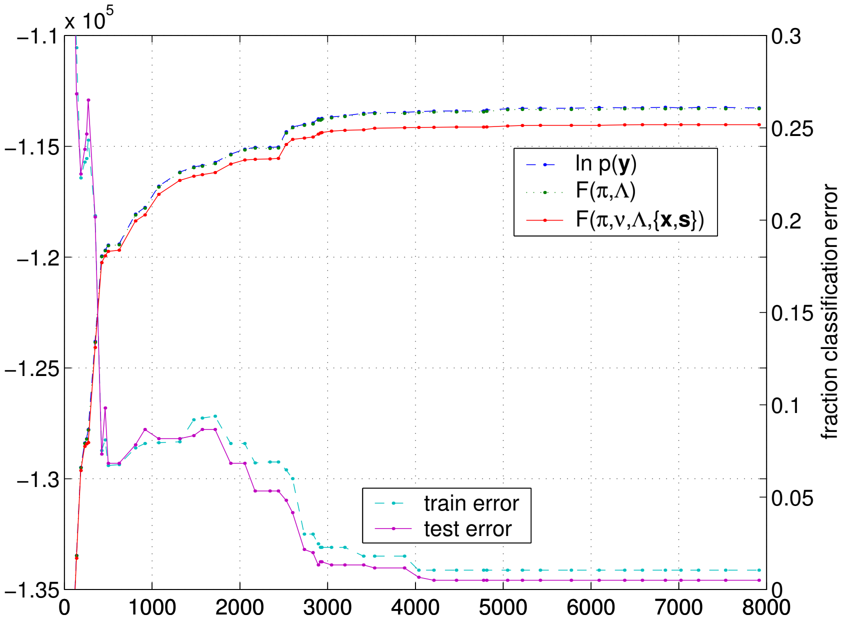
Figure 4.12(a) shows these results plotted alongside the training and test classification errors. We can see that for the most part the lower bound, calculated during the optimisation and denoted $\mathcal{F}(\boldsymbol{\pi}, \boldsymbol{\nu}, \tilde{\Lambda}, \mathbf{x}, \mathbf{s})$ to indicate that it is computed from variational distributions over parameters and hidden variables, is close to the estimate of the log marginal likelihood $\ln p(\mathbf{y})$, and more importantly remains roughly in tandem with it throughout the optimisation. The training and test errors are roughly equal and move together, suggesting that the variational Bayesian model is not overfitting the data. Furthermore, upward changes to the log marginal likelihood are for the most part accompanied by downward changes to the test error rate, suggesting that the marginal likelihood is a good measure for classification performance in this scenario. Lastly, the estimate of the lower bound $\mathcal{F}(\boldsymbol{\pi}, \tilde{\Lambda})$, which is computed by inserting the importance weights into (4.109), is very close to the estimate of the log marginal likelihood (the difference is made more clear in the accompanying figure 4.12(b)). This means that the KL divergence between the variational and exact posteriors over $(\boldsymbol{\pi}, \tilde{\Lambda})$ is fairly small, suggesting that the majority of the gap between $\ln p(\mathbf{y})$ and $\mathcal{F}(\boldsymbol{\pi}, \boldsymbol{\nu}, \tilde{\Lambda}, \mathbf{x}, \mathbf{s})$ is due to the KL divergence between the variational posterior and exact posteriors over the hidden variables $(\boldsymbol{\nu}, \mathbf{x}, \mathbf{s})$.

Aside: efficiency of the structure search

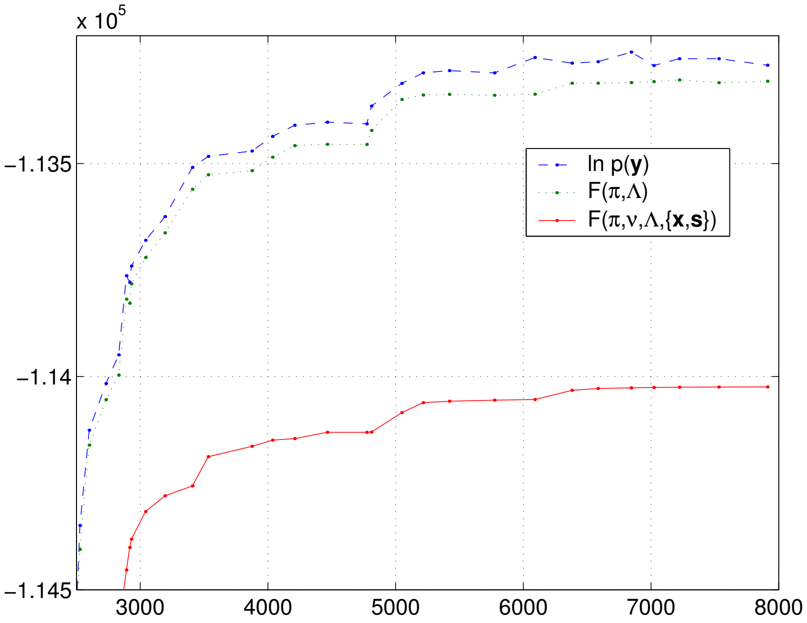
During the optimisation, there were 52 accepted epochs, and a total of 692 proposed component splits (an acceptance rate of only about 7%), resulting in 36 components. However it is clear from the graph (see also figure 4.13(c)) that the model structure does not change appreciably after about 5000 iterations, at which point 41 epochs have been accepted from 286 proposals. This corresponds to an acceptance rate of 14% which suggests that our heuristics for choosing which component to split and how to split it are performing well, given the number of components to chose from and the dimensionality of the data space.

Analysis of the lower bound gap

Given that 100 samples may be too few to obtain reliable estimates, the experiment was repeated with 6 runs of importance sampling, each with 100 samples as before. Figures 4.13(a) and 4.13(b) show the KL divergence measuring the distance between the log marginal likelihood estimate and the lower bounds $\mathcal{F}(\boldsymbol{\pi}, \boldsymbol{\nu}, \tilde{\Lambda}, \mathbf{x}, \mathbf{s})$ and $\mathcal{F}(\boldsymbol{\pi}, \tilde{\Lambda})$, respectively, as the optimisation proceeds. Figure 4.13(c) plots the number of components, S, in the mixture with iterations of EM, and it is quite clear that the KL divergences in the previous two graphs correlate closely



(a)



(b)

Figure 4.12: **(a)** Log marginal likelihood estimates from importance sampling with iterations of VBEM. Each point corresponds to the model at the end of a successful epoch of learning. The fraction of training and test classification errors are shown on the right vertical axis, and the lower bound $\mathcal{F}(\boldsymbol{\pi}, \boldsymbol{\nu}, \tilde{\Lambda}, \mathbf{x}, \mathbf{s})$ that guides the optimisation on the left vertical axis. Also plotted is $\mathcal{F}(\boldsymbol{\pi}, \tilde{\Lambda})$, but this is indistinguishable from the other lower bound. The second plot **(b)** is exactly the same as **(a)** except the log marginal likelihood axis has been rescaled to make clear the difference between the log marginal likelihood and the bound $\mathcal{F}(\boldsymbol{\pi}, \tilde{\Lambda})$.

with the number of components. This observation is borne out explicitly in figures 4.13(d) and 4.13(d) where it is clear that the KL divergence between the lower bound $\mathcal{F}(\boldsymbol{\pi}, \boldsymbol{\nu}, \tilde{\Lambda}, \mathbf{x}, \mathbf{s})$ and the marginal likelihood is roughly proportional to the number of components in the mixture. This is true to an extent also for the lower bound estimate $\mathcal{F}(\boldsymbol{\pi}, \tilde{\Lambda})$ although this quantity is more noisy. These two observations are unlikely to be artifacts of the sampling process, as the variances are much smaller than the trend. In section 2.3.2 we noted that if the KL discrepancy increases with S then the model exploration may be biased to simpler models. Here we have found some evidence of this, which suggests that variational Bayesian methods may suffer from a tendency to underfit the model structure.

4.7.3 Extending simple importance sampling

Why importance sampling is dangerous

Unfortunately, the importance sampling procedure that we have used is notoriously bad in high dimensions. Moreover, it is easy to show that importance sampling can fail even for just one dimension: consider computing expectations under a one dimensional Gaussian $p(\theta)$ with precision ν_p using an importance distribution $q(\theta)$ which is also a Gaussian with precision ν_q and the same mean. Although importance sampling can give us unbiased estimates, it is simple to show that if $\nu_q > 2\nu_p$ then the variance of the importance weights will be infinite! We briefly derive this result here. The importance weight for the sample drawn from $q(\theta)$ is given by

$$\omega(\theta) = \frac{p(\theta)}{q(\theta)} \, , \tag{4.117}$$

and the variance of the importance weights can be written

$$\begin{aligned}
var(\omega) &= \langle \omega^2 \rangle_{q(\theta)} - \langle \omega \rangle^2_{q(\theta)} && (4.118)\\
&= \int d\theta \; q(\theta) \left(\frac{p(\theta)}{q(\theta)} \right)^2 - \left(\int d\theta \; q(\theta) \frac{p(\theta)}{q(\theta)} \right)^2 && (4.119)\\
&= \frac{\nu_p}{\nu_q^{1/2}} \int d\theta \; \exp \left[- \left(\nu_p - \frac{1}{2}\nu_q \right) \theta^2 + k\theta + k' \right] \quad - 1 \, , && (4.120)\\
&= \begin{cases} \nu_p \nu_q^{-1/2} \left(2\nu_p - \nu_q\right)^{-1/2} - 1 & \text{for } \; 2\nu_p > \nu_q \\ \infty & \text{for } \; 2\nu_p \leq \nu_q \end{cases} \, . && (4.121)
\end{aligned}$$

where k and k' are constants independent of x. For $2\nu_p \leq \nu_q$, the integral diverges and the variance of the weights is infinite. Indeed this problem is exacerbated in higher dimensions, where if this condition is not met in any dimension of parameter space, then the importance weights will have infinite variance. The intuition behind this is that we need the tails of the sampling distribution $q(\theta)$ to fall off slower than the true distribution $p(\theta)$, otherwise there exists some probability

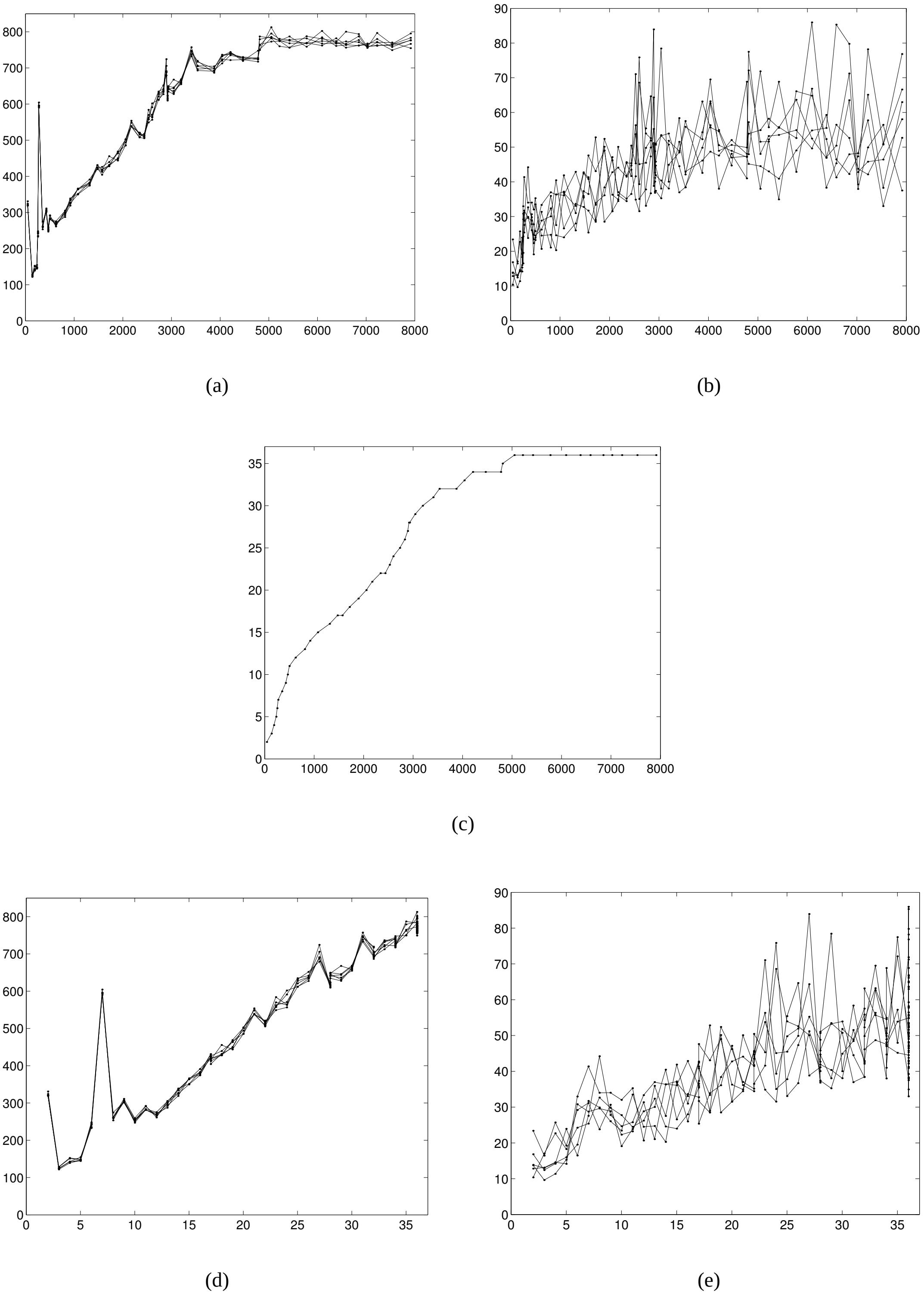


Figure 4.13: At the end of every accepted epoch 6 estimates of the log marginal likelihood were calculated (see text). **(a)** Differences between the log marginal likelihood estimate and the lower bound $\mathcal{F}(\boldsymbol{\pi}, \boldsymbol{\nu}, \tilde{\Lambda}, \mathbf{x}, \mathbf{s})$, as a function of iterations of VBEM. **(b)** Differences between the log marginal likelihood estimate and the lower bound $\mathcal{F}(\boldsymbol{\pi}, \tilde{\Lambda})$. **(c)** Number of components S in the mixture model with iterations of VBEM. **(d)** The same data as in (a), plotted against the number of components S, as given in (c). **(e)** As for (d) but using the data from (b) instead of (a).

that we obtain a very high importance weight. This result is clearly a setback for importance sampling using the variational posterior distribution, since the variational posterior tends to be tighter than the exact posterior, having neglected correlations between some parameters in order to make inference tractable. To complete the argument, we should mention that importance sampling becomes very difficult in high dimensions even if this condition is met, since: firstly, samples from the typical set of the $q(\theta)$ are unlikely to have high probability under $p(\theta)$, unless the distributions are very similar; secondly, even if the distributions are well matched, the weights have a wide range that scales order $\exp(r^{1/2})$, where r is the dimensionality (MacKay, 1999).

The above result (4.121) is extended in Miskin (2000, chapter 4), where the finite variance condition is derived for general $p(\theta)$ and $q(\theta)$ in the exponential family. Also in that work, a bound is derived for the variance of the importance weights when using a finite mixture distribution as the importance distribution (equation 4.31 of that manuscript). This mixture is made from the variational posterior distribution mixed with a set of broader distributions from the *same* exponential family. The rationale for this approach is precisely to create heavier-tailed importance distributions. Unfortunately the bound is not very tight, and the simulations therein report no increase in convergence to the correct expectation.

In addition to these problems, the exact posterior over the parameters can be very multi-modal. The most benign form of such multi-modality is due to aliases arising from having likelihood functions which are invariant to exchanges of labelling of hidden variables, for example indicator variables for components in a mixture. In such cases the variational posterior tends to lock on to one mode and so, when used in an importance sampler, the estimate represents only a fraction of the marginal likelihood. If the modes are well-separated then simple degeneracies of this sort can be accounted for by multiplying the result by the number of aliases. If the modes are overlapping, then a correction should not be needed as we expect the importance distribution to be broad enough. However if the modes are only partially separated then the correction factor is difficult to compute. In general, these corrections cannot be made precise and should be avoided.

Using heavy-tailed and mixture distributions

Here we investigate the effect of two modifications to the naive use of the variational posterior as importance distribution. The first modification considers replacing the variational posterior entirely by a related heavy-tailed Student-t distribution. The second modification uses a stochastic *mixture* distribution for the importance distribution, with each component being the variational posterior obtained from a different VBEM optimisation.

The Student-t can be derived by considering the marginal probability of Gaussian distributed variables under a conjugate gamma distribution for the precision, γ, which is for the univariate case:

$$\begin{aligned} q_{\mathrm{St}}(\theta) &= \int d\gamma \; p(\gamma \,|\, a, b) p(\theta \,|\, \mu, \gamma^{-1}) && (4.122) \\ &= \int d\gamma \; \mathrm{Ga}(\gamma \,|\, a, b) \mathrm{N}(\theta \,|\, \mu, \gamma^{-1}) && (4.123) \\ &= \frac{b^a}{\Gamma(a)\sqrt{2\pi}} \int d\gamma \; e^{-(b+(\theta-\mu)^2/2)\gamma} \gamma^{a-1/2} && (4.124) \\ &= \frac{1}{Z_{\mathrm{St}}(a,b)} \left(1 + \frac{(\theta-\mu)^2}{2b}\right)^{-(a+1/2)} && (4.125) \end{aligned}$$

where a and b are the shape and inverse-scale respectively of the precision distribution, and Z_{St} is given by

$$Z_{\mathrm{St}}(a,b) = \frac{\Gamma(a + \frac{1}{2})}{\Gamma(a)\sqrt{2\pi b}} \,, \qquad \text{for } \; a > 0, \; b > 0 \,. \qquad (4.126)$$

It is straightforward to show that the variance of θ is given by $b/(a-1)$ and the kurtosis by $3(a-1)/(a-2)$ (see appendix A). The degrees of freedom ν and dispersion parameter σ^2 can be arrived at with the following equivalence:

$$\nu = 2a \,, \qquad \sigma^2 = \frac{b}{a} \,. \qquad (4.127)$$

The attraction of using this distribution for sampling is that it has heavier tails, with a polynomial rather than exponential decay. In the limit of $\nu \to \infty$ the Student-t is a Gaussian distribution, while for $\nu = 1$ it is a Cauchy distribution.

Three 2-dimensional data sets were generated by drawing 150 samples from 4 Gaussian clusters, with varying separations of their centres, as shown in figure 4.14. For each data set, 10 randomly initialised VBEM algorithms were run to learn a model of the data. If any of the learnt models contained fewer or more than 4 components, that optimisation was discarded and replaced with another. We would expect that for the well-separated data set the exact posterior distribution over the parameters would consist of tight, well-separated modes. Conversely, for the overlapping data set we would expect the posterior to be very broad consisting of several weakly-defined peaks. In the intermediately-spaced data set we would expect the posterior to be mostly separated modes with some overlap.

The following importance samplers were constructed, separately for each data set, and are summarised in table 4.3: (1) a single model out of the 10 that were trained was randomly chosen (once) and its variational posterior $q(\boldsymbol{\pi})q(\tilde{\Lambda})$ used as the importance distribution; (2) the covariance parameters of the variational posterior $q(\tilde{\Lambda})$ of that same model were used as the covariance parameters in t-distributions with 3 degrees of freedom to form $q^{(3)}(\tilde{\Lambda})$, and this used in conjunction with the same $q(\boldsymbol{\pi})$ to form the importance distribution $q(\boldsymbol{\pi})q^{(3)}(\tilde{\Lambda})$; (3) the same as

sampler	type of	each component's			
key	importance dist.	form	dof	relative variance	kurtosis
1	single 1	Gaussian	∞	1	3
2	single 1	Student-t	3	3	3.75
3	single 1	Student-t	2	∞	4.5
4	mixture of 10	Gaussian	∞	ditto	ditto
5	mixture of 10	Student-t	3		
6	mixture of 10	Student-t	2		

Table 4.3: The specifications of six importance sampling distributions.

(2) but using 2 degrees of freedom; samplers (4,5,6) are the same as (1,2,3) except the operations are carried out on every one of the 10 models returned, to generate a mixture model with 10 equally weighted mixture components.

Recall that the covariance matrix for the entries of the $\tilde{\Lambda}^s$ matrix for each analyser is of block diagonal form, and so each row can be sampled from independently to produce the importance samples. Furthermore, generating the multivariate Student-t samples from these covariances is a straightforward procedure using standard methods.

Figure 4.14 shows the results of attempting to estimate the marginal likelihood of the three different data sets, using the 6 differently constructed importance samplers given in table 4.3, which are denoted by the labels 1–6. The axis marks F and F' correspond to lower bounds on the log marginal likelihood: F is the lower bound reported by the single model used for the single set of importance samplers (i.e. 1,2,3); and F' is the highest reported lower bound of all 10 of the models trained on that data set. The error bars correspond to the unbiased estimate of the standard deviation in the estimates from five separate runs of importance sampling. We can see several interesting features. First, all the estimates (1-6) using different importance distributions yield estimates greater than the highest lower bound (F'). Second, the use of heavier-tailed and broader Student-t distributions for the most part increases the estimate, whether based on single or mixture importance distributions. Also, the move from 3 to 2 degrees of freedom (i.e. (2) to (3), or (5) to (6) in the plot) for the most part increases the estimate further. These observations suggest that there exists mass outside of the variational posterior that is neglected with the Gaussian implementations (1,4). Third, using mixture distributions increases the estimates. However, this increase from (1,2,3) to (4,5,6) is roughly the same as the increase in lower bounds from F to F'. This implies that the single estimates are affected if using a sub-optimal solution, whereas the mixture distribution can perform approximately as well as its best constituent solution. It should be noted that only the highest lower bound, F', was plotted for each data set, as plotting the remaining 9 lower bounds would have extended the graphs' y-axes too much to be able to visually resolve the differences in the methods (in all three data sets there were at least two poor optimisations).

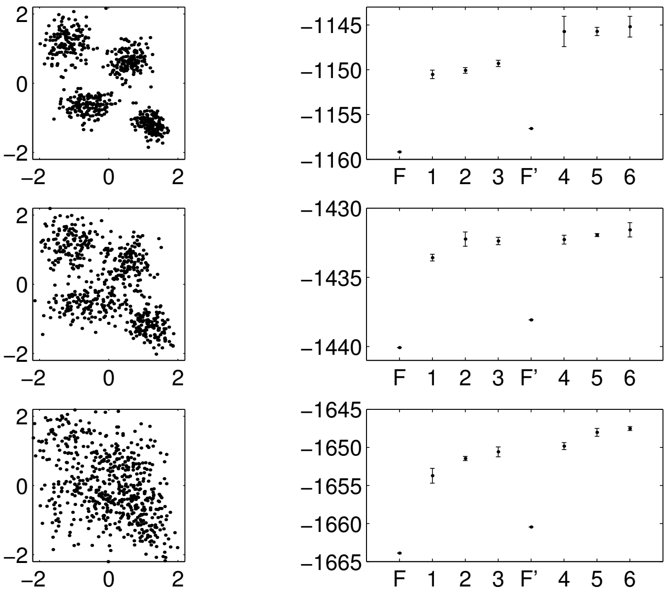


Figure 4.14: **(right)** Importance sampling estimates of the marginal likelihoods of VBMFA models trained on **(left)** three data sets of differently spaced Gaussian clusters. In the plots in the right column, the vertical axis is the log of the marginal likelihood estimate, and the horizontal axis denotes which importance sampling method is used for the estimate, as given in table 4.3. The estimates are taken from five separate runs of importance sampling, with each run consisting of 4000 samples; the error bars are the standard errors in the estimate, assuming the logarithm of the estimates from the five runs are Gaussian distributed. The axis mark F corresponds to the lower bound from the model used for the single samplers (1,2,3), and the mark F' corresponds to the highest lower bound from the 10 models used in the mixture samplers (4,5,6).

4.8 Summary

In this chapter we have shown that how the marginal likelihood of a mixture of factor analysers is intractable, and derived a tractable deterministic variational lower bound which can be optimised using a variational EM algorithm. We can use the lower bound to guide a search among model structures using birth and death moves. We can also use the lower bound to obtain a distribution over structures if desired: $p(m \mid \mathbf{y}) \propto p(m)p(\mathbf{y} \mid m) \approx p(m) \cdot e^{\mathcal{F}_{opt}(m)}$, with the caveat that there is no guarantee that the best achieved lower bound, $\mathcal{F}_{opt}(m)$, is similarly tight across different models m. Indeed we have found that the KL divergence between the variational and exact posterior over parameters increases approximately linearly with the number of components in the mixture, which suggests a systematic tendency to underfit (refer to page 60).

We have derived a generally applicable importance sampler based on the variational solution, which gives us consistent estimates of the exact marginal likelihood, the exact predictive density, and the KL divergence between the variational posterior and the exact posterior. We have also investigated the use of heavy-tailed and mixture distributions for improving the importance sampler estimates, but there are theoretical reasons for why methods more sophisticated than importance sampling are required for reliable estimates.

It is also possible to integrate the variational optimisation into the proposal distribution for an MCMC sampling method (NIPS workshop: *Advanced Mean Field Methods*, Denver CO, December 1999; personal communication with N. de Freitas, July 2000). The combined procedures combine the relative advantages of the two methods, namely the asymptotic correctness of sampling, and the rapid and deterministic convergence of variational methods. Since the variational optimisation can quickly provide us with an approximation to the shape of the local posterior landscape, the MCMC transition kernel can be adapted to utilise this information to more accurately explore and update that approximation. One would hope that this refined knowledge could then be used to update the variational posterior, and the process iterated. Unfortunately, in its simplest form, this MCMC *adaption* can not be done infinitely often, as it disrupts the stationary distribution of the chain (although see Gilks et al., 1998, for a *regeneration* technique). In de Freitas et al. (2001), a variational MCMC method that includes mixture transition kernels is described and applied to the task of finding the moments of posterior distributions in a sigmoid belief network. There remain plenty of directions of research for such combinations of variational and MCMC methods.

The VB mixtures formalism has been applied to more complicated variants of MFA models recently, with a view to determining the number of components and the local manifold dimensionalities. For example, mixtures of independent components analysers (Choudrey and Roberts, 2002), and mixtures of independent components analysers with non-symmetric sources (Chan et al., 2002).

There have been other Bayesian approaches to modelling densities using mixture distributions. One notable example is the infinite Gaussian mixture model of Rasmussen (2000), which uses sampling to entertain a countably infinite number of mixture components, rather than any particular finite number. In that work, when training on the Spiral data set (examined in section 4.5.3 of this thesis), it was found that on average about 18–20 of the infinitely many Gaussian components had data associated with them. Our VB method usually found between 12–14 analyser components. Examining the differences between the models returned, and perhaps more importantly the predictions made, by these two algorithms is an interesting direction of research.

Search over model structures for MFAs is computationally intractable if each factor analyser is allowed to have different intrinsic dimensionalities. In this chapter we have shown how the variational Bayesian approach can be used to efficiently infer the structure of the model whilst avoiding overfitting and other deficiencies of ML approaches. We have also shown how we can simultaneously infer both the number of analysers and their dimensionalities using birth-death steps and ARD methods, all based on a variational lower bound on the marginal likelihood.