15 Discrete Latent Variables

We have seen how complex distributions can be constructed by combining multiple simple distributions and how the resulting models can be described by directed graphs. In addition to the observed variables, which form part of the data set, such models often introduce additional hidden, or latent, variables. These might correspond to specific quantities involved in the data generation process, such as the unknown orientation of an object in three-dimensional space in the case of images, or they may be introduced simply as modelling constructs to allow much richer models to be created. If we define a joint distribution over observed and latent variables, the corresponding distribution of the observed variables alone is obtained by marginalization. This allows relatively complex marginal distributions over observed variables to be expressed in terms of more tractable joint distributions over the expanded space of observed and latent variables.

Chapter 11

In this chapter, we will see that marginalizing over discrete latent variables gives rise to mixture distributions. Our focus will be on mixtures of Gaussians that provide a good illustration of mixture distributions and that are also widely used in machine learning. One simple application for mixture models is to discover clusters in data, and we begin our discussion by considering a technique for clustering called the K-means algorithm, which corresponds to a particular non-probabilistic limit of Gaussian mixtures. Then we introduce the latent-variable view of mixture distributions in which the discrete latent variables can be interpreted as defining assignments of data points to specific components of the mixture.

A general technique for finding maximum likelihood estimators in latent-variable models is the expectation–maximization (EM) algorithm. We first use the Gaussian mixture distribution to motivate the EM algorithm in an informal way, and then we give a more careful treatment based on the latent-variable viewpoint. Finally we provide a general perspective by introducing the evidence lower bound (ELBO), which will play an important role in generative models such as variational autoencoders and diffusion models.

15.1. K-means Clustering

We begin by considering the problem of identifying groups, or clusters, of data points in a multi-dimensional space. Suppose we have a data set $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ consisting of N observations of a D-dimensional Euclidean variable $\mathbf{x}$. Our goal is to partition the data set into some number K of clusters, where we will suppose for the moment that the value of K is given. Intuitively, we might think of a cluster as comprising a group of data points whose inter-point distances are small compared with the distances to points outside the cluster. We can formalize this notion by first introducing a set of D-dimensional vectors $\boldsymbol{\mu}_k$, where $k = 1, \ldots, K$, in which $\boldsymbol{\mu}_k$ is a 'prototype' associated with the kth cluster. As we will see shortly, we can think of the $\boldsymbol{\mu}_k$ as representing the centres of the clusters. Our goal is then to find a set of cluster vectors $\{\boldsymbol{\mu}_k\}$, along with an assignment of data points to clusters, such that the sum of the squares of the distances of each data point to its closest cluster vector $\boldsymbol{\mu}_k$ is a minimum.

It is convenient at this point to define some notation to describe the assignment of data points to clusters. For each data point $\mathbf{x}_n$, we introduce a corresponding set of binary indicator variables $r_{nk} \in \{0, 1\}$, where $k = 1, \ldots, K$. These indicators describe which of the K clusters the data point $\mathbf{x}_n$ is assigned to, so that if data point $\mathbf{x}_n$ is assigned to cluster k then $r_{nk} = 1$, and $r_{nj} = 0$ for $j \neq k$. This is an example of the 1-of-K coding scheme. We can then define an error function:

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2, \tag{15.1}$$

which represents the sum of the squares of the distances of each data point to its assigned vector $\boldsymbol{\mu}_k$. Our goal is to find values for the $\{r_{nk}\}$ and the $\{\boldsymbol{\mu}_k\}$ so as to minimize J. We can do this through an iterative procedure in which each iteration involves two successive steps corresponding to successive optimizations with respect to the $\{r_{nk}\}$ and the $\{\boldsymbol{\mu}_k\}$. First we choose some initial values for the $\{\boldsymbol{\mu}_k\}$. Then in the first step, we minimize J with respect to the $\{r_{nk}\}$, keeping the $\{\boldsymbol{\mu}_k\}$ fixed. In the second step, we minimize J with respect to the $\{\boldsymbol{\mu}_k\}$, keeping $\{r_{nk}\}$ fixed. This two-step optimization is then repeated until convergence. We will see that these two stages of updating $\{r_{nk}\}$ and updating $\{\boldsymbol{\mu}_k\}$ correspond, respectively, to the E (expectation) and M (maximization) steps of the EM algorithm, and to emphasize this, we will use the terms E step and M step in the context of the K-means algorithm.

Section 15.3

Consider first the determination of the $\{r_{nk}\}$ with the $\{\boldsymbol{\mu}_k\}$ held fixed (the E step). Because J in (15.1) is a linear function of the $\{r_{nk}\}$, this optimization can be performed easily to give a closed-form solution. The terms involving different n are independent, and so we can optimize for each n separately by choosing r_{nk} to be 1 for whichever value of k gives the minimum value of $\|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$. In other words, we simply assign the nth data point to the closest cluster centre. More formally, this can be expressed as

$$r_{nk} = \begin{cases} 1, & \text{if } k = \arg\min_j \|\mathbf{x}_n - \boldsymbol{\mu}_j\|^2, \\ 0, & \text{otherwise.} \end{cases} \tag{15.2}$$

Now consider the optimization of the $\{\boldsymbol{\mu}_k\}$ with the $\{r_{nk}\}$ held fixed (the M step). The objective function J is a quadratic function of $\boldsymbol{\mu}_k$, and it can be minimized by setting its derivative with respect to $\boldsymbol{\mu}_k$ to zero giving

$$2\sum_{n=1}^{N} r_{nk}(\mathbf{x}_n - \boldsymbol{\mu}_k) = 0, \tag{15.3}$$

which we can easily solve for $\boldsymbol{\mu}_k$ to give

$$\boldsymbol{\mu}_k = \frac{\sum_n r_{nk}\mathbf{x}_n}{\sum_n r_{nk}}. \tag{15.4}$$

The denominator in this expression is equal to the number of points assigned to cluster k, and so this result has a simple interpretation, namely that $\boldsymbol{\mu}_k$ is equal to the mean of all the data points $\mathbf{x}_n$ assigned to cluster k. For this reason, the procedure is known as the *K-means* algorithm (Lloyd, 1982). It is summarized in Algorithm 15.1. Because the assignments $\{r_{nk}\}$ are discrete and each iteration will not lead to an increase in the error function, the K-means algorithm is guaranteed to converge in a finite number of steps.

Exercise 15.1

The two phases of reassigning data points to clusters and recomputing the cluster means are repeated in turn until there is no further change in the assignments (or until some maximum number of iterations is exceeded). However, this approach may converge to a local rather than a global minimum of J. The convergence properties of the K-means algorithm were studied by MacQueen (1967).

The K-means algorithm is illustrated in Figure 15.1 using data derived from eruptions of the Old Faithful geyser in Yellowstone National Park. The data set consists of 272 data points, each of which gives the duration of an eruption on the

Section 3.2.9

Algorithm 15.1: K-means algorithm

```
Input: Initial prototype vectors $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$
       Data set $\mathbf{x}_1, \ldots, \mathbf{x}_N$
Output: Final prototype vectors $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$
---
$\{r_{nk} \leftarrow 0\}$ // Initially set all assignments to zero
repeat
    $\{r_{nk}^{(\text{old})}\} \leftarrow \{r_{nk}\}$
    // Update assignments
    for $N \in \{1, \ldots, N\}$ do
        $k \leftarrow \arg\min_j \|\mathbf{x}_n - \boldsymbol{\mu}_j\|^2$
        $r_{nk} \leftarrow 1$
        $r_{nj} \leftarrow 0, \quad j \in \{1, \ldots, K\}, \ j \neq k$
    end for
    // Update prototype vectors
    for $k \in \{1, \ldots, K\}$ do
        $\boldsymbol{\mu}_k \leftarrow \sum_n r_{nk}\mathbf{x}_n / \sum_n r_{nk}$
    end for
until $\{r_{nk}\} = \{r_{nk}^{(\text{old})}\}$ // Assignments unchanged
return $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K, \{r_{nk}\}$
```

horizontal axis and the time to the next eruption on the vertical axis. Here we have made a linear re-scaling of the data, known as *standardizing*, such that each of the variables has zero mean and unit standard deviation.

For this example, we have chosen $K = 2$ and so the assignment of each data point to the nearest cluster centre is equivalent to a classification of the data points according to which side they lie of the perpendicular bisector of the two cluster centres. A plot of the cost function J given by (15.1) for the Old Faithful example is shown in Figure 15.2. Note that we have deliberately chosen poor initial values for the cluster centres so that the algorithm takes several steps before convergence. In practice, a better initialization procedure would be to choose the cluster centres $\boldsymbol{\mu}_k$ to be equal to a random subset of K data points. Also note that the K-means algorithm is often used to initialize the parameters in a Gaussian mixture model before applying the EM algorithm. *Section 15.2.2*

So far, we have considered a batch version of K-means in which the whole data set is used together to update the prototype vectors. We can also derive a sequential update in which, for each data point $\mathbf{x}_n$ in turn, we update the nearest prototype $\boldsymbol{\mu}_k$ using *Exercise 15.2*

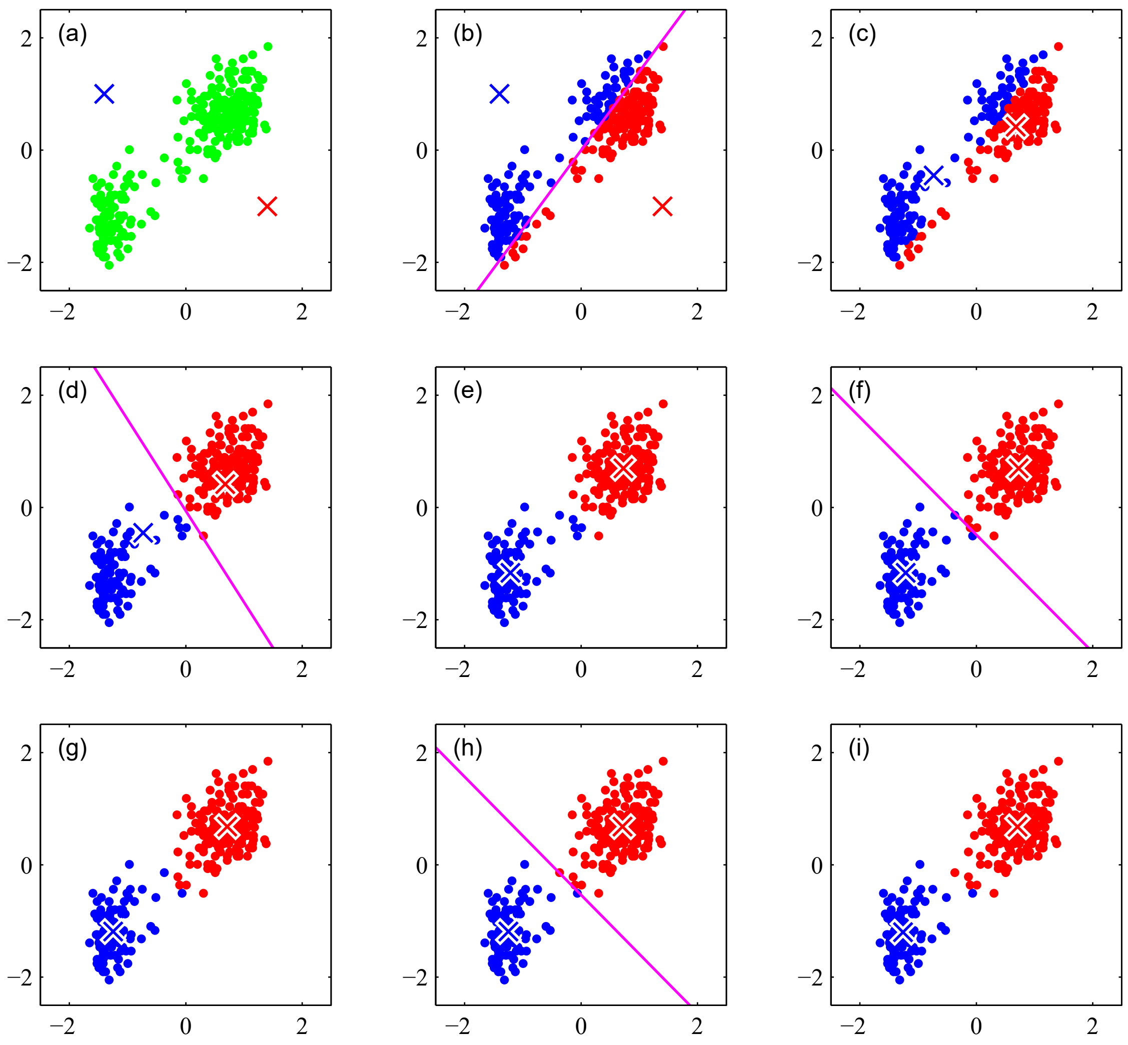


Figure 15.1 Illustration of the K-means algorithm using the re-scaled Old Faithful data set. (a) Green points denote the data set in a two-dimensional Euclidean space. The initial choices for centres $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ are shown by the red and blue crosses, respectively. (b) In the initial E step, each data point is assigned either to the red cluster or to the blue cluster, according to which cluster centre is nearer. This is equivalent to classifying the points according to which side of the perpendicular bisector of the two cluster centres, shown by the magenta line, they lie. (c) In the subsequent M step, each cluster centre is recomputed to be the mean of the points assigned to the corresponding cluster. (d)–(i) show successive E and M steps through to final convergence of the algorithm.

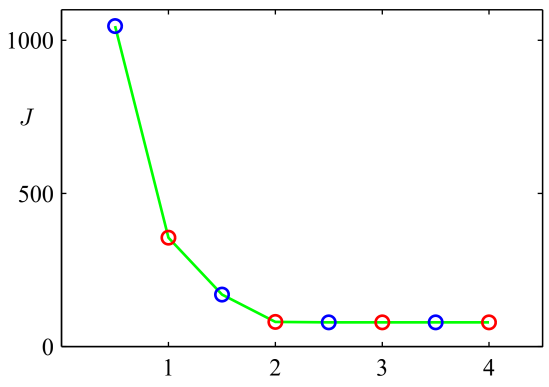


Figure 15.2 Plot of the cost function J given by (15.1) after each E step (blue points) and M step (red points) of the K-means algorithm for the example shown in Figure 15.1. The algorithm has converged after the third M step, and the final EM cycle produces no changes in either the assignments or the prototype vectors.

$$\boldsymbol{\mu}_k^{\text{new}} = \boldsymbol{\mu}_k^{\text{old}} + \frac{1}{N_k}(\mathbf{x}_n - \boldsymbol{\mu}_k^{\text{old}}) \tag{15.5}$$

where N_k is the number of data points that have so far been used to update $\boldsymbol{\mu}_k$. This allows each data point to be used once and then discarded before seeing the next data point.

One notable feature of the K-means algorithm is that at each iteration, every data point is assigned to one, and only one, of the clusters. Although some data points will be much closer to a particular centre $\boldsymbol{\mu}_k$ than to any other centre, there may be other data points that lie roughly midway between cluster centres. In the latter case, it is not clear that the hard assignment to the nearest cluster is the most appropriate. We will see that by adopting a probabilistic approach, we obtain 'soft' assignments of data points to clusters in a way that reflects the level of uncertainty over the most appropriate assignment. This probabilistic formulation has numerous benefits.

Section 15.2

15.1.1 Image segmentation

As an illustration of the application of the K-means algorithm, we consider the related problems of image segmentation and image compression. The goal of segmentation is to partition an image into regions such that each region has a reasonably homogeneous visual appearance or which corresponds to objects or parts of objects (Forsyth and Ponce, 2003). Each pixel in an image is a point in a three-dimensional space comprising the intensities of the red, blue, and green channels, and our segmentation algorithm simply treats each pixel in the image as a separate data point. Note that strictly this space is not Euclidean because the channel intensities are bounded by the interval $[0, 1]$. Nevertheless, we can apply the K-means algorithm without difficulty. We illustrate the result of running K-means to convergence, for any particular value of K, by redrawing the image in which we replace each pixel vector with the $\{R, G, B\}$ intensity triplet given by the centre $\boldsymbol{\mu}_k$ to which that pixel has been assigned. Results for various values of K are shown in Figure 15.3. We see that for a given value of K, the algorithm represents the image

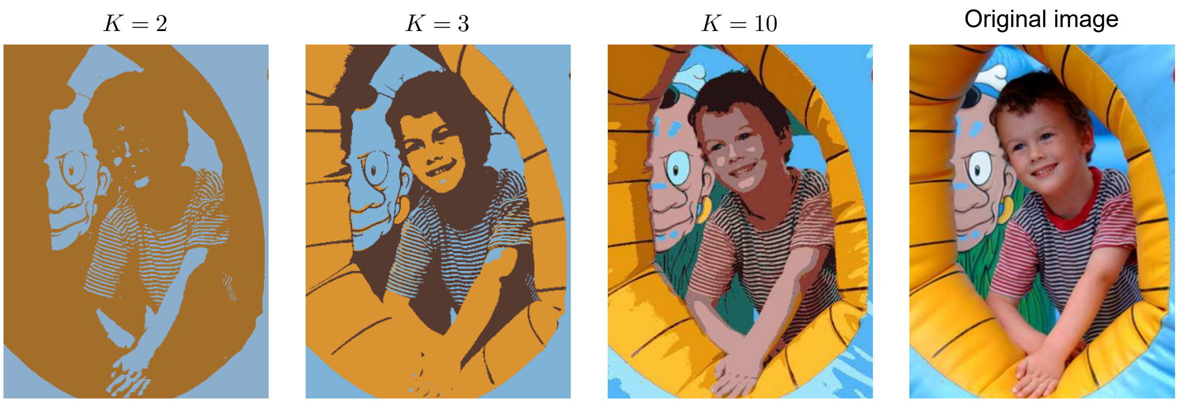


Figure 15.3 An example of the application of the K-means clustering algorithm to image segmentation showing an initial image together with their K-means segmentations obtained using various values of K. This also illustrates the use of vector quantization for data compression, in which smaller values of K give higher compression at the expense of poorer image quality.

using a palette of only K colours. It should be emphasized that this use of K-means is not a particularly sophisticated approach to image segmentation, not least because it takes no account of the spatial proximity of different pixels. Image segmentation is in general extremely difficult and remains the subject of active research and is introduced here simply to illustrate the behaviour of the K-means algorithm.

We can also use a clustering algorithm to perform data compression. It is important to distinguish between *lossless data compression*, in which the goal is to be able to reconstruct the original data exactly from the compressed representation, and *lossy data compression*, in which we accept some errors in the reconstruction in return for higher levels of compression than can be achieved in the lossless case. We can apply the K-means algorithm to the problem of lossy data compression as follows. For each of the N data points, we store only the identity k of the cluster to which it is assigned. We also store the values of the K cluster centres $\{\boldsymbol{\mu}_k\}$, which typically requires significantly less data, provided we choose $K \ll N$. Each data point is then approximated by its nearest centre $\boldsymbol{\mu}_k$. New data points can similarly be compressed by first finding the nearest $\boldsymbol{\mu}_k$ and then storing the label k instead of the original data vector. This framework is often called *vector quantization*, and the vectors $\{\boldsymbol{\mu}_k\}$ are called *codebook vectors*.

The image segmentation problem discussed above also provides an illustration of the use of clustering for data compression. Suppose the original image has N pixels comprising $\{R, G, B\}$ values, each of which is stored with 8 bits of precision. Directly transmitting the whole image would cost $24N$ bits. Now suppose we first run K-means on the image data, and then instead of transmitting the original pixel intensity vectors, we transmit the identity of the nearest vector $\boldsymbol{\mu}_k$. Because there are K such vectors, this requires $\log_2 K$ bits per pixel. We must also transmit the K code book vectors $\{\boldsymbol{\mu}_k\}$, which requires $24K$ bits, and so the total number of bits required to transmit the image is $24K + N \log_2 K$ (rounding up to the nearest integer). The original image shown in Figure 15.3 has $240 \times 180 = 43{,}200$ pixels and so requires $24 \times 43{,}200 = 1{,}036{,}800$ bits to transmit directly. By comparison, the compressed images require 43,248 bits ($K = 2$), 86,472 bits ($K = 3$), and 173,040 bits ($K = 10$), respectively, to transmit. These represent compression ratios compared to the original image of 4.2%, 8.3%, and 16.7%, respectively. We see that there is a trade-off between the degree of compression and image quality. Note that our aim in this example is to illustrate the K-means algorithm. If we had been aiming to produce a good image compressor, then it would be more fruitful to consider small blocks of adjacent pixels, for instance 5×5, and thereby exploit the correlations that exist in natural images between nearby pixels.

15.2. Mixtures of Gaussians

Section 3.2.9

We have previously motivated the Gaussian mixture model as a simple linear superposition of Gaussian components, aimed at providing a richer class of density models than a single Gaussian. We now turn to a formulation of Gaussian mixtures in terms of discrete latent variables. This will provide us with a deeper insight into this important distribution and will also serve to motivate the expectation–maximization algorithm.

Recall from (3.111) that the Gaussian mixture distribution can be written as a linear superposition of Gaussians in the form

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k). \tag{15.6}$$

Let us introduce a K-dimensional binary random variable $\mathbf{z}$ having a 1-of-K representation in which one of the elements is equal to 1 and all other elements are equal to 0. The values of z_k therefore satisfy $z_k \in \{0, 1\}$ and $\sum_k z_k = 1$, and we see that there are K possible states for the vector $\mathbf{z}$ according to which element is non-zero. We will define the joint distribution $p(\mathbf{x}, \mathbf{z})$ in terms of a marginal distribution $p(\mathbf{z})$ and a conditional distribution $p(\mathbf{x}|\mathbf{z})$. The marginal distribution over $\mathbf{z}$ is specified in terms of the mixing coefficients π_k, such that

$$p(z_k = 1) = \pi_k$$

where the parameters $\{\pi_k\}$ must satisfy

$$0 \leqslant \pi_k \leqslant 1 \tag{15.7}$$

together with

$$\sum_{k=1}^{K} \pi_k = 1 \tag{15.8}$$

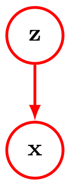


Figure 15.4 Graphical representation of a mixture model, in which the joint distribution is expressed in the form $p(\mathbf{x}, \mathbf{z}) = p(\mathbf{z})p(\mathbf{x}|\mathbf{z})$.

if they are to be valid probabilities. Because $\mathbf{z}$ uses a 1-of-K representation, we can also write this distribution in the form

$$p(\mathbf{z}) = \prod_{k=1}^{K} \pi_k^{z_k}. \tag{15.9}$$

Similarly, the conditional distribution of $\mathbf{x}$ given a particular value for $\mathbf{z}$ is a Gaussian:

$$p(\mathbf{x}|z_k = 1) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k),$$

which can also be written in the form

$$p(\mathbf{x}|\mathbf{z}) = \prod_{k=1}^{K} \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)^{z_k}. \tag{15.10}$$

The joint distribution is given by $p(\mathbf{z})p(\mathbf{x}|\mathbf{z})$ and is described by the graphical model in Figure 15.4. The marginal distribution of $\mathbf{x}$ is then obtained by summing the joint distribution over all possible states of $\mathbf{z}$ to give

Exercise 15.3

$$p(\mathbf{x}) = \sum_{\mathbf{z}} p(\mathbf{z})p(\mathbf{x}|\mathbf{z}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{15.11}$$

where we have made use of (15.9) and (15.10). Thus, the marginal distribution of $\mathbf{x}$ is a Gaussian mixture of the form (15.6). If we have several observations $\mathbf{x}_1, \ldots, \mathbf{x}_N$, then, because we have represented the marginal distribution in the form $p(\mathbf{x}) = \sum_{\mathbf{z}} p(\mathbf{x}, \mathbf{z})$, it follows that for every observed data point $\mathbf{x}_n$ there is a corresponding latent variable $\mathbf{z}_n$.

We have therefore found an equivalent formulation of the Gaussian mixture involving explicit latent variables. It might seem that we have not gained much by doing so. However, we are now able to work with the joint distribution $p(\mathbf{x}, \mathbf{z})$ instead of the marginal distribution $p(\mathbf{x})$, and this will lead to significant simplifications, most notably through the introduction of the EM algorithm.

Another quantity that will play an important role is the conditional probability of $\mathbf{z}$ given $\mathbf{x}$. We will use $\gamma(z_k)$ to denote $p(z_k = 1|\mathbf{x})$, whose value can be found using Bayes' theorem:

$$\begin{aligned}\gamma(z_k) \equiv p(z_k = 1|\mathbf{x}) &= \frac{p(z_k = 1)p(\mathbf{x}|z_k = 1)}{\sum_{j=1}^{K} p(z_j = 1)p(\mathbf{x}|z_j = 1)} \\ &= \frac{\pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}. \qquad (15.12)\end{aligned}$$

We will view π_k as the prior probability of $z_k = 1$, and the quantity $\gamma(z_k)$ as the corresponding posterior probability once we have observed $\mathbf{x}$. As we will see later, $\gamma(z_k)$ can also be viewed as the *responsibility* that component k takes for 'explaining' the observation $\mathbf{x}$.

Section 14.2.5

We can use ancestral sampling to generate random samples distributed according to the Gaussian mixture model. To do this, we first generate a value for $\mathbf{z}$, which we denote $\widehat{\mathbf{z}}$, from the marginal distribution $p(\mathbf{z})$ and then generate a value for $\mathbf{x}$ from the conditional distribution $p(\mathbf{x}|\widehat{\mathbf{z}})$. We can depict samples from the joint distribution $p(\mathbf{x}, \mathbf{z})$ by plotting points at the corresponding values of $\mathbf{x}$ and then colouring them according to the value of $\mathbf{z}$, in other words according to which Gaussian component was responsible for generating them, as shown in Figure 15.5(a). Similarly samples from the marginal distribution $p(\mathbf{x})$ are obtained by taking the samples from the joint distribution and ignoring the values of $\mathbf{z}$. These are illustrated in Figure 15.5(b) by plotting the $\mathbf{x}$ values without any coloured labels.

We can also use this synthetic data set to illustrate the 'responsibilities' by evaluating, for every data point, the posterior probability for each component in the mixture distribution from which this data set was generated. In particular, we can represent the value of the responsibilities $\gamma(z_{nk})$ associated with data point $\mathbf{x}_n$ by plotting the corresponding point using proportions of red, blue, and green ink given by $\gamma(z_{nk})$ for $k = 1, 2, 3$, respectively, as shown in Figure 15.5(c). So, for instance, a data point for which $\gamma(z_{n1}) = 1$ will be coloured red, whereas one for which $\gamma(z_{n2}) = \gamma(z_{n3}) = 0.5$ will be coloured with equal proportions of blue and green ink and so will appear cyan. This should be compared with Figure 15.5(a) in which the data points were labelled using the true identity of the component from which they were generated.

15.2.1 Likelihood function

Suppose we have a data set of observations $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, and we wish to model this data using a mixture of Gaussians. We can represent this data set as an $N \times D$ matrix $\mathbf{X}$ in which the nth row is given by $\mathbf{x}_n^{\mathrm{T}}$. From (15.6) the log of the likelihood function is given by

$$\ln p(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\}. \qquad (15.13)$$

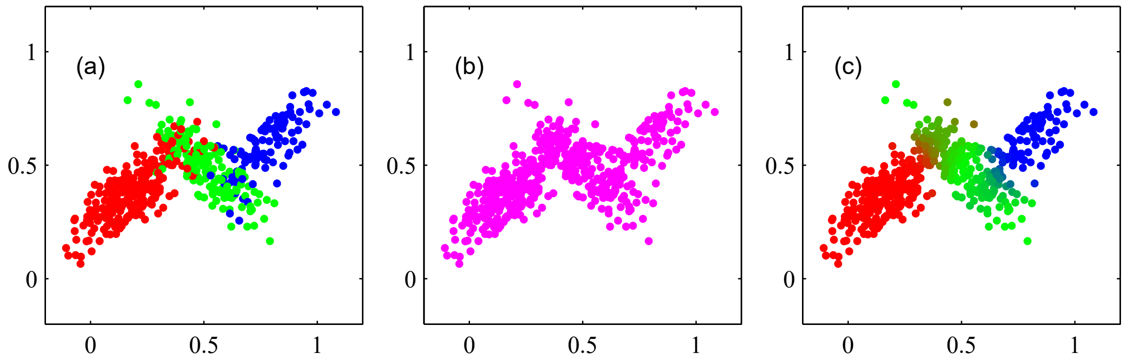


Figure 15.5 Example of 500 points drawn from the mixture of three Gaussians shown in Figure 3.8. (a) Samples from the joint distribution $p(\mathbf{z})p(\mathbf{x}|\mathbf{z})$ in which the three states of $\mathbf{z}$, corresponding to the three components of the mixture, are depicted in red, green, and blue, and (b) the corresponding samples from the marginal distribution $p(\mathbf{x})$, which is obtained by simply ignoring the values of $\mathbf{z}$ and just plotting the $\mathbf{x}$ values. The data set in (a) is said to be *complete*, whereas that in (b) is *incomplete*, as discussed further in Section 15.3. (c) The same samples in which the colours represent the value of the responsibilities $\gamma(z_{nk})$ associated with data point $\mathbf{x}_n$, obtained by plotting the corresponding point using proportions of red, blue, and green ink given by $\gamma(z_{nk})$ for $k = 1, 2, 3$, respectively.

Maximizing this log likelihood function (15.13) is a more complex problem than for a single Gaussian. The difficulty arises from the presence of the summation over k that appears inside the logarithm in (15.13), so that the logarithm function no longer acts directly on the Gaussian. If we set the derivatives of the log likelihood to zero, we will no longer obtain a closed-form solution, as we will see shortly.

Before discussing how to maximize this function, it is worth emphasizing that there is a significant problem associated with the maximum likelihood framework when applied to Gaussian mixture models, due to the presence of singularities. For simplicity, consider a Gaussian mixture whose components have covariance matrices given by $\mathbf{\Sigma}_k = \sigma_k^2\mathbf{I}$, where $\mathbf{I}$ is the unit matrix, although the conclusions will hold for general covariance matrices. Suppose that one of the components of the mixture model, let us say the jth component, has its mean $\boldsymbol{\mu}_j$ exactly equal to one of the data points so that $\boldsymbol{\mu}_j = \mathbf{x}_n$ for some value of n. This data point will then contribute a term in the likelihood function of the form

$$\mathcal{N}(\mathbf{x}_n|\mathbf{x}_n, \sigma_j^2\mathbf{I}) = \frac{1}{(2\pi)^{1/2}}\frac{1}{\sigma_j}. \tag{15.14}$$

If we consider the limit $\sigma_j \to 0$, then we see that this term goes to infinity and so the log likelihood function will also go to infinity. Thus, the maximization of the log likelihood function is not a well posed-problem because such singularities will always be present and will occur whenever one of the Gaussian components 'collapses' onto a specific data point. Recall that this problem did not arise with a single Gaussian distribution. To understand the difference, note that if a single Gaussian collapses onto a data point, it will contribute multiplicative factors to the

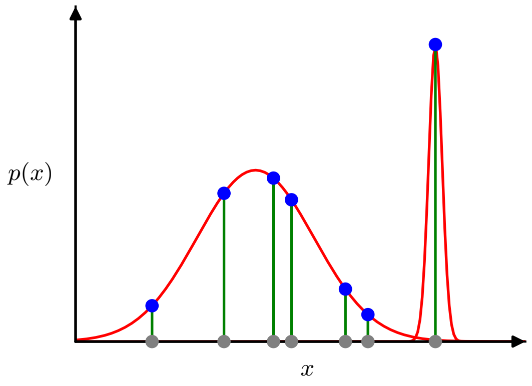


Figure 15.6 Illustration of how singularities in the likelihood function arise with mixtures of Gaussians. This should be compared with a single Gaussian shown in Figure 2.9 for which no singularities arise.

likelihood function arising from the other data points, and these factors will go to zero exponentially fast, giving an overall likelihood that goes to zero rather than infinity. However, once we have (at least) two components in the mixture, one of the components can have a finite variance and therefore assign finite probability to all the data points while the other component can shrink onto one specific data point and thereby contribute an ever increasing additive value to the log likelihood. This is illustrated in Figure 15.6. These singularities provide an example of the over-fitting that can occur in a maximum likelihood approach. When applying maximum likelihood to Gaussian mixture models, we must take steps to avoid finding such pathological solutions and instead seek local maxima of the likelihood function that are well behaved. We can try to avoid the singularities by using suitable heuristics, for instance by detecting when a Gaussian component is collapsing and resetting its mean to a randomly chosen value while also resetting its covariance to some large value and then continuing with the optimization. The singularities can also be avoided by adding a regularization term to the log likelihood corresponding to a prior distribution over the parameters. *Section 15.4.3*

A further issue in finding maximum likelihood solutions arises because for any given maximum likelihood solution, a K-component mixture will have a total of $K!$ equivalent solutions corresponding to the $K!$ ways of assigning K sets of parameters to K components. In other words, for any given (non-degenerate) point in the space of parameter values, there will be a further $K! - 1$ additional points all of which give rise to exactly the same distribution. This problem is known as *identifiability* (Casella and Berger, 2002) and is an important issue when we wish to interpret the parameter values discovered by a model. Identifiability will also arise when we discuss models having continuous latent variables. However, when finding a good density model, it is irrelevant because any of the equivalent solutions is as good as any other. *Chapter 16*

15.2.2 Maximum likelihood

An elegant and powerful method for finding maximum likelihood solutions for models with latent variables is called the *expectation–maximization* algorithm or *EM* algorithm (Dempster, Laird, and Rubin, 1977; McLachlan and Krishnan, 1997). In this chapter we will give three different derivations of the EM algorithm, each more general than the previous. We begin here with a relatively informal treatment in the context of a Gaussian mixture model. We emphasize, however, that EM has broad applicability, and the underlying concepts will be encountered in the context of several different models in this book.

We begin by writing down the conditions that must be satisfied at a maximum of the likelihood function. Setting the derivatives of $\ln p(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$ in (15.13) with respect to the means $\boldsymbol{\mu}_k$ of the Gaussian components to zero, we obtain

$$0 = \sum_{n=1}^{N} \underbrace{\frac{\pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_j \pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}}_{\gamma(z_{nk})} \boldsymbol{\Sigma}_k^{-1}(\mathbf{x}_n - \boldsymbol{\mu}_k) \tag{15.15}$$

where we have made use of the form (3.26) for the Gaussian distribution. Note that the posterior probabilities, or responsibilities, $\gamma(z_{nk})$ given by (15.12) appear naturally on the right-hand side. Multiplying by $\boldsymbol{\Sigma}_k$ (which we assume to be non-singular) and rearranging we obtain

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) \mathbf{x}_n \tag{15.16}$$

where we have defined

$$N_k = \sum_{n=1}^{N} \gamma(z_{nk}). \tag{15.17}$$

We can interpret N_k as the effective number of points assigned to cluster k. Note carefully the form of this solution. We see that the mean $\boldsymbol{\mu}_k$ for the kth Gaussian component is obtained by taking a weighted mean of all the points in the data set, in which the weighting factor for data point $\mathbf{x}_n$ is given by the posterior probability $\gamma(z_{nk})$ that component k was responsible for generating $\mathbf{x}_n$.

If we set the derivative of $\ln p(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$ with respect to $\boldsymbol{\Sigma}_k$ to zero and follow a similar line of reasoning by making use of the result for the maximum likelihood solution for the covariance matrix of a single Gaussian, we obtain

Section 3.2.7

$$\boldsymbol{\Sigma}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) (\mathbf{x}_n - \boldsymbol{\mu}_k)(\mathbf{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}}, \tag{15.18}$$

which has the same form as the corresponding result for a single Gaussian fitted to the data set, but again with each data point weighted by the corresponding posterior probability and with the denominator given by the effective number of points associated with the corresponding component.

Finally, we maximize $\ln p(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$ with respect to the mixing coefficients π_k. Here we must take account of the constraint (15.8), which requires the mixing coefficients to sum to one. This can be achieved using a Lagrange multiplier λ and

Appendix C

maximizing the following quantity:

$$\ln p(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) + \lambda \left(\sum_{k=1}^{K} \pi_k - 1 \right), \tag{15.19}$$

which gives

$$0 = \sum_{n=1}^{N} \frac{\mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_j \pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} + \lambda \tag{15.20}$$

where again we see the appearance of the responsibilities. If we now multiply both sides by π_k and sum over k making use of the constraint (15.8), we find $\lambda = -N$. Using this to eliminate λ and rearranging, we obtain

$$\pi_k = \frac{N_k}{N} \tag{15.21}$$

so that the mixing coefficient for the kth component is given by the average responsibility which that component takes for explaining the data points.

Note that the results (15.16), (15.18), and (15.21) do not constitute a closed-form solution for the parameters of the mixture model because the responsibilities $\gamma(z_{nk})$ depend on those parameters in a complex way through (15.12). However, these results do suggest a simple iterative scheme for finding a solution to the maximum likelihood problem, which as we will see turns out to be an instance of the EM algorithm for the particular case of the Gaussian mixture model. We first choose some initial values for the means, covariances, and mixing coefficients. Then we alternate between the following two updates, which we will call the E step and the M step for reasons that will become apparent shortly. In the *expectation* step, or E step, we use the current values for the parameters to evaluate the posterior probabilities, or responsibilities, given by (15.12). We then use these probabilities in the *maximization* step, or M step, to re-estimate the means, covariances, and mixing coefficients using the results (15.16), (15.18), and (15.21). Note that in so doing, we first evaluate the new means using (15.16) and then use these new values to find the covariances using (15.18), in keeping with the corresponding result for a single Gaussian distribution. We will show that each update to the parameters resulting from an E step followed by an M step is guaranteed to increase the log likelihood function. In practice, the algorithm is deemed to have converged when the change in the log likelihood function, or alternatively in the parameters, falls below some threshold.

Section 15.3

We illustrate the EM algorithm for a mixture of two Gaussians applied to the re-scaled Old Faithful data in Figure 15.7. Here a mixture of two Gaussians is used, with centres initialized using the same values as for the K-means algorithm in Figure 15.1 and with covariance matrices initialized to be proportional to the unit matrix. Plot (a) shows the data points in green, together with the initial configuration of the mixture model in which the one standard-deviation contours for the two Gaussian components are shown as blue and red circles. Plot (b) shows the result of the initial E step, in which each data point is depicted using a proportion of blue ink equal to the posterior probability of having been generated from the blue component and a

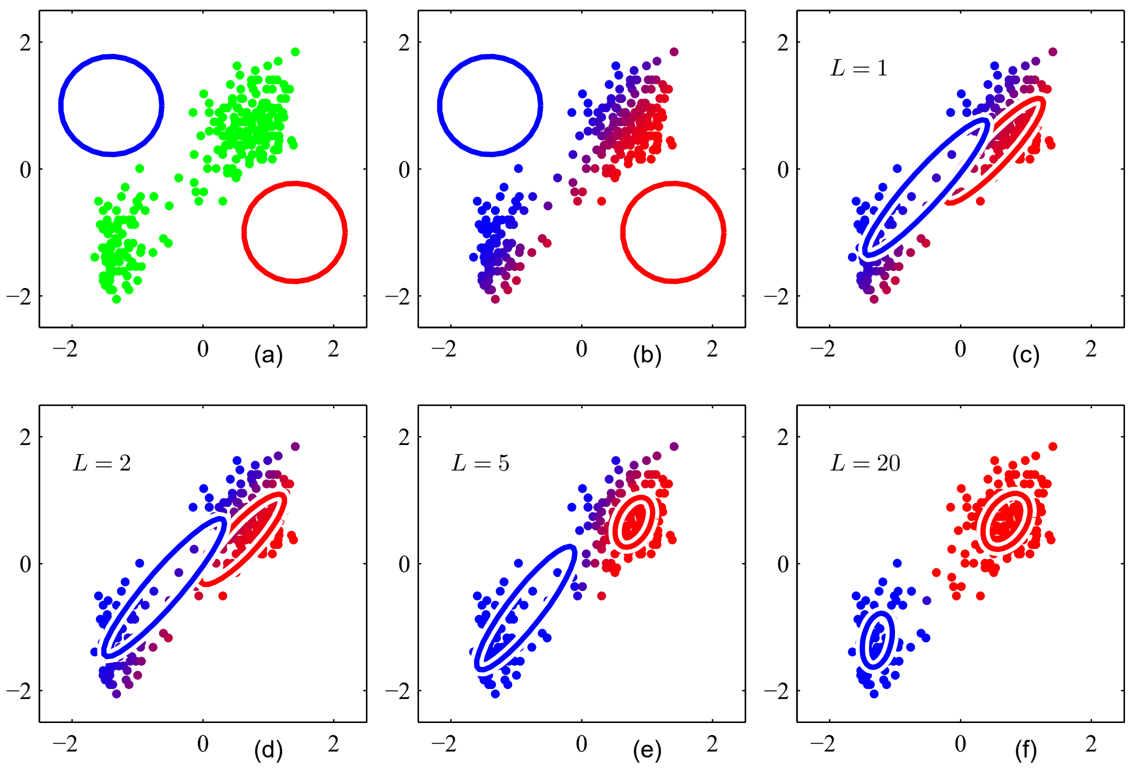


Figure 15.7 Application of the EM algorithm to the Old Faithful data set as used for the illustration of the K-means algorithm in Figure 15.1. See the text for details.

corresponding proportion of red ink given by the posterior probability of having been generated by the red component. Thus, points that have a roughly equal probability for belonging to either cluster appear purple. The situation after the first M step is shown in plot (c), in which the mean of the blue Gaussian has moved to the mean of the data set, weighted by the probabilities of each data point belonging to the blue cluster. In other words it has moved to the centre of mass of the blue ink. Similarly, the covariance of the blue Gaussian is set equal to the covariance of the blue ink. Analogous results hold for the red component. Plots (d), (e), and (f) show the results after 2, 5, and 20 complete cycles of EM, respectively. In plot (f) the algorithm is close to convergence.

Note that the EM algorithm takes many more iterations to reach (approximate) convergence compared with the K-means algorithm and that each cycle requires significantly more computation. It is therefore common to run the K-means algorithm to find a suitable initialization for a Gaussian mixture model that is subsequently adapted using EM. The covariance matrices can conveniently be initialized to the sample covariances of the clusters found by the K-means algorithm, and the mixing coefficients can be set to the fractions of data points assigned to the respective

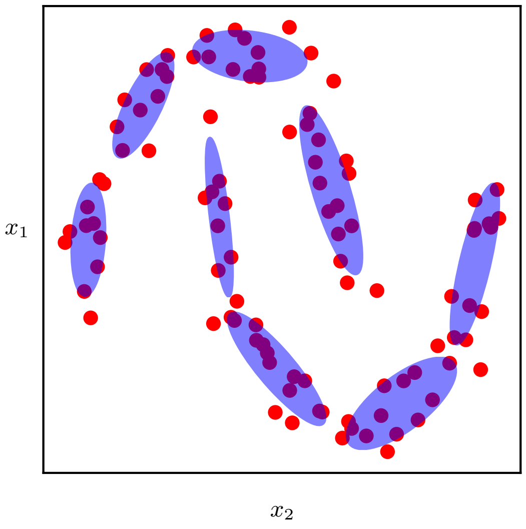


Figure 15.8 A Gaussian mixture model fitted to the 'two-moons' data set, showing that a large number of mixture components may be required to give an accurate representation of a complex data distribution. Here the ellipses represent the contours of constant density for the corresponding mixture components. As we move to spaces of larger dimensionality, the number of components required to model a distribution accurately can become unacceptably large.

clusters. Techniques such as parameter regularization must be employed to avoid singularities of the likelihood function in which a Gaussian component collapses onto a particular data point. It should be emphasized that there will generally be multiple local maxima of the log likelihood function and that EM is not guaranteed to find the largest of these maxima. Because the EM algorithm for Gaussian mixtures plays such an important role, we summarize it in Algorithm 15.2.

Mixture models are very flexible and can approximate complicated distributions to high accuracy given a sufficient number of components if the model parameters are chosen appropriately. In practice, however, the number of components can be extremely large, especially in spaces of high dimensionality. This problem is illustrated for the two-moons data set in Figure 15.8. Nevertheless, mixture models are useful in many applications. Also, an understanding of mixture models lays the foundations for models with continuous latent variables and for generative models based on deep neural networks, which have much better scaling to spaces of high dimensionality.

Chapter 16

15.3. Expectation–Maximization Algorithm

We turn now to a more general view of the EM algorithm in which we focus on the role of latent variables. As before we denote the set of all observed data points by $\mathbf{X}$, in which the nth row represents $\mathbf{x}_n^{\mathrm{T}}$. Similarly, the corresponding latent variables will be denoted by an $N \times K$ matrix $\mathbf{Z}$ with rows $\mathbf{z}_n^{\mathrm{T}}$. If we assume that the data points are drawn independently from the distribution, then we can express the Gaussian mixture model for this i.i.d. data set using the graphical representation shown in Figure 15.9. The set of all model parameters is denoted by $\boldsymbol{\theta}$, and so the log likelihood function

Algorithm 15.2: EM algorithm for a Gaussian mixture model

Input: Initial model parameters $\{\boldsymbol{\mu}_k\}, \{\boldsymbol{\Sigma}_k\}, \{\pi_k\}$
Data set $\{\mathbf{x}_1, \dots, \mathbf{x}_N\}$
Output: Final model parameters $\{\boldsymbol{\mu}_k\}, \{\boldsymbol{\Sigma}_k\}, \{\pi_k\}$

```
repeat
    // E step
    for $n \in \{1, \dots, N\}$ do
        for $k \in \{1, \dots, K\}$ do
            $\gamma(z_{nk}) \leftarrow \dfrac{\pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}$
        end for
    end for
    // M step
    for $k \in \{1, \dots, K\}$ do
        $N_k \leftarrow \sum_{n=1}^{N} \gamma(z_{nk})$
        $\boldsymbol{\mu}_k \leftarrow \dfrac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) \mathbf{x}_n$
        $\boldsymbol{\Sigma}_k \leftarrow \dfrac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) (\mathbf{x}_n - \boldsymbol{\mu}_k) (\mathbf{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}}$
        $\pi_k \leftarrow \dfrac{N_k}{N}$
    end for
    // Log likelihood
    $\mathcal{L} \leftarrow \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\}$
until convergence
return $\{\mathbf{x}_k\}, \{\boldsymbol{\Sigma}_k\}, \{\pi_k\}$
```

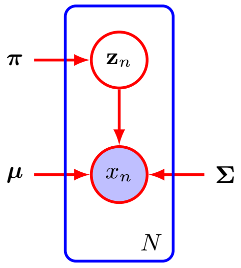


Figure 15.9 Graphical representation of a Gaussian mixture model for a set of N i.i.d. data points $\{\mathbf{x}_n\}$, with corresponding latent points $\{\mathbf{z}_n\}$, where $n = 1, \ldots, N$.

is given by

$$\ln p(\mathbf{X}|\boldsymbol{\theta}) = \ln \left\{ \sum_{\mathbf{Z}} p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta}) \right\}. \tag{15.22}$$

Chapter 16

Note that our discussion will apply equally well to continuous latent variables simply by replacing the sum over $\mathbf{Z}$ with an integral.

A key observation is that the summation over the latent variables appears inside the logarithm. Even if the joint distribution $p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$ belongs to the exponential family, the marginal distribution $p(\mathbf{X}|\boldsymbol{\theta})$ typically does not as a result of this summation. The presence of the sum prevents the logarithm from acting directly on the joint distribution, resulting in complicated expressions for the maximum likelihood solution.

Now suppose that, for each observation in $\mathbf{X}$, we were told the corresponding value of the latent variable $\mathbf{Z}$. We will call $\{\mathbf{X}, \mathbf{Z}\}$ the *complete* data set, and we will refer to the actual observed data $\mathbf{X}$ as *incomplete*, as illustrated in Figure 15.5. The likelihood function for the complete data set simply takes the form $\ln p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$, and we will suppose that maximization of this complete-data log likelihood function is straightforward.

In practice, however, we are not given the complete data set $\{\mathbf{X}, \mathbf{Z}\}$ but only the incomplete data $\mathbf{X}$. Our state of knowledge of the values of the latent variables in $\mathbf{Z}$ is given only by the posterior distribution $p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta})$. Because we cannot use the complete-data log likelihood, we consider instead its expected value under the posterior distribution of the latent variables, which corresponds (as we will see) to the E step of the EM algorithm. In the subsequent M step, we maximize this expectation. If the current estimate for the parameters is denoted by $\boldsymbol{\theta}^{\text{old}}$, then a pair of successive E and M steps gives rise to a revised estimate $\boldsymbol{\theta}^{\text{new}}$. The algorithm is initialized by choosing some starting value for the parameters $\boldsymbol{\theta}_0$. Although this use of the expectation may seem somewhat arbitrary, we will see the motivation for this choice when we give a deeper treatment of EM in Section 15.4.

In the E step, we use the current parameter values $\boldsymbol{\theta}^{\text{old}}$ to find the posterior distribution of the latent variables given by $p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{\text{old}})$. We then use this posterior distribution to find the expectation of the complete-data log likelihood evaluated for some general parameter value $\boldsymbol{\theta}$. This expectation, denoted by $\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}})$, is given by

$$\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}) = \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{\text{old}}) \ln p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta}). \tag{15.23}$$

Algorithm 15.3: General EM algorithm

```
Input: Joint distribution $p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$
       Initial parameters $\boldsymbol{\theta}^{\text{old}}$
       Data set $\mathbf{x}_1, \ldots, \mathbf{x}_N$
Output: Final parameters $\boldsymbol{\theta}$

repeat
    $\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}) \leftarrow \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{\text{old}}) \ln p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$ // E step
    $\boldsymbol{\theta}^{\text{new}} \leftarrow \arg\max_{\boldsymbol{\theta}} \mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}})$ // M step
    $\mathcal{L} \leftarrow p(\mathbf{X}|\boldsymbol{\theta}^{\text{new}})$ // Evaluate log likelihood
    $\boldsymbol{\theta}^{\text{old}} \leftarrow \boldsymbol{\theta}^{\text{new}}$ // Update the parameters
until convergence
return $\boldsymbol{\theta}^{\text{new}}$
```

In the M step, we determine the revised parameter estimate $\boldsymbol{\theta}^{\text{new}}$ by maximizing this function:

$$\boldsymbol{\theta}^{\text{new}} = \arg\max_{\boldsymbol{\theta}} \mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}). \tag{15.24}$$

Note that in the definition of $\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}})$, the logarithm acts directly on the joint distribution $p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$, and so the corresponding M-step maximization will, according to our assumption, be tractable. The general EM algorithm is summarized in Algorithm 15.3. It has the property, as we will show later, that each cycle of EM will increase the incomplete-data log likelihood (unless it is already at a local maximum). *Section 15.4.1*

The EM algorithm can also be used to find MAP (maximum posterior) solutions for models in which a prior $p(\boldsymbol{\theta})$ is defined over the parameters. *Exercise 15.5* In this case the E step remains the same as in the maximum likelihood case, whereas in the M step the quantity to be maximized is given by $\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}) + \ln p(\boldsymbol{\theta})$. Suitable choices for the prior will remove the singularities of the kind illustrated in Figure 15.6.

Here we have considered the use of the EM algorithm to maximize a likelihood function when there are discrete latent variables. However, it can also be applied when the unobserved variables correspond to missing values in the data set. The distribution of the observed values is obtained by taking the joint distribution of all the variables and then marginalizing over the missing ones. EM can then be used to maximize the corresponding likelihood function. This will be a valid procedure if the data values are *missing at random*, meaning that the mechanism causing values to be missing does not depend on the unobserved values. In many situations this will not be the case, for instance if a sensor fails to return a value whenever the quantity it is measuring exceeds some threshold.

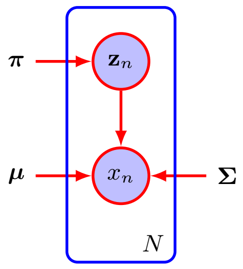


Figure 15.10 This shows the same graph as in Figure 15.9 except that we now suppose that the discrete variables $\mathbf{z}_n$ are observed, as well as the data variables $\mathbf{x}_n$.

15.3.1 Gaussian mixtures

We now consider the application of this latent-variable view of EM to the specific case of a Gaussian mixture model. Recall that our goal is to maximize the log likelihood function (15.13), which is computed using the observed data set $\mathbf{X}$, and we saw that this was more difficult than with a single Gaussian distribution due to the summation over k that occurs inside the logarithm. Suppose then that in addition to the observed data set $\mathbf{X}$, we were also given the values of the corresponding discrete variables $\mathbf{Z}$. Recall that Figure 15.5(a) shows a *complete* data set (i.e., one that includes labels showing which component generated each data point) whereas Figure 15.5(b) shows the corresponding *incomplete* data set. A graphical model for the complete data is shown in Figure 15.10.

Now consider the problem of maximizing the likelihood for the complete data set $\{\mathbf{X}, \mathbf{Z}\}$. From (15.9) and (15.10), this likelihood function takes the form

$$p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}) = \prod_{n=1}^{N}\prod_{k=1}^{K} \pi_k^{z_{nk}} \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)^{z_{nk}} \tag{15.25}$$

where z_{nk} denotes the kth component of $\mathbf{z}_n$. Taking the logarithm, we obtain

$$\ln p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}) = \sum_{n=1}^{N}\sum_{k=1}^{K} z_{nk} \left\{\ln \pi_k + \ln \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\right\}. \tag{15.26}$$

Comparison with the log likelihood function (15.13) for the incomplete data shows that the summation over k and the logarithm have been interchanged. The logarithm now acts directly on the Gaussian distribution, which itself is a member of the exponential family. Not surprisingly, this leads to a much simpler solution to the maximum likelihood problem, as we now show. Consider first the maximization with respect to the means and covariances. Because $\mathbf{z}_n$ is a K-dimensional vector with all elements equal to 0 except for a single element having the value 1, the complete-data log likelihood function is simply a sum of K independent contributions, one for each mixture component. Thus, the maximization with respect to a mean or a covariance is exactly as for a single Gaussian, except that it involves only the subset of data points that are 'assigned' to that component. For the maximization with respect to the mixing coefficients, note that these are coupled for different values of k by virtue of the summation constraint (15.8). Again, this can be enforced using a Lagrange multiplier as before, which leads to the result

$$\pi_k = \frac{1}{N}\sum_{n=1}^{N} z_{nk} \tag{15.27}$$

so that the mixing coefficients are equal to the fractions of data points assigned to the corresponding components.

Thus, we see that the complete-data log likelihood function can be maximized trivially in closed form. In practice, however, we do not have values for the latent variables. Therefore, as discussed earlier, we consider the expectation, with respect to the posterior distribution of the latent variables, of the complete-data log likelihood. Using (15.9) and (15.10) together with Bayes' theorem, we see that this posterior distribution takes the form

$$p(\mathbf{Z}|\mathbf{X},\boldsymbol{\mu},\boldsymbol{\Sigma},\boldsymbol{\pi}) \propto \prod_{n=1}^{N}\prod_{k=1}^{K}\left[\pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)\right]^{z_{nk}}. \tag{15.28}$$

Exercise 15.6
Section 11.2

We see that this factorizes over n so that under the posterior distribution, the $\{\mathbf{z}_n\}$ are independent. This is easily verified by inspecting the directed graph in Figure 15.9 and making use of the d-separation criterion. The expected value of the indicator variable z_{nk} under this posterior distribution is then given by

$$\begin{aligned}\mathbb{E}[z_{nk}] &= \frac{\displaystyle\sum_{\mathbf{z}_n} z_{nk}\prod_{k'}\left[\pi_{k'}\mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_{k'},\boldsymbol{\Sigma}_{k'})\right]^{z_{nk'}}}{\displaystyle\sum_{\mathbf{z}_n}\prod_{j}\left[\pi_j\mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j)\right]^{z_{nj}}} \\ &= \frac{\pi_k\mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)}{\displaystyle\sum_{j=1}^{K}\pi_j\mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j)} = \gamma(z_{nk}),\end{aligned} \tag{15.29}$$

which is just the responsibility of component k for data point $\mathbf{x}_n$. The expected value of the complete-data log likelihood function is therefore given by

$$\mathbb{E}_{\mathbf{Z}}[\ln p(\mathbf{X},\mathbf{Z}|\boldsymbol{\mu},\boldsymbol{\Sigma},\boldsymbol{\pi})] = \sum_{n=1}^{N}\sum_{k=1}^{K}\gamma(z_{nk})\left\{\ln\pi_k + \ln\mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)\right\}. \tag{15.30}$$

Exercise 15.9

We can now proceed as follows. First we choose some initial values for the parameters $\boldsymbol{\mu}^{\text{old}}$, $\boldsymbol{\Sigma}^{\text{old}}$, and $\boldsymbol{\pi}^{\text{old}}$, and we use these to evaluate the responsibilities (the E step). We then keep the responsibilities fixed and maximize (15.30) with respect to $\boldsymbol{\mu}_k$, $\boldsymbol{\Sigma}_k$, and π_k (the M step). This leads to closed-form solutions for $\boldsymbol{\mu}^{\text{new}}$, $\boldsymbol{\Sigma}^{\text{new}}$, and $\boldsymbol{\pi}^{\text{new}}$ given by (15.16), (15.18), and (15.21) as before. This is precisely the EM algorithm for Gaussian mixtures as derived earlier. We will gain more insight into the role of the expected complete-data log likelihood function when discuss the convergence of the EM algorithm in Section 15.4.

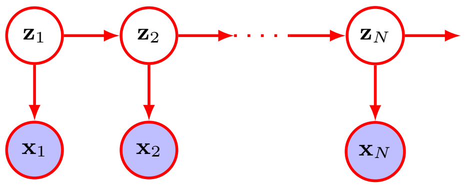


Figure 15.11 The probabilistic graphical model for sequential data corresponding to a hidden Markov model. The discrete latent variables are no longer independent but form a Markov chain.

Throughout this chapter we assume that the data observations are i.i.d. For ordered observations that form a sequence, the mixture model can be extended by connecting the latent variables in a Markov chain to give a *hidden Markov model* whose graphical structure is shown in Figure 15.11. The EM algorithm can be extended to this more complex model in which the E step involves a sequential calculation in which messages are passed along the chain of latent variables (Bishop, 2006).

15.3.2 Relation to K-means

Comparison of the K-means algorithm with the EM algorithm for Gaussian mixtures shows that there is a close similarity. Whereas the K-means algorithm performs a *hard* assignment of data points to clusters in which each data point is associated uniquely with one cluster, the EM algorithm makes a *soft* assignment based on the posterior probabilities. In fact, we can derive the K-means algorithm as a particular limit of EM for Gaussian mixtures as follows.

Consider a Gaussian mixture model in which the covariance matrices of the mixture components are given by $\epsilon\mathbf{I}$, where ϵ is a variance parameter that is shared by all the components, and $\mathbf{I}$ is the identity matrix, so that

$$p(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \frac{1}{(2\pi\epsilon)^{D/2}} \exp\left\{-\frac{1}{2\epsilon}\|\mathbf{x}-\boldsymbol{\mu}_k\|^2\right\}. \tag{15.31}$$

We now consider the EM algorithm for a mixture of K Gaussians of this form in which we treat ϵ as a fixed constant, instead of a parameter to be re-estimated. From (15.12) the posterior probabilities, or responsibilities, for a particular data point $\mathbf{x}_n$ are given by

$$\gamma(z_{nk}) = \frac{\pi_k \exp\left\{-\|\mathbf{x}_n-\boldsymbol{\mu}_k\|^2/2\epsilon\right\}}{\sum_j \pi_j \exp\left\{-\|\mathbf{x}_n-\boldsymbol{\mu}_j\|^2/2\epsilon\right\}}. \tag{15.32}$$

Consider the limit $\epsilon \to 0$. The denominator consists of a sum of terms indexed by j each of which goes to zero. The particular term for which $\|\mathbf{x}_n - \boldsymbol{\mu}_j\|^2$ is smallest, say $j = l$, will go to zero most slowly and will then dominate this sum. Therefore, the responsibilities $\gamma(z_{nk})$ for the data point $\mathbf{x}_n$ all go to zero except for term l, for which the responsibility $\gamma(z_{nl})$ will go to unity. Note that this holds independently of the values of the π_k so long as none of the π_k is zero. Thus, in this limit, we obtain a hard assignment of data points to clusters, just as in the K-means algorithm, so that $\gamma(z_{nk}) \to r_{nk}$ where r_{nk} is defined by (15.2). Each data point is thereby assigned to the cluster having the closest mean. The EM re-estimation equation for the $\boldsymbol{\mu}_k$, given by (15.16), then reduces to the K-means result (15.4). Note that the re-estimation formula for the mixing coefficients (15.21) simply resets the value of π_k to be equal to the fraction of data points assigned to cluster k, although these parameters no longer play an active role in the algorithm.

Finally, in the limit $\epsilon \to 0$, the expected complete-data log likelihood, given by (15.30), becomes

Exercise 15.12

$$\mathbb{E}_{\mathbf{Z}}[\ln p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi})] \to -\frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk}\|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2 + \text{const.} \tag{15.33}$$

Thus, we see that in this limit, maximizing the expected complete-data log likelihood is equivalent to minimizing the error measure J for the K-means algorithm given by (15.1). Note that the K-means algorithm does not estimate the covariances of the clusters but only the cluster means.

15.3.3 Mixtures of Bernoulli distributions

So far in this chapter, we have focused on distributions over continuous variables described by mixtures of Gaussians. As a further example of mixture modelling and to illustrate the EM algorithm in a different context, we now discuss mixtures of discrete binary variables described by Bernoulli distributions. This model is also known as *latent class analysis* (Lazarsfeld and Henry, 1968; McLachlan and Peel, 2000).

Consider a set of D binary variables x_i, where $i = 1, \ldots, D$, each of which is governed by a Bernoulli distribution with parameter μ_i, so that

Section 3.1.1

$$p(\mathbf{x}|\boldsymbol{\mu}) = \prod_{i=1}^{D} \mu_i^{x_i}(1-\mu_i)^{(1-x_i)} \tag{15.34}$$

where $\mathbf{x} = (x_1, \ldots, x_D)^{\mathrm{T}}$ and $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_D)^{\mathrm{T}}$. We see that the individual variables x_i are independent, given $\boldsymbol{\mu}$. The mean and covariance of this distribution are easily seen to be

Exercise 15.13

$$\mathbb{E}[\mathbf{x}] = \boldsymbol{\mu} \tag{15.35}$$

$$\text{cov}[\mathbf{x}] = \text{diag}\{\mu_i(1-\mu_i)\}. \tag{15.36}$$

Now let us consider a finite mixture of these distributions given by

$$p(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\pi}) = \sum_{k=1}^{K} \pi_k p(\mathbf{x}|\boldsymbol{\mu}_k) \tag{15.37}$$

where $\boldsymbol{\mu} = \{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K\}$, $\boldsymbol{\pi} = \{\pi_1, \ldots, \pi_K\}$, and

$$p(\mathbf{x}|\boldsymbol{\mu}_k) = \prod_{i=1}^{D} \mu_{ki}^{x_i}(1-\mu_{ki})^{(1-x_i)}. \tag{15.38}$$

The mixing coefficients satisfy (15.7) and (15.8). The mean and covariance of this mixture distribution are given by

Exercise 15.14

$$\mathbb{E}[\mathbf{x}] = \sum_{k=1}^{K} \pi_k \boldsymbol{\mu}_k \tag{15.39}$$

$$\mathrm{cov}[\mathbf{x}] = \sum_{k=1}^{K} \pi_k \left\{ \boldsymbol{\Sigma}_k + \boldsymbol{\mu}_k \boldsymbol{\mu}_k^{\mathrm{T}} \right\} - \mathbb{E}[\mathbf{x}]\mathbb{E}[\mathbf{x}]^{\mathrm{T}} \tag{15.40}$$

where $\boldsymbol{\Sigma}_k = \mathrm{diag}\left\{\mu_{ki}(1-\mu_{ki})\right\}$. Because the covariance matrix $\mathrm{cov}[\mathbf{x}]$ is no longer diagonal, the mixture distribution can capture correlations between the variables, unlike a single Bernoulli distribution.

If we are given a data set $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ then the log likelihood function for this model is given by

$$\ln p(\mathbf{X}|\boldsymbol{\mu}, \boldsymbol{\pi}) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k p(\mathbf{x}_n|\boldsymbol{\mu}_k) \right\}. \tag{15.41}$$

Again we see the appearance of the summation inside the logarithm, so that the maximum likelihood solution no longer has closed form.

We now derive the EM algorithm for maximizing the likelihood function for the mixture of Bernoulli distributions. To do this, we first introduce an explicit discrete latent variable $\mathbf{z}$ associated with each instance of $\mathbf{x}$. As with the Gaussian mixture, $\mathbf{z}$ has a 1-of-K coding so that $\mathbf{z} = (z_1, \ldots, z_K)^{\mathrm{T}}$ is a binary K-dimensional vector having a single component equal to 1, with all other components equal to 0. We can then write the conditional distribution of $\mathbf{x}$, given the latent variable, as

$$p(\mathbf{x}|\mathbf{z}, \boldsymbol{\mu}) = \prod_{k=1}^{K} p(\mathbf{x}|\boldsymbol{\mu}_k)^{z_k} \tag{15.42}$$

whereas the prior distribution for the latent variables is the same as for the mixture-of-Gaussians model, so that

$$p(\mathbf{z}|\boldsymbol{\pi}) = \prod_{k=1}^{K} \pi_k^{z_k}. \tag{15.43}$$

Exercise 15.16

If we form the product of $p(\mathbf{x}|\mathbf{z}, \boldsymbol{\mu})$ and $p(\mathbf{z}|\boldsymbol{\pi})$ and then marginalize over $\mathbf{z}$, then we recover (15.37).

To derive the EM algorithm, we first write down the complete-data log likelihood function, which is given by

$$\ln p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\mu}, \boldsymbol{\pi}) = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} \left\{ \ln \pi_k \right. \\ \left. + \sum_{i=1}^{D} \left[x_{ni} \ln \mu_{ki} + (1-x_{ni}) \ln(1-\mu_{ki}) \right] \right\} \tag{15.44}$$

where $\mathbf{X} = \{\mathbf{x}_n\}$ and $\mathbf{Z} = \{\mathbf{z}_n\}$. Next we take the expectation of the complete-data log likelihood with respect to the posterior distribution of the latent variables to give

$$\mathbb{E}_{\mathbf{Z}}[\ln p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\mu}, \boldsymbol{\pi})] = \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma(z_{nk}) \left\{ \ln \pi_k \right. \\ \left. + \sum_{i=1}^{D} [x_{ni} \ln \mu_{ki} + (1 - x_{ni}) \ln(1 - \mu_{ki})] \right\} \tag{15.45}$$

where $\gamma(z_{nk}) = \mathbb{E}[z_{nk}]$ is the posterior probability, or responsibility, of component k given data point $\mathbf{x}_n$. In the E step, these responsibilities are evaluated using Bayes' theorem, which takes the form

$$\gamma(z_{nk}) = \mathbb{E}[z_{nk}] = \frac{\sum_{\mathbf{z}_n} z_{nk} \prod_{k'} [\pi_{k'} p(\mathbf{x}_n|\boldsymbol{\mu}_{k'})]^{z_{nk'}}}{\sum_{\mathbf{z}_n} \prod_{j} [\pi_j p(\mathbf{x}_n|\boldsymbol{\mu}_j)]^{z_{nj}}} \\ = \frac{\pi_k p(\mathbf{x}_n|\boldsymbol{\mu}_k)}{\sum_{j=1}^{K} \pi_j p(\mathbf{x}_n|\boldsymbol{\mu}_j)}. \tag{15.46}$$

If we consider the sum over n in (15.45), we see that the responsibilities enter only through two terms, which can be written as

$$N_k = \sum_{n=1}^{N} \gamma(z_{nk}) \tag{15.47}$$

$$\overline{\mathbf{x}}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) \mathbf{x}_n \tag{15.48}$$

where N_k is the effective number of data points associated with component k. In the M step, we maximize the expected complete-data log likelihood with respect to the parameters $\boldsymbol{\mu}_k$ and $\boldsymbol{\pi}$. If we set the derivative of (15.45) with respect to $\boldsymbol{\mu}_k$ equal to zero and rearrange the terms, we obtain

Exercise 15.17

$$\boldsymbol{\mu}_k = \overline{\mathbf{x}}_k. \tag{15.49}$$

We see that this sets the mean of component k equal to a weighted mean of the data, with weighting coefficients given by the responsibilities that component k takes for each of the data points. For the maximization with respect to π_k, we need to introduce a Lagrange multiplier to enforce the constraint $\sum_k \pi_k = 1$. Following analogous steps to those used for the mixture of Gaussians, we then obtain

Exercise 15.18

$$\pi_k = \frac{N_k}{N}, \tag{15.50}$$

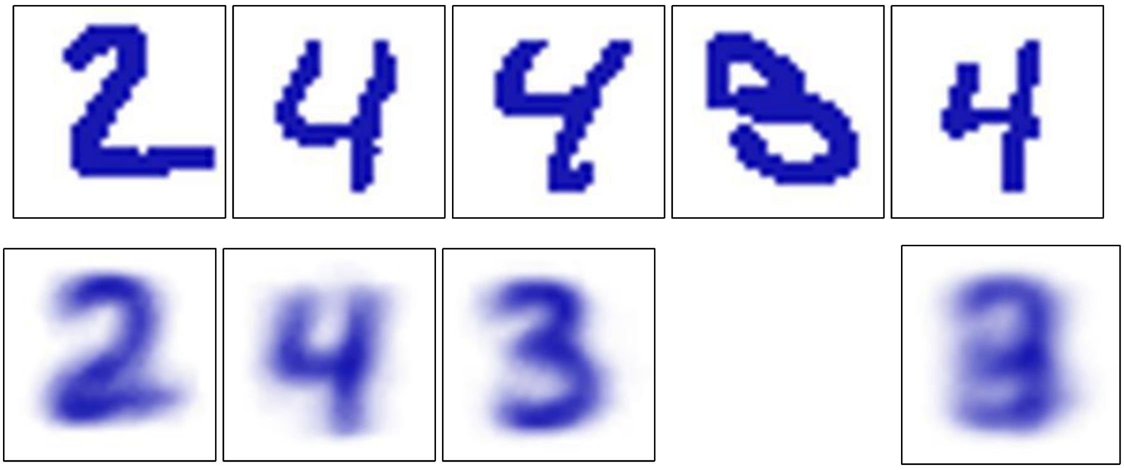


Figure 15.12 Illustration of the Bernoulli mixture model in which the top row shows examples from the digits data set after converting the pixel values from grey scale to binary using a threshold of 0.5. On the bottom row the first three images show the parameters μ_{ki} for each of the three components in the mixture model. As a comparison, we also fit the same data set using a single multivariate Bernoulli distribution, again using maximum likelihood. This amounts to simply averaging the counts in each pixel and is shown by the right-most image on the bottom row.

which represents the intuitively reasonable result that the mixing coefficient for component k is given by the effective fraction of points in the data set explained by that component.

Note that in contrast to the mixture of Gaussians, there are no singularities in which the likelihood function goes to infinity. This can be seen by noting that the likelihood function is bounded above because $0 \leqslant p(\mathbf{x}_n|\boldsymbol{\mu}_k) \leqslant 1$. There exist solutions for which the likelihood function is zero, but these will not be found by EM provided it is not initialized to a pathological starting point, because the EM algorithm always increases the value of the likelihood function, until a local maximum is found.

Exercise 15.19

Section 15.3

We illustrate the Bernoulli mixture model in Figure 15.12 by using it to model handwritten digits. Here the digit images have been turned into binary vectors by setting all elements whose values exceed 0.5 to 1 and setting the remaining elements to 0. We now fit a data set of $N = 600$ such digits, comprising the digits '2', '3', and '4', with a mixture of $K = 3$ Bernoulli distributions by running 10 iterations of the EM algorithm. The mixing coefficients were initialized to $\pi_k = 1/K$, and the parameters μ_{kj} were set to random values chosen uniformly in the range $(0.25, 0.75)$ and then normalized to satisfy the constraint that $\sum_j \mu_{kj} = 1$. We see that a mixture of three Bernoulli distributions is able to find the three clusters in the data set corresponding to the different digits. It is straightforward to extend the analysis of Bernoulli mixtures to the case of multinomial binary variables having $M > 2$ states by making use of the discrete distribution (3.14).

Exercise 15.20

15.4. Evidence Lower Bound

We now present an even more general perspective on the EM algorithm by deriving a lower bound on the log likelihood function, which is known as the *evidence lower bound* or *ELBO*. It is sometimes called a *variational lower bound*. Here the term evidence refers to the (log) likelihood function, which is sometimes called the 'model evidence' in a Bayesian setting as it allows different models to be compared without the use of hold-out data (Bishop, 2006). As an illustration of this bound, we use it to re-derive the EM algorithm for Gaussian mixtures from a third perspective. The ELBO will play an important role in several of the deep generative models discussed in later chapters. It also provides an example of a *variational* framework in which we introduce a distribution $q(\mathbf{Z})$ over the latent variables and then optimize with respect to this distribution using the calculus of variations.

Appendix B

Consider a probabilistic model in which we collectively denote all the observed variables by $\mathbf{X}$ and all the hidden variables by $\mathbf{Z}$. The joint distribution $p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$ is governed by a set of parameters denoted by $\boldsymbol{\theta}$. Our goal is to maximize the likelihood function:

$$p(\mathbf{X}|\boldsymbol{\theta}) = \sum_{\mathbf{Z}} p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta}). \tag{15.51}$$

Here we are assuming that $\mathbf{Z}$ is discrete, although the discussion is identical if $\mathbf{Z}$ comprises continuous variables or a combination of discrete and continuous variables, with summation replaced by integration as appropriate.

We will suppose that direct optimization of $p(\mathbf{X}|\boldsymbol{\theta})$ is difficult, but that optimization of the complete-data likelihood function $p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$ is significantly easier. Next we introduce a distribution $q(\mathbf{Z})$ defined over the latent variables, and we observe that, for any choice of $q(\mathbf{Z})$, the following decomposition holds:

$$\ln p(\mathbf{X}|\boldsymbol{\theta}) = \mathcal{L}(q, \boldsymbol{\theta}) + \mathrm{KL}(q\|p) \tag{15.52}$$

where we have defined

$$\mathcal{L}(q, \boldsymbol{\theta}) = \sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})}{q(\mathbf{Z})} \right\} \tag{15.53}$$

$$\mathrm{KL}(q\|p) = -\sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta})}{q(\mathbf{Z})} \right\}. \tag{15.54}$$

Appendix B

Note that $\mathcal{L}(q, \boldsymbol{\theta})$ is a functional of the distribution $q(\mathbf{Z})$ and a function of the parameters $\boldsymbol{\theta}$. It is worth studying carefully the forms of the expressions (15.53) and (15.54), and in particular noting that they differ in sign and also that $\mathcal{L}(q, \boldsymbol{\theta})$ contains the joint distribution of $\mathbf{X}$ and $\mathbf{Z}$ whereas $\mathrm{KL}(q\|p)$ contains the conditional distribution of $\mathbf{Z}$ given $\mathbf{X}$. To verify the decomposition (15.52), we first make use of the product rule of probability to give

Exercise 15.21

$$\ln p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta}) = \ln p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}) + \ln p(\mathbf{X}|\boldsymbol{\theta}), \tag{15.55}$$

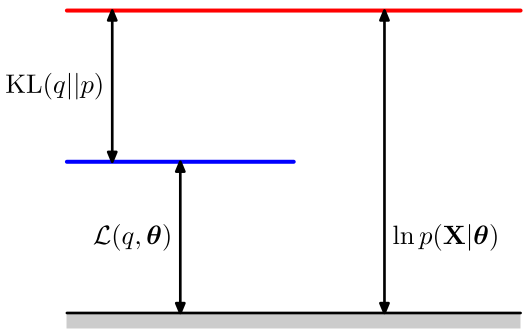

$\mathrm{KL}(q\|p)$
$\mathcal{L}(q, \boldsymbol{\theta})$
$\ln p(\mathbf{X}|\boldsymbol{\theta})$

Figure 15.13 Illustration of the decomposition given by (15.52), which holds for any choice of distribution $q(\mathbf{Z})$. Because the Kullback–Leibler divergence satisfies $\mathrm{KL}(q\|p) \geqslant 0$, we see that the quantity $\mathcal{L}(q, \boldsymbol{\theta})$ is a lower bound on the log likelihood function $\ln p(\mathbf{X}|\boldsymbol{\theta})$.

which we then substitute into the expression for $\mathcal{L}(q, \boldsymbol{\theta})$. This gives rise to two terms, one of which cancels $\mathrm{KL}(q\|p)$ whereas the other gives the required log likelihood $\ln p(\mathbf{X}|\boldsymbol{\theta})$ after noting that $q(\mathbf{Z})$ is a normalized distribution that sums to 1.

From (15.54), we see that $\mathrm{KL}(q\|p)$ is the Kullback–Leibler divergence between $q(\mathbf{Z})$ and the posterior distribution $p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta})$. Recall that the Kullback–Leibler divergence satisfies $\mathrm{KL}(q\|p) \geqslant 0$, with equality if, and only if, $q(\mathbf{Z}) = p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta})$. It therefore follows from (15.52) that $\mathcal{L}(q, \boldsymbol{\theta}) \leqslant \ln p(\mathbf{X}|\boldsymbol{\theta})$, in other words that $\mathcal{L}(q, \boldsymbol{\theta})$ is a lower bound on $\ln p(\mathbf{X}|\boldsymbol{\theta})$. The decomposition (15.52) is illustrated in Figure 15.13. *Section 2.5.7*

15.4.1 EM revisited

We can use the decomposition (15.52) to derive the EM algorithm and to demonstrate that it does indeed maximize the log likelihood. Suppose that the current value of the parameter vector is $\boldsymbol{\theta}^{\text{old}}$. In the E step, the lower bound $\mathcal{L}(q, \boldsymbol{\theta}^{\text{old}})$ is maximized with respect to $q(\mathbf{Z})$ while holding $\boldsymbol{\theta}^{\text{old}}$ fixed. The solution to this maximization problem is easily seen by noting that the value of $\ln p(\mathbf{X}|\boldsymbol{\theta}^{\text{old}})$ does not depend on $q(\mathbf{Z})$ and so the largest value of $\mathcal{L}(q, \boldsymbol{\theta}^{\text{old}})$ will occur when the Kullback–Leibler divergence vanishes, in other words when $q(\mathbf{Z})$ is equal to the posterior distribution $p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{\text{old}})$. In this case, the lower bound will equal the log likelihood, as illustrated in Figure 15.14.

In the subsequent M step, the distribution $q(\mathbf{Z})$ is held fixed and the lower bound

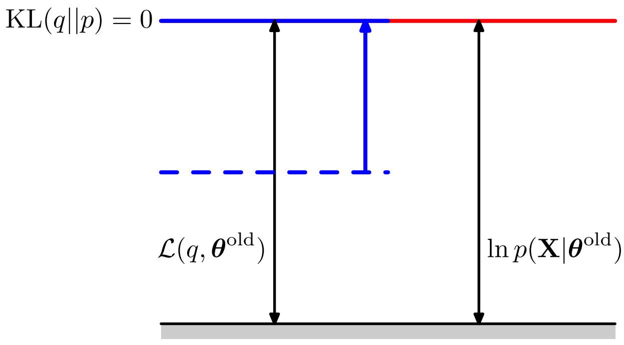

$\mathrm{KL}(q\|p) = 0$
$\mathcal{L}(q, \boldsymbol{\theta}^{\text{old}})$
$\ln p(\mathbf{X}|\boldsymbol{\theta}^{\text{old}})$

Figure 15.14 Illustration of the E step of the EM algorithm. The q distribution is set equal to the posterior distribution for the current parameter values $\boldsymbol{\theta}^{\text{old}}$, causing the lower bound to move up to the same value as the log likelihood function, with the KL divergence vanishing.

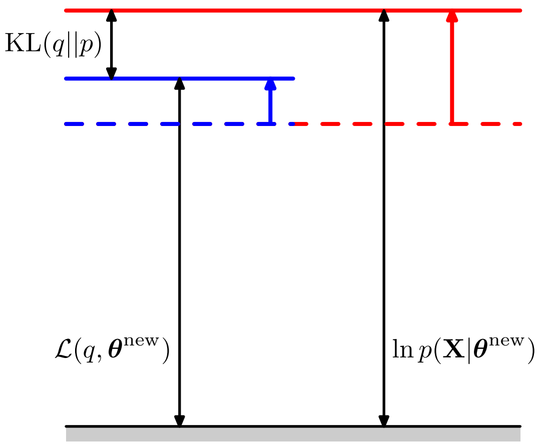

$\mathrm{KL}(q\|p)$
$\mathcal{L}(q, \boldsymbol{\theta}^{\mathrm{new}})$
$\ln p(\mathbf{X}|\boldsymbol{\theta}^{\mathrm{new}})$

Figure 15.15 Illustration of the M step of the EM algorithm. The distribution $q(\mathbf{Z})$ is held fixed and the lower bound $\mathcal{L}(q, \boldsymbol{\theta})$ is maximized with respect to the parameter vector $\boldsymbol{\theta}$ to give a revised value $\boldsymbol{\theta}^{\mathrm{new}}$. Because the Kullback–Leibler divergence is non-negative, this causes the log likelihood $\ln p(\mathbf{X}|\boldsymbol{\theta})$ to increase by at least as much as the lower bound does.

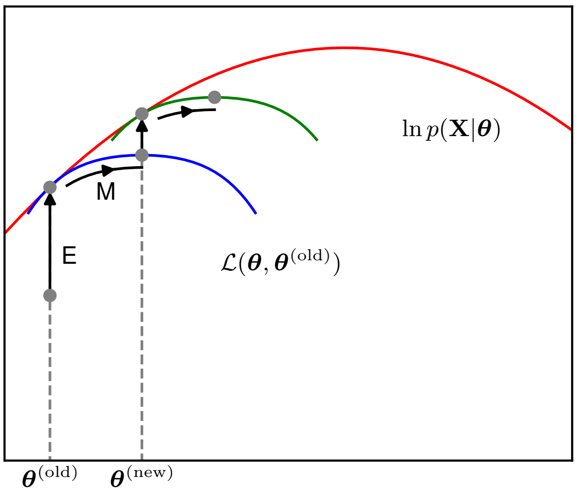
$\mathcal{L}(q, \boldsymbol{\theta})$ is maximized with respect to $\boldsymbol{\theta}$ to give some new value $\boldsymbol{\theta}^{\mathrm{new}}$. This will cause the lower bound $\mathcal{L}$ to increase (unless it is already at a maximum), which will necessarily cause the corresponding log likelihood function to increase. Because the distribution q is determined using the old parameter values rather than the new values and is held fixed during the M step, it will not equal the new posterior distribution $p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{\mathrm{new}})$, and hence there will be a non-zero Kullback–Leibler divergence. The increase in the log likelihood function is therefore greater than the increase in the lower bound, as shown in Figure 15.15. If we substitute $q(\mathbf{Z}) = p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{\mathrm{old}})$ into (15.53), we see that, after the E step, the lower bound takes the form

$$\begin{aligned}\mathcal{L}(q, \boldsymbol{\theta}) &= \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{\mathrm{old}}) \ln p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta}) - \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{\mathrm{old}}) \ln p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{\mathrm{old}}) \\ &= \mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{\mathrm{old}}) + \mathrm{const} \qquad (15.56)\end{aligned}$$

where the constant is simply the negative entropy of the q distribution and is therefore independent of $\boldsymbol{\theta}$. Here we recognize $\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{\mathrm{old}})$ as the expected complete-data log-likelihood defined by (15.23), and it is therefore the quantity that is being maximized in the M step, as we saw earlier distribution for mixtures of Gaussians. Note that the variable $\boldsymbol{\theta}$ over which we are optimizing appears only inside the logarithm. If the joint distribution $p(\mathbf{Z}, \mathbf{X}|\boldsymbol{\theta})$ is a member of the exponential family or a product of such members, then we see that the logarithm will cancel the exponential and lead to an M step that will be typically much simpler than the maximization of the corresponding incomplete-data log likelihood function $p(\mathbf{X}|\boldsymbol{\theta})$.

Section 15.3

The operation of the EM algorithm can also be viewed in the space of parameters, as illustrated schematically in Figure 15.16. Here the red curve depicts the (incomplete-data) log likelihood function whose value we wish to maximize. We start with some initial parameter value $\boldsymbol{\theta}^{\mathrm{old}}$, and in the first E step we evaluate the posterior distribution over latent variables, which gives rise to a lower bound $\mathcal{L}(q, \boldsymbol{\theta})$ whose value equals the log likelihood at $\boldsymbol{\theta}^{(\mathrm{old})}$, as shown by the blue curve. Note that the bound makes a tangential contact with the log likelihood at $\boldsymbol{\theta}^{(\mathrm{old})}$, so that both



$\ln p(\mathbf{X}|\boldsymbol{\theta})$
M
E
$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\theta}^{(\text{old})})$
$\boldsymbol{\theta}^{(\text{old})}$
$\boldsymbol{\theta}^{(\text{new})}$

Figure 15.16 The EM algorithm involves alternately computing a lower bound on the log likelihood for the current parameter values and then maximizing this bound to obtain the new parameter values. See the text for a full discussion.

Exercise 15.22

curves have the same gradient. This bound is a convex function having a unique maximum (for mixture components from the exponential family). In the M step, the bound is maximized giving the value $\boldsymbol{\theta}^{(\text{new})}$, which gives a larger value of the log likelihood than $\boldsymbol{\theta}^{(\text{old})}$. The subsequent E step then constructs a bound that is tangential at $\boldsymbol{\theta}^{(\text{new})}$ as shown by the green curve.

We have seen that both the E and the M steps of the EM algorithm increase the value of a well-defined bound on the log likelihood function and that the complete EM cycle will change the model parameters in such a way as to cause the log likelihood to increase (unless it is already at a maximum, in which case the parameters remain unchanged).

15.4.2 Independent and identically distributed data

For the particular case of an i.i.d. data set, $\mathbf{X}$ will comprise N data points $\{\mathbf{x}_n\}$ whereas $\mathbf{Z}$ will comprise N corresponding latent variables $\{\mathbf{z}_n\}$, where $n = 1, \ldots, N$. From the independence assumption, we have $p(\mathbf{X}, \mathbf{Z}) = \prod_n p(\mathbf{x}_n, \mathbf{z}_n)$, and by marginalizing over the $\{\mathbf{z}_n\}$ we have $p(\mathbf{X}) = \prod_n p(\mathbf{x}_n)$. Using the sum and product rules, we see that the posterior probability that is evaluated in the E step takes the form

$$p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}) = \frac{p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})}{\sum_{\mathbf{Z}} p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})} = \frac{\prod_{n=1}^{N} p(\mathbf{x}_n, \mathbf{z}_n|\boldsymbol{\theta})}{\sum_{\mathbf{Z}} \prod_{n=1}^{N} p(\mathbf{x}_n, \mathbf{z}_n|\boldsymbol{\theta})} = \prod_{n=1}^{N} p(\mathbf{z}_n|\mathbf{x}_n, \boldsymbol{\theta}) \quad (15.57)$$

and so the posterior distribution also factorizes with respect to n. For a Gaussian mixture model, this simply says that the responsibility that each of the mixture components takes for a particular data point $\mathbf{x}_n$ depends only on the value of $\mathbf{x}_n$ and on the parameters $\boldsymbol{\theta}$ of the mixture components, not on the values of the other data points.

15.4.3 Parameter priors

We can also use the EM algorithm to maximize the posterior distribution $p(\boldsymbol{\theta}|\mathbf{X})$ for models in which we have introduced a prior $p(\boldsymbol{\theta})$ over the parameters. To see this, note that as a function of $\boldsymbol{\theta}$, we have $p(\boldsymbol{\theta}|\mathbf{X}) = p(\boldsymbol{\theta}, \mathbf{X})/p(\mathbf{X})$ and so

$$\ln p(\boldsymbol{\theta}|\mathbf{X}) = \ln p(\boldsymbol{\theta}, \mathbf{X}) - \ln p(\mathbf{X}). \tag{15.58}$$

Making use of the decomposition (15.52), we have

$$\begin{aligned} \ln p(\boldsymbol{\theta}|\mathbf{X}) &= \mathcal{L}(q, \boldsymbol{\theta}) + \mathrm{KL}(q\|p) + \ln p(\boldsymbol{\theta}) - \ln p(\mathbf{X}) \\ &\geqslant \mathcal{L}(q, \boldsymbol{\theta}) + \ln p(\boldsymbol{\theta}) - \ln p(\mathbf{X}) \end{aligned} \tag{15.59}$$

where $\ln p(\mathbf{X})$ is a constant. We can again optimize the right-hand side alternately with respect to q and $\boldsymbol{\theta}$. The optimization with respect to q gives rise to the same E-step equations as for the standard EM algorithm, because q appears only in $\mathcal{L}(q, \boldsymbol{\theta})$. The M-step equations are modified through the introduction of the prior term $\ln p(\boldsymbol{\theta})$, which typically requires only a small modification to the standard maximum likelihood M-step equations. The additional term represents a form of regularization and has the effect of removing the singularities of the likelihood function for Gaussian mixture models.

Chapter 9

15.4.4 Generalized EM

The EM algorithm breaks down the potentially difficult problem of maximizing the likelihood function into two stages, the E step and the M step, each of which will often prove simpler to implement. Nevertheless, for complex models it may be the case that either the E step or the M step, or indeed both, remain intractable. This leads to two possible extensions of the EM algorithm, as follows.

The *generalized EM*, or *GEM*, algorithm addresses the problem of an intractable M step. Instead of aiming to maximize $\mathcal{L}(q, \boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$, it seeks instead to change the parameters in such a way as to increase its value. Again, because $\mathcal{L}(q, \boldsymbol{\theta})$ is a lower bound on the log likelihood function, each complete EM cycle of the GEM algorithm is guaranteed to increase the value of the log likelihood (unless the parameters already correspond to a local maximum). One way to exploit the GEM approach would be to use gradient-based iterative optimization algorithms during the M step. Another form of GEM algorithm, known as the *expectation conditional maximization* algorithm, involves making several constrained optimizations within each M step (Meng and Rubin, 1993). For instance, the parameters might be partitioned into groups and the M step broken down into multiple steps each of which involves optimizing one of the groups with the remainder held fixed.

We can similarly generalize the E step of the EM algorithm by performing a partial, rather than complete, optimization of $\mathcal{L}(q, \boldsymbol{\theta})$ with respect to $q(\mathbf{Z})$ (Neal and Hinton, 1999). As we have seen, for any given value of $\boldsymbol{\theta}$ there is a unique maximum of $\mathcal{L}(q, \boldsymbol{\theta})$ with respect to $q(\mathbf{Z})$ that corresponds to the posterior distribution $q_{\boldsymbol{\theta}}(\mathbf{Z}) =$

$p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta})$ and that for this choice of $q(\mathbf{Z})$, the bound $\mathcal{L}(q, \boldsymbol{\theta})$ is equal to the log likelihood function $\ln p(\mathbf{X}|\boldsymbol{\theta})$. It follows that any algorithm that converges to the global maximum of $\mathcal{L}(q, \boldsymbol{\theta})$ will find a value of $\boldsymbol{\theta}$ that is also a global maximum of the log likelihood $\ln p(\mathbf{X}|\boldsymbol{\theta})$. Provided $p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$ is a continuous function of $\boldsymbol{\theta}$ then, by continuity, any local maximum of $\mathcal{L}(q, \boldsymbol{\theta})$ will also be a local maximum of $\ln p(\mathbf{X}|\boldsymbol{\theta})$.

15.4.5 Sequential EM

Consider N independent data points $\mathbf{x}_1, \ldots, \mathbf{x}_N$ with corresponding latent variables $\mathbf{z}_1, \ldots, \mathbf{z}_N$. The joint distribution $p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$ factorizes over the data points, and this structure can be exploited in an incremental form of EM in which at each EM cycle, only one data point is processed at a time. In the E step, instead of recomputing the responsibilities for all the data points, we just re-evaluate the responsibilities for one data point. It might appear that the subsequent M step would require a computation involving the responsibilities for all the data points. However, if the mixture components are members of the exponential family, then the responsibilities enter only through simple sufficient statistics, and these can be updated efficiently. Consider, for instance, a Gaussian mixture, and suppose we perform an update for data point m in which the corresponding old and new values of the responsibilities are denoted by $\gamma^{\text{old}}(z_{mk})$ and $\gamma^{\text{new}}(z_{mk})$. In the M step, the required sufficient statistics can be updated incrementally. For instance, for the means, the sufficient statistics are defined by (15.16) and (15.17) from which we obtain

Exercise 15.23

$$\boldsymbol{\mu}_k^{\text{new}} = \boldsymbol{\mu}_k^{\text{old}} + \left(\frac{\gamma^{\text{new}}(z_{mk}) - \gamma^{\text{old}}(z_{mk})}{N_k^{\text{new}}}\right)\left(\mathbf{x}_m - \boldsymbol{\mu}_k^{\text{old}}\right) \tag{15.60}$$

together with

$$N_k^{\text{new}} = N_k^{\text{old}} + \gamma^{\text{new}}(z_{mk}) - \gamma^{\text{old}}(z_{mk}). \tag{15.61}$$

The corresponding results for the covariances and the mixing coefficients are analogous.

Thus, both the E step and the M step take a fixed time that is independent of the total number of data points. Because the parameters are revised after each data point, rather than waiting until after the whole data set is processed, this incremental version can converge faster than the batch version. Each E or M step in this incremental algorithm increases the value of $\mathcal{L}(q, \boldsymbol{\theta})$, and as we have shown above, if the algorithm converges to a local (or global) maximum of $\mathcal{L}(q, \boldsymbol{\theta})$, this will correspond to a local (or global) maximum of the log likelihood function $\ln p(\mathbf{X}|\boldsymbol{\theta})$.

Exercises

15.1 ($\star$) Consider the K-means algorithm discussed in Section 15.1. Show that as a consequence of there being a finite number of possible assignments for the set of discrete indicator variables r_{nk} and that for each such assignment there is a unique optimum for the $\{\boldsymbol{\mu}_k\}$, the K-means algorithm must converge after a finite number of iterations.