21 *Variational inference*

21.1 Introduction

We have now seen several algorithms for computing (functions of) a posterior distribution. For discrete graphical models, we can use the junction tree algorithm to perform exact inference, as explained in Section 20.4. However, this takes time exponential in the treewidth of the graph, rendering exact inference often impractical. For the case of Gaussian graphical models, exact inference is cubic in the treewidth. However, even this can be too slow if we have many variables. In addition, the JTA does not work for continuous random variables outside of the Gaussian case, nor for mixed discrete-continuous variables, outside of the conditionally Gaussian case.

For some simple two node graphical models, of the form $\mathbf{x} \rightarrow \mathcal{D}$, we can compute the exact posterior $p(\mathbf{x}|\mathcal{D})$ in closed form, provided the prior $p(\mathbf{x})$ is conjugate to the likelihood, $p(\mathcal{D}|\mathbf{x})$ (which means the likelihood must be in the exponential family). See Chapter 5 for some examples of this. (Note that in this chapter, $\mathbf{x}$ represent the unknown variables, whereas in Chapter 5, we used $\boldsymbol{\theta}$ to represent the unknowns.)

In more general settings, we must use approximate inference methods. In Section 8.4.1, we discussed the Gaussian approximation, which is useful for inference in two node models of the form $\mathbf{x} \rightarrow \mathcal{D}$, where the prior is not conjugate. (For example, Section 8.4.3 applied the method to logistic regression.)

The Gaussian approximation is simple. However, some posteriors are not naturally modelled using Gaussians. For example, when inferring multinomial parameters, a Dirichlet distribution is a better choice, and when inferring states in a discrete graphical model, a categorical distribution is a better choice.

In this chapter, we will study a more general class of deterministic approximate inference algorithms based on **variational inference** (Jordan et al. 1998; Jaakkola and Jordan 2000; Jaakkola 2001; Wainwright and Jordan 2008a). The basic idea is to pick an approximation $q(\mathbf{x})$ to the distribution from some tractable family, and then to try to make this approximation as close as possible to the true posterior, $p^*(\mathbf{x}) \triangleq p(\mathbf{x}|\mathcal{D})$. This reduces inference to an optimization problem. By relaxing the constraints and/or approximating the objective, we can trade accuracy for speed. The bottom line is that variational inference often gives us the speed benefits of MAP estimation but the statistical benefits of the Bayesian approach.

21.2 Variational inference

Suppose $p^*(\mathbf{x})$ is our true but intractable distribution and $q(\mathbf{x})$ is some approximation, chosen from some tractable family, such as a multivariate Gaussian or a factored distribution. We assume q has some free parameters which we want to optimize so as to make q "similar to" p^*.

An obvious cost function to try to minimize is the KL divergence:

$$\mathbb{KL}\left(p^*||q\right) = \sum_{\mathbf{x}} p^*(\mathbf{x}) \log \frac{p^*(\mathbf{x})}{q(\mathbf{x})} \tag{21.1}$$

However, this is hard to compute, since taking expectations wrt p^* is assumed to be intractable. A natural alternative is the reverse KL divergence:

$$\mathbb{KL}\left(q||p^*\right) = \sum_{\mathbf{x}} q(\mathbf{x}) \log \frac{q(\mathbf{x})}{p^*(\mathbf{x})} \tag{21.2}$$

The main advantage of this objective is that computing expectations wrt q is tractable (by choosing a suitable form for q). We discuss the statistical differences between these two objectives in Section 21.2.2.

Unfortunately, Equation 21.2 is still not tractable as written, since even evaluating $p^*(\mathbf{x}) = p(\mathbf{x}|\mathcal{D})$ pointwise is hard, since it requires evaluating the intractable normalization constant $Z = p(\mathcal{D})$. However, usually the unnormalized distribution $\tilde{p}(\mathbf{x}) \triangleq p(\mathbf{x}, \mathcal{D}) = p^*(\mathbf{x})Z$ is tractable to compute. We therefore define our new objective function as follows:

$$J(q) \quad \triangleq \quad \mathbb{KL}\left(q||\tilde{p}\right) \tag{21.3}$$

where we are slightly abusing notation, since $\tilde{p}$ is not a normalized distribution. Plugging in the definition of KL, we get

$$\begin{aligned} J(q) &= \sum_{\mathbf{x}} q(\mathbf{x}) \log \frac{q(\mathbf{x})}{\tilde{p}(\mathbf{x})} && (21.4)\\ &= \sum_{\mathbf{x}} q(\mathbf{x}) \log \frac{q(\mathbf{x})}{Zp^*(\mathbf{x})} && (21.5)\\ &= \sum_{\mathbf{x}} q(\mathbf{x}) \log \frac{q(\mathbf{x})}{p^*(\mathbf{x})} - \log Z && (21.6)\\ &= \mathbb{KL}\left(q||p^*\right) - \log Z && (21.7) \end{aligned}$$

Since Z is a constant, by minimizing $J(q)$, we will force q to become close to p^*.

Since KL divergence is always non-negative, we see that $J(q)$ is an upper bound on the NLL (negative log likelihood):

$$J(q) = \mathbb{KL}\left(q||p^*\right) - \log Z \geq -\log Z = -\log p(\mathcal{D}) \tag{21.8}$$

Alternatively, we can try to *maximize* the following quantity (in (Koller and Friedman 2009), this is referred to as the **energy functional**), which is a lower bound on the log likelihood of the data:

$$L(q) \triangleq -J(q) = -\mathbb{KL}\left(q||p^*\right) + \log Z \leq \log Z = \log p(\mathcal{D}) \tag{21.9}$$

Since this bound is tight when $q = p^*$, we see that variational inference is closely related to EM (see Section 11.4.7).

21.2.1 Alternative interpretations of the variational objective

There are several equivalent ways of writing this objective that provide different insights. One formulation is as follows:

$$J(q) = \mathbb{E}_q\left[\log q(\mathbf{x})\right] + \mathbb{E}_q\left[-\log \tilde{p}(\mathbf{x})\right] = -\mathbb{H}\left(q\right) + \mathbb{E}_q\left[E(\mathbf{x})\right] \tag{21.10}$$

which is the expected energy (recall $E(\mathbf{x}) = -\log \tilde{p}(\mathbf{x})$) minus the entropy of the system. In statistical physics, $J(q)$ is called the **variational free energy** or the **Helmholtz free energy**.[1]

Another formulation of the objective is as follows:

$$\begin{aligned} J(q) &= \mathbb{E}_q\left[\log q(\mathbf{x}) - \log p(\mathbf{x})p(\mathcal{D}|\mathbf{x})\right] & (21.11)\\ &= \mathbb{E}_q\left[\log q(\mathbf{x}) - \log p(\mathbf{x}) - \log p(\mathcal{D}|\mathbf{x})\right] & (21.12)\\ &= \mathbb{E}_q\left[-\log p(\mathcal{D}|\mathbf{x})\right] + \mathbb{KL}\left(q(\mathbf{x})||p(\mathbf{x})\right) & (21.13) \end{aligned}$$

This is the expected NLL, plus a penalty term that measures how far the approximate posterior is from the exact prior.

We can also interpret the variational objective from the point of view of information theory (the so-called bits-back argument). See (Hinton and Camp 1993; Honkela and Valpola 2004), for details.

21.2.2 Forward or reverse KL? *

Since the KL divergence is not symmetric in its arguments, minimizing $\mathbb{KL}\left(q||p\right)$ wrt q will give different behavior than minimizing $\mathbb{KL}\left(p||q\right)$. Below we discuss these two different methods.

First, consider the reverse KL, $\mathbb{KL}\left(q||p\right)$, also known as an **I-projection** or **information projection**. By definition, we have

$$\mathbb{KL}\left(q||p\right) = \sum_{\mathbf{x}} q(\mathbf{x}) \ln \frac{q(\mathbf{x})}{p(\mathbf{x})} \tag{21.14}$$

This is infinite if $p(\mathbf{x}) = 0$ and $q(\mathbf{x}) > 0$. Thus if $p(\mathbf{x}) = 0$ we must ensure $q(\mathbf{x}) = 0$. We say that the reverse KL is **zero forcing** for q. Hence q will typically under-estimate the support of p.

Now consider the forwards KL, also known as an **M-projection** or **moment projection**:

$$\mathbb{KL}\left(p||q\right) = \sum_{\mathbf{x}} p(\mathbf{x}) \ln \frac{p(\mathbf{x})}{q(\mathbf{x})} \tag{21.15}$$

This is infinite if $q(\mathbf{x}) = 0$ and $p(\mathbf{x}) > 0$. So if $p(\mathbf{x}) > 0$ we must ensure $q(\mathbf{x}) > 0$. We say that the forwards KL is **zero avoiding** for q. Hence q will typically over-estimate the support of p.

The difference between these methods is illustrated in Figure 21.1. We see that when the true distribution is multimodal, using the forwards KL is a bad idea (assuming q is constrained to be unimodal), since the resulting posterior mode/mean will be in a region of low density, right between the two peaks. In such contexts, the reverse KL is not only more tractable to compute, but also more sensible statistically.

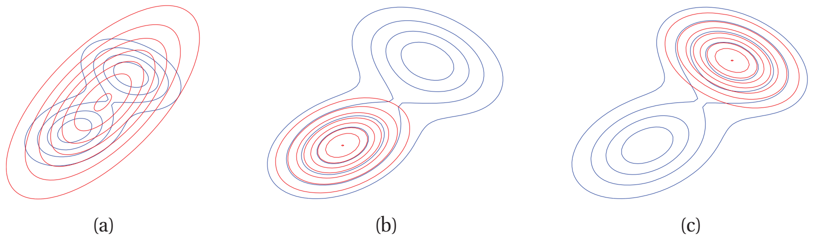


Figure 21.1 Illustrating forwards vs reverse KL on a bimodal distribution. The blue curves are the contours of the true distribution p. The red curves are the contours of the unimodal approximation q. (a) Minimizing forwards KL: q tends to "cover" p. (b-c) Minimizing reverse KL: q locks on to one of the two modes. Based on Figure 10.3 of (Bishop 2006b). Figure generated by `KLfwdReverseMixGauss`.

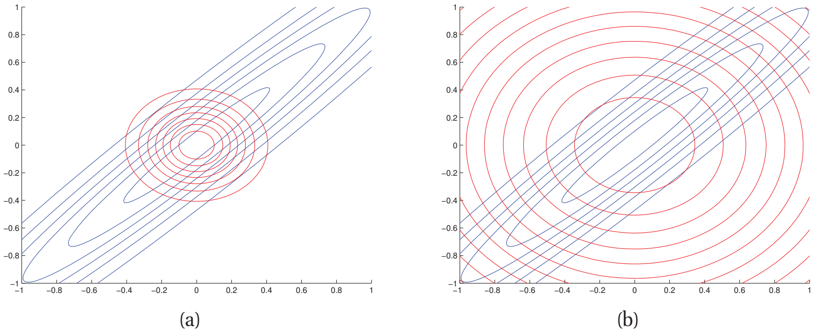


Figure 21.2 Illustrating forwards vs reverse KL on a symmetric Gaussian. The blue curves are the contours of the true distribution p. The red curves are the contours of a factorized approximation q. (a) Minimizing $\mathbb{KL}(q||p)$. (b) Minimizing $\mathbb{KL}(p||q)$. Based on Figure 10.2 of (Bishop 2006b). Figure generated by `KLpqGauss`.

Another example of the difference is shown in Figure 21.2, where the target distribution is an elongated 2d Gaussian and the approximating distribution is a product of two 1d Gaussians. That is, $p(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Lambda}^{-1})$, where

$$\boldsymbol{\mu} = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \quad \boldsymbol{\Lambda} = \begin{pmatrix} \Lambda_{11} & \Lambda_{12} \\ \Lambda_{21} & \Lambda_{22} \end{pmatrix} \tag{21.16}$$

In Figure 21.2(a) we show the result of minimizing $\mathbb{KL}(q||p)$. In this simple example, one can show that the solution has the form

$$\begin{aligned} q(\mathbf{x}) &= \mathcal{N}(x_1|m_1, \Lambda_{11}^{-1})\mathcal{N}(x_2|m_2, \Lambda_{22}^{-1}) && (21.17) \\ m_1 &= \mu_1 - \Lambda_{11}^{-1}\Lambda_{12}(m_2 - \mu_2) && (21.18) \\ m_2 &= \mu_2 - \Lambda_{22}^{-1}\Lambda_{21}(m_1 - \mu_1) && (21.19) \end{aligned}$$

[1] It is called "free" because the variables $\mathbf{x}$ are free to vary, rather than being fixed. The variational free energy is a function of the distribution q, whereas the regular energy is a function of the state vector $\mathbf{x}$.

Figure 21.2(a) shows that we have correctly captured the mean, but the approximation is too compact: its variance is controlled by the direction of smallest variance of p. In fact, it is often the case (although not always (Turner et al. 2008)) that minimizing $\mathbb{KL}(q||p)$, where q is factorized, results in an approximation that is overconfident.

In Figure 21.2(b), we show the result of minimizing $\mathbb{KL}(p||q)$. As we show in Exercise 21.7, the optimal solution when minimizing the forward KL wrt a factored approximation is to set q to be the product of marginals. Thus the solution has the form

$$q(\mathbf{x}) = \mathcal{N}(x_1|\mu_1, \Lambda_{11}^{-1})\mathcal{N}(x_2|\mu_2, \Lambda_{22}^{-1}) \tag{21.20}$$

Figure 21.2(b) shows that this is too broad, since it is an over-estimate of the support of p.

For the rest of this chapter, and for most of the next, we will focus on minimizing $\mathbb{KL}(q||p)$. In Section 22.5, when we discuss expectation proagation, we will discuss ways to locally optimize $\mathbb{KL}(p||q)$.

One can create a family of divergence measures indexed by a parameter $\alpha \in \mathbb{R}$ by defining the **alpha divergence** as follows:

$$D_\alpha(p||q) \triangleq \frac{4}{1-\alpha^2}\left(1 - \int p(x)^{(1+\alpha)/2} q(x)^{(1-\alpha)/2} dx\right) \tag{21.21}$$

This measure satisfies $D_\alpha(p||q) = 0$ iff $p = q$, but is obviously not symmetric, and hence is not a metric. $\mathbb{KL}(p||q)$ corresponds to the limit $\alpha \to 1$, whereas $\mathbb{KL}(q||p)$ corresponds to the limit $\alpha \to -1$. When $\alpha = 0$, we get a symmetric divergence measure that is linearly related to the **Hellinger distance**, defined by

$$D_H(p||q) \triangleq \int \left(p(x)^{\frac{1}{2}} - q(x)^{\frac{1}{2}}\right)^2 dx \tag{21.22}$$

Note that $\sqrt{D_H(p||q)}$ is a valid distance metric, that is, it is symmetric, non-negative and satisfies the triangle inequality. See (Minka 2005) for details.

21.3 The mean field method

One of the most popular forms of variational inference is called the **mean field** approximation (Opper and Saad 2001). In this approach, we assume the posterior is a fully factorized approximation of the form

$$q(\mathbf{x}) = \prod_i q_i(\mathbf{x}_i) \tag{21.23}$$

Our goal is to solve this optimization problem:

$$\min_{q_1,\ldots,q_D} \mathbb{KL}(q||p) \tag{21.24}$$

where we optimize over the parameters of each marginal distribution q_i. In Section 21.3.1, we derive a coordinate descent method, where at each step we make the following update:

$$\log q_j(\mathbf{x}_j) = \mathbb{E}_{-q_j}[\log \tilde{p}(\mathbf{x})] + \text{const} \tag{21.25}$$

Model	Section
Ising model	Section 21.3.2
Factorial HMM	Section 21.4.1
Univariate Gaussian	Section 21.5.1
Linear regression	Section 21.5.2
Logistic regression	Section 21.8.1.1
Mixtures of Gaussians	Section 21.6.1
Latent Dirichlet allocation	Section 27.3.6.3

Table 21.1 Some models in this book for which we provide detailed derivations of the mean field inference algorithm.

where $\tilde{p}(\mathbf{x}) = p(\mathbf{x}, \mathcal{D})$ is the unnormalized posterior and the notation $\mathbb{E}_{-q_j}[f(\mathbf{x})]$ means to take the expectation over $f(\mathbf{x})$ with respect to all the variables except for x_j. For example, if we have three variables, then

$$\mathbb{E}_{-q_2}[f(\mathbf{x})] = \sum_{x_1}\sum_{x_3} q(x_1)q_3(x_3)f(x_1, x_2, x_3) \tag{21.26}$$

where sums get replaced by integrals where necessary.

When updating q_j, we only need to reason about the variables which share a factor with x_j, i.e., the terms in j's Markov blanket (see Section 10.5.3); the other terms get absorbed into the constant term. Since we are replacing the neighboring values by their mean value, the method is known as mean field. This is very similar to Gibbs sampling (Section 24.2), except instead of sending sampled values between neighboring nodes, we send mean values between nodes. This tends to be more efficient, since the mean can be used as a proxy for a large number of samples. (On the other hand, mean field messages are dense, whereas samples are sparse; this can make sampling more scalable to very large models.)

Of course, updating one distribution at a time can be slow, since it is a form of coordinate descent. Several methods have been proposed to speed up this basic approach, including using pattern search (Honkela et al. 2003), and techniques based on parameter expansion (Qi and Jaakkola 2008). However, we will not consider these methods in this chapter.

It is important to note that the mean field method can be used to infer discrete or continuous latent quantities, using a variety of parametric forms for q_i, as we will see below. This is in contrast to some of the other variational methods we will encounter later, which are more restricted in their applicability. Table 21.1 lists some of the examples of mean field that we cover in this book.

21.3.1 Derivation of the mean field update equations

Recall that the goal of variational inference is to minimize the upper bound $J(q) \geq -\log p(\mathcal{D})$. Equivalently, we can try to maximize the lower bound

$$L(q) \triangleq -J(q) = \sum_{\mathbf{x}} q(\mathbf{x}) \log \frac{\tilde{p}(\mathbf{x})}{q(\mathbf{x})} \leq \log p(\mathcal{D}) \tag{21.27}$$

We will do this one term at a time.

If we write the objective singling out the terms that involve q_j, and regarding all the other terms as constants, we get

$$\begin{aligned}
L(q_j) &= \sum_{\mathbf{x}} \prod_i q_i(\mathbf{x}_i) \left[\log \tilde{p}(\mathbf{x}) - \sum_k \log q_k(\mathbf{x}_k)\right] && (21.28) \\
&= \sum_{\mathbf{x}_j} \sum_{\mathbf{x}_{-j}} q_j(\mathbf{x}_j) \prod_{i \neq j} q_i(\mathbf{x}_i) \left[\log \tilde{p}(\mathbf{x}) - \sum_k \log q_k(\mathbf{x}_k)\right] && (21.29) \\
&= \sum_{\mathbf{x}_j} q_j(\mathbf{x}_j) \sum_{\mathbf{x}_{-j}} \prod_{i \neq j} q_i(\mathbf{x}_i) \log \tilde{p}(\mathbf{x}) \\
&\quad - \sum_{\mathbf{x}_j} q_j(\mathbf{x}_j) \sum_{\mathbf{x}_{-j}} \prod_{i \neq j} q_i(\mathbf{x}_i) \left[\sum_{k \neq j} \log q_k(\mathbf{x}_k) + q_j(\mathbf{x}_j)\right] && (21.30) \\
&= \sum_{\mathbf{x}_j} q_j(\mathbf{x}_j) \log f_j(\mathbf{x}_j) - \sum_{\mathbf{x}_j} q_j(\mathbf{x}_j) \log q_j(\mathbf{x}_j) + \text{const} && (21.31)
\end{aligned}$$

where

$$\log f_j(\mathbf{x}_j) \triangleq \sum_{\mathbf{x}_{-j}} \prod_{i \neq j} q_i(\mathbf{x}_i) \log \tilde{p}(\mathbf{x}) = \mathbb{E}_{-q_j}\left[\log \tilde{p}(\mathbf{x})\right] \tag{21.32}$$

So we average out all the hidden variables except for $\mathbf{x}_j$. Thus we can rewrite $L(q_j)$ as follows:

$$L(q_j) = -\mathbb{KL}\left(q_j || f_j\right) \tag{21.33}$$

We can maximize L by minimizing this KL, which we can do by setting $q_j = f_j$, as follows:

$$q_j(\mathbf{x}_j) = \frac{1}{Z_j} \exp\left(\mathbb{E}_{-q_j}\left[\log \tilde{p}(\mathbf{x})\right]\right) \tag{21.34}$$

We can usually ignore the local normalization constant Z_j, since we know q_j must be a normalized distribution. Hence we usually work with the form

$$\log q_j(\mathbf{x}_j) = \mathbb{E}_{-q_j}\left[\log \tilde{p}(\mathbf{x})\right] + \text{const} \tag{21.35}$$

The functional form of the q_j distributions will be determined by the type of variables $\mathbf{x}_j$, as well as the form of the model. (This is sometimes called **free-form optimization**.) If x_j is a discrete random variable, then q_j will be a discrete distribution; if $\mathbf{x}_j$ is a continuous random variable, then q_j will be some kind of pdf. We will see examples of this below.

21.3.2 Example: mean field for the Ising model

Consider the image denoising example from Section 19.4.1, where $x_i \in \{-1, +1\}$ are the hidden pixel values of the "clean" image. We have a joint model of the form

$$p(\mathbf{x}, \mathbf{y}) = p(\mathbf{x}) p(\mathbf{y}|\mathbf{x}) \tag{21.36}$$

where the prior has the form

$$p(\mathbf{x}) = \frac{1}{Z_0}\exp(-E_0(\mathbf{x})) \tag{21.37}$$

$$E_0(\mathbf{x}) = -\sum_{i=1}^{D}\sum_{j\in\text{nbr}_i} W_{ij}x_ix_j \tag{21.38}$$

and the likelihood has the form

$$p(\mathbf{y}|\mathbf{x}) = \prod_i p(\mathbf{y}_i|x_i) = \sum_i \exp(-L_i(x_i)) \tag{21.39}$$

Therefore the posterior has the form

$$p(\mathbf{x}|\mathbf{y}) = \frac{1}{Z}\exp(-E(\mathbf{x})) \tag{21.40}$$

$$E(\mathbf{x}) = E_0(\mathbf{x}) - \sum_i L_i(x_i) \tag{21.41}$$

We will now approximate this by a fully factored approximation

$$q(\mathbf{x}) = \prod_i q(x_i, \mu_i) \tag{21.42}$$

where μ_i is the mean value of node i. To derive the update for the variational parameter μ_i, we first write out $\log\tilde{p}(\mathbf{x}) = -E(\mathbf{x})$, dropping terms that do not involve x_i:

$$\log\tilde{p}(\mathbf{x}) = x_i\sum_{j\in\text{nbr}_i} W_{ij}x_j + L_i(x_i) + \text{const} \tag{21.43}$$

This only depends on the states of the neighboring nodes. Now we take expectations of this wrt $\prod_{j\neq i} q_j(x_j)$ to get

$$q_i(x_i) \propto \exp\left(x_i\sum_{j\in\text{nbr}_i} W_{ij}\mu_j + L_i(x_i)\right) \tag{21.44}$$

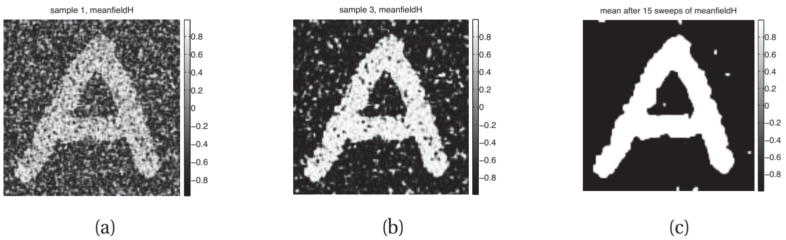
Thus we replace the states of the neighbors by their average values. Let

$$m_i = \sum_{j\in\text{nbr}_i} W_{ij}\mu_j \tag{21.45}$$

be the mean field influence on node i. Also, let $L_i^+ \triangleq L_i(+1)$ and $L_i^- \triangleq L_i(-1)$. The approximate marginal posterior is given by

$$q_i(x_i = 1) = \frac{e^{m_i+L_i^+}}{e^{m_i+L_i^+} + e^{-m_i+L_i^-}} = \frac{1}{1+e^{-2m_i+L_i^- - L_i^+}} = \text{sigm}(2a_i) \tag{21.46}$$

$$a_i \triangleq m_i + 0.5(L_i^+ - L_i^-) \tag{21.47}$$



(a) (b) (c)

Figure 21.3 Example of image denoising using mean field (with parallel updates and a damping factor of 0.5). We use an Ising prior with $W_{ij} = 1$ and a Gaussian noise model with $\sigma = 2$. We show the results after 1, 3 and 15 iterations across the image. Compare to Figure 24.1. Figure generated by `isingImageDenoiseDemo`.

Similarly, we have $q_i(x_i = -1) = \text{sigm}(-2a_i)$. From this we can compute the new mean for site i:

$$\mu_i = \mathbb{E}_{q_i}[x_i] = q_i(x_i = +1) \cdot (+1) + q_i(x_i = -1) \cdot (-1) \tag{21.48}$$

$$= \frac{1}{1+e^{-2a_i}} - \frac{1}{1+e^{2a_i}} = \frac{e^{a_i}}{e^{a_i}+e^{-a_i}} - \frac{e^{-a_i}}{e^{-a_i}+e^{a_i}} = \tanh(a_i) \tag{21.49}$$

Hence the update equation becomes

$$\mu_i = \tanh\left(\sum_{j \in \text{nbr}_i} W_{ij}\mu_j + 0.5(L_i^+ - L_i^-)\right) \tag{21.50}$$

See also Exercise 21.6 for an alternative derivation of these equations.

We can turn the above equations in to a fixed point algorithm by writing

$$\mu_i^t = \tanh\left(\sum_{j \in \text{nbr}_i} W_{ij}\mu_j^{t-1} + 0.5(L_i^+ - L_i^-)\right) \tag{21.51}$$

It is usually better to use **damped updates** of the form

$$\mu_i^t = (1-\lambda)\mu_i^{t-1} + \lambda \tanh\left(\sum_{j \in \text{nbr}_i} W_{ij}\mu_j^{t-1} + 0.5(L_i^+ - L_i^-)\right) \tag{21.52}$$

for $0 < \lambda < 1$. We can update all the nodes in parallel, or update them asychronously.

Figure 21.3 shows the method in action, applied to a 2d Ising model with homogeneous attractive potentials, $W_{ij} = 1$. We use parallel updates with a damping factor of $\lambda = 0.5$. (If we don't use damping, we tend to get "checkerboard" artefacts.)

21.4 Structured mean field *

Assuming that all the variables are independent in the posterior is a very strong assumption that can lead to poor results. Sometimes we can exploit **tractable substructure** in our problem, so

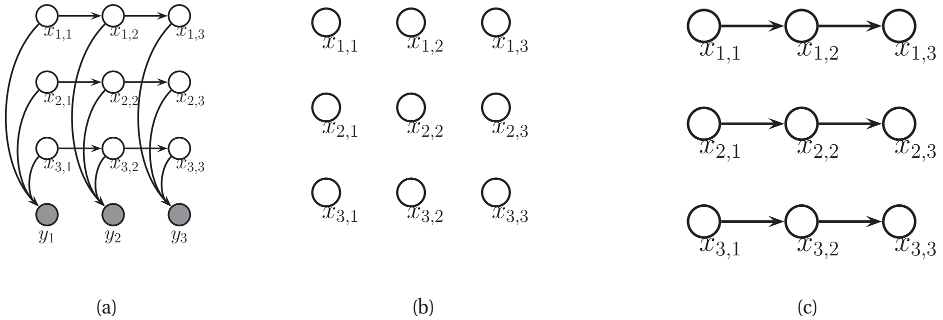


Figure 21.4 (a) A factorial HMM with 3 chains. (b) A fully factorized approximation. (c) A product-of-chains approximation. Based on Figure 2 of (Ghahramani and Jordan 1997).

that we can efficiently handle some kinds of dependencies. This is called the **structured mean field** approach (Saul and Jordan 1995). The approach is the same as before, except we group sets of variables together, and we update them simultaneously. (This follows by simply treating all the variables in the i'th group as a single "mega-variable", and then repeating the derivation in Section 21.3.1.) As long as we can perform efficient inference in each q_i, the method is tractable overall. We give an example below. See (Bouchard-Cote and Jordan 2009) for some more recent work in this area.

21.4.1 Example: factorial HMM

Consider the factorial HMM model (Ghahramani and Jordan 1997) introduced in Section 17.6.5. Suppose there are M chains, each of length T, and suppose each hidden node has K states. The model is defined as follows

$$p(\mathbf{x}, \mathbf{y}) = \prod_m \prod_t p(x_{tm}|x_{t-1,m}) p(\mathbf{y}_t|x_{tm}) \tag{21.53}$$

where $p(x_{tm} = k|x_{t-1,m} = j) = A_{mjk}$ is an entry in the transition matrix for chain m, $p(x_{1m} = k|x_{0m}) = p(x_{1m} = k) = \pi_{mk}$, is the initial state distribution for chain m, and

$$p(\mathbf{y}_t|\mathbf{x}_t) = \mathcal{N}\left(\mathbf{y}_t| \sum_{m=1}^{M} \mathbf{W}_m \mathbf{x}_{tm}, \mathbf{\Sigma}\right) \tag{21.54}$$

is the observation model, where $\mathbf{x}_{tm}$ is a 1-of-K encoding of x_{tm} and $\mathbf{W}_m$ is a $D \times K$ matrix (assuming $\mathbf{y}_t \in \mathbb{R}^D$). Figure 21.4(a) illustrates the model for the case where $M = 3$. Even though each chain is a priori independent, they become coupled in the posterior due to having an observed common child, $\mathbf{y}_t$. The junction tree algorithm applied to this graph takes $O(TMK^{M+1})$ time. Below we will derive a structured mean field algorithm that takes $O(TMK^2I)$ time, where I is the number of mean field iterations (typically $I \sim 10$ suffices for good performance).

We can write the exact posterior in the following form:

$$p(\mathbf{x}|\mathbf{y}) = \frac{1}{Z}\exp(-E(\mathbf{x},\mathbf{y})) \tag{21.55}$$

$$\begin{aligned} E(\mathbf{x},\mathbf{y}) &= \frac{1}{2}\sum_{t=1}^{T}\left(\mathbf{y}_t - \sum_m \mathbf{W}_m\mathbf{x}_{tm}\right)^T \mathbf{\Sigma}^{-1}\left(\mathbf{y}_t - \sum_m \mathbf{W}_m\mathbf{x}_{tm}\right) \\ &\quad -\sum_m \mathbf{x}_{1m}^T\tilde{\boldsymbol{\pi}}_m - \sum_{t=2}^{T}\sum_m \mathbf{x}_{tm}^T\tilde{\mathbf{A}}_m\mathbf{x}_{t-1,m} \end{aligned} \tag{21.56}$$

where $\tilde{\mathbf{A}}_m \triangleq \log \mathbf{A}_m$ and $\tilde{\boldsymbol{\pi}}_m \triangleq \log \boldsymbol{\pi}_m$ (both interpreted elementwise).

We can approximate the posterior as a product of marginals, as in Figure 21.4(b), but a better approximation is to use a product of chains, as in Figure 21.4(c). Each chain can be tractably updated individually, using the forwards-backwards algorithm. More precisely, we assume

$$q(\mathbf{x}|\mathbf{y}) = \frac{1}{Z_q}\prod_{m=1}^{M} q(x_{1m}|\boldsymbol{\xi}_{1m})\prod_{t=2}^{T} q(x_{tm}|x_{t-1,m},\boldsymbol{\xi}_{tm}) \tag{21.57}$$

$$q(x_{1m}|\boldsymbol{\xi}_{1m}) = \prod_{k=1}^{K}(\xi_{1mk}\pi_{mk})^{x_{1mk}} \tag{21.58}$$

$$q(x_{tm}|x_{t-1,m},\boldsymbol{\xi}_{tm}) = \prod_{k=1}^{K}\left(\xi_{tmk}\prod_{j=1}^{K}(A_{mjk})^{x_{t-1,m,j}}\right)^{x_{tmk}} \tag{21.59}$$

We see that the ξ_{tmk} parameters play the role of an approximate local evidence, averaging out the effects of the other chains. This is contrast to the exact local evidence, which couples all the chains together.

We can rewrite the approximate posterior as $q(\mathbf{x}) = \frac{1}{Z_q}\exp(-E_q(\mathbf{x}))$, where

$$E_q(\mathbf{x}) = -\sum_{t=1}^{T}\sum_{m=1}^{M}\mathbf{x}_{tm}^T\tilde{\boldsymbol{\xi}}_{tm} - \sum_{m=1}^{M}\mathbf{x}_{1m}^T\tilde{\boldsymbol{\pi}}_m - \sum_{t=2}^{T}\sum_{m=1}^{M}\mathbf{x}_{tm}^T\tilde{\mathbf{A}}_m\mathbf{x}_{t-1,m} \tag{21.60}$$

where $\tilde{\boldsymbol{\xi}}_{tm} = \log\boldsymbol{\xi}_{tm}$. We see that this has the same temporal factors as the exact posterior, but the local evidence term is different. The objective function is given by

$$\mathbb{KL}\left(q||p\right) = \mathbb{E}\left[E\right] - \mathbb{E}\left[E_q\right] - \log Z_q + \log Z \tag{21.61}$$

where the expectations are taken wrt q. One can show (Exercise 21.8) that the update has the form

$$\boldsymbol{\xi}_{tm} = \exp\left(\mathbf{W}_m^T\mathbf{\Sigma}^{-1}\tilde{\mathbf{y}}_{tm} - \frac{1}{2}\boldsymbol{\delta}_m\right) \tag{21.62}$$

$$\boldsymbol{\delta}_m \triangleq \text{diag}(\mathbf{W}_m^T\mathbf{\Sigma}^{-1}\mathbf{W}_m) \tag{21.63}$$

$$\tilde{\mathbf{y}}_{tm} \triangleq \mathbf{y}_t - \sum_{\ell\neq m}^{M}\mathbf{W}_\ell\mathbb{E}\left[\mathbf{x}_{t,\ell}\right] \tag{21.64}$$

The $\boldsymbol{\xi}_{tm}$ parameter plays the role of the local evidence, averaging over the neighboring chains. Having computed this for each chain, we can perform forwards-backwards in parallel, using these approximate local evidence terms to compute $q(\mathbf{x}_{t,m}|\mathbf{y}_{1:T})$ for each m and t.

The update cost is $O(TMK^2)$ for a full "sweep" over all the variational parameters, since we have to run forwards-backwards M times, for each chain independently. This is the same cost as a fully factorized approximation, but is much more accurate.

21.5 Variational Bayes

So far we have been concentrating on inferring latent variables $\mathbf{z}_i$ assuming the parameters $\boldsymbol{\theta}$ of the model are known. Now suppose we want to infer the parameters themselves. If we make a fully factorized (i.e., mean field) approximation, $p(\boldsymbol{\theta}|\mathcal{D}) \approx \prod_k q(\boldsymbol{\theta}_k)$, we get a method known as **variational Bayes** or **VB** (Hinton and Camp 1993; MacKay 1995a; Attias 2000; Beal and Ghahramani 2006; Smidl and Quinn 2005).[2] We give some examples of VB below, assuming that there are no latent variables. If we want to infer both latent variables and parameters, and we make an approximation of the form $p(\boldsymbol{\theta}, \mathbf{z}_{1:N}|\mathcal{D}) \approx q(\boldsymbol{\theta}) \prod_i q_i(\mathbf{z}_i)$, we get a method known as variational Bayes EM, which we described in Section 21.6.

21.5.1 Example: VB for a univariate Gaussian

Following (MacKay 2003, p429), let us consider how to apply VB to infer the posterior over the parameters for a 1d Gaussian, $p(\mu, \lambda|\mathcal{D})$, where $\lambda = 1/\sigma^2$ is the precision. For convenience, we will use a conjugate prior of the form

$$p(\mu, \lambda) = \mathcal{N}(\mu|\mu_0, (\kappa_0\lambda)^{-1})\text{Ga}(\lambda|a_0, b_0) \tag{21.65}$$

However, we will use an approximate factored posterior of the form

$$q(\mu, \lambda) = q_\mu(\mu)q_\lambda(\lambda) \tag{21.66}$$

We do not need to specify the forms for the distributions q_μ and q_λ; the optimal forms will "fall out" automatically during the derivation (and conveniently, they turn out to be Gaussian and Gamma respectively).

You might wonder why we would want to do this, since we know how to compute the exact posterior for this model (Section 4.6.3.7). There are two reasons. First, it is a useful pedagogical exercise, since we can compare the quality of our approximation to the exact posterior. Second, it is simple to modify the method to handle a semi-conjugate prior of the form $p(\mu, \lambda) = \mathcal{N}(\mu|\mu_0, \tau_0)\text{Ga}(\lambda|a_0, b_0)$, for which exact inference is no longer possible.

[2] This method was originally called **ensemble learning** (MacKay 1995a), since we are using an ensemble of parameters (a distribution) instead of a point estimate. However, the term "ensemble learning" is also used to describe methods such as boosting, so we prefer the term VB.

21.5.1.1 Target distribution

The unnormalized log posterior has the form

$$
\begin{aligned}
\log \tilde{p}(\mu, \lambda) &= \log p(\mu, \lambda, \mathcal{D}) = \log p(\mathcal{D}|\mu, \lambda) + \log p(\mu|\lambda) + \log p(\lambda) & (21.67)\\
&= \frac{N}{2} \log \lambda - \frac{\lambda}{2} \sum_{i=1}^{N} (x_i - \mu)^2 - \frac{\kappa_0 \lambda}{2} (\mu - \mu_0)^2 \\
&\quad + \frac{1}{2} \log(\kappa_0 \lambda) + (a_0 - 1) \log \lambda - b_0 \lambda + \text{const} & (21.68)
\end{aligned}
$$

21.5.1.2 Updating $q_\mu(\mu)$

The optimal form for $q_\mu(\mu)$ is obtained by averaging over λ:

$$
\begin{aligned}
\log q_\mu(\mu) &= \mathbb{E}_{q_\lambda} [\log p(\mathcal{D}|\mu, \lambda) + \log p(\mu|\lambda)] + \text{const} & (21.69)\\
&= -\frac{\mathbb{E}_{q_\lambda}[\lambda]}{2} \left\{ \kappa_0 (\mu - \mu_0)^2 + \sum_{i=1}^{N} (x_i - \mu)^2 \right\} + \text{const} & (21.70)
\end{aligned}
$$

By completing the square one can show that $q_\mu(\mu) = \mathcal{N}(\mu|\mu_N, \kappa_N^{-1})$, where

$$
\mu_N = \frac{\kappa_0 \mu_0 + N\overline{x}}{\kappa_0 + N}, \ \kappa_N = (\kappa_0 + N)\mathbb{E}_{q_\lambda}[\lambda] \tag{21.71}
$$

At this stage we don't know what $q_\lambda(\lambda)$ is, and hence we cannot compute $\mathbb{E}[\lambda]$, but we will derive this below.

21.5.1.3 Updating $q_\lambda(\lambda)$

The optimal form for $q_\lambda(\lambda)$ is given by

$$
\begin{aligned}
\log q_\lambda(\lambda) &= \mathbb{E}_{q_\mu} [\log p(\mathcal{D}|\mu, \lambda) + \log p(\mu|\lambda) + \log p(\lambda)] + \text{const} & (21.72)\\
&= (a_0 - 1) \log \lambda - b_0 \lambda + \frac{1}{2} \log \lambda + \frac{N}{2} \log \lambda \\
&\quad - \frac{\lambda}{2} \mathbb{E}_{q_\mu} \left[\kappa_0 (\mu - \mu_0)^2 + \sum_{i=1}^{N} (x_i - \mu)^2 \right] + \text{const} & (21.73)
\end{aligned}
$$

We recognize this as the log of a Gamma distribution, hence $q_\lambda(\lambda) = \text{Ga}(\lambda|a_N, b_N)$, where

$$
\begin{aligned}
a_N &= a_0 + \frac{N+1}{2} & (21.74)\\
b_N &= b_0 + \frac{1}{2} \mathbb{E}_{q_\mu} \left[\kappa_0 (\mu - \mu_0)^2 + \sum_{i=1}^{N} (x_i - \mu)^2 \right] & (21.75)
\end{aligned}
$$

21.5.1.4 Computing the expectations

To implement the updates, we have to specify how to compute the various expectations. Since $q(\mu) = \mathcal{N}(\mu|\mu_N, \kappa_N^{-1})$, we have

$$\mathbb{E}_{q(\mu)}\left[\mu\right] = \mu_N \tag{21.76}$$

$$\mathbb{E}_{q(\mu)}\left[\mu^2\right] = \frac{1}{\kappa_N} + \mu_N^2 \tag{21.77}$$

Since $q(\lambda) = \text{Ga}(\lambda|a_N, b_N)$, we have

$$\mathbb{E}_{q(\lambda)}\left[\lambda\right] = \frac{a_N}{b_N} \tag{21.78}$$

We can now give explicit forms for the update equations. For $q(\mu)$ we have

$$\mu_N = \frac{\kappa_0\mu_0 + N\overline{x}}{\kappa_0 + N} \tag{21.79}$$

$$\kappa_N = (\kappa_0 + N)\frac{a_N}{b_N} \tag{21.80}$$

and for $q(\lambda)$ we have

$$a_N = a_0 + \frac{N+1}{2} \tag{21.81}$$

$$b_N = b_0 + \kappa_0(\mathbb{E}\left[\mu^2\right] + \mu_0^2 - 2\mathbb{E}\left[\mu\right]\mu_0) + \frac{1}{2}\sum_{i=1}^{N}\left(x_i^2 + \mathbb{E}\left[\mu^2\right] - 2\mathbb{E}\left[\mu\right]x_i\right) \tag{21.82}$$

We see that μ_N and a_N are in fact fixed constants, and only κ_N and b_N need to be updated iteratively. (In fact, one can solve for the fixed points of κ_N and b_N analytically, but we don't do this here in order to illustrate the iterative updating scheme.)

21.5.1.5 Illustration

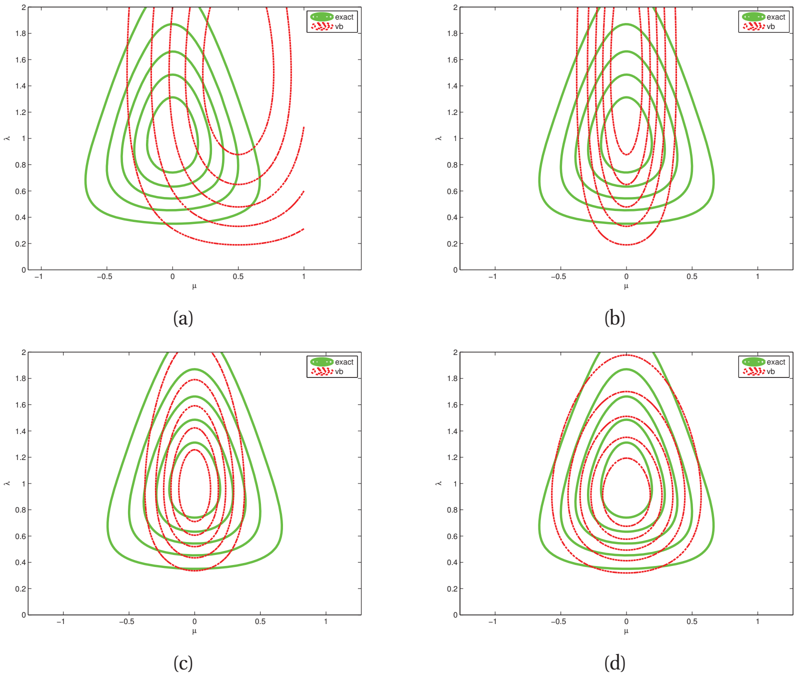
Figure 21.5 gives an example of this method in action. The green contours represent the exact posterior, which is Gaussian-Gamma. The dotted red contours represent the variational approximation over several iterations. We see that the final approximation is reasonably close to the exact solution. However, it is more "compact" than the true distribution. It is often the case that mean field inference underestimates the posterior uncertainty; See Section 21.2.2 for more discussion of this point.

21.5.1.6 Lower bound *

In VB, we are maximizing $L(q)$, which is a lower bound on the log marginal likelihood:

$$L(q) \leq \log p(\mathcal{D}) = \log \int\int p(\mathcal{D}|\mu,\lambda)p(\mu,\lambda)d\mu d\lambda \tag{21.83}$$

It is very useful to compute the lower bound itself, for three reasons. First, it can be used to assess convergence of the algorithm. Second, it can be used to assess the correctness of one's



(a)

(b)

(c)

(d)

Figure 21.5 Factored variational approximation (red) to the Gaussian-Gamma distribution (green). (a) Initial guess. (b) After updating q_μ. (c) After updating q_λ. (d) At convergence (after 5 iterations). Based on 10.4 of (Bishop 2006b). Figure generated by `unigaussVbDemo`.

code: as with EM, if the bound does not increase monotonically, there must be a bug. Third, the bound can be used as an approximation to the marginal likelihood, which can be used for Bayesian model selection.

Unfortunately, computing this lower bound involves a fair amount of tedious algebra. We work out the details for this example, but for other models, we will just state the results without proof, or even omit discussion of the bound altogether, for brevity.

For this model, $L(q)$ can be computed as follows:

$$L(q) = \int\int q(\mu,\lambda)\log\frac{p(\mathcal{D},\mu,\lambda)}{q(\mu,\lambda)}d\mu d\lambda \tag{21.84}$$

$$= \mathbb{E}\left[\log p(\mathcal{D}|\mu,\lambda)\right] + \mathbb{E}\left[\log p(\mu|\lambda)\right] + \mathbb{E}\left[\log p(\lambda)\right] - \mathbb{E}\left[\log q(\mu)\right] - \mathbb{E}\left[\log q(\lambda)\right] \tag{21.85}$$

where all expectations are wrt $q(\mu,\lambda)$. We recognize the last two terms as the entropy of a Gaussian and the entropy of a Gamma distribution, which are given by

$$\mathbb{H}\left(\mathcal{N}(\mu_N,\kappa_N^{-1})\right) = -\frac{1}{2}\log\kappa_N + \frac{1}{2}(1+\log(2\pi)) \tag{21.86}$$

$$\mathbb{H}\left(\mathrm{Ga}(a_N,b_N)\right) = \log\Gamma(a_N) - (a_N-1)\psi(a_N) - \log(b_N) + a_N \tag{21.87}$$

where $\psi()$ is the digamma function.

To compute the other terms, we need the following facts:

$$\begin{aligned}
\mathbb{E}\left[\log x | x \sim \mathrm{Ga}(a,b)\right] &= \psi(a) - \log(b) && (21.88)\\
\mathbb{E}\left[x | x \sim \mathrm{Ga}(a,b)\right] &= \frac{a}{b} && (21.89)\\
\mathbb{E}\left[x | x \sim \mathcal{N}(\mu,\sigma^2)\right] &= \mu && (21.90)\\
\mathbb{E}\left[x^2 | x \sim \mathcal{N}(\mu,\sigma^2)\right] &= \mu + \sigma^2 && (21.91)
\end{aligned}$$

For the expected log likelihood, one can show that

$$\begin{aligned}
&\mathbb{E}_{q(\mu,\lambda)}\left[\log p(\mathcal{D}|\mu,\lambda)\right] && (21.92)\\
&= -\frac{N}{2}\log(2\pi) + \frac{N}{2}\mathbb{E}_{q(\lambda)}\left[\log\lambda\right] - \frac{\mathbb{E}\left[\lambda\right]_{q(\lambda)}}{2}\sum_{i=1}^N \mathbb{E}_{q(\mu)}\left[(x_i-\mu)^2\right]\\
&= -\frac{N}{2}\log(2\pi) + \frac{N}{2}\left(\psi(a_N) - \log b_N\right) && (21.93)\\
&\quad -\frac{Na_N}{2b_N}\left(\hat{\sigma}^2 + \overline{x}^2 - 2\mu_N\overline{x} + \mu_N^2 + \frac{1}{\kappa_N}\right) && (21.94)
\end{aligned}$$

where $\overline{x}$ and $\hat{\sigma}^2$ are the empirical mean and variance.

For the expected log prior of λ, we have

$$\begin{aligned}
\mathbb{E}_{q(\lambda)}\left[\log p(\lambda)\right] &= (a_0-1)\mathbb{E}\left[\log\lambda\right] - b_0\mathbb{E}\left[\lambda\right] + a_0\log b_0 - \log\Gamma(a_0) && (21.95)\\
&= (a_0-1)(\psi(a_N) - \log b_N) - b_0\frac{a_N}{b_N} + a_0\log b_0 - \log\Gamma(a_0) && (21.96)
\end{aligned}$$

For the expected log prior of μ, one can show that

$$\begin{aligned}
\mathbb{E}_{q(\mu,\lambda)}\left[\log p(\mu|\lambda)\right] &= \frac{1}{2}\log\frac{\kappa_0}{2\pi} + \frac{1}{2}\mathbb{E}\left[\log\lambda\right]q(\lambda) - \frac{1}{2}\mathbb{E}_{q(\mu,\lambda)}\left[(\mu-\mu_0)^2\kappa_0\lambda\right]\\
&= \frac{1}{2}\log\frac{\kappa_0}{2\pi} + \frac{1}{2}\left(\psi(a_N) - \log b_N\right)\\
&\quad -\frac{\kappa_0}{2}\frac{a_N}{b_N}\left[\frac{1}{\kappa_N} + (\mu_N-\mu_0)^2\right] && (21.97)
\end{aligned}$$

Putting it altogether, one can show that

$$L(q) = \frac{1}{2}\log\frac{1}{\kappa_N} + \log\Gamma(a_N) - a_N\log b_N + \mathrm{const} \tag{21.98}$$

This quantity monotonically increases after each VB update.

21.5.2 Example: VB for linear regression

In Section 7.6.4, we discussed an empirical Bayes approach to setting the hyper-parameters for ridge regression known as the evidence procedure. In particular, we assumed a likelihood of the form $p(\mathbf{y}|\mathbf{X},\boldsymbol{\theta}) = \mathcal{N}(\mathbf{X}\mathbf{w},\lambda^{-1})$ and a prior of the form $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{0},\alpha^{-1}\mathbf{I})$. We then computed a type II estimate of α and λ. The same approach was extended in Section 13.7 to handle a prior of the form $\mathcal{N}(\mathbf{w}|\mathbf{0}, \text{diag}(\boldsymbol{\alpha})^{-1})$, which allows one hyper-parameter per feature, a technique known as automatic relevancy determination.

In this section, we derive a VB algorithm for this model. We follow the presentation of (Drugowitsch 2008).[3] Initially we will use the following prior:

$$p(\mathbf{w}, \lambda, \alpha) = \mathcal{N}(\mathbf{w}|\mathbf{0}, (\lambda\alpha)^{-1}\mathbf{I})\text{Ga}(\lambda|a_0^\lambda, b_0^\lambda)\text{Ga}(\alpha|a_0^\alpha, b_0^\alpha) \tag{21.99}$$

We choose to use the following factorized approximation to the posterior:

$$q(\mathbf{w}, \alpha, \lambda) = q(\mathbf{w}, \lambda)q(\alpha) \tag{21.100}$$

Given these assumptions, one can show (see (Drugowitsch 2008)) that the optimal form for the posterior is

$$q(\mathbf{w}, \alpha, \lambda) = \mathcal{N}(\mathbf{w}|\mathbf{w}_N, \lambda^{-1}\mathbf{V}_N)\text{Ga}(\lambda|a_N^\lambda, b_N^\lambda)\text{Ga}(\alpha|a_N^\alpha, b_N^\alpha) \tag{21.101}$$

where

$$\mathbf{V}_N^{-1} = \overline{\mathbf{A}} + \mathbf{X}^{\mathbf{X}} \tag{21.102}$$

$$\mathbf{w}_N = \mathbf{V}_N\mathbf{X}^T\mathbf{y} \tag{21.103}$$

$$a_N^\lambda = a_0^\lambda + \frac{N}{2} \tag{21.104}$$

$$b_N^\lambda = b_0^\lambda + \frac{1}{2}(||\mathbf{y} - \mathbf{X}\mathbf{w}||^2 + \mathbf{w}_N^T\overline{\mathbf{A}}\mathbf{w}_N) \tag{21.105}$$

$$a_N^\alpha = a_0^\alpha + \frac{D}{2} \tag{21.106}$$

$$b_N^\alpha = b_0^\alpha + \frac{1}{2}\left(\frac{a_N^\lambda}{b_N^\lambda}\mathbf{w}_N^T\mathbf{w}_N + \text{tr}(\mathbf{V}_N)\right) \tag{21.107}$$

$$\overline{\mathbf{A}} = \langle\alpha\rangle\mathbf{I} = \frac{a_N^\alpha}{b_N^\alpha}\mathbf{I} \tag{21.108}$$

This method can be extended to the ARD case in a straightforward way, by using the following priors:

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{0}, \text{diag}(\alpha)^{-1}) \tag{21.109}$$

$$p(\boldsymbol{\alpha}) = \prod_{j=1}^{D}\text{Ga}(\alpha_j|a_0^\alpha, b_0^\alpha) \tag{21.110}$$

The posterior for $\mathbf{w}$ and λ is computed as before, except we use $\overline{\mathbf{A}} = \text{diag}(a_N^\alpha/b_{N_j}^\alpha)$ instead of

[3] Note that Drugowitsch uses a_0, b_0 as the hyper-parameters for $p(\lambda)$ and c_0, d_0 as the hyper-parameters for $p(\alpha)$, whereas (Bishop 2006b, Sec 10.3) uses a_0, b_0 as the hyper-parameters for $p(\alpha)$ and treats λ as fixed. To (hopefully) avoid confusion, I use a_0^λ, b_0^λ as the hyper-parameters for $p(\lambda)$, and a_0^α, b_0^α as the hyper-parameters for $p(\alpha)$.

$a_N^\alpha/b_N^\alpha\mathbf{I}$. The posterior for $\boldsymbol{\alpha}$ has the form

$$\begin{aligned} q(\boldsymbol{\alpha}) &= \prod_j \mathrm{Ga}(\alpha_j|a_N^\alpha, b_{N_j}^\alpha) && (21.111) \\ a_N^\alpha &= a_0^\alpha + \frac{1}{2} && (21.112) \\ b_{N_j}^\alpha &= b_0^\alpha + \frac{1}{2}\left(\frac{a_N^\lambda}{b_N^\lambda} w_{N,j}^2 + (\mathbf{V}_N)_{jj}\right) && (21.113) \end{aligned}$$

The algorithm alternates between updating $q(\mathbf{w}, \lambda)$ and $q(\boldsymbol{\alpha})$. Once $\mathbf{w}$ and λ have been inferred, the posterior predictive is a Student distribution, as shown in Equation 7.76. Specifically, for a single data case, we have

$$p(y|\mathbf{x}, \mathcal{D}) = \mathcal{T}(y|\mathbf{w}_N^T\mathbf{x}, \frac{b_N^\lambda}{a_N^\lambda}(1 + \mathbf{x}^T\mathbf{V}_N\mathbf{x}), 2a_N^\lambda) \tag{21.114}$$

The exact marginal likelihood, which can be used for model selection, is given by

$$p(\mathcal{D}) = \int\int\int p(\mathbf{y}|\mathbf{X}, \mathbf{w}, \lambda)p(\mathbf{w}|\alpha)p(\lambda)d\mathbf{w}d\alpha d\lambda \tag{21.115}$$

We can compute a lower bound on $\log p(\mathcal{D})$ as follows:

$$\begin{aligned} L(q) &= -\frac{N}{2}\log(2\pi) - \frac{1}{2}\sum_{i=1}^N \left(\frac{a_N^\lambda}{b_N^\lambda}(y_i - \mathbf{w}_N^T\mathbf{x}_i)^2 + \mathbf{x}_i^T\mathbf{V}_N\mathbf{x}_i\right) \\ &\quad + \frac{1}{2}\log|\mathbf{V}_N| + \frac{D}{2} \\ &\quad - \log\Gamma(a_0^\lambda) + a_0^\lambda \log b_0^\lambda - b_0^\lambda \frac{a_N^\lambda}{b_N^\lambda} + \log\Gamma(a_N^\lambda) - a_N^\lambda \log b_N^\lambda + a_N^\lambda \\ &\quad - \log\Gamma(a_0^\alpha) + a_0^\alpha \log b_0^\alpha + \log\Gamma(a_N^\alpha) - a_N^\alpha \log b_N^\alpha && (21.116) \end{aligned}$$

In the ARD case, the last line becomes

$$\sum_{j=1}^D \left[-\log\Gamma(a_0^\alpha) + a_0^\alpha \log b_0^\alpha + \log\Gamma(a_N^\alpha) - a_N^\alpha \log b_{N_j}^\alpha\right] \tag{21.117}$$

Figure 21.6 compare VB and EB on a model selection problem for polynomial regression. We see that VB gives similar results to EB, but the precise behavior depends on the sample size. When $N = 5$, VB's estimate of the posterior over models is more diffuse than EB's, since VB models uncertainty in the hyper-parameters. When $N = 30$, the posterior estimate of the hyper-parameters becomes more well-determined. Indeed, if we compute $\mathbb{E}[\alpha|\mathcal{D}]$ when we have an uninformative prior, $a_0^\alpha = b_0^\alpha = 0$, we get

$$\overline{\alpha} = \frac{a_N^\alpha}{b_N^\alpha} = \frac{D/2}{\frac{1}{2}(\frac{a_N^\lambda}{b_N^\lambda}\mathbf{w}_N^T\mathbf{w}_N + \mathrm{tr}(\mathbf{V}_N))} \tag{21.118}$$

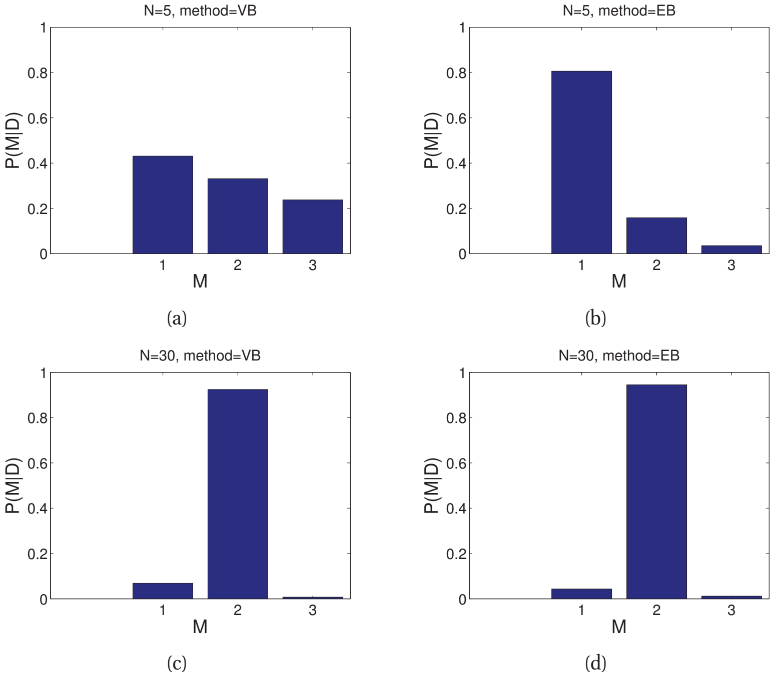


Figure 21.6 We plot the posterior over models (polynomials of degree 1, 2 and 3) assuming a uniform prior $p(m) \propto 1$. We approximate the marginal likelihood using (a,c) VB and (b,d) EB. In (a-b), we use $N = 5$ data points (shown in Figure 5.7). In (c-d), we use $N = 30$ data points (shown in Figure 5.8). Figure generated by `linregEbModelSelVsN`.

Compare this to Equation 13.167 for EB:

$$\hat{\alpha} = \frac{D}{\mathbb{E}\left[\mathbf{w}^T\mathbf{w}\right]} = \frac{D}{\mathbf{w}_N^T\mathbf{w}_N + \text{tr}(\mathbf{V}_N)} \tag{21.119}$$

Modulo the a_N^λ and b_N^λ terms, these are the same. In hindsight this is perhaps not that surprising, since EB is trying to maximize $\log p(\mathcal{D})$, and VB is trying to maximize a lower bound on $\log p(\mathcal{D})$.

21.6 Variational Bayes EM

Now consider latent variable models of the form $\mathbf{z}_i \rightarrow \mathbf{x}_i \leftarrow \boldsymbol{\theta}$. This includes mixtures models, PCA, HMMs, etc. There are now two kinds of unknowns: parameters, $\boldsymbol{\theta}$, and latent variables, $\mathbf{z}_i$. As we saw in Section 11.4, it is common to fit such models using EM, where in the E step we infer the posterior over the latent variables, $p(\mathbf{z}_i|\mathbf{x}_i, \boldsymbol{\theta})$, and in the M step, we compute a point estimate of the parameters, $\boldsymbol{\theta}$. The justification for this is two-fold. First, it results in simple algorithms. Second, the posterior uncertainty in $\boldsymbol{\theta}$ is usually less than in $\mathbf{z}_i$, since the $\boldsymbol{\theta}$ are informed by all N data cases, whereas $\mathbf{z}_i$ is only informed by $\mathbf{x}_i$; this makes a MAP estimate of

$\boldsymbol{\theta}$ more reasonable than a MAP estimate of $\mathbf{z}_i$.

However, VB provides a way to be "more Bayesian", by modeling uncertainty in the parameters $\boldsymbol{\theta}$ as well in the latent variables $\mathbf{z}_i$, at a computational cost that is essentially the same as EM. This method is known as **variational Bayes EM** or **VBEM**. The basic idea is to use mean field, where the approximate posterior has the form

$$p(\boldsymbol{\theta}, \mathbf{z}_{1:N}|\mathcal{D}) \approx q(\boldsymbol{\theta})q(\mathbf{z}) = q(\boldsymbol{\theta}) \prod_i q(\mathbf{z}_i) \tag{21.120}$$

The first factorization, between $\boldsymbol{\theta}$ and $\mathbf{z}$, is a crucial assumption to make the algorithm tractable. The second factorization follows from the model, since the latent variables are iid conditional on $\boldsymbol{\theta}$.

In VBEM, we alternate between updating $q(\mathbf{z}_i|\mathcal{D})$ (the variational E step) and updating $q(\boldsymbol{\theta}|\mathcal{D})$ (the variational M step). We can recover standard EM from VBEM by approximating the parameter posterior using a delta function, $q(\boldsymbol{\theta}|\mathcal{D}) \approx \delta_{\hat{\boldsymbol{\theta}}}(\boldsymbol{\theta})$.

The variational E step is similar to a standard E step, except instead of plugging in a MAP estimate of the parameters and computing $p(\mathbf{z}_i|\mathcal{D}, \hat{\boldsymbol{\theta}})$, we need to average over the parameters. Roughly speaking, this can be computed by plugging in the posterior mean of the parameters instead of the MAP estimate, and then computing $p(\mathbf{z}_i|\mathcal{D}, \overline{\boldsymbol{\theta}})$ using standard algorithms, such as forwards-backwards. Unfortunately, things are not quite this simple, but this is the basic idea. The details depend on the form of the model; we give some examples below.

The variational M step is similar to a standard M step, except instead of computing a point estimate of the parameters, we update the hyper-parameters, using the expected sufficient statistics. This process is usually very similar to MAP estimation in regular EM. Again, the details on how to do this depend on the form of the model.

The principle advantage of VBEM over regular EM is that by marginalizing out the parameters, we can compute a lower bound on the marginal likelihood, which can be used for model selection. We will see an example of this in Section 21.6.1.6. VBEM is also "egalitarian", since it treats parameters as "first class citizens", just like any other unknown quantity, whereas EM makes an artificial distinction between parameters and latent variables.

21.6.1 Example: VBEM for mixtures of Gaussians *

Let us consider how to "fit" a mixture of Gaussians using VBEM. (We use scare quotes since we are not estimating the model parameters, but inferring a posterior over them.) We will follow the presentation of (Bishop 2006b, Sec 10.2). Unfortunately, the details are rather complicated. Fortunately, as with EM, one gets used to it after a bit of practice. (As usual with math, simply reading the equations won't help much, you should really try deriving these results yourself (or try some of the exercises) if you want to learn this stuff in depth.)

21.6.1.1 The variational posterior

The likelihood function is the usual one for Gaussian mixture models:

$$p(\mathbf{z}, \mathbf{X}|\boldsymbol{\theta}) = \prod_i \prod_k \pi_k^{z_{ik}} \mathcal{N}(\mathbf{x}_i|\boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k^{-1})^{z_{ik}} \tag{21.121}$$

where $z_{ik} = 1$ if data point i belongs to cluster k, and $z_{ik} = 0$ otherwise.

We will assume the following factored conjugate prior

$$p(\boldsymbol{\theta}) = \text{Dir}(\boldsymbol{\pi}|\boldsymbol{\alpha}_0) \prod_k \mathcal{N}(\boldsymbol{\mu}_k|\mathbf{m}_0, (\beta_0 \boldsymbol{\Lambda}_k)^{-1}) \text{Wi}(\boldsymbol{\Lambda}_k|\mathbf{L}_0, \nu_0) \tag{21.122}$$

where $\boldsymbol{\Lambda}_k$ is the precision matrix for cluster k. The subscript 0 means these are parameters of the prior; we assume all the prior parameters are the same for all clusters. For the mixing weights, we usually use a symmetric prior, $\boldsymbol{\alpha}_0 = \alpha_0 \mathbf{1}$.

The exact posterior $p(\mathbf{z}, \boldsymbol{\theta}|\mathcal{D})$ is a mixture of K^N distributions, corresponding to all possible labelings $\mathbf{z}$. We will try to approximate the volume around one of these modes. We will use the standard VB approximation to the posterior:

$$p(\boldsymbol{\theta}, \mathbf{z}_{1:N}|\mathcal{D}) \approx q(\boldsymbol{\theta}) \prod_i q(\mathbf{z}_i) \tag{21.123}$$

At this stage we have not specified the forms of the q functions; these will be determined by the form of the likelihood and prior. Below we will show that the optimal form is as follows:

$$\begin{aligned} q(\mathbf{z}, \boldsymbol{\theta}) &= q(\mathbf{z}|\boldsymbol{\theta}) q(\boldsymbol{\theta}) = \left[\prod_i \text{Cat}(\mathbf{z}_i|\mathbf{r}_i)\right] & (21.124) \\ & \left[\text{Dir}(\boldsymbol{\pi}|\boldsymbol{\alpha}) \prod_k \mathcal{N}(\boldsymbol{\mu}_k|\mathbf{m}_k, (\beta_k \boldsymbol{\Lambda}_k)^{-1}) \text{Wi}(\boldsymbol{\Lambda}_k|\mathbf{L}_k, \nu_k)\right] & (21.125) \end{aligned}$$

(The lack of 0 subscript means these are parameters of the posterior, not the prior.) Below we will derive the update equations for these variational parameters.

21.6.1.2 Derivation of $q(\mathbf{z})$ (variational E step)

The form for $q(\mathbf{z})$ can be obtained by looking at the complete data log joint, ignoring terms that do not involve $\mathbf{z}$, and taking expectations of what's left over wrt all the hidden variables except for $\mathbf{z}$. We have

$$\begin{aligned} \log q(\mathbf{z}) &= \mathbb{E}_{q(\boldsymbol{\theta})} \left[\log p(\mathbf{x}, \mathbf{z}, \boldsymbol{\theta})\right] + \text{const} & (21.126) \\ &= \sum_i \sum_i z_{ik} \log \rho_{ik} + \text{const} & (21.127) \end{aligned}$$

where we define

$$\begin{aligned} \log \rho_{ik} &\triangleq \mathbb{E}_{q(\boldsymbol{\theta})} \left[\log \pi_k\right] + \frac{1}{2} \mathbb{E}_{q(\boldsymbol{\theta})} \left[\log |\boldsymbol{\Lambda}_k|\right] - \frac{D}{2} \log(2\pi) \\ & -\frac{1}{2} \mathbb{E}_{q(\boldsymbol{\theta})} \left[(\mathbf{x}_i - \boldsymbol{\mu}_k)^T \boldsymbol{\Lambda}_k (\mathbf{x}_i - \boldsymbol{\mu}_k)\right] & (21.128) \end{aligned}$$

Using the fact that $q(\boldsymbol{\pi}) = \text{Dir}(\boldsymbol{\pi})$, we have

$$\log \tilde{\pi}_k \triangleq \mathbb{E}\left[\log \pi_k\right] = \psi(\alpha_k) - \psi(\sum_{k'} \alpha_{k'}) \tag{21.129}$$

where $\psi()$ is the digamma function. (See Exercise 21.5 for the detailed derivation.) Next, we use the fact that

$$q(\boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k) = \mathcal{N}(\boldsymbol{\mu}_k|\mathbf{m}_k, (\beta_k \boldsymbol{\Lambda}_k)^{-1})\mathrm{Wi}(\boldsymbol{\Lambda}_k|\mathbf{L}_k, \nu_k) \tag{21.130}$$

to get

$$\log \tilde{\Lambda}_k \triangleq \mathbb{E}\left[\log |\boldsymbol{\Lambda}_k|\right] = \sum_{j=1}^{D} \psi\left(\frac{\nu_k + 1 - j}{2}\right) + D\log 2 + \log |\boldsymbol{\Lambda}_k| \tag{21.131}$$

Finally, for the expected value of the quadratic form, we get

$$\mathbb{E}\left[(\mathbf{x}_i - \boldsymbol{\mu}_k)^T \boldsymbol{\Lambda}_k (\mathbf{x}_i - \boldsymbol{\mu}_k)\right] = D\beta_k^{-1} + \nu_k(\mathbf{x}_i - \mathbf{m}_k)^T \boldsymbol{\Lambda}_k (\mathbf{x}_i - \mathbf{m}_k) \tag{21.132}$$

Putting it altogether, we get that the posterior responsibility of cluster k for datapoint i is

$$r_{ik} \quad \propto \quad \tilde{\pi}_k \tilde{\Lambda}_k^{\frac{1}{2}} \exp\left(-\frac{D}{2\beta_k} - \frac{\nu_k}{2}(\mathbf{x}_i - \mathbf{m}_k)^T \boldsymbol{\Lambda}_k (\mathbf{x}_i - \mathbf{m}_k)\right) \tag{21.133}$$

Compare this to the expression used in regular EM:

$$r_{ik}^{EM} \quad \propto \quad \hat{\pi}_k |\hat{\boldsymbol{\Lambda}}|_k^{\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_k)^T \hat{\boldsymbol{\Lambda}}_k (\mathbf{x}_i - \hat{\boldsymbol{\mu}}_k)\right) \tag{21.134}$$

The significance of this difference is discussed further in Section 21.6.1.7.

21.6.1.3 Derivation of $q(\theta)$ (variational M step)

Using the mean field recipe, we have

$$\begin{aligned} \log q(\boldsymbol{\theta}) \quad &= \quad \log p(\boldsymbol{\pi}) + \sum_k \log p(\mu_k, \boldsymbol{\Lambda}_k) + \sum_i \mathbb{E}_{q(\mathbf{z})}\left[\log p(\mathbf{z}_i|\boldsymbol{\pi})\right] \\ &\quad + \sum_k \sum_i \mathbb{E}_{q(\mathbf{z})}\left[z_{ik}\right] \log \mathcal{N}(\mathbf{x}_i|\boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k^{-1}) + \text{const} \end{aligned} \tag{21.135}$$

We see this factorizes into the form

$$q(\boldsymbol{\theta}) \quad = \quad q(\boldsymbol{\pi}) \prod_k q(\boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k) \tag{21.136}$$

For the $\boldsymbol{\pi}$ term, we have

$$\log q(\boldsymbol{\pi}) \quad = \quad (\alpha_0 - 1)\sum_k \log \pi_k + \sum_k \sum_i r_{ik} \log \pi_k + \text{const} \tag{21.137}$$

Exponentiating, we recognize this as a Dirichlet distribution:

$$q(\boldsymbol{\pi}) \quad = \quad \mathrm{Dir}(\boldsymbol{\pi}|\boldsymbol{\alpha}) \tag{21.138}$$

$$\alpha_k \quad = \quad \alpha_0 + N_k \tag{21.139}$$

$$N_k \quad = \quad \sum_i r_{ik} \tag{21.140}$$

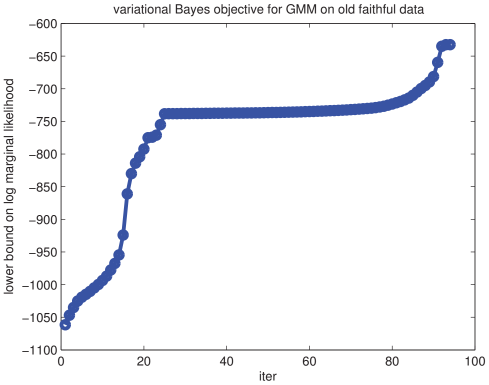


Figure 21.7 Lower bound vs iterations for the VB algorithm in Figure 21.8. The steep parts of the curve correspond to places where the algorithm figures out that it can increase the bound by "killing off" unnecessary mixture components, as described in Section 21.6.1.6. The plateaus correspond to slowly moving the clusters around. Figure generated by `mixGaussVbDemoFaithful`.

For the $\boldsymbol{\mu}_k$ and $\boldsymbol{\Lambda}_k$ terms, we have

$$
\begin{aligned}
q(\boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k) &= \mathcal{N}(\boldsymbol{\mu}_k|\mathbf{m}_k, (\beta_k \boldsymbol{\Lambda}_k)^{-1})\mathrm{Wi}(\boldsymbol{\Lambda}_k|\mathbf{L}_k, \nu_k) &(21.141)\\
\beta_k &= \beta_0 + N_k &(21.142)\\
\mathbf{m}_k &= (\beta_0 \mathbf{m}_0 + N_k \overline{\mathbf{x}}_k)/\beta_k &(21.143)\\
\mathbf{L}_k^{-1} &= \mathbf{L}_0^{-1} + N_k \mathbf{S}_k + \frac{\beta_0 N_k}{\beta_0 + N_k}(\overline{\mathbf{x}}_k - \mathbf{m}_0)(\overline{\mathbf{x}}_k - \mathbf{m}_0)^T &(21.144)\\
\nu_k &= \nu_0 + N_k + 1 &(21.145)\\
\overline{\mathbf{x}}_k &= \frac{1}{N_k}\sum_i r_{ik}\mathbf{x}_i &(21.146)\\
\mathbf{S}_k &= \frac{1}{N_k}\sum_i r_{ik}(\mathbf{x}_i - \overline{\mathbf{x}}_k)(\mathbf{x}_i - \overline{\mathbf{x}}_k)^T &(21.147)
\end{aligned}
$$

This is very similar to the M step for MAP estimation discussed in Section 11.4.2.8, except here we are computing the parameters of the posterior over $\boldsymbol{\theta}$, rather than MAP estimates of $\boldsymbol{\theta}$.

21.6.1.4 Lower bound on the marginal likelihood

The algorithm is trying to maximize the following lower bound

$$\mathcal{L} = \sum_{\mathbf{z}} \int q(\mathbf{z}, \boldsymbol{\theta}) \log \frac{p(\mathbf{x}, \mathbf{z}, \boldsymbol{\theta})}{q(\mathbf{z}, \boldsymbol{\theta})} d\boldsymbol{\theta} \leq \log p(\mathcal{D}) \tag{21.148}$$

This quantity should increase monotonically with each iteration, as shown in Figure 21.7. Unfortunately, deriving the bound is a bit messy, because we need to compute expectations of the unnormalized log posterior as well as entropies of the q distribution. We leave the details (which are similar to Section 21.5.1.6) to Exercise 21.4.

21.6.1.5 Posterior predictive distribution

We showed that the approximate posterior has the form

$$q(\boldsymbol{\theta}) = \mathrm{Dir}(\boldsymbol{\pi}|\boldsymbol{\alpha}) \prod_k \mathcal{N}(\boldsymbol{\mu}_k|\mathbf{m}_k, (\beta_k \boldsymbol{\Lambda}_k)^{-1}) \mathrm{Wi}(\boldsymbol{\Lambda}_k|\mathbf{L}_k, \nu_k) \tag{21.149}$$

Consequently the posterior predictive density can be approximated as follows, using the results from Section 4.6.3.6:

$$\begin{aligned}
p(\mathbf{x}|\mathcal{D}) &\approx \sum_z \int p(\mathbf{x}|z, \boldsymbol{\theta}) p(z|\boldsymbol{\theta}) q(\boldsymbol{\theta}) d\boldsymbol{\theta} && (21.150)\\
&= \sum_k \int \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k^{-1}) q(\boldsymbol{\theta}) d\boldsymbol{\theta} && (21.151)\\
&= \sum_k \frac{\alpha_k}{\sum_{k'} \alpha_{k'}} \mathcal{T}(\mathbf{x}|\mathbf{m}_k, \mathbf{M}_k, \nu_k + 1 - D) && (21.152)\\
\mathbf{M}_k &= \frac{(\nu_k + 1 - D)\beta_k}{1 + \beta_k} \mathbf{L}_k && (21.153)
\end{aligned}$$

This is just a weighted sum of Student distributions. If instead we used a plug-in approximation, we would get a weighted sum of Gaussian distributions.

21.6.1.6 Model selection using VBEM

The simplest way to select K when using VB is to fit several models, and then to use the variational lower bound to the log marginal likelihood, $\mathcal{L}(K) \leq \log p(\mathcal{D}|K)$, to approximate $p(K|\mathcal{D})$:

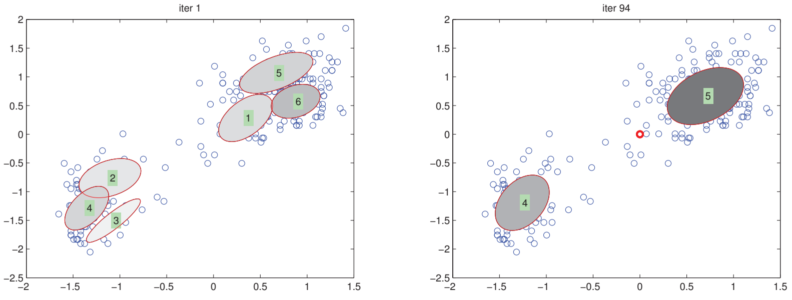
$$p(K|\mathcal{D}) = \frac{e^{\mathcal{L}(K)}}{\sum_{K'} e^{\mathcal{L}(K')}} \tag{21.154}$$

However, the lower bound needs to be modified somewhat to take into account the lack of identifiability of the parameters (Section 11.3.1). In particular, although VB will approximate the volume occupied by the parameter posterior, it will only do so around one of the local modes. With K components, there are $K!$ equivalent modes, which differ merely by permuting the labels. Therefore we should use $\log p(\mathcal{D}|K) \approx \mathcal{L}(K) + \log(K!)$.

21.6.1.7 Automatic sparsity inducing effects of VBEM

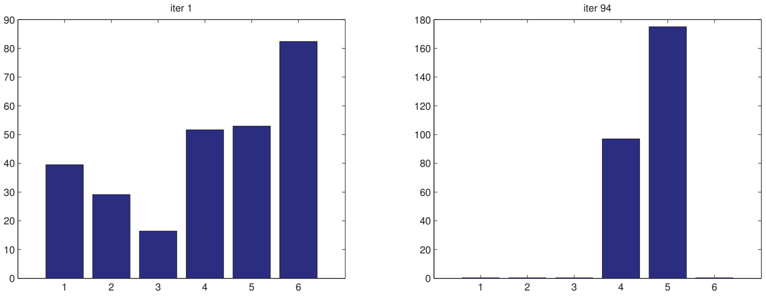
Although VB provides a reasonable approximation to the marginal likelihood (better than BIC (Beal and Ghahramani 2006)), this method still requires fitting multiple models, one for each value of K being considered. A faster alternative is to fit a single model, where K is set large, but where α_0 is set very small, $\alpha_0 \ll 1$. From Figure 2.14(d), we see that the resulting prior for the mixing weights $\boldsymbol{\pi}$ has "spikes" near the corners of the simplex, encouraging a sparse mixing weight vector.

In regular EM, the MAP estimate of the mixing weights will have the form $\hat{\pi}_k \propto (\alpha_k - 1)$, where $\alpha_k = \alpha_0 + N_k$. Unforuntately, this can be negative if $\alpha_0 = 0$ and $N_k = 0$ (Figueiredo



(a) (b)

Figure 21.8 We visualize the posterior mean parameters at various stages of the VBEM algorithm applied to a mixture of Gaussians model on the Old Faithful data. Shading intensity is proportional to the mixing weight. We initialize with K-means and use $\alpha_0 = 0.001$ as the Dirichlet hyper-parameter. Based on Figure 10.6 of (Bishop 2006b). Figure generated by `mixGaussVbDemoFaithful`, based on code by Emtiyaz Khan.



(a) (b)

Figure 21.9 We visualize the posterior values of $\boldsymbol{\alpha}_k$ for the model in Figure 21.8. We see that unnecessary components get "killed off". Figure generated by `mixGaussVbDemoFaithful`.

and Jain 2002). However, in VBEM, we use

$$\tilde{\pi}_k = \frac{\exp[\Psi(\alpha_k)]}{\exp[\Psi(\sum_{k'} \alpha_{k'})]} \tag{21.155}$$

Now $\exp(\Psi(x)) \approx x - 0.5$ for $x > 1$. So if $\alpha_k = 0$, when we compute $\tilde{\pi}_k$, it's like we substract 0.5 from the posterior counts. This will hurt small clusters more than large clusters (like a regressive tax).[4] The effect is that clusters which have very few (weighted) members become more and more empty over successive iterations, whereas the popular clusters get more and more members. This is called the **rich get richer** phenomenon; we will encounter it again in Section 25.2, when we discuss Dirichlet process mixture models.

This automatic pruning method is demonstrated in Figure 21.8. We fit a mixture of 6 Gaussians to the Old Faithful dataset, but the data only really "needs" 2 clusters, so the rest get "killed off".

[4] For more details, see (Liang et al. 2007).

In this example, we used $\alpha_0 = 0.001$; if we use a larger α_0, we do not get a sparsity effect. In Figure 21.9, we plot $q(\boldsymbol{\alpha}|\mathcal{D})$ at various iterations; we see that the unwanted components get extinguished. This provides an efficient alternative to performing a discrete search over the number of clusters.

21.7 Variational message passing and VIBES

We have seen that mean field methods, at least of the fully-factorized variety, are all very similar: just compute each node's full conditional, and average out the neighbors. This is very similar to Gibbs sampling (Section 24.2), except the derivation of the equations is usually a bit more work. Fortunately it is possible to derive a general purpose set of update equations that work for any DGM for which all CPDs are in the exponential family, and for which all parent nodes have conjugate distributions (Ghahramani and Beal 2001). (See (Wand et al. 2011) for a recent extension to handle non-conjugate priors.) One can then sweep over the graph, updating nodes one at a time, in a manner similar to Gibbs sampling. This is known as **variational message passing** or **VMP** (Winn and Bishop 2005), and has been implemented in the open-source program **VIBES**[5]. This is a VB analog to BUGS, which is a popular generic program for Gibbs sampling discussed in Section 24.2.6.

VMP/ mean field is best-suited to inference where one or more of the hidden nodes are continuous (e.g., when performing "Bayesian learning"). For models where all the hidden nodes are discrete, more accurate approximate inference algorithms can be used, as we discuss in Chapter 22.

21.8 Local variational bounds *

So far, we have been focusing on mean field inference, which is a form of variational inference based on minimizing $\mathbb{KL}(q||\tilde{p})$, where q is the approximate posterior, assumed to be factorized, and $\tilde{p}$ is the exact (but unnormalized) posterior. However, there is another kind of variational inference, where we replace a specific term in the joint distribution with a simpler function, to simplify computation of the posterior. Such an approach is sometimes called a **local variational approximation**, since we are only modifying one piece of the model, unlike mean field, which is a global approximation. In this section, we study several examples of this method.

21.8.1 Motivating applications

Before we explain how to derive local variational bounds, we give some examples of where this is useful.

21.8.1.1 Variational logistic regression

Consider the problem of how to approximate the parameter posterior for multiclass logistic regression model under a Gaussian prior. One approach is to use a Gaussian (Laplace) approximation, as discussed in Section 8.4.3. However, a variational approach can produce a more

[5] Available at `http://vibes.sourceforge.net/`.

accurate approximation to the posterior, since it has tunable parameters. Another advantage is that the variational approach monotonically optimizes a lower bound on the likelihood of the data, as we will see.

To see why we need a bound, note that the likelihood can be written as follows:

$$p(\mathbf{y}|\mathbf{X}, \mathbf{w}) = \prod_{i=1}^{N} \exp\left[\mathbf{y}_i^T \boldsymbol{\eta}_i - \text{lse}(\boldsymbol{\eta}_i)\right] \tag{21.156}$$

where $\boldsymbol{\eta}_i = \mathbf{X}_i \mathbf{w}_i = [\mathbf{x}_i^T \mathbf{w}_1, \ldots, \mathbf{x}_i^T \mathbf{w}_M]$, where $M = C - 1$ (since we set $\mathbf{w}_C = \mathbf{0}$ for identifiability), and where we define the **log-sum-exp** or **lse** function as follows:

$$\text{lse}(\boldsymbol{\eta}_i) \triangleq \log\left(1 + \sum_{m=1}^{M} e^{\eta_{im}}\right) \tag{21.157}$$

The main problem is that this likelihood is not conjugate to the Gaussian prior. Below we discuss how to compute "Gaussian-like" lower bounds to this likelihood, which give rise to approximate Gaussian posteriors.

21.8.1.2 Multi-task learning

One important application of Bayesian inference for logistic regression is where we have multiple related classifiers we want to fit. In this case, we want to share information between the parameters for each classifier; this requires that we maintain a posterior distibution over the parameters, so we have a measure of confidence as well as an estimate of the value. We can embed the above variational method inside of a larger hierarchical model in order to perform such multi-task learning, as described in e.g., (Braun and McAuliffe 2010).

21.8.1.3 Discrete factor analysis

Another situation where variational bounds are useful arises when we fit a factor analysis model to discrete data. This model is just like multinomial logistic regression, except the input variables are hidden factors. We need to perform inference on the hidden variables as well as the regression weights. For simplicity, we might perform point estimation of the weights, and just integrate out the hidden variables. We can do this using variational EM, where we use the variational bound in the E step. See Section 12.4 for details.

21.8.1.4 Correlated topic model

A topic model is a latent variable model for text documents and other forms of discrete data; see Section 27.3 for details. Often we assume the distribution over topics has a Dirichlet prior, but a more powerful model, known as the correlated topic model, uses a Gaussian prior, which can model correlations more easily (see Section 27.4.1 for details). Unfortunately, this also involves the lse function. However, we can use our variational bounds in the context of a variational EM algorithm, as we will see later.

21.8.2 Bohning's quadratic bound to the log-sum-exp function

All of the above examples require dealing with multiplying a Gaussian prior by a multinomial likelihood; this is difficult because of the log-sum-exp (lse) term. In this section, we derive a way to derive a "Gaussian-like" lower bound on this likelihood.

Consider a Taylor series expansion of the lse function around $\boldsymbol{\psi}_i \in \mathbb{R}^M$:

$$\begin{aligned}
\text{lse}(\boldsymbol{\eta}_i) &= \text{lse}(\boldsymbol{\psi}_i) + (\boldsymbol{\eta}_i - \boldsymbol{\psi}_i)^T \mathbf{g}(\boldsymbol{\psi}_i) + \frac{1}{2}(\boldsymbol{\eta}_i - \boldsymbol{\psi}_i)^T \mathbf{H}(\boldsymbol{\psi}_i)(\boldsymbol{\eta}_i - \boldsymbol{\psi}_i) && (21.158)\\
\mathbf{g}(\boldsymbol{\psi}_i) &= \exp[\boldsymbol{\psi}_i - \text{lse}(\boldsymbol{\psi}_i)] = \mathcal{S}(\boldsymbol{\psi}_i) && (21.159)\\
\mathbf{H}(\boldsymbol{\psi}_i) &= \text{diag}(\mathbf{g}(\boldsymbol{\psi}_i)) - \mathbf{g}(\boldsymbol{\psi}_i)\mathbf{g}(\boldsymbol{\psi}_i)^T && (21.160)
\end{aligned}$$

where $\mathbf{g}$ and $\mathbf{H}$ are the gradient and Hessian of lse, and $\boldsymbol{\psi}_i \in \mathbb{R}^M$ is chosen such that equality holds. An upper bound to lse can be found by replacing the Hessian matrix $\mathbf{H}(\boldsymbol{\psi}_i)$ with a matrix $\mathbf{A}_i$ such that $\mathbf{A}_i \prec \mathbf{H}(\boldsymbol{\psi}_i)$. (Bohning 1992) showed that this can be achieved if we use the matrix $\mathbf{A}_i = \frac{1}{2}\left[\mathbf{I}_M - \frac{1}{M+1}\mathbf{1}_M\mathbf{1}_M^T\right]$. (Recall that $M + 1 = C$ is the number of classes.) Note that $\mathbf{A}_i$ is independent of $\boldsymbol{\psi}_i$; however, we still write it as $\mathbf{A}_i$ (rather than dropping the i subscript), since other bounds that we consider below will have a data-dependent curvature term. The upper bound on lse therefore becomes

$$\begin{aligned}
\text{lse}(\boldsymbol{\eta}_i) &\leq \frac{1}{2}\boldsymbol{\eta}_i^T \mathbf{A}_i \boldsymbol{\eta}_i - \mathbf{b}_i^T \boldsymbol{\eta}_i + c_i && (21.161)\\
\mathbf{A}_i &= \frac{1}{2}\left[\mathbf{I}_M - \frac{1}{M+1}\mathbf{1}_M\mathbf{1}_M^T\right] && (21.162)\\
\mathbf{b}_i &= \mathbf{A}_i\boldsymbol{\psi}_i - \mathbf{g}(\boldsymbol{\psi}_i) && (21.163)\\
c_i &= \frac{1}{2}\boldsymbol{\psi}_i^T \mathbf{A}_i \boldsymbol{\psi}_i - \mathbf{g}(\boldsymbol{\psi}_i)^T \boldsymbol{\psi}_i + \text{lse}(\boldsymbol{\psi}_i) && (21.164)
\end{aligned}$$

where $\boldsymbol{\psi}_i \in \mathbb{R}^M$ is a vector of variational parameters.

We can use the above result to get the following lower bound on the softmax likelihood:

$$\log p(y_i = c|\mathbf{x}_i, \mathbf{w}) \geq \left[\mathbf{y}_i^T \mathbf{X}_i \mathbf{w} - \frac{1}{2}\mathbf{w}^T \mathbf{X}_i \mathbf{A}_i \mathbf{X}_i \mathbf{w} + \mathbf{b}_i^T \mathbf{X}_i \mathbf{w} - c_i\right]_c \tag{21.165}$$

To simplify notation, define the pseudo-measurement

$$\tilde{\mathbf{y}}_i \triangleq \mathbf{A}_i^{-1}(\mathbf{b}_i + \mathbf{y}_i) \tag{21.166}$$

Then we can get a "Gaussianized" version of the observation model:

$$p(\mathbf{y}_i|\mathbf{x}_i, \mathbf{w}) \geq f(\mathbf{x}_i, \boldsymbol{\psi}_i)\, \mathcal{N}(\tilde{\mathbf{y}}_i|\mathbf{X}_i\mathbf{w}, \mathbf{A}_i^{-1}) \tag{21.167}$$

where $f(\mathbf{x}_i, \boldsymbol{\psi}_i)$ is some function that does not depend on $\mathbf{w}$. Given this, it is easy to compute the posterior $q(\mathbf{w}) = \mathcal{N}(\mathbf{m}_N, \mathbf{V}_N)$, using Bayes rule for Gaussians. Below we will explain how to update the variational parameters $\boldsymbol{\psi}_i$.

21.8.2.1 Applying Bohning's bound to multinomial logistic regression

Let us see how to apply this bound to multinomial logistic regression. From Equation 21.13, we can define the goal of variational inference as maximizing

$$\begin{aligned} L(q) &\triangleq -\mathbb{KL}\left(q(\mathbf{w})||p(\mathbf{w}|\mathcal{D})\right) + \mathbb{E}_q\left[\sum_{i=1}^{N} \log p(y_i|\mathbf{x}_i, \mathbf{w})\right] && (21.168)\\ &= -\mathbb{KL}\left(q(\mathbf{w})||p(\mathbf{w}|\mathcal{D})\right) + \mathbb{E}_q\left[\sum_{i=1}^{N} \mathbf{y}_i^T \boldsymbol{\eta}_i - \text{lse}(\boldsymbol{\eta}_i)\right] && (21.169)\\ &= -\mathbb{KL}\left(q(\mathbf{w})||p(\mathbf{w}|\mathcal{D})\right) + \sum_{i=1}^{N} \mathbf{y}_i^T \mathbb{E}_q\left[\boldsymbol{\eta}_i\right] - \sum_{i=1}^{N} \mathbb{E}_q\left[\text{lse}(\boldsymbol{\eta}_i)\right] && (21.170) \end{aligned}$$

where $q(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_N, \mathbf{V}_N)$ is the approximate posterior. The first term is just the KL divergence between two Gaussians, which is given by

$$\begin{aligned} -\mathbb{KL}\left(\mathcal{N}(\mathbf{m}_0, \mathbf{V}_0)||\mathcal{N}(\mathbf{m}_N, \mathbf{V}_N)\right) &= -\frac{1}{2}\left[\text{tr}(\mathbf{V}_N \mathbf{V}_0^{-1}) - \log|\mathbf{V}_N \mathbf{V}_0^{-1}|\right.\\ &\qquad \left. +(\mathbf{m}_N - \mathbf{m}_0)^T \mathbf{V}_0^{-1}(\mathbf{m}_N - \mathbf{m}_0) - DM\right] && (21.171) \end{aligned}$$

where DM is the dimensionality of the Gaussian, and we assume a prior of the form $p(\mathbf{w}) = \mathcal{N}(\mathbf{m}_0, \mathbf{V}_0)$, where typically $\boldsymbol{\mu}_0 = \mathbf{0}_{DM}$, and $\mathbf{V}_0$ is block diagonal. The second term is simply

$$\sum_{i=1}^{N} \mathbf{y}_i^T \mathbb{E}_q\left[\boldsymbol{\eta}_i\right] = \sum_{i=1}^{N} \mathbf{y}_i^T \tilde{\mathbf{m}}_i \tag{21.172}$$

where $\tilde{\mathbf{m}}_i \triangleq \mathbf{X}_i \mathbf{m}_N$. The final term can be lower bounded by taking expectations of our quadratic upper bound on lse as follows:

$$-\sum_{i=1}^{N} \mathbb{E}_q\left[\text{lse}(\boldsymbol{\eta}_i)\right] \geq -\frac{1}{2}\text{tr}(\mathbf{A}_i \tilde{\mathbf{V}}_i) - \frac{1}{2}\tilde{\mathbf{m}}_i \mathbf{A}_i \tilde{\mathbf{m}}_i + \mathbf{b}_i^T \tilde{\mathbf{m}}_i - c_i \tag{21.173}$$

where $\tilde{\mathbf{V}}_i \triangleq \mathbf{X}_i \mathbf{V}_N \mathbf{X}_i^T$. Putting it altogether, we have

$$\begin{aligned} L_{QJ}(q) &\geq -\frac{1}{2}\left[\text{tr}(\mathbf{V}_N \mathbf{V}_0^{-1}) - \log|\mathbf{V}_N \mathbf{V}_0^{-1}| + (\mathbf{m}_N - \mathbf{m}_0)^T \mathbf{V}_0^{-1}(\mathbf{m}_N - \mathbf{m}_0)\right]\\ &\quad -\frac{1}{2}DM + \sum_{i=1}^{N} \mathbf{y}_i^T \tilde{\mathbf{m}}_i - \frac{1}{2}\text{tr}(\mathbf{A}_i \tilde{\mathbf{V}}_i) - \frac{1}{2}\tilde{\mathbf{m}}_i \mathbf{A}_i \tilde{\mathbf{m}}_i + \mathbf{b}_i^T \tilde{\mathbf{m}}_i - c_i && (21.174) \end{aligned}$$

This lower bound combines Jensen's inequality (as in mean field inference), plus the quadratic lower bound due to the lse term, so we write it as L_{QJ}.

We will use coordinate ascent to optimize this lower bound. That is, we update the variational posterior parameters $\mathbf{V}_N$ and $\mathbf{m}_N$, and then the variational likelihood parameters $\boldsymbol{\psi}_i$. We leave the detailed derivation as an exercise, and just state the results. We have

$$\mathbf{V}_N = \left(\mathbf{V}_0 + \sum_{i=1}^{N} \mathbf{X}_i^T \mathbf{A}_i \mathbf{X}_i\right)^{-1} \tag{21.175}$$

$$\mathbf{m}_N = \mathbf{V}_n \left(\mathbf{V}_0^{-1}\mathbf{m}_0 + \sum_{i=1}^{N} \mathbf{X}_i^T (\mathbf{y}_i + \mathbf{b}_i)\right) \tag{21.176}$$

$$\boldsymbol{\psi}_i = \tilde{\mathbf{m}}_i = \mathbf{X}_i \mathbf{m}_N \tag{21.177}$$

We can exploit the fact that $\mathbf{A}_i$ is a constant matrix, plus the fact that $\mathbf{X}_i$ has block structure, to simplify the first two terms as follows:

$$\mathbf{V}_N = \left(\mathbf{V}_0 + \mathbf{A} \otimes \sum_{i=1}^{N} \mathbf{x}_i \mathbf{x}_i^T\right)^{-1} \tag{21.178}$$

$$\mathbf{m}_N = \mathbf{V}_n \left(\mathbf{V}_0^{-1}\mathbf{m}_0 + \sum_{i=1}^{N} (\mathbf{y}_i + \mathbf{b}_i) \otimes \mathbf{x}_i\right) \tag{21.179}$$

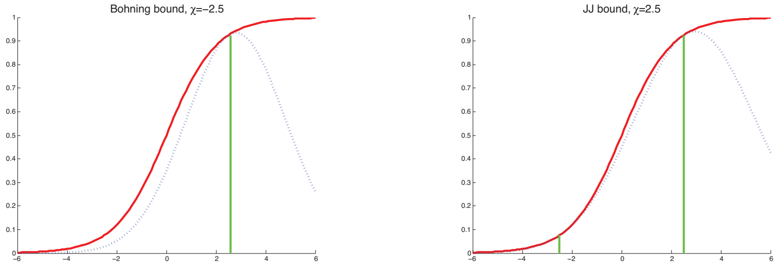
where $\otimes$ denotes the kronecker product. See Algorithm 15 for some pseudocode, and `http://www.cs.ubc.ca/~emtiyaz/software/catLGM.html` for some Matlab code.

Algorithm 21.1: Variational inference for multi-class logistic regression using Bohning's bound

```
Input: $y_i \in \{1, \ldots, C\}$, $\mathbf{x}_i \in \mathbb{R}^D$, $i = 1 : N$, prior $\mathbf{m}_0$, $\mathbf{V}_0$ ;
Define $M := C - 1$; dummy encode $\mathbf{y}_i \in \{0, 1\}^M$; define $\mathbf{X}_i = \text{blockdiag}(\mathbf{x}_i^T)$ ;
Define $\mathbf{y} := [\mathbf{y}_1; \ldots; \mathbf{y}_N]$, $\mathbf{X} := [\mathbf{X}_1; \ldots; \mathbf{X}_N]$ and $\mathbf{A} := \frac{1}{2}\left[\mathbf{I}_M - \frac{1}{M+1}\mathbf{1}_M \mathbf{1}_M^T\right]$;
$\mathbf{V}_N := \left(\mathbf{V}_0^{-1} + \sum_{i=1}^{n} \mathbf{X}_i^T \mathbf{A} \mathbf{X}_i\right)^{-1}$;
Initialize $\mathbf{m}_N := \mathbf{m}_0$;
repeat
    $\boldsymbol{\psi} := \mathbf{X}\mathbf{m}_N$;
    $\boldsymbol{\Psi} := \text{reshape}(\mathbf{m}, M, N)$;
    $\mathbf{G} := \exp(\boldsymbol{\Psi} - \text{lse}(\boldsymbol{\Psi}))$;
    $\mathbf{B} := \mathbf{A}\boldsymbol{\Psi} - \mathbf{G}$;
    $\mathbf{b} := (\mathbf{B})$ ;
    $\mathbf{m}_N := \mathbf{V}_N \left(\mathbf{V}_0^{-1}\mathbf{m}_0 + \mathbf{X}^T(\mathbf{y} + \mathbf{b})\right)$;
    Compute the lower bound $L_{QJ}$ using Equation 21.174;
until converged;
Return $\mathbf{m}_N$ and $\mathbf{V}_N$;
```

21.8.3 Bounds for the sigmoid function

In many models, we just have binary data. In this case, we have $y_i \in \{0, 1\}$, $M = 1$ and $\eta_i = \mathbf{w}^T \mathbf{x}_i$ where $\mathbf{w} \in \mathbb{R}^D$ is a weight vector (not matrix). In this case, the Bohning bound



(a) (b)

Figure 21.10 Quadratic lower bounds on the sigmoid (logistic) function. In solid red, we plot $\text{sigm}(x)$ vs x. In dotted blue, we plot the lower bound $L(x, \xi)$ vs x for $\xi = 2.5$. (a) Bohning bound. This is tight at $-\xi = 2.5$. (b) JJ bound. This is tight at $\xi = \pm 2.5$. Figure generated by `sigmoidLowerBounds`.

becomes

$$\begin{aligned}
\log(1 + e^{\eta}) &\leq \frac{1}{2} a\eta^2 - b\eta + c && (21.180)\\
a &= \frac{1}{4} && (21.181)\\
b &= A\psi - (1 + e^{-\psi})^{-1} && (21.182)\\
c &= \frac{1}{2} A\psi^2 - (1 + e^{-\psi})^{-1}\psi + \log(1 + e^{\psi}) && (21.183)
\end{aligned}$$

It is possible to derive an alternative quadratic bound for this case, as shown in (Jaakkola and Jordan 1996b, 2000). This has the following form

$$\begin{aligned}
\log(1 + e^{\eta}) &\leq \lambda(\xi)(\eta^2 - \xi^2) + \frac{1}{2}(\eta - \xi) + \log(1 + e^{\xi}) && (21.184)\\
\lambda(\xi) &\triangleq \frac{1}{4\xi} \tanh(\xi/2) = \frac{1}{2\xi}\left[\text{sigm}(\xi) - \frac{1}{2}\right] && (21.185)
\end{aligned}$$

We shall refer to this as the **JJ bound**, after its inventors, (Jaakkola and Jordan 1996b, 2000).

To facilitate comparison with Bohning's bound, let us rewrite the JJ bound as a quadratic form as follows

$$\begin{aligned}
\log(1 + e^{\eta}) &\leq \frac{1}{2} a(\xi)\eta^2 - b(\xi)\eta + c(\xi) && (21.186)\\
a(\xi) &= 2\lambda(\xi) && (21.187)\\
b(\xi) &= -\frac{1}{2} && (21.188)\\
c(\xi) &= -\lambda(\xi)\xi^2 - \frac{1}{2}\xi + \log(1 + e^{\xi}) && (21.189)
\end{aligned}$$

The JJ bound has an adaptive curvature term, since a depends on ξ. In addition, it is tight at two points, as is evident from Figure 21.10(b). By contrast, the Bohning bound is a constant curvature bound, and is only tight at one point, as is evident from Figure 21.10(a).

If we wish to use the JJ bound for binary logistic regression, we can make some small modifications to Algorithm 15. First, we use the new definitions for a_i, b_i and c_i. The fact that a_i is not constant when using the JJ bound, unlike when using the Bohning bound, means we cannot compute $\mathbf{V}_N$ outside of the main loop, making the method a constant factor slower. Next we note that $\mathbf{X}_i = \mathbf{x}_i^T$, so the updates for the posterior become

$$\mathbf{V}_N^{-1} = \mathbf{V}_0^{-1} + 2\sum_{i=1}^N \lambda(\xi_i)\mathbf{x}_i\mathbf{x}_i^T \tag{21.190}$$

$$\mathbf{m}_N = \mathbf{V}_N \left(\mathbf{V}_0^{-1}\mathbf{m}_0 + \sum_{i=1}^N (y_i - \frac{1}{2})\mathbf{x}_i \right) \tag{21.191}$$

Finally, to compute the update for ξ_i, we isolate the terms in L_{QJ} that depend on ξ_i to get

$$L(\boldsymbol{\xi}) = \sum_{i=1}^N \left\{ \ln \mathrm{sigm}(\xi_i) - \xi_i/2 - \lambda(\xi_i)(\mathbf{x}_i^T \mathbb{E}_q \left[\mathbf{w}\mathbf{w}^T \right] \mathbf{x}_i - \xi_i^2) \right\} + \mathrm{const} \tag{21.192}$$

Optimizing this wrt ξ_i gives the equation

$$0 = \lambda'(\xi_i)(\mathbf{x}_i^T \mathbb{E}_q \left[\mathbf{w}\mathbf{w}^T \right] \mathbf{x}_i - \xi_i^2) \tag{21.193}$$

Now $\lambda'(\xi_i)$ is monotonic for $\xi_i \geq 0$, and we do not need to consider negative values of ξ_i by symmetry of the bound around $\xi_i = 0$ (see Figure 21.10). Hence the only way to make the above expression 0 is if we have $(\mathbf{x}_i^T \mathbb{E} \left[\mathbf{w}\mathbf{w}^T \right] \mathbf{x}_i - \xi_i^2) = 0$. Hence the update becomes

$$(\xi_i^{new})^2 = \mathbf{x}_i^T (\mathbf{V}_N + \mathbf{m}_N \mathbf{m}_N^T)\mathbf{x}_i \tag{21.194}$$

Although the JJ bound is tighter than the Bohning bound, sometimes it is not tight enough in order to estimate the posterior covariance accurately. A more accurate approach, which uses a piecewise quadratic upper bound to lse, is described in (Marlin et al. 2011). By increasing the number of pieces, the bound can be made arbitrarily tight.

21.8.4 Other bounds and approximations to the log-sum-exp function *

There are several other bounds and approximations to the multiclass lse function which we can use, which we briefly summarize below. Note, however, that all of these require numerical optimization methods to compute $\mathbf{m}_N$ and $\mathbf{V}_N$, making them more complicated to implement.

21.8.4.1 Product of sigmoids

The approach in (Bouchard 2007) exploits the fact that

$$\log \left(\sum_{k=1}^K e^{\eta_k} \right) \leq \alpha + \sum_{k=1}^K \log(1 + e^{\eta_k - \alpha}) \tag{21.195}$$

It then applies the JJ bound to the term on the right.

21.8.4.2 Jensen's inequality

The approach in (Blei and Lafferty 2006a, 2007) uses Jensen's inequality as follows:

$$\begin{aligned}\mathbb{E}_q\left[\text{lse}(\boldsymbol{\eta}_i)\right] &= \mathbb{E}_q\left[\log\left(1+\sum_{c=1}^{M}\exp(\mathbf{x}_i^T\mathbf{w}_c)\right)\right] && (21.196)\\ &\leq \log\left(1+\sum_{c=1}^{M}\mathbb{E}_q\left[\exp(\mathbf{x}_i^T\mathbf{w}_c)\right]\right) && (21.197)\\ &\leq \log\left(1+\sum_{c=1}^{M}\exp(\mathbf{x}_i^T\mathbf{m}_{N,c}+\frac{1}{2}\mathbf{x}_i^T\mathbf{V}_{N,cc}\mathbf{x}_i)\right) && (21.198)\end{aligned}$$

where the last term follows from the mean of a log-normal distribution, which is $e^{\mu+\sigma^2/2}$.

21.8.4.3 Multivariate delta method

The approach in (Ahmed and Xing 2007; Braun and McAuliffe 2010) uses the **multivariate delta method**, which is a way to approximate moments of a function using a Taylor series expansion. In more detail, let $f(\mathbf{w})$ be the function of interest. Using a second-order approximation around $\mathbf{m}$ we have

$$f(\mathbf{w}) \approx f(\mathbf{m})+(\mathbf{w}-\mathbf{m})^T\mathbf{g}(\mathbf{w}-\mathbf{m})+\frac{1}{2}(\mathbf{w}-\mathbf{m})^T\mathbf{H}(\mathbf{w}-\mathbf{m}) \quad (21.199)$$

where $\mathbf{g}$ and $\mathbf{H}$ are the gradient and Hessian evaluated at $\mathbf{m}$. If $q(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{m},\mathbf{V})$, we have

$$\mathbb{E}_q\left[f(\mathbf{w})\right] \approx f(\mathbf{m})+\frac{1}{2}\text{tr}[\mathbf{H}\mathbf{V}] \quad (21.200)$$

If we use $f(\mathbf{w}) = \text{lse}(\mathbf{X}_i\mathbf{w})$, we get

$$\mathbb{E}_q\left[\text{lse}(\mathbf{X}_i\mathbf{w})\right] \approx \text{lse}(\mathbf{X}_i\mathbf{m})+\frac{1}{2}\text{tr}[\mathbf{X}_i\mathbf{H}\mathbf{X}_i^T\mathbf{V}] \quad (21.201)$$

where $\mathbf{g}$ and $\mathbf{H}$ for the lse function are defined in Equations 21.159 and 21.160.

21.8.5 Variational inference based on upper bounds

So far, we have been concentrating on lower bounds. However, sometimes we need to use an upper bound. For example, (Saul et al. 1996) derives a mean field algorithm for sigmoid belief nets, which are DGMs in which each CPD is a logistic regression function (Neal 1992). Unlike the case of Ising models, the resulting MRF is not pairwise, but contains higher order interactions. This makes the standard mean field updates intractable. In particular, they turn out to involve computing an expression which requires evaluating

$$\mathbb{E}\left[\log(1+e^{-\sum_{j\in\text{pa}_i}w_{ij}x_j})\right] = \mathbb{E}\left[-\log\text{sigm}(\mathbf{w}_i^T\mathbf{x}_{\text{pa}(i)})\right] \quad (21.202)$$

(Notice the minus sign in front.) (Saul et al. 1996) show how to derive an upper bound on the sigmoid function so as to make this update tractable, resulting in a monotonically convergent inference procedure.

Exercises

Exercise 21.1 Laplace approximation to $p(\mu, \log\sigma|\mathcal{D})$ for a univariate Gaussian

Compute a Laplace approximation of $p(\mu, \log\sigma|\mathcal{D})$ for a Gaussian, using an uninformative prior $p(\mu, \log\sigma) \propto 1$.

Exercise 21.2 Laplace approximation to normal-gamma

Consider estimating μ and $\ell = \log\sigma$ for a Gaussian using an uniformative normal-Gamma prior. The log posterior is

$$\log p(\mu, \ell|\mathcal{D}) = -n\log\sigma - \frac{1}{2\sigma^2}[ns^2 + n(\overline{y}-\mu)^2] \tag{21.203}$$

a. Show that the first derivatives are

$$\frac{\partial}{\partial\mu}\log p(\mu,\ell|\mathcal{D}) = \frac{n(\overline{y}-\mu)}{\sigma^2} \tag{21.204}$$

$$\frac{\partial}{\partial\ell}\log p(\mu,\ell|\mathcal{D}) = -n + \frac{ns^2 + n(\overline{y}-\mu)^2}{\sigma^2} \tag{21.205}$$

b. Show that the Hessian matrix is given by

$$\mathbf{H} = \begin{pmatrix} \frac{\partial^2}{\partial\mu^2}\log p(\mu,\ell|\mathcal{D}) & \frac{\partial^2}{\partial\mu\partial\ell}\log p(\mu,\ell|\mathcal{D}) \\ \frac{\partial^2}{\partial\ell^2}\log p(\mu,\ell|\mathcal{D}) & \frac{\partial^2}{\partial\ell^2}\log p(\mu,\ell|\mathcal{D}) \end{pmatrix} \tag{21.206}$$

$$= \begin{pmatrix} -\frac{n}{\sigma^2} & -2n\frac{\overline{y}-\mu}{\sigma^2} \\ -2n\frac{\overline{y}-\mu}{\sigma^2} & -\frac{2}{\sigma^2}(ns^2 + n(\overline{y}-\mu)^2) \end{pmatrix} \tag{21.207}$$

c. Use this to derive a Laplace approximation to the posterior $p(\mu, \ell|\mathcal{D})$.

Exercise 21.3 Variational lower bound for VB for univariate Gaussian

Fill in the details of the derivation in Section 21.5.1.6.

Exercise 21.4 Variational lower bound for VB for GMMs

Consider VBEM for GMMs as in Section 21.6.1.4. Show that the lower bound has the following form

$$\begin{aligned} \mathcal{L} = {} & \mathbb{E}\left[\ln p(\mathbf{x}|\mathbf{z}, \boldsymbol{\mu}, \boldsymbol{\Lambda})\right] + \mathbb{E}\left[\ln p(\mathbf{z}|\boldsymbol{\pi})\right] + \mathbb{E}\left[\ln p(\boldsymbol{\pi})\right] + \mathbb{E}\left[\ln p(\boldsymbol{\mu}, \boldsymbol{\Lambda})\right] \\ & -\mathbb{E}\left[\ln q(\mathbf{z})\right] - \mathbb{E}\left[\ln q(\boldsymbol{\pi})\right] - \mathbb{E}\left[\ln q(\boldsymbol{\mu}, \boldsymbol{\Lambda})\right] \end{aligned} \tag{21.208}$$

where

$$\begin{aligned}
\mathbb{E}\left[\ln p(\mathbf{x}|\mathbf{z}, \boldsymbol{\mu}, \boldsymbol{\Lambda})\right] &= \frac{1}{2}\sum_k N_k \Big\{\ln \tilde{\Lambda}_k - D\beta_k^{-1} - \nu_k \text{tr}(\mathbf{S}_k \mathbf{L}_k) \\
&\quad -\nu_k(\overline{\mathbf{x}}_k - \mathbf{m}_k)^T \mathbf{L}_k(\overline{\mathbf{x}}_k - \mathbf{m}_k) - D\ln(2\pi)\Big\} && (21.209) \\
\mathbb{E}\left[\ln p(\mathbf{z}|\boldsymbol{\pi})\right] &= \sum_i \sum_k r_{ik} \ln \tilde{\pi}_k && (21.210) \\
\mathbb{E}\left[\ln p(\boldsymbol{\pi})\right] &= \ln C_{dir}(\boldsymbol{\alpha}_0) + (\alpha_0 - 1)\sum_k \ln \tilde{\pi}_k && (21.211) \\
\mathbb{E}\left[\ln p(\boldsymbol{\mu}, \boldsymbol{\Lambda})\right] &= \frac{1}{2}\sum_k \Big\{D\ln(\beta_0/2\pi) + \ln \tilde{\Lambda}_k - \frac{D\beta_0}{\beta_k} \\
&\quad -\beta_0 \nu_k(\mathbf{m}_k - \mathbf{m}_0)^T \mathbf{L}_k(\mathbf{m}_k - \mathbf{m}_0) \\
&\quad +\ln C_{Wi}(\mathbf{L}_0, \nu_0) + \frac{\nu_0 - D - 1}{2}\ln \tilde{\Lambda}_k - \frac{1}{2}\nu_k \text{tr}(\mathbf{L}_0^{-1}\mathbf{L}_k)\Big\} && (21.212) \\
\mathbb{E}\left[\ln q(\mathbf{z})\right] &= \sum_i \sum_k r_{ik} \ln r_{ik} && (21.213) \\
\mathbb{E}\left[\ln q(\boldsymbol{\pi})\right] &= \sum_k (\alpha_k - 1)\ln \tilde{\pi}_k + \ln C_{dir}(\boldsymbol{\alpha}) && (21.214) \\
\mathbb{E}\left[\ln q(\boldsymbol{\mu}, \boldsymbol{\Lambda})\right] &= \sum_k \left\{\frac{1}{2}\ln \tilde{\Lambda}_k + \frac{D}{2}\ln\left(\frac{\beta_k}{2\pi}\right) - \frac{D}{2} - \mathbb{H}\left(q(\boldsymbol{\Lambda}_k)\right)\right\} && (21.215)
\end{aligned}$$

where the normalization constant for the Dirichlet and Wishart is given by

$$\begin{aligned}
C_{dir}(\boldsymbol{\alpha}) &\triangleq \frac{\Gamma(\sum_k \alpha_k)}{\prod_k \Gamma(\alpha_k)} && (21.216) \\
C_{Wi}(\mathbf{L}, \nu) &\triangleq |\mathbf{L}|^{-\nu/2}\left(2^{\nu D/2}\Gamma_D(\nu/2)\right)^{-1} && (21.217) \\
\Gamma_D(\alpha) &\triangleq \pi^{D(D-1)/4}\prod_{j=1}^{D}\Gamma\left(\alpha + (1-j)/2\right) && (21.218)
\end{aligned}$$

where $\Gamma_D(\nu)$ is the multivariate Gamma function. Finally, the entropy of the Wishart is given by

$$\mathbb{H}\left(\text{Wi}(\mathbf{L}, \nu)\right) = -\ln C_{Wi}(\mathbf{L}, \nu) - \frac{\nu - D - 1}{2}\mathbb{E}\left[\ln |\boldsymbol{\Lambda}|\right] + \frac{\nu D}{2} \quad (21.219)$$

where $\mathbb{E}\left[\ln |\boldsymbol{\Lambda}|\right]$ is given in Equation 21.131.

Exercise 21.5 Derivation of $\mathbb{E}\left[\log \pi_k\right]$ under a Dirichlet distribution
Show that

$$\exp(\mathbb{E}\left[\log \pi_k\right]) = \frac{\exp(\Psi(\alpha_k))}{\exp(\Psi(\sum_{k'} \alpha_{k'}))} \quad (21.220)$$

where $\boldsymbol{\pi} \sim \text{Dir}(\boldsymbol{\alpha})$.

Exercise 21.6 Alternative derivation of the mean field updates for the Ising model
Derive Equation 21.50 by directly optimizing the variational free energy one term at a time.

Exercise 21.7 Forwards vs reverse KL divergence

(Source: Exercise 33.7 of (MacKay 2003).) Consider a factored approximation $q(x, y) = q(x)q(y)$ to a joint distribution $p(x, y)$. Show that to minimize the forwards KL $\mathbb{KL}\left(p||q\right)$ we should set $q(x) = p(x)$ and $q(y) = p(y)$, i.e., the optimal approximation is a product of marginals

Now consider the following joint distribution, where the rows represent y and the columns x.

	x			
	1	2	3	4
1	1/8	1/8	0	0
2	1/8	1/8	0	0
3	0	0	1/4	0
4	0	0	0	1/4

Show that the reverse KL $\mathbb{KL}\left(q||p\right)$ for this p has three distinct minima. Identify those minima and evaluate $\mathbb{KL}\left(q||p\right)$ at each of them. What is the value of $\mathbb{KL}\left(q||p\right)$ if we set $q(x, y) = p(x)p(y)$?

Exercise 21.8 Derivation of the structured mean field updates for FHMM

Derive the updates in Section 21.4.1.

Exercise 21.9 Variational EM for binary FA with sigmoid link

Consider the binary FA model:

$$p(\mathbf{x}_i|\mathbf{z}_i, \boldsymbol{\theta}) = \prod_{j=1}^{D} \text{Ber}(x_{ij}|\text{sigm}(\mathbf{w}_j^T \mathbf{z}_i + \beta_j)) = \prod_{j=1}^{D} \text{Ber}(x_{ij}|\text{sigm}(\eta_{ij})) \tag{21.221}$$

$$\boldsymbol{\eta}_i = \tilde{\mathbf{W}} \tilde{\mathbf{z}}_i \tag{21.222}$$

$$\tilde{\mathbf{z}}_i \triangleq (\mathbf{z}_i; 1) \tag{21.223}$$

$$\tilde{\mathbf{W}} \triangleq (\mathbf{W}, \boldsymbol{\beta}) \tag{21.224}$$

$$p(\mathbf{z}_i) = \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{21.225}$$

Derive an EM algorithm to fit this model, using the Jaakkola-Jordan bound. Hint: the answer is in (Tipping 1998), but the exercise asks you to derive these equations.

Exercise 21.10 VB for binary FA with probit link

In Section 11.4.6, we showed how to use EM to fit probit regression, using a model of the form $p(y_i = 1|z_i) = \mathbb{I}(z_i > 0)$, where $z_i \sim \mathcal{N}(\mathbf{w}^T \mathbf{x}_i, 1)$ is latent. Now consider the case where the inputs $\mathbf{x}_i$ are also unknown, as in binary factor analysis. Show how to fit this model using variational Bayes, making an approximation to the posterior of the form $q(\mathbf{x}, \mathbf{z}, \mathbf{W}) = \prod_{i=1}^{N} q(\mathbf{x}_i) q(z_i) \prod_{l=1}^{L} q(\mathbf{w}_l)$. Hint: $q(\mathbf{x}_i)$ and $q(\mathbf{w}_i)$ will be Gaussian, and $q(z_i)$ will be a truncated univariate Gaussian.