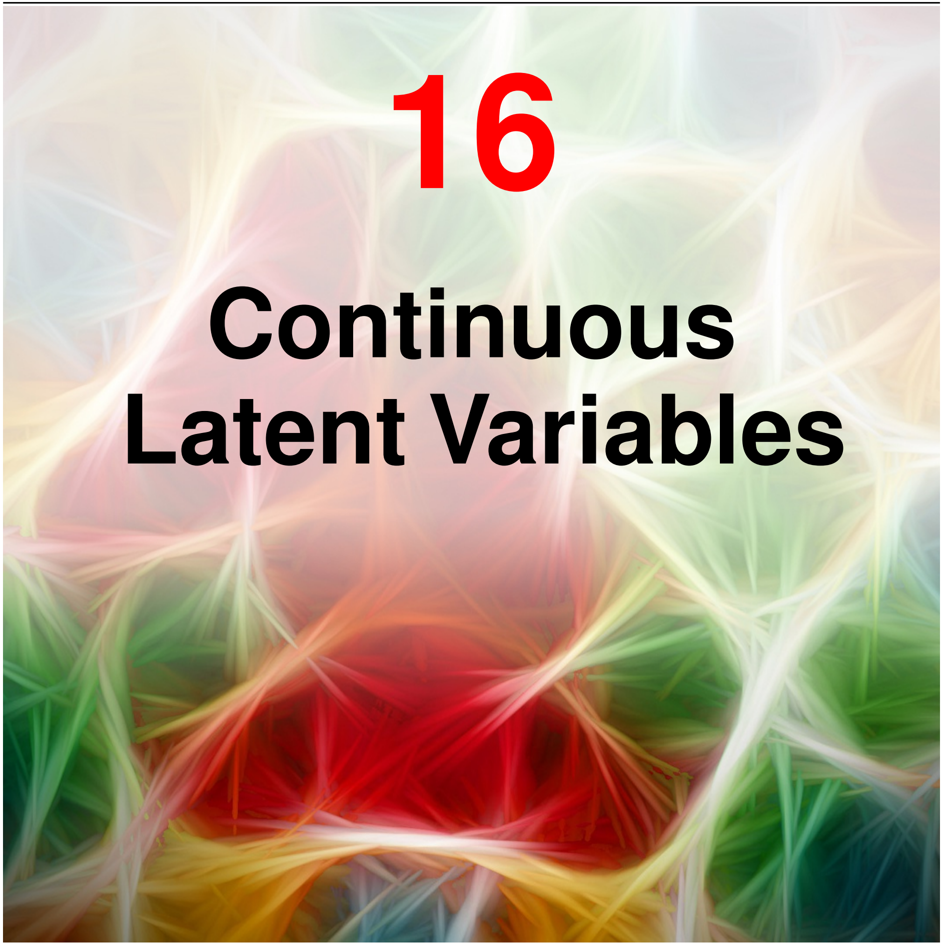


16 Continuous Latent Variables

In the previous chapter we discussed probabilistic models having discrete latent variables, such as a mixture of Gaussians. We now explore models in which some, or all, of the latent variables are continuous. An important motivation for such models

Section 6.1.3

is that many data sets have the property that the data points lie close to a manifold of much lower dimensionality than that of the original data space. To see why this might arise, consider an artificial data set constructed by taking a handwritten digit from the MNIST data set (LeCun *et al.*, 1998), represented by a 64×64 pixel grey-level image, and embedding it in a larger image of size 100×100 by padding with pixels having the value zero (corresponding to white pixels) in which the location and orientation of the digit are varied at random, as illustrated in Figure 16.1. Each of the resulting images is represented by a point in the $100 \times 100 = 10{,}000$-dimensional data space. However, across a data set of such images, there are only three *degrees of freedom* of variability, corresponding to vertical and horizontal translations and

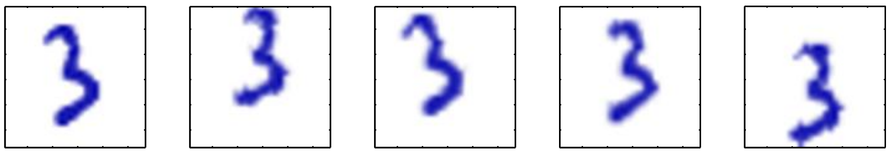


Figure 16.1 A synthetic data set obtained by taking an image of a handwritten digit and creating multiple copies in each of which the digit has undergone a random displacement and rotation within some larger image field. The resulting images each have $100 \times 100 = 10{,}000$ pixels.

rotations. The data points will therefore live on a subspace of the data space whose *intrinsic dimensionality* is three. Note that the manifold will be nonlinear because, for instance, if we translate the digit past a particular pixel, that pixel value will go from zero (white) to one (black) and back to zero again, which is clearly a nonlinear function of the digit position. In this example, the translation and rotation parameters are latent variables because we observe only the image vectors and are not told which values of the translation or rotation variables were used to create them.

For real data sets of handwritten digits, there will be further degrees of freedom arising from scaling and other variations due, for example, to the variability in an individual's writing as well as the differences in writing styles between individuals. Nevertheless, the number of such degrees of freedom will be small compared to the dimensionality of the data set.

In practice, the data points will not be confined precisely to a smooth low-dimensional manifold, and we can interpret the departures of data points from the manifold as 'noise'. This leads naturally to a generative view of such models in which we first select a point within the manifold according to some latent-variable distribution and then generate an observed data point by adding noise drawn from some conditional distribution of the data variables given the latent variables.

The simplest continuous latent-variable model assumes Gaussian distributions for both the latent and observed variables and makes use of a linear-Gaussian dependence of the observed variables on the state of the latent variables. This leads to a probabilistic formulation of the well-known technique of principal component analysis (PCA) as well as to a related model called factor analysis. In this chapter we will begin with a standard, non-probabilistic treatment of PCA, and then we show how PCA arises naturally as the maximum likelihood solution for a linear-Gaussian latent-variable model. This probabilistic reformulation brings many advantages, such as the use of EM for parameter estimation, principled extensions to mixtures of PCA models, and Bayesian formulations that allow the number of principal components to be determined automatically from the data (Bishop, 2006). This chapter also lays the foundations for nonlinear models having continuous latent variables including normalizing flows, variational autoencoders, and diffusion models.

Section 11.1.4

Section 16.1

Section 16.2

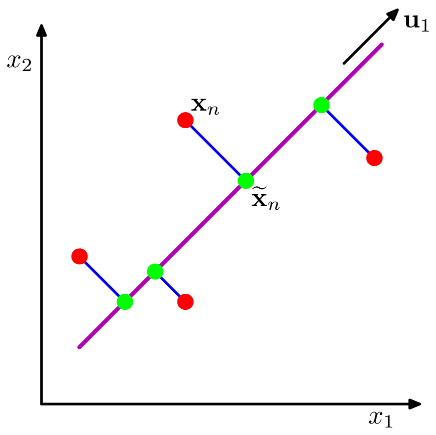


Figure 16.2 Principal component analysis seeks a space of lower dimensionality, known as the principal subspace and denoted by the magenta line, such that the orthogonal projection of the data points (red dots) onto this subspace maximizes the variance of the projected points (green dots). An alternative definition of PCA is based on minimizing the sum-of-squares of the projection errors, indicated by the blue lines.

16.1. Principal Component Analysis

Principal component analysis, or PCA, is widely used for applications such as dimensionality reduction, lossy data compression, feature extraction, and data visualization (Jolliffe, 2002). It is also known as the *Kosambi–Karhunen–Loève* transform.

Consider the orthogonal projection of a data set onto a lower-dimensional linear space, known as the *principal subspace*, as shown in Figure 16.2. PCA can be defined as the linear projection that maximizes the variance of the projected data (Hotelling, 1933). Equivalently, it can be defined as the linear projection that minimizes the average projection cost, defined as the mean squared distance between the data points and their projections (Pearson, 1901). We consider each of these definitions in turn.

16.1.1 Maximum variance formulation

Consider a data set of observations $\{\mathbf{x}_n\}$ where $n = 1, \ldots, N$, and $\mathbf{x}_n$ is a Euclidean variable with dimensionality D. Our goal is to project the data onto a space having dimensionality $M < D$ while maximizing the variance of the projected data. For the moment, we will assume that the value of M is given. Later in this chapter, we will consider techniques to determine an appropriate value of M from the data.

To begin with, consider the projection onto a one-dimensional space ($M = 1$). We can define the direction of this space using a D-dimensional vector $\mathbf{u}_1$, which for convenience (and without loss of generality) we will choose to be a unit vector so that $\mathbf{u}_1^{\mathrm{T}}\mathbf{u}_1 = 1$ (note that we are interested only in the direction defined by $\mathbf{u}_1$, not in the magnitude of $\mathbf{u}_1$ itself). Each data point $\mathbf{x}_n$ is then projected onto a scalar value $\mathbf{u}_1^{\mathrm{T}}\mathbf{x}_n$. The mean of the projected data is $\mathbf{u}_1^{\mathrm{T}}\overline{\mathbf{x}}$ where $\overline{\mathbf{x}}$ is the sample set mean given by

$$\overline{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n \tag{16.1}$$

and the variance of the projected data is given by

$$\frac{1}{N}\sum_{n=1}^{N}\left\{\mathbf{u}_1^{\mathrm{T}}\mathbf{x}_n - \mathbf{u}_1^{\mathrm{T}}\overline{\mathbf{x}}\right\}^2 = \mathbf{u}_1^{\mathrm{T}}\mathbf{S}\mathbf{u}_1 \tag{16.2}$$

where $\mathbf{S}$ is the data covariance matrix defined by

$$\mathbf{S} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n - \overline{\mathbf{x}})(\mathbf{x}_n - \overline{\mathbf{x}})^{\mathrm{T}}. \tag{16.3}$$

We now maximize the projected variance $\mathbf{u}_1^{\mathrm{T}}\mathbf{S}\mathbf{u}_1$ with respect to $\mathbf{u}_1$. Clearly, this has to be a constrained maximization to prevent $\|\mathbf{u}_1\| \to \infty$. The appropriate constraint comes from the normalization condition $\mathbf{u}_1^{\mathrm{T}}\mathbf{u}_1 = 1$. To enforce this constraint, we introduce a Lagrange multiplier that we will denote by λ_1, and then make an unconstrained maximization of

Appendix C

$$\mathbf{u}_1^{\mathrm{T}}\mathbf{S}\mathbf{u}_1 + \lambda_1\left(1 - \mathbf{u}_1^{\mathrm{T}}\mathbf{u}_1\right). \tag{16.4}$$

By setting the derivative with respect to $\mathbf{u}_1$ equal to zero, we see that this quantity will have a stationary point when

$$\mathbf{S}\mathbf{u}_1 = \lambda_1\mathbf{u}_1, \tag{16.5}$$

which says that $\mathbf{u}_1$ must be an eigenvector of $\mathbf{S}$. If we left-multiply by $\mathbf{u}_1^{\mathrm{T}}$ and make use of $\mathbf{u}_1^{\mathrm{T}}\mathbf{u}_1 = 1$, we see that the variance is given by

$$\mathbf{u}_1^{\mathrm{T}}\mathbf{S}\mathbf{u}_1 = \lambda_1 \tag{16.6}$$

and so the variance will be a maximum when we set $\mathbf{u}_1$ equal to the eigenvector having the largest eigenvalue λ_1. This eigenvector is known as the first principal component.

We can define additional principal components in an incremental fashion by choosing each new direction to be that which maximizes the projected variance amongst all possible directions orthogonal to those already considered. If we consider the general case of an M-dimensional projection space, the optimal linear projection for which the variance of the projected data is maximized is now defined by the M eigenvectors $\mathbf{u}_1, \ldots, \mathbf{u}_M$ of the data covariance matrix $\mathbf{S}$ corresponding to the M largest eigenvalues $\lambda_1, \ldots, \lambda_M$. This is easily shown using proof by induction.

Exercise 16.1

To summarize, PCA involves evaluating the mean $\overline{\mathbf{x}}$ and the covariance matrix $\mathbf{S}$ of a data set and then finding the M eigenvectors of $\mathbf{S}$ corresponding to the M largest eigenvalues. Algorithms for finding eigenvectors and eigenvalues, as well as additional theorems related to eigenvector decomposition, can be found in Golub and Van Loan (1996). Note that the computational cost of computing the full eigenvector decomposition for a matrix of size $D \times D$ is $\mathcal{O}(D^3)$. If we plan to project our data onto the first M principal components, then we only need to find the first M eigenvalues and eigenvectors. This can be done with more efficient techniques, such as the *power method* (Golub and Van Loan, 1996), that scale like $\mathcal{O}(MD^2)$, or alternatively we can make use of the EM algorithm.

Section 16.3.2

16.1.2 Minimum-error formulation

Appendix A

We now discuss an alternative formulation of PCA based on projection error minimization. To do this, we introduce a complete orthonormal set of D-dimensional basis vectors $\{\mathbf{u}_i\}$ where $i = 1, \dots, D$ that satisfy

$$\mathbf{u}_i^{\mathrm{T}}\mathbf{u}_j = \delta_{ij}. \tag{16.7}$$

Because this basis is complete, each data point can be represented exactly by a linear combination of the basis vectors

$$\mathbf{x}_n = \sum_{i=1}^{D} \alpha_{ni}\mathbf{u}_i \tag{16.8}$$

where the coefficients α_{ni} will be different for different data points. This simply corresponds to a rotation of the coordinate system to a new system defined by the $\{\mathbf{u}_i\}$, and the original D components $\{x_{n1}, \dots, x_{nD}\}$ are replaced by an equivalent set $\{\alpha_{n1}, \dots, \alpha_{nD}\}$. Taking the inner product with $\mathbf{u}_j$, and making use of the orthonormality property, we obtain $\alpha_{nj} = \mathbf{x}_n^{\mathrm{T}}\mathbf{u}_j$, and so without loss of generality we can write

$$\mathbf{x}_n = \sum_{i=1}^{D} \left(\mathbf{x}_n^{\mathrm{T}}\mathbf{u}_i\right)\mathbf{u}_i. \tag{16.9}$$

Our goal, however, is to approximate this data point using a representation involving a restricted number $M < D$ of variables corresponding to a projection onto a lower-dimensional subspace. The M-dimensional linear subspace can be represented, without loss of generality, by the first M of the basis vectors, and so we approximate each data point $\mathbf{x}_n$ by

$$\widetilde{\mathbf{x}}_n = \sum_{i=1}^{M} z_{ni}\mathbf{u}_i + \sum_{i=M+1}^{D} b_i\mathbf{u}_i \tag{16.10}$$

where the $\{z_{ni}\}$ depend on the particular data point, whereas the $\{b_i\}$ are constants that are the same for all data points. We are free to choose the $\{\mathbf{u}_i\}$, the $\{z_{ni}\}$, and the $\{b_i\}$ so as to minimize the error introduced by the reduction in dimensionality. As our error measure, we will use the squared distance between the original data point $\mathbf{x}_n$ and its approximation $\widetilde{\mathbf{x}}_n$, averaged over the data set, so that our goal is to minimize

$$J = \frac{1}{N}\sum_{n=1}^{N} \|\mathbf{x}_n - \widetilde{\mathbf{x}}_n\|^2. \tag{16.11}$$

Consider first the minimization with respect to the quantities $\{z_{ni}\}$. Substituting for $\widetilde{\mathbf{x}}_n$, setting the derivative with respect to z_{nj} to zero, and making use of the orthonormality conditions, we obtain

$$z_{nj} = \mathbf{x}_n^{\mathrm{T}}\mathbf{u}_j \tag{16.12}$$

where $j = 1, \ldots, M$. Similarly, setting the derivative of J with respect to b_i to zero and again making use of the orthonormality relations, gives

$$b_j = \overline{\mathbf{x}}^{\mathrm{T}} \mathbf{u}_j \tag{16.13}$$

where $j = M+1, \ldots, D$. If we substitute for z_{ni} and b_i and make use of the general expansion (16.9), we obtain

$$\mathbf{x}_n - \widetilde{\mathbf{x}}_n = \sum_{i=M+1}^{D} \left\{ (\mathbf{x}_n - \overline{\mathbf{x}})^{\mathrm{T}} \mathbf{u}_i \right\} \mathbf{u}_i \tag{16.14}$$

from which we see that the displacement vector from $\mathbf{x}_n$ to $\widetilde{\mathbf{x}}_n$ lies in the space orthogonal to the principal subspace, because it is a linear combination of $\{\mathbf{u}_i\}$ for $i = M + 1, \ldots, D$, as illustrated in Figure 16.2. This is to be expected because the projected points $\widetilde{\mathbf{x}}_n$ must lie within the principal subspace, but we can move them freely within that subspace, and so the minimum error is given by the orthogonal projection.

We therefore obtain an expression for the error measure J as a function purely of the $\{\mathbf{u}_i\}$ in the form

$$J = \frac{1}{N} \sum_{n=1}^{N} \sum_{i=M+1}^{D} \left(\mathbf{x}_n^{\mathrm{T}} \mathbf{u}_i - \overline{\mathbf{x}}^{\mathrm{T}} \mathbf{u}_i \right)^2 = \sum_{i=M+1}^{D} \mathbf{u}_i^{\mathrm{T}} \mathbf{S} \mathbf{u}_i. \tag{16.15}$$

There remains the task of minimizing J with respect to the $\{\mathbf{u}_i\}$, which must be a constrained minimization otherwise we will obtain the vacuous result $\mathbf{u}_i = 0$. The constraints arise from the orthonormality conditions, and as we will see, the solution will be expressed in terms of the eigenvector expansion of the covariance matrix. Before considering a formal solution, let us try to obtain some intuition about the result by considering a two-dimensional data space $D = 2$ and a one-dimensional principal subspace $M = 1$. We have to choose a direction $\mathbf{u}_2$ so as to minimize $J = \mathbf{u}_2^{\mathrm{T}} \mathbf{S} \mathbf{u}_2$, subject to the normalization constraint $\mathbf{u}_2^{\mathrm{T}} \mathbf{u}_2 = 1$. Using a Lagrange multiplier λ_2 to enforce the constraint, we consider the minimization of

$$\widetilde{J} = \mathbf{u}_2^{\mathrm{T}} \mathbf{S} \mathbf{u}_2 + \lambda_2 \left(1 - \mathbf{u}_2^{\mathrm{T}} \mathbf{u}_2 \right). \tag{16.16}$$

Setting the derivative with respect to $\mathbf{u}_2$ to zero, we obtain $\mathbf{S}\mathbf{u}_2 = \lambda_2 \mathbf{u}_2$ so that $\mathbf{u}_2$ is an eigenvector of $\mathbf{S}$ with eigenvalue λ_2. Thus, any eigenvector will define a stationary point of the error measure. To find the value of J at the minimum, we back-substitute the solution for $\mathbf{u}_2$ into the error measure to give $J = \lambda_2$. We therefore obtain the minimum value of J by choosing $\mathbf{u}_2$ to be the eigenvector corresponding to the smaller of the two eigenvalues. Thus, we should choose the principal subspace to be aligned with the eigenvector having the *larger* eigenvalue. This result accords with our intuition that, to minimize the average squared projection distance, we should choose the principal component subspace so that it passes through the mean of the data points and is aligned with the directions of maximum variance. If the eigenvalues are equal, any choice of principal direction will give rise to the same value of J.

Exercise 16.2

The general solution to the minimization of J for arbitrary D and arbitrary $M < D$ is obtained by choosing the $\{\mathbf{u}_i\}$ to be eigenvectors of the covariance matrix given by

$$\mathbf{S}\mathbf{u}_i = \lambda_i \mathbf{u}_i \tag{16.17}$$

where $i = 1, \ldots, D$, and as usual the eigenvectors $\{\mathbf{u}_i\}$ are chosen to be orthonormal. The corresponding value of the error measure is then given by

$$J = \sum_{i=M+1}^{D} \lambda_i, \tag{16.18}$$

which is simply the sum of the eigenvalues of those eigenvectors that are orthogonal to the principal subspace. We therefore obtain the minimum value of J by selecting these eigenvectors to be those having the $D - M$ smallest eigenvalues, and hence the eigenvectors defining the principal subspace are those corresponding to the M largest eigenvalues.

Although we have considered $M < D$, the PCA analysis still holds if $M = D$, in which case there is no dimensionality reduction but simply a rotation of the coordinate axes to align with the principal components.

Finally, note that there is a related linear dimensionality reduction technique called *canonical correlation analysis* (Hotelling, 1936; Bach and Jordan, 2002). Whereas PCA works with a single random variable, canonical correlation analysis considers two (or more) variables and tries to find a corresponding pair of linear subspaces that have high cross-correlation, so that each component within one of the subspaces is correlated with a single component from the other subspace. Its solution can be expressed in terms of a generalized eigenvector problem.

16.1.3 Data compression

One application for PCA is data compression, and we can illustrate this by considering a data set of images of handwritten digits. Because each eigenvector of the covariance matrix is a vector in the original D-dimensional space, we can represent the eigenvectors as images of the same size as the data points. The mean image and the first four eigenvectors, along with their corresponding eigenvalues, are shown in Figure 16.3.

A plot of the complete spectrum of eigenvalues, sorted into decreasing order, is shown in Figure 16.4(a). The error measure J associated with choosing a particular value of M is given by the sum of the eigenvalues from $M+1$ up to D and is plotted for different values of M in Figure 16.4(b).

If we substitute (16.12) and (16.13) into (16.10), we can write the PCA approx-

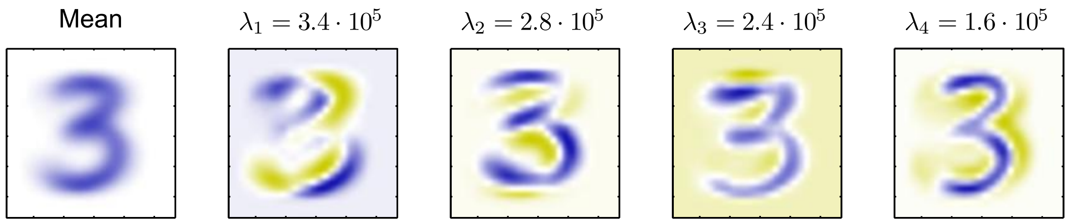

Mean
$\lambda_1 = 3.4 \cdot 10^5$
$\lambda_2 = 2.8 \cdot 10^5$
$\lambda_3 = 2.4 \cdot 10^5$
$\lambda_4 = 1.6 \cdot 10^5$

Figure 16.3 Illustration of PCA applied to a data set of 6,000 images of size 28×28, each comprising a hand-written image of the numeral '3' , showing the mean vector $\overline{\mathbf{x}}$ along with the first four PCA eigenvectors $\mathbf{u}_1, \ldots, \mathbf{u}_4$, together with their corresponding eigenvalues.

imation to a data vector $\mathbf{x}_n$ in the form

$$\widetilde{\mathbf{x}}_n = \sum_{i=1}^{M}(\mathbf{x}_n^{\mathrm{T}}\mathbf{u}_i)\mathbf{u}_i + \sum_{i=M+1}^{D}(\overline{\mathbf{x}}^{\mathrm{T}}\mathbf{u}_i)\mathbf{u}_i \tag{16.19}$$

$$= \overline{\mathbf{x}} + \sum_{i=1}^{M}\left(\mathbf{x}_n^{\mathrm{T}}\mathbf{u}_i - \overline{\mathbf{x}}^{\mathrm{T}}\mathbf{u}_i\right)\mathbf{u}_i \tag{16.20}$$

where we have made use of the relation

$$\overline{\mathbf{x}} = \sum_{i=1}^{D}\left(\overline{\mathbf{x}}^{\mathrm{T}}\mathbf{u}_i\right)\mathbf{u}_i, \tag{16.21}$$

which follows from the completeness of the $\{\mathbf{u}_i\}$. This represents a compression of the data set, because for each data point we have replaced the D-dimensional vector $\mathbf{x}_n$ with an M-dimensional vector having components $\left(\mathbf{x}_n^{\mathrm{T}}\mathbf{u}_i - \overline{\mathbf{x}}^{\mathrm{T}}\mathbf{u}_i\right)$. The smaller the value of M, the greater the degree of compression. Examples of PCA reconstructions of data points for the digits data set are shown in Figure 16.5.

16.1.4 Data whitening

Another application of PCA is to data pre-processing. In this case, the goal is not dimensionality reduction but rather the transformation of a data set to standardize certain of its properties. This can be important in allowing subsequent machine learning algorithms to be applied successfully to the data set. Typically, it is done when the original variables are measured in various different units or have significantly different variabilities. For instance in the Old Faithful data set, the time between eruptions is typically an order of magnitude greater than the duration of an eruption. When we applied the K-means algorithm to this data set, we first made a separate linear re-scaling of the individual variables such that each variable had zero

Section 15.1

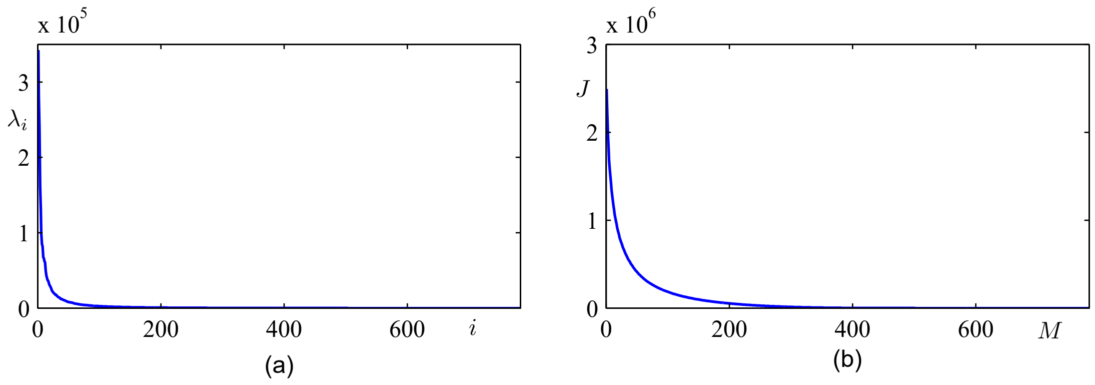


Figure 16.4 (a) Plot of the eigenvalue spectrum for the data set of handwritten digits used in Figure 16.3. (b) Plot of the sum of the discarded eigenvalues, which represents the sum-of-squares error J introduced by projecting the data onto a principal component subspace of dimensionality M.

mean and unit variance. This is known as *standardizing* the data, and the covariance matrix for the standardized data has components

$$\rho_{ij} = \frac{1}{N}\sum_{n=1}^{N}\frac{(x_{ni}-\overline{x}_i)}{\sigma_i}\frac{(x_{nj}-\overline{x}_j)}{\sigma_j} \tag{16.22}$$

where σ_i is the standard deviation of x_i. This is known as the *correlation* matrix of the original data and has the property that if two components x_i and x_j of the data are perfectly correlated, then $\rho_{ij} = 1$, and if they are uncorrelated, then $\rho_{ij} = 0$.

However, using PCA we can make a more substantial normalization of the data to give it zero mean and unit covariance, so that different variables become decorre-

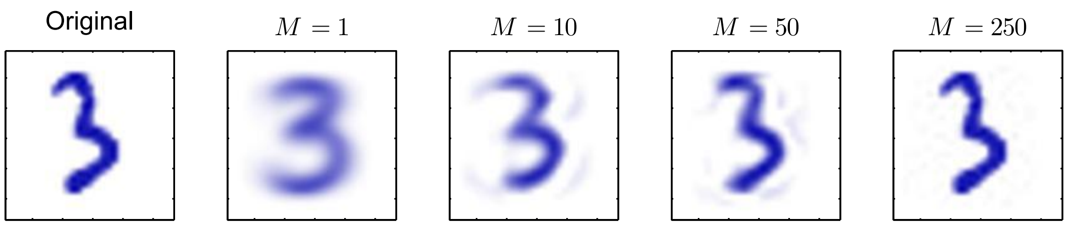


Figure 16.5 An example from the data set of handwritten digits together with its PCA reconstructions obtained by retaining M principal components for various values of M. As M increases, the reconstruction becomes more accurate and would become perfect when $M = D = 28 \times 28 = 784$.

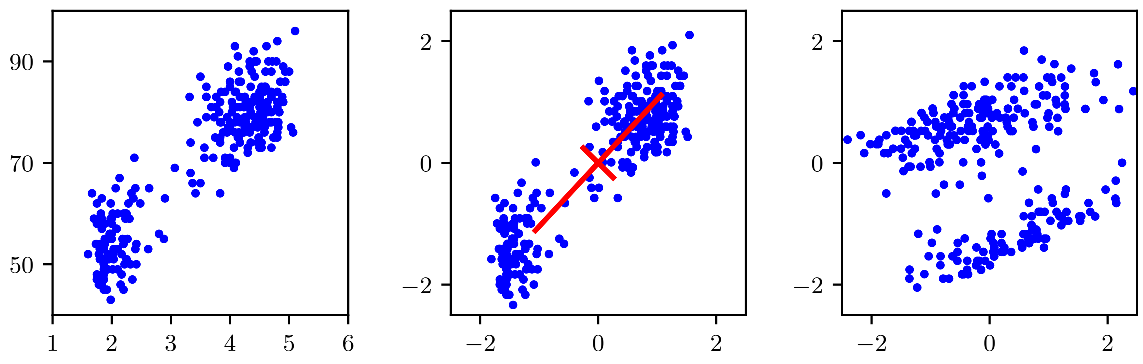


Figure 16.6 Illustration of the effects of linear pre-processing applied to the Old Faithful data set. The plot on the left shows the original data. The centre plot shows the result of standardizing the individual variables to zero mean and unit variance. Also shown are the principal axes of this normalized data set, plotted over the range $\pm\lambda_i^{1/2}$. The plot on the right shows the result of whitening the data to give it zero mean and unit covariance.

lated. To do this, we first write the eigenvector equation (16.17) in the form

$$\mathbf{SU} = \mathbf{UL} \tag{16.23}$$

where $\mathbf{L}$ is a $D \times D$ diagonal matrix with elements λ_i, and $\mathbf{U}$ is a $D \times D$ orthogonal matrix with columns given by $\mathbf{u}_i$. Then we define, for each data point $\mathbf{x}_n$, a transformed value given by

$$\mathbf{y}_n = \mathbf{L}^{-1/2}\mathbf{U}^{\mathrm{T}}(\mathbf{x}_n - \overline{\mathbf{x}}) \tag{16.24}$$

where $\overline{\mathbf{x}}$ is the sample mean defined by (16.1). Clearly, the set $\{\mathbf{y}_n\}$ has zero mean, and its covariance is given by the identity matrix because

$$\begin{aligned}\frac{1}{N}\sum_{n=1}^{N}\mathbf{y}_n\mathbf{y}_n^{\mathrm{T}} &= \frac{1}{N}\sum_{n=1}^{N}\mathbf{L}^{-1/2}\mathbf{U}^{\mathrm{T}}(\mathbf{x}_n - \overline{\mathbf{x}})(\mathbf{x}_n - \overline{\mathbf{x}})^{\mathrm{T}}\mathbf{U}\mathbf{L}^{-1/2} \\ &= \mathbf{L}^{-1/2}\mathbf{U}^{\mathrm{T}}\mathbf{SUL}^{-1/2} = \mathbf{L}^{-1/2}\mathbf{LL}^{-1/2} = \mathbf{I}. \end{aligned} \tag{16.25}$$

This operation is known as *whitening* or *sphering* the data and is illustrated for the Old Faithful data set in Figure 16.6. *Section 15.1*

16.1.5 High-dimensional data

In some applications of PCA, the number of data points is smaller than the dimensionality of the data space. For example, we might want to apply PCA to a data set of a few hundred images, each of which corresponds to a vector in a space of potentially several million dimensions (corresponding to three colour values for each of the pixels in the image). Note that in a D-dimensional space, a set of N points, where $N < D$, defines a linear subspace whose dimensionality is at most $N - 1$, and so there is little point in applying PCA for values of M that are greater than $N - 1$. Indeed, if we perform PCA we will find that at least $D - N + 1$ of the eigenvalues are zero, corresponding to eigenvectors along whose directions the data set has zero variance. Furthermore, typical algorithms for finding the eigenvectors of a $D \times D$ matrix have a computational cost that scales like $\mathcal{O}(D^3)$, and so for applications such as the image example, a direct application of PCA will be computationally infeasible.

We can resolve this problem as follows. First, let us define $\mathbf{X}$ to be the $(N \times D)$-dimensional centred data matrix, whose nth row is given by $(\mathbf{x}_n - \overline{\mathbf{x}})^{\mathrm{T}}$. The covariance matrix (16.3) can then be written as $\mathbf{S} = N^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{X}$, and the corresponding eigenvector equation becomes

$$\frac{1}{N}\mathbf{X}^{\mathrm{T}}\mathbf{X}\mathbf{u}_i = \lambda_i \mathbf{u}_i. \tag{16.26}$$

Now pre-multiply both sides by $\mathbf{X}$ to give

$$\frac{1}{N}\mathbf{X}\mathbf{X}^{\mathrm{T}}(\mathbf{X}\mathbf{u}_i) = \lambda_i(\mathbf{X}\mathbf{u}_i). \tag{16.27}$$

If we now define $\mathbf{v}_i = \mathbf{X}\mathbf{u}_i$, we obtain

$$\frac{1}{N}\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{v}_i = \lambda_i \mathbf{v}_i, \tag{16.28}$$

which is an eigenvector equation for the $N \times N$ matrix $N^{-1}\mathbf{X}\mathbf{X}^{\mathrm{T}}$. We see that this has the same $N-1$ eigenvalues as the original covariance matrix (which itself has an additional $D-N+1$ eigenvalues of value zero). Thus, we can solve the eigenvector problem in spaces of lower dimensionality with computational cost $\mathcal{O}(N^3)$ instead of $\mathcal{O}(D^3)$. To determine the eigenvectors, we multiply both sides of (16.28) by $\mathbf{X}^{\mathrm{T}}$ to give

$$\left(\frac{1}{N}\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)(\mathbf{X}^{\mathrm{T}}\mathbf{v}_i) = \lambda_i(\mathbf{X}^{\mathrm{T}}\mathbf{v}_i) \tag{16.29}$$

from which we see that $(\mathbf{X}^{\mathrm{T}}\mathbf{v}_i)$ is an eigenvector of $\mathbf{S}$ with eigenvalue λ_i. Note, however, that these eigenvectors are not necessarily normalized. To determine the appropriate normalization, we re-scale $\mathbf{u}_i \propto \mathbf{X}^{\mathrm{T}}\mathbf{v}_i$ by a constant such that $\|\mathbf{u}_i\| = 1$, which, assuming $\mathbf{v}_i$ has been normalized to unit length, gives

Exercise 16.3

$$\mathbf{u}_i = \frac{1}{(N\lambda_i)^{1/2}}\mathbf{X}^{\mathrm{T}}\mathbf{v}_i. \tag{16.30}$$

In summary, to apply this approach we first evaluate $\mathbf{X}\mathbf{X}^{\mathrm{T}}$ and then find its eigenvectors and eigenvalues and then compute the eigenvectors in the original data space using (16.30).

16.2. Probabilistic Latent Variables

We have seen in the previous section that PCA can be defined in terms of a linear projection of the data onto a subspace of lower dimensionality than the original data space. Each data point projects to a unique value of the quantities z_{nj} defined by (16.12), and we can view these quantities as deterministic latent variables. To introduce and motivate probabilistic continuous latent variables, we now show that PCA can also be expressed as the maximum likelihood solution of a probabilistic latent-variable model. This reformulation of PCA, known as *probabilistic PCA*, has several advantages compared with conventional PCA:

- A probabilistic PCA model represents a constrained form of a Gaussian distribution in which the number of free parameters can be restricted while still allowing the model to capture the dominant correlations in a data set.
- We can derive an EM algorithm for PCA that is computationally efficient in situations where only a few leading eigenvectors are required and that avoids having to evaluate the data covariance matrix as an intermediate step. *Section 16.3.2*
- The combination of a probabilistic model and EM allows us to deal with missing values in the data set.
- Mixtures of probabilistic PCA models can be formulated in a principled way and trained using the EM algorithm.
- The existence of a likelihood function allows direct comparison with other probabilistic density models. By contrast, conventional PCA will assign a low reconstruction cost to data points that are close to the principal subspace even if they lie arbitrarily far from the training data.
- Probabilistic PCA can be used to model class-conditional densities and hence be applied to classification problems.
- A probabilistic PCA model can be run generatively to provide samples from the distribution.
- Probabilistic PCA forms the basis for a Bayesian treatment of PCA in which the dimensionality of the principal subspace can be found automatically from the data (Bishop, 2006).

This formulation of PCA as a probabilistic model was proposed independently by Tipping and Bishop 1997; 1999 and by Roweis (1998). As we will see later, it is closely related to *factor analysis* (Basilevsky, 1994).

16.2.1 Generative model

Section 11.1.4

Probabilistic PCA is a simple example of the linear-Gaussian framework in which all the marginal and conditional distributions are Gaussian. We can formulate probabilistic PCA by first introducing an explicit M-dimensional latent variable

$\mathbf{z}$ corresponding to the principal-component subspace. Next we define a Gaussian prior distribution $p(\mathbf{z})$ over the latent variable, together with a Gaussian conditional distribution $p(\mathbf{x}|\mathbf{z})$ for the D-dimensional observed variable $\mathbf{x}$ conditioned on the value of the latent variable. Specifically, the prior distribution over $\mathbf{z}$ is given by a zero-mean unit-covariance Gaussian:

$$p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I}). \tag{16.31}$$

Similarly, the conditional distribution of the observed variable $\mathbf{x}$, conditioned on the value of the latent variable $\mathbf{z}$, is again Gaussian:

$$p(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}|\mathbf{W}\mathbf{z} + \boldsymbol{\mu}, \sigma^2\mathbf{I}) \tag{16.32}$$

in which the mean of $\mathbf{x}$ is a general linear function of $\mathbf{z}$ governed by the $D \times M$ matrix $\mathbf{W}$ and the D-dimensional vector $\boldsymbol{\mu}$. Note that this factorizes with respect to the elements of $\mathbf{x}$. In other words this is an example of a naive Bayes model. As we will see shortly, the columns of $\mathbf{W}$ span a linear subspace within the data space that corresponds to the principal subspace. The other parameter in this model is the scalar σ^2 governing the variance of the conditional distribution. Note that there is no loss of generality in assuming a zero-mean unit-covariance Gaussian for the latent distribution $p(\mathbf{z})$ because a more general Gaussian distribution would give rise to an equivalent probabilistic model.

Section 11.2.3

Exercise 16.4

We can view the probabilistic PCA model from a generative viewpoint in which a sampled value of the observed variable is obtained by first choosing a value for the latent variable and then sampling the observed variable conditioned on this latent value. Specifically, the D-dimensional observed variable $\mathbf{x}$ is defined by a linear transformation of the M-dimensional latent variable $\mathbf{z}$ plus additive Gaussian noise, so that

$$\mathbf{x} = \mathbf{W}\mathbf{z} + \boldsymbol{\mu} + \boldsymbol{\epsilon} \tag{16.33}$$

where $\mathbf{z}$ is an M-dimensional Gaussian latent variable, and $\boldsymbol{\epsilon}$ is a D-dimensional zero-mean Gaussian-distributed noise variable with covariance $\sigma^2\mathbf{I}$. This generative process is illustrated in Figure 16.7. Note that this framework is based on a mapping from latent space to data space, in contrast to the more conventional view of PCA discussed above. The reverse mapping, from data space to the latent space, will be obtained shortly using Bayes' theorem.

16.2.2 Likelihood function

Suppose we wish to determine the values of the parameters $\mathbf{W}$, $\boldsymbol{\mu}$, and σ^2 using maximum likelihood. To write down the likelihood function, we need an expression for the marginal distribution $p(\mathbf{x})$ of the observed variable. This is expressed, from the sum and product rules of probability, in the form

$$p(\mathbf{x}) = \int p(\mathbf{x}|\mathbf{z})p(\mathbf{z})\,\mathrm{d}\mathbf{z}. \tag{16.34}$$

Because this corresponds to a linear-Gaussian model, this marginal distribution is again Gaussian, and is given by

Exercise 16.6

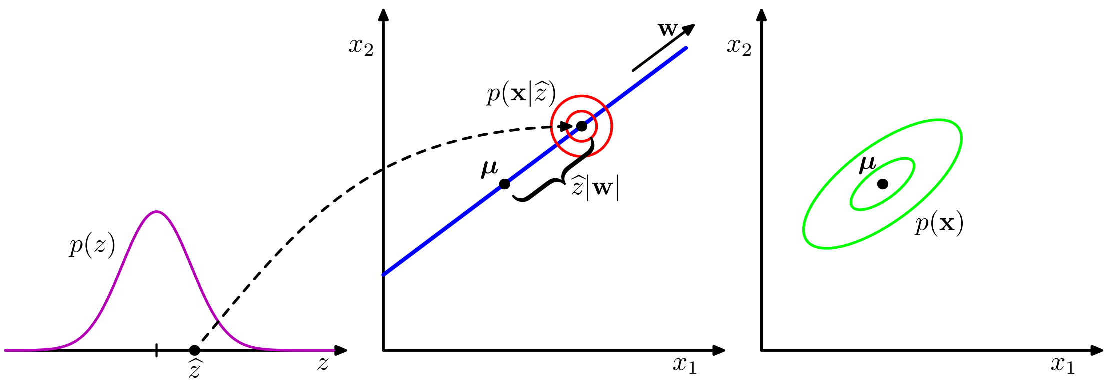


Figure 16.7 An illustration of the generative view of a probabilistic PCA model for a two-dimensional data space and a one-dimensional latent space. An observed data point $\mathbf{x}$ is generated by first drawing a value $\widehat{z}$ for the latent variable from its prior distribution $p(z)$ and then drawing a value for $\mathbf{x}$ from an isotropic Gaussian distribution (illustrated by the red circles) having mean $\mathbf{w}\widehat{z} + \boldsymbol{\mu}$ and covariance $\sigma^2\mathbf{I}$. The green ellipses show the density contours for the marginal distribution $p(\mathbf{x})$.

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \mathbf{C}) \tag{16.35}$$

where the $D \times D$ covariance matrix $\mathbf{C}$ is defined by

$$\mathbf{C} = \mathbf{W}\mathbf{W}^{\mathrm{T}} + \sigma^2\mathbf{I}. \tag{16.36}$$

This result can also be derived more directly by noting that the predictive distribution will be Gaussian and then evaluating its mean and covariance using (16.33). This gives

$$\mathbb{E}[\mathbf{x}] = \mathbb{E}[\mathbf{W}\mathbf{z} + \boldsymbol{\mu} + \boldsymbol{\epsilon}] = \boldsymbol{\mu} \tag{16.37}$$

$$\begin{aligned}
\operatorname{cov}[\mathbf{x}] &= \mathbb{E}\left[(\mathbf{W}\mathbf{z} + \boldsymbol{\epsilon})(\mathbf{W}\mathbf{z} + \boldsymbol{\epsilon})^{\mathrm{T}}\right] \\
&= \mathbb{E}\left[\mathbf{W}\mathbf{z}\mathbf{z}^{\mathrm{T}}\mathbf{W}^{\mathrm{T}}\right] + \mathbb{E}[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^{\mathrm{T}}] \qquad (16.38) \\
&= \mathbf{W}\mathbf{W}^{\mathrm{T}} + \sigma^2\mathbf{I} \qquad (16.39)
\end{aligned}$$

where we have used the fact that $\mathbf{z}$ and $\boldsymbol{\epsilon}$ are independent random variables and hence are uncorrelated.

Intuitively, we can think of the distribution $p(\mathbf{x})$ as being defined by taking an isotropic Gaussian 'spray can' and moving it across the principal subspace spraying Gaussian ink with density determined by σ^2 and weighted by the prior distribution. The accumulated ink density gives rise to a 'pancake' shaped distribution representing the marginal density $p(\mathbf{x})$.

The predictive distribution $p(\mathbf{x})$ is governed by the parameters $\boldsymbol{\mu}$, $\mathbf{W}$, and σ^2. However, there is redundancy in this parameterization corresponding to rotations of the latent space coordinates. To see this, consider a matrix $\widetilde{\mathbf{W}} = \mathbf{W}\mathbf{R}$ where $\mathbf{R}$ is an orthogonal matrix. Using the orthogonality property $\mathbf{R}\mathbf{R}^{\mathrm{T}} = \mathbf{I}$, we see that the quantity $\widetilde{\mathbf{W}}\widetilde{\mathbf{W}}^{\mathrm{T}}$ that appears in the covariance matrix $\mathbf{C}$ takes the form

$$\widetilde{\mathbf{W}}\widetilde{\mathbf{W}}^{\mathrm{T}} = \mathbf{W}\mathbf{R}\mathbf{R}^{\mathrm{T}}\mathbf{W}^{\mathrm{T}} = \mathbf{W}\mathbf{W}^{\mathrm{T}} \tag{16.40}$$

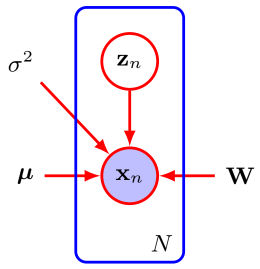


Figure 16.8 The probabilistic PCA model for a data set of N observations of $\mathbf{x}$ can be expressed as a directed graph in which each observation $\mathbf{x}_n$ is associated with a value $\mathbf{z}_n$ of the latent variable.

and hence is independent of $\mathbf{R}$. Thus, there is a whole family of matrices $\widetilde{\mathbf{W}}$ all of which give rise to the same predictive distribution. This invariance can be understood in terms of rotations within the latent space. We will return to a discussion of the number of independent parameters in this model later.

When we evaluate the predictive distribution, we require $\mathbf{C}^{-1}$, which involves the inversion of a $D \times D$ matrix. The computation required to do this can be reduced by making use of the matrix inversion identity (A.7) to give

$$\mathbf{C}^{-1} = \sigma^{-2}\mathbf{I} - \sigma^{-2}\mathbf{W}\mathbf{M}^{-1}\mathbf{W}^{\mathrm{T}} \tag{16.41}$$

where the $M \times M$ matrix $\mathbf{M}$ is defined by

$$\mathbf{M} = \mathbf{W}^{\mathrm{T}}\mathbf{W} + \sigma^2\mathbf{I}. \tag{16.42}$$

Because we invert $\mathbf{M}$ rather than inverting $\mathbf{C}$ directly, the cost of evaluating $\mathbf{C}^{-1}$ is reduced from $\mathcal{O}(D^3)$ to $\mathcal{O}(M^3)$.

As well as the predictive distribution $p(\mathbf{x})$, we will also require the posterior distribution $p(\mathbf{z}|\mathbf{x})$, which can again be written down directly using the result (3.100) for linear-Gaussian models to give

Exercise 16.8

$$p(\mathbf{z}|\mathbf{x}) = \mathcal{N}\left(\mathbf{z}|\mathbf{M}^{-1}\mathbf{W}^{\mathrm{T}}(\mathbf{x}-\boldsymbol{\mu}), \sigma^2\mathbf{M}^{-1}\right). \tag{16.43}$$

Note that the posterior mean depends on $\mathbf{x}$, whereas the posterior covariance is independent of $\mathbf{x}$.

16.2.3 Maximum likelihood

We next consider the determination of the model parameters using maximum likelihood. Given a data set $\mathbf{X} = \{\mathbf{x}_n\}$ of observed data points, the probabilistic PCA model can be expressed as a directed graph, as shown in Figure 16.8. The corresponding log likelihood function is given, from (16.35), by

$$\begin{aligned}\ln p(\mathbf{X}|\boldsymbol{\mu}, \mathbf{W}, \sigma^2) &= \sum_{n=1}^{N} \ln p(\mathbf{x}_n|\mathbf{W}, \boldsymbol{\mu}, \sigma^2) \\ &= -\frac{ND}{2}\ln(2\pi) - \frac{N}{2}\ln|\mathbf{C}| - \frac{1}{2}\sum_{n=1}^{N}(\mathbf{x}_n-\boldsymbol{\mu})^{\mathrm{T}}\mathbf{C}^{-1}(\mathbf{x}_n-\boldsymbol{\mu}). \end{aligned} \tag{16.44}$$

Exercise 16.9

Setting the derivative of the log likelihood with respect to $\boldsymbol{\mu}$ equal to zero gives the expected result $\boldsymbol{\mu} = \overline{\mathbf{x}}$ where $\overline{\mathbf{x}}$ is the data mean defined by (16.1). Because the log likelihood is a quadratic function of $\boldsymbol{\mu}$, this solution represents the unique maximum, as can be confirmed by computing second derivatives. Back-substituting, we can then write the log likelihood function in the form

$$\ln p(\mathbf{X}|\mathbf{W}, \boldsymbol{\mu}, \sigma^2) = -\frac{N}{2}\left\{D\ln(2\pi) + \ln|\mathbf{C}| + \mathrm{Tr}\left(\mathbf{C}^{-1}\mathbf{S}\right)\right\} \tag{16.45}$$

where $\mathbf{S}$ is the data covariance matrix defined by (16.3).

Maximization with respect to $\mathbf{W}$ and σ^2 is more complex but nonetheless has an exact closed-form solution. It was shown by Tipping and Bishop (1999) that all the stationary points of the log likelihood function can be written as

$$\mathbf{W}_{\mathrm{ML}} = \mathbf{U}_M(\mathbf{L}_M - \sigma^2\mathbf{I})^{1/2}\mathbf{R} \tag{16.46}$$

where $\mathbf{U}_M$ is a $D \times M$ matrix whose columns are given by any subset (of size M) of the eigenvectors of the data covariance matrix $\mathbf{S}$. The $M \times M$ diagonal matrix $\mathbf{L}_M$ has elements given by the corresponding eigenvalues λ_i, and $\mathbf{R}$ is an arbitrary $M \times M$ orthogonal matrix.

Furthermore, Tipping and Bishop (1999) showed that the *maximum* of the likelihood function is obtained when the M eigenvectors are chosen to be those whose eigenvalues are the M largest (all other solutions being saddle points). A similar result was conjectured independently by Roweis (1998), although no proof was given. Again, we will assume that the eigenvectors have been arranged in order of decreasing values of the corresponding eigenvalues, so that the M principal eigenvectors are $\mathbf{u}_1, \ldots, \mathbf{u}_M$. In this case, the columns of $\mathbf{W}$ define the principal subspace of standard PCA. The corresponding maximum likelihood solution for σ^2 is then given by

$$\sigma^2_{\mathrm{ML}} = \frac{1}{D-M}\sum_{i=M+1}^{D}\lambda_i \tag{16.47}$$

so that σ^2_{ML} is the average variance associated with the discarded dimensions.

Because $\mathbf{R}$ is orthogonal, it can be interpreted as a rotation matrix in the M-dimensional latent space. If we substitute the solution for $\mathbf{W}$ into the expression for $\mathbf{C}$ and make use of the orthogonality property $\mathbf{R}\mathbf{R}^{\mathrm{T}} = \mathbf{I}$, we see that $\mathbf{C}$ is independent of $\mathbf{R}$. This simply says that the predictive density is unchanged by rotations in the latent space as discussed earlier. For the particular case $\mathbf{R} = \mathbf{I}$, we see that the columns of $\mathbf{W}$ are the principal component eigenvectors scaled by the variance parameters $\lambda_i - \sigma^2$. The interpretation of these scaling factors is clear once we recognize that for a convolution of independent Gaussian distributions (in this case the latent space distribution and the noise model) the variances are additive. Thus, the variance λ_i in the direction of an eigenvector $\mathbf{u}_i$ is composed of the sum of a contribution $\lambda_i - \sigma^2$ from the projection of the unit-variance latent space distribution into data space through the corresponding column of $\mathbf{W}$ plus an isotropic contribution of variance σ^2, which is added in all directions by the noise model.

It is worth taking a moment to study the form of the covariance matrix given by (16.36). Consider the variance of the predictive distribution along some direction specified by the unit vector $\mathbf{v}$, where $\mathbf{v}^{\mathrm{T}}\mathbf{v} = 1$, which is given by $\mathbf{v}^{\mathrm{T}}\mathbf{C}\mathbf{v}$. First suppose that $\mathbf{v}$ is orthogonal to the principal subspace, in other words it is given by some linear combination of the discarded eigenvectors. Then $\mathbf{v}^{\mathrm{T}}\mathbf{U} = \mathbf{0}$ and hence $\mathbf{v}^{\mathrm{T}}\mathbf{C}\mathbf{v} = \sigma^2$. Thus, the model predicts a noise variance orthogonal to the principal subspace, which from (16.47) is just the average of the discarded eigenvalues. Now suppose that $\mathbf{v} = \mathbf{u}_i$ where $\mathbf{u}_i$ is one of the retained eigenvectors defining the principal subspace. Then $\mathbf{v}^{\mathrm{T}}\mathbf{C}\mathbf{v} = (\lambda_i - \sigma^2) + \sigma^2 = \lambda_i$. In other words, this model correctly captures the variance of the data along the principal axes and approximates the variance in all remaining directions with a single average value σ^2.

One way to construct the maximum likelihood density model would simply be to find the eigenvectors and eigenvalues of the data covariance matrix and then to evaluate $\mathbf{W}$ and σ^2 using the results given above. In this case, we would choose $\mathbf{R} = \mathbf{I}$ for convenience. However, if the maximum likelihood solution is found by numerical optimization of the likelihood function, for instance using an algorithm such as conjugate gradients (Fletcher, 1987; Nocedal and Wright, 1999) or through the EM algorithm, then the resulting value of $\mathbf{R}$ is essentially arbitrary. This implies that the columns of $\mathbf{W}$ need not be orthogonal. If an orthogonal basis is required, the matrix $\mathbf{W}$ can be post-processed appropriately (Golub and Van Loan, 1996). Alternatively, the EM algorithm can be modified in such a way as to yield orthonormal principal directions, sorted in descending order of the corresponding eigenvalues, directly (Ahn and Oh, 2003).

Section 16.3.2

The rotational invariance in latent space represents a form of statistical non-identifiability, analogous to that encountered for mixture models for discrete latent variables. Here there is a continuum of parameters, any value of which leads to the same predictive density, in contrast to the discrete non-identifiability associated with component relabelling in the mixture setting.

If we consider $M = D$, so that there is no reduction of dimensionality, then $\mathbf{U}_M = \mathbf{U}$ and $\mathbf{L}_M = \mathbf{L}$. Making use of the orthogonality properties $\mathbf{U}\mathbf{U}^{\mathrm{T}} = \mathbf{I}$ and $\mathbf{R}\mathbf{R}^{\mathrm{T}} = \mathbf{I}$, we see that the covariance $\mathbf{C}$ of the marginal distribution for $\mathbf{x}$ becomes

$$\mathbf{C} = \mathbf{U}(\mathbf{L} - \sigma^2\mathbf{I})^{1/2}\mathbf{R}\mathbf{R}^{\mathrm{T}}(\mathbf{L} - \sigma^2\mathbf{I})^{1/2}\mathbf{U}^{\mathrm{T}} + \sigma^2\mathbf{I} = \mathbf{U}\mathbf{L}\mathbf{U}^{\mathrm{T}} = \mathbf{S} \tag{16.48}$$

and so we obtain the standard maximum likelihood solution for an unconstrained Gaussian distribution in which the covariance matrix is given by the sample covariance.

Conventional PCA is generally formulated as a projection of points from the D-dimensional data space onto an M-dimensional linear subspace. Probabilistic PCA, however, is most naturally expressed as a mapping from the latent space into the data space via (16.33). For applications such as visualization and data compression, we can reverse this mapping using Bayes' theorem. Any point $\mathbf{x}$ in data space can then be summarized by its posterior mean and covariance in latent space. From (16.43) the mean is given by

$$\mathbb{E}[\mathbf{z}|\mathbf{x}] = \mathbf{M}^{-1}\mathbf{W}_{\mathrm{ML}}^{\mathrm{T}}(\mathbf{x} - \overline{\mathbf{x}}) \tag{16.49}$$

where $\mathbf{M}$ is given by (16.42). This projects to a point in data space given by

$$\mathbf{W}\,\mathbb{E}[\mathbf{z}|\mathbf{x}] + \boldsymbol{\mu}. \tag{16.50}$$

Section 4.1.6

Note that this takes the same form as the equations for regularized linear regression and is a consequence of maximizing the likelihood function for a linear-Gaussian model. Similarly, from (16.43) the posterior covariance is given by $\sigma^2\mathbf{M}^{-1}$ and is independent of $\mathbf{x}$.

If we take the limit $\sigma^2 \to 0$, then the posterior mean reduces to

$$(\mathbf{W}_{\mathrm{ML}}^{\mathrm{T}}\mathbf{W}_{\mathrm{ML}})^{-1}\mathbf{W}_{\mathrm{ML}}^{\mathrm{T}}(\mathbf{x} - \overline{\mathbf{x}}), \tag{16.51}$$

Exercise 16.11

Exercise 16.12

which represents an orthogonal projection of the data point onto the latent space, and so we recover the standard PCA model. The posterior covariance in this limit is zero, however, and the density becomes singular. For $\sigma^2 > 0$, the latent projection is shifted towards the origin, relative to the orthogonal projection.

Section 3.2

Finally, note that an important role for the probabilistic PCA model is in defining a multivariate Gaussian distribution in which the number of degrees of freedom, in other words the number of independent parameters, can be controlled while still allowing the model to capture the dominant correlations in the data. Recall that a general Gaussian distribution has $D(D+1)/2$ independent parameters in its covariance matrix (plus another D parameters in its mean). Thus, the number of parameters scales quadratically with D and can become excessive in spaces of high dimensionality. If we restrict the covariance matrix to be diagonal, then it has only D independent parameters, and so the number of parameters now grows linearly with dimensionality. However, it now treats the variables as if they were independent and hence can no longer express any correlations between them. Probabilistic PCA provides an elegant compromise in which the M most significant correlations can be captured while still ensuring that the total number of parameters grows only linearly with D. We can see this by evaluating the number of degrees of freedom in the probabilistic PCA model as follows. The covariance matrix $\mathbf{C}$ depends on the parameters $\mathbf{W}$, which has size $D \times M$, and σ^2, giving a total parameter count of $DM + 1$. However, we have seen that there is some redundancy in this parameterization associated with rotations of the coordinate system in the latent space. The orthogonal matrix $\mathbf{R}$ that expresses these rotations has size $M \times M$. In the first column of this matrix, there are $M - 1$ independent parameters, because the column vector must be normalized to unit length. In the second column, there are $M - 2$ independent parameters, because the column must be normalized and also must be orthogonal to the previous column, and so on. Summing this arithmetic series, we see that $\mathbf{R}$ has a total of $M(M-1)/2$ independent parameters. Thus, the number of degrees of freedom in the covariance matrix $\mathbf{C}$ is given by

$$DM + 1 - M(M-1)/2. \tag{16.52}$$

Exercise 16.14

The number of independent parameters in this model therefore only grows linearly with D, for fixed M. If we take $M = D - 1$, then we recover the standard result for a full covariance Gaussian. In this case, the variance along $D - 1$ linearly independent directions is controlled by the columns of $\mathbf{W}$, and the variance along the remaining direction is given by σ^2. If $M = 0$, the model is equivalent to the isotropic covariance case.

16.2.4 Factor analysis

Factor analysis is a linear-Gaussian latent-variable model that is closely related to probabilistic PCA. Its definition differs from that of probabilistic PCA only in that the conditional distribution of the observed variable $\mathbf{x}$ given the latent variable $\mathbf{z}$ has a diagonal rather than an isotropic covariance so that

$$p(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}|\mathbf{W}\mathbf{z} + \boldsymbol{\mu}, \boldsymbol{\Psi}) \tag{16.53}$$

where $\boldsymbol{\Psi}$ is a $D \times D$ diagonal matrix. Note that the factor analysis model, in common with probabilistic PCA, assumes that the observed variables $x_1, \ldots, x_D$ are independent, given the latent variable $\mathbf{z}$. In essence, a factor analysis model explains the observed covariance structure of the data by representing the independent variance associated with each coordinate in the matrix $\boldsymbol{\Psi}$ and capturing the covariance between variables in the matrix $\mathbf{W}$. In the factor analysis literature, the columns of $\mathbf{W}$, which capture the correlations between observed variables, are called *factor loadings*, and the diagonal elements of $\boldsymbol{\Psi}$, which represent the independent noise variances for each of the variables, are called *uniquenesses*.

The origins of factor analysis are as old as those of PCA, and discussions of factor analysis can be found in the books by Everitt (1984), Bartholomew (1987), and Basilevsky (1994). Links between factor analysis and PCA were investigated by Lawley (1953) and Anderson (1963), who showed that at stationary points of the likelihood function, for a factor analysis model with $\boldsymbol{\Psi} = \sigma^2\mathbf{I}$, the columns of $\mathbf{W}$ are scaled eigenvectors of the sample covariance matrix and σ^2 is the average of the discarded eigenvalues. Later, Tipping and Bishop (1999) showed that the maximum of the log likelihood function occurs when the eigenvectors comprising $\mathbf{W}$ are chosen to be the principal eigenvectors.

Making use of (16.34), we see that the marginal distribution for the observed variable is given by $p(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \mathbf{C})$ where now

$$\mathbf{C} = \mathbf{W}\mathbf{W}^{\mathrm{T}} + \boldsymbol{\Psi}. \tag{16.54}$$

Exercise 16.16 As with probabilistic PCA, this model is invariant to rotations in the latent space.

Historically, factor analysis has been the subject of controversy when attempts have been made to place an interpretation on the individual factors (the coordinates in $\mathbf{z}$-space), which has proven problematic due to the non-identifiability of factor analysis associated with rotations in this space. From our perspective, however, we shall view factor analysis as a form of latent-variable density model, in which the form of the latent space is of interest but not the particular choice of coordinates used to describe it. If we wish to remove the degeneracy associated with latent-space rotations, we must consider non-Gaussian latent-variable distributions, giving rise to independent component analysis models. *Section 16.2.5*

Another difference between probabilistic PCA and factor analysis is their behaviour under transformations of the data set. For PCA and probabilistic PCA, if we *Exercise 16.17* rotate the coordinate system in data space, then we obtain exactly the same fit to the data but with the $\mathbf{W}$ matrix transformed by the corresponding rotation matrix. However, for factor analysis, the analogous property is that if we make a component-wise re-scaling of the data vectors, then this is absorbed into a corresponding re-scaling of the elements of $\mathbf{\Psi}$.

16.2.5 Independent component analysis

One generalization of the linear-Gaussian latent-variable model is to consider models in which the observed variables are related linearly to the latent variables, but for which the latent distribution is non-Gaussian. An important class of such models, known as *independent component analysis*, or ICA, arises when we consider a distribution over the latent variables that factorizes, so that

$$p(\mathbf{z}) = \prod_{j=1}^{M} p(z_j). \tag{16.55}$$

To understand the role of such models, consider a situation in which two people are talking at the same time, and we record their voices using two microphones. If we ignore effects such as time delay and echoes, then the signals received by the microphones at any point in time will be given by linear combinations of the amplitudes of the two voices. The coefficients of this linear combination will be constant, and if we can infer their values from sample data, then we can invert the mixing process (assuming it is non-singular) and thereby obtain two clean signals each of which contains the voice of just one person. This is an example of a problem called *blind source separation* in which 'blind' refers to the fact that we are given only the mixed data, and neither the original sources nor the mixing coefficients are observed (Cardoso, 1998).

This type of problem is sometimes addressed using the following approach (MacKay, 2003) in which we ignore the temporal nature of the signals and treat the successive samples as i.i.d. We consider a generative model in which there are two latent variables corresponding to the unobserved speech signal amplitudes, and there are two observed variables given by the signal values at the microphones. The latent variables have a joint distribution that factorizes as above, and the observed variables are given by a linear combination of the latent variables. There is no need to include a noise distribution because the number of latent variables equals the number of observed variables, and therefore the marginal distribution of the observed variables will not in general be singular, so the observed variables are simply deterministic linear combinations of the latent variables. Given a data set of observations, the likelihood function for this model is a function of the coefficients in the linear combination. The log likelihood can be maximized using gradient-based optimization giving rise to a particular version of ICA.

The success of this approach requires that the latent variables have non-Gaussian distributions. To see this, recall that in probabilistic PCA (and in factor analysis) the latent-space distribution is given by a zero-mean isotropic Gaussian. The model therefore cannot distinguish between two different choices for the latent variables if these differ simply by a rotation in latent space. This can be verified directly by noting that the marginal density (16.35), and hence the likelihood function, is unchanged if we make the transformation $\mathbf{W} \rightarrow \mathbf{WR}$ where $\mathbf{R}$ is an orthogonal matrix satisfying $\mathbf{RR}^{\mathrm{T}} = \mathbf{I}$, because the matrix $\mathbf{C}$ given by (16.36) is itself invariant. Extending the model to allow more general Gaussian latent distributions does not change this conclusion because, as we have seen, such a model is equivalent to the zero-mean isotropic Gaussian latent-variable model.

Another way to see why a Gaussian latent-variable distribution in a linear model is insufficient to find independent components is to note that the principal components represent a rotation of the coordinate system in data space so as to diagonalize the covariance matrix. The data distribution in the new coordinates is then uncorrelated. Although zero correlation is a necessary condition for independence it is not, however, sufficient. In practice, a common choice for the latent-variable distribution is given by

Exercise 2.39

$$p(z_j) = \frac{1}{\pi \cosh(z_j)} = \frac{2}{\pi(e^{z_j} + e^{-z_j})}, \tag{16.56}$$

which has heavy tails compared to a Gaussian, reflecting the observation that many real-world distributions also exhibit this property.

The original ICA model (Bell and Sejnowski, 1995) was based on the optimization of an objective function defined by information maximization. One advantage of a probabilistic latent-variable formulation is that it helps to motivate and formulate generalizations of basic ICA. For instance, *independent factor analysis* (Attias, 1999) considers a model in which the number of latent and observed variables can differ, the observed variables are noisy, and the individual latent variables have flexible distributions modelled by mixtures of Gaussians. The log likelihood for this model is maximized using EM, and the reconstruction of the latent variables is approximated using a variational approach. Many other types of model have been considered, and there is now a huge literature on ICA and its applications (Jutten and Herault, 1991; Comon, Jutten, and Herault, 1991; Amari, Cichocki, and Yang, 1996; Pearlmutter and Parra, 1997; Hyvärinen and Oja, 1997; Hinton *et al.*, 2001; Miskin and MacKay, 2001; Hojen-Sorensen, Winther, and Hansen, 2002; Choudrey and Roberts, 2003; Chan, Lee, and Sejnowski, 2003; Stone, 2004).

16.2.6 Kalman filters

So far we have assumed that the data values are i.i.d. A common situation in which this assumption does not hold is when the data points form an ordered sequence. We have seen that a hidden Markov model can be viewed as an extension of the mixture models to allow for sequential correlations in the data. In a similar way, a continuous latent-variable model can be extended to handle sequential data by connecting the latent variables to form a Markov chain, as shown in the graphical model of Figure 16.9. This is known as a *linear dynamical system* or *Kalman filter* (Zarchan and Musoff, 2005). Note that this is the same graphical structure as a hidden Markov model. It is interesting to note that, historically, hidden Markov models and linear dynamical systems were developed independently. Once they are both expressed as graphical models, however, the deep relationship between them

Section 15.3.1

Section 15.3.1

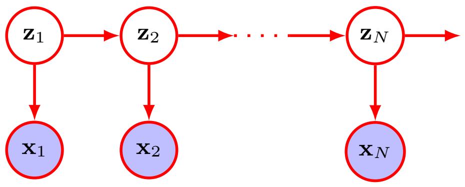


Figure 16.9 A probabilistic graphical model for sequential data, known as a linear dynamical system, or Kalman filter, in which the latent variables form a Markov chain.

immediately becomes apparent. Kalman filters are widely used in many real-time tracking applications, for example to track aircraft using radar reflections.

In the simplest such model, the distributions $p(\mathbf{x}_n|\mathbf{z}_n)$ in Figure 16.9 represent a linear-Gaussian latent-variable model for that particular observation, of the kind we have discussed previously for i.i.d. data. However, the latent variables $\{\mathbf{z}_n\}$ are no longer treated as independent but now form a Markov chain in which the distribution $p(\mathbf{z}_n|\mathbf{z}_{n-1})$ of each latent variable is conditioned on the state of the previous latent variable in the chain. Again these can be chosen to be linear-Gaussian in which the distribution of $\mathbf{z}_n$ is Gaussian with a mean given by a linear function of $\mathbf{z}_{n-1}$. Typically the parameters of all the distributions $p(\mathbf{x}_n|\mathbf{z}_n)$ are shared, and likewise the parameters of the distributions $p(\mathbf{z}_n|\mathbf{z}_{n-1})$ are shared, so that the total number of parameters in the model is fixed, independently of the length of the sequence. These parameters can be learned from data by maximum likelihood with efficient algorithms that involve propagating messages around the graph (Bishop, 2006). For the rest of this chapter, however, we will focus on i.i.d. data.

16.3. Evidence Lower Bound

Section 15.4

In our discussion of models with discrete latent variables, we derived the evidence lower bound (ELBO) on the marginal log likelihood and showed how this forms the basis for deriving the expectation–maximization (EM) algorithm including its generalizations such as variational inference. The same framework applies to continuous latent variables as well as to models that combine both discrete and continuous variables. Here we present a slightly different derivation of the ELBO, and we assume that the latent variables $\mathbf{z}$ are continuous.

Consider a model $p(\mathbf{x}, \mathbf{z}|\mathbf{w})$ with an observed variable $\mathbf{x}$, a latent variable $\mathbf{z}$, and a learnable parameter vector $\mathbf{w}$. If we introduce an arbitrary distribution $q(\mathbf{z})$ over the latent variable then we can write the log likelihood function $\ln p(\mathbf{x}|\mathbf{w})$ as a sum of two terms in the form

Exercise 16.18

$$\ln p(\mathbf{x}|\mathbf{w}) = \mathcal{L}(\mathbf{w}) + \mathrm{KL}\left(q(\mathbf{z})\|p(\mathbf{z}|\mathbf{x}, \mathbf{w})\right) \tag{16.57}$$

where we have defined

$$\mathcal{L}(q, \mathbf{w}) = \int q(\mathbf{z}) \ln \left\{ \frac{p(\mathbf{x}, \mathbf{z}|\mathbf{w})}{q(\mathbf{z})} \right\} \mathrm{d}\mathbf{z} \tag{16.58}$$

$$\mathrm{KL}\left(q(\mathbf{z})\|p(\mathbf{z}|\mathbf{x}, \mathbf{w})\right) = -\int q(\mathbf{z}) \ln \left\{ \frac{p(\mathbf{z}|\mathbf{x}, \mathbf{w})}{q(\mathbf{z})} \right\} \mathrm{d}\mathbf{z}. \tag{16.59}$$

Since $\mathrm{KL}\left(q(\mathbf{z})\|p(\mathbf{z}|\mathbf{x},\mathbf{w})\right)$ is a Kullback–Leibler divergence, it satisfies the property $\mathrm{KL}\left(\cdot\|\cdot\right) \geqslant 0$ from which it follows that

Section 2.5.5

$$\ln p(\mathbf{x}|\mathbf{w}) \geqslant \mathcal{L}(\mathbf{w}) \tag{16.60}$$

and we therefore see that $\mathcal{L}(q,\mathbf{w})$ given by (16.58) forms a lower bound on the log likelihood, known as the *evidence lower bound* or ELBO. We see that $\mathcal{L}(q,\mathbf{w})$ takes the same form (15.53) as derived for the discrete case but with summations replaced by integrals.

We can maximize the log likelihood function using a two-stage iterative procedure called the *expectation maximization* algorithm, or EM algorithm, in which we alternately maximize $\mathcal{L}(q,\mathbf{w})$ with respect to $q(\mathbf{z})$ (the E step) and $\mathbf{w}$ (the M step). We first initialize the parameters $\mathbf{w}^{(\mathrm{old})}$. Then in the E step we keep $\mathbf{w}$ fixed and we maximize the lower bound with respect to $q(\mathbf{z})$. This is easily done by noting that the highest value for the bound is obtained by minimizing the Kullback–Leibler divergence in (16.59) and hence is achieved when $q(\mathbf{z}) = p(\mathbf{z}|\mathbf{x},\mathbf{w}^{(\mathrm{old})})$ for which the Kullback–Leibler divergence is zero. In the M step, we keep this choice of $q(\mathbf{z})$ fixed and maximize $\mathcal{L}(q,\mathbf{w})$ with respect to $\mathbf{w}$. Substituting for $q(\mathbf{z})$ in (16.58) we obtain

$$\begin{aligned}\mathcal{L}(q,\mathbf{w}) = &\int p(\mathbf{z}|\mathbf{x},\mathbf{w}^{(\mathrm{old})})\ln p(\mathbf{x},\mathbf{z}|\mathbf{w})\,\mathrm{d}\mathbf{z}\\ &-\int p(\mathbf{z}|\mathbf{x},\mathbf{w}^{(\mathrm{old})})\ln p(\mathbf{z}|\mathbf{x},\mathbf{w}^{(\mathrm{old})})\,\mathrm{d}\mathbf{z}.\end{aligned} \tag{16.61}$$

We now maximize this with respect to $\mathbf{w}$ in the M step while keeping $\mathbf{w}^{(\mathrm{old})}$ fixed. Note that the second term on the right-hand side of (16.61) is independent of $\mathbf{w}$ and so can be ignored during the M step. The first term on the right-hand side is the expectation of the *complete data log likelihood* where the expectation is taken with respect to the posterior distribution of $\mathbf{z}$ computed using $\mathbf{w}^{(\mathrm{old})}$.

Section 15.3

If we have a data set $\mathbf{x}_1,\ldots,\mathbf{x}_N$ of i.i.d. observations then the likelihood function takes the form

$$\ln p(\mathbf{X}|\mathbf{w}) = \sum_{n=1}^{N}\ln p(\mathbf{x}_n|\mathbf{w}) \tag{16.62}$$

where the data matrix $\mathbf{X}$ comprises $\mathbf{x}_1,\ldots,\mathbf{x}_N$, and the parameters $\mathbf{w}$ are shared across all data points. For each data point we introduce a corresponding latent variable $\mathbf{z}_n$ with its associated distribution $q(\mathbf{z}_n)$, and by following similar steps to those used to derive (16.58), we obtain the ELBO in the form

Exercise 16.19

$$\mathcal{L}(q,\mathbf{w}) = \sum_{n=1}^{N}\int q(\mathbf{z}_n)\ln\left\{\frac{p(\mathbf{x}_n,\mathbf{z}_n|\mathbf{w})}{q(\mathbf{z}_n)}\right\}\mathrm{d}\mathbf{z}_n. \tag{16.63}$$

Section 19.2

When we discuss variational autoencoders, we will encounter a model for which an exact solution to the E step is not feasible so instead a partial maximization is performed by modelling $q(\mathbf{z})$ using a deep neural network and then using the ELBO to learn the parameters of the network.

16.3.1 Expectation maximization

We can now use the EM algorithm, derived by iteratively maximizing the evidence lower bound, to learn the parameters of the probabilistic PCA model. This may seem rather pointless because we have already obtained an exact closed-form solution for the maximum likelihood parameter values. However, in spaces of high dimensionality, there may be computational advantages in using an iterative EM procedure rather than working directly with the sample covariance matrix. This EM procedure can also be extended to the factor analysis model, for which there is no closed-form solution. Finally, it allows missing data to be handled in a principled way.

Section 16.2.4

We can derive the EM algorithm for probabilistic PCA by following the general framework for EM. Thus, we write down the complete-data log likelihood and take its expectation with respect to the posterior distribution of the latent distribution evaluated using 'old' parameter values. Maximization of this expected complete-data log likelihood then yields the 'new' parameter values. Because the data points are assumed independent, the complete-data log likelihood function takes the form

Section 15.3

$$\ln p\left(\mathbf{X}, \mathbf{Z}|\boldsymbol{\mu}, \mathbf{W}, \sigma^2\right) = \sum_{n=1}^{N} \{\ln p(\mathbf{x}_n|\mathbf{z}_n) + \ln p(\mathbf{z}_n)\} \tag{16.64}$$

where the nth row of the matrix $\mathbf{Z}$ is given by $\mathbf{z}_n$. We already know that the exact maximum likelihood solution for $\boldsymbol{\mu}$ is given by the sample mean $\overline{\mathbf{x}}$ defined by (16.1), and it is convenient to substitute for $\boldsymbol{\mu}$ at this stage. Making use of the expressions (16.31) and (16.32) for the latent and conditional distributions, respectively, and taking the expectation with respect to the posterior distribution over the latent variables, we obtain

$$\begin{aligned}\mathbb{E}[\ln p\left(\mathbf{X}, \mathbf{Z}|\boldsymbol{\mu}, \mathbf{W}, \sigma^2\right)] = -\sum_{n=1}^{N} \Bigg\{ \frac{D}{2}\ln(2\pi\sigma^2) + \frac{1}{2}\mathrm{Tr}\left(\mathbb{E}[\mathbf{z}_n\mathbf{z}_n^{\mathrm{T}}]\right) \\ +\frac{1}{2\sigma^2}\|\mathbf{x}_n - \boldsymbol{\mu}\|^2 - \frac{1}{\sigma^2}\mathbb{E}[\mathbf{z}_n]^{\mathrm{T}}\mathbf{W}^{\mathrm{T}}(\mathbf{x}_n - \boldsymbol{\mu}) \\ + \frac{1}{2\sigma^2}\mathrm{Tr}\left(\mathbb{E}[\mathbf{z}_n\mathbf{z}_n^{\mathrm{T}}]\mathbf{W}^{\mathrm{T}}\mathbf{W}\right) + \frac{M}{2}\ln(2\pi)\Bigg\}.\end{aligned} \tag{16.65}$$

Note that this depends on the posterior distribution only through the sufficient statistics of the Gaussian. Thus, in the E step, we use the old parameter values to evaluate

$$\mathbb{E}[\mathbf{z}_n] = \mathbf{M}^{-1}\mathbf{W}^{\mathrm{T}}(\mathbf{x}_n - \overline{\mathbf{x}}) \tag{16.66}$$

$$\mathbb{E}[\mathbf{z}_n\mathbf{z}_n^{\mathrm{T}}] = \sigma^2\mathbf{M}^{-1} + \mathbb{E}[\mathbf{z}_n]\mathbb{E}[\mathbf{z}_n]^{\mathrm{T}}, \tag{16.67}$$

which follow directly from the posterior distribution (16.43) together with the standard result $\mathbb{E}[\mathbf{z}_n\mathbf{z}_n^{\mathrm{T}}] = \mathrm{cov}[\mathbf{z}_n] + \mathbb{E}[\mathbf{z}_n]\mathbb{E}[\mathbf{z}_n]^{\mathrm{T}}$. Here $\mathbf{M}$ is defined by (16.42).

In the M step, we maximize with respect to $\mathbf{W}$ and σ^2, keeping the posterior statistics fixed. Maximization with respect to σ^2 is straightforward. For the maximization with respect to $\mathbf{W}$, we make use of (A.24) to obtain the M-step equations:

Exercise 16.21

$$\mathbf{W}_{\text{new}} = \left[\sum_{n=1}^{N}(\mathbf{x}_n - \overline{\mathbf{x}})\mathbb{E}[\mathbf{z}_n]^{\mathrm{T}}\right]\left[\sum_{n=1}^{N}\mathbb{E}[\mathbf{z}_n\mathbf{z}_n^{\mathrm{T}}]\right]^{-1} \tag{16.68}$$

$$\sigma^2_{\text{new}} = \frac{1}{ND}\sum_{n=1}^{N}\left\{\|\mathbf{x}_n - \overline{\mathbf{x}}\|^2 - 2\mathbb{E}[\mathbf{z}_n]^{\mathrm{T}}\mathbf{W}_{\text{new}}^{\mathrm{T}}(\mathbf{x}_n - \overline{\mathbf{x}}) + \mathrm{Tr}\left(\mathbb{E}[\mathbf{z}_n\mathbf{z}_n^{\mathrm{T}}]\mathbf{W}_{\text{new}}^{\mathrm{T}}\mathbf{W}_{\text{new}}\right)\right\}. \tag{16.69}$$

The EM algorithm for probabilistic PCA proceeds by initializing the parameters and then alternately computing the sufficient statistics of the latent space posterior distribution using (16.66) and (16.67) in the E step and revising the parameter values using (16.68) and (16.69) in the M step.

One of the benefits of the EM algorithm for PCA is its computational efficiency for large-scale applications (Roweis, 1998). Unlike conventional PCA based on an eigenvector decomposition of the sample covariance matrix, the EM approach is iterative and so might appear to be less attractive. However, each cycle of the EM algorithm can be computationally much more efficient than conventional PCA in spaces of high dimensionality. To see this, note that the eigendecomposition of the covariance matrix requires $\mathcal{O}(D^3)$ computation. Often we are interested only in the first M eigenvectors and their corresponding eigenvalues, in which case we can use algorithms that are $\mathcal{O}(MD^2)$. However, evaluating the covariance matrix requires $\mathcal{O}(ND^2)$ computations, where N is the number of data points. Algorithms such as the snapshot method (Sirovich, 1987), which assume that the eigenvectors are linear combinations of the data vectors, avoid a direct evaluation of the covariance matrix but are $\mathcal{O}(N^3)$ and hence unsuited to large data sets. The EM algorithm described here also does not construct the covariance matrix explicitly. Instead, the most computationally demanding steps are those involving sums over the data set that are $\mathcal{O}(NDM)$. For large D, and $M \ll D$, this can be a significant saving compared to $\mathcal{O}(ND^2)$ and can offset the iterative nature of the EM algorithm.

Note that this EM algorithm can be implemented in an online form in which each D-dimensional data point is read in and processed and then discarded before the next data point is considered. To see this, note that the quantities evaluated in the E step (an M-dimensional vector and an $M \times M$ matrix) can be computed for each data point separately, and in the M step we need to accumulate sums over data points, which we can do incrementally. This approach can be advantageous if both N and D are large.

Because we now have a fully probabilistic model for PCA, we can deal with missing data, provided that it is *missing at random*, in other words that the process that determines which values are missing does not depend on the values of any observed or unobserved variables. Such data sets can be handled by marginalizing over the distribution of the unobserved variables, and the resulting likelihood function can be maximized using the EM algorithm.

Exercise 16.22

16.3.2 EM for PCA

Another elegant feature of the EM approach is that we can take the limit $\sigma^2 \to 0$, corresponding to standard PCA, and still obtain a valid EM-like algorithm (Roweis, 1998). From (16.67), we see that the only quantity we need to compute in the E step is $\mathbb{E}[\mathbf{z}_n]$. Furthermore, the M step is simplified because $\mathbf{M} = \mathbf{W}^{\mathrm{T}}\mathbf{W}$. To emphasize the simplicity of the algorithm, let us define $\widetilde{\mathbf{X}}$ to be a matrix of size $N \times D$ whose nth row is given by the vector $\mathbf{x}_n - \overline{\mathbf{x}}$ and similarly define $\boldsymbol{\Omega}$ to be a matrix of size $M \times N$ whose nth column is given by the vector $\mathbb{E}[\mathbf{z}_n]$. The E step (16.66) of the EM algorithm for PCA then becomes

$$\boldsymbol{\Omega} = (\mathbf{W}_{\mathrm{old}}^{\mathrm{T}}\mathbf{W}_{\mathrm{old}})^{-1}\mathbf{W}_{\mathrm{old}}^{\mathrm{T}}\widetilde{\mathbf{X}}^{\mathrm{T}} \tag{16.70}$$

and the M step (16.68) takes the form

$$\mathbf{W}_{\mathrm{new}} = \widetilde{\mathbf{X}}^{\mathrm{T}}\boldsymbol{\Omega}^{\mathrm{T}}(\boldsymbol{\Omega}\boldsymbol{\Omega}^{\mathrm{T}})^{-1}. \tag{16.71}$$

Again these can be implemented in an online form. These equations have a simple interpretation as follows. From our earlier discussion, we see that the E step involves an orthogonal projection of the data points onto the current estimate for the principal subspace. Correspondingly, the M step represents a re-estimation of the principal subspace to minimize the reconstruction error in which the projections are fixed.

Exercise 16.23

We can give a simple physical analogy for this EM algorithm, which is easily visualized for $D = 2$ and $M = 1$. Consider a collection of data points in two dimensions, and let the one-dimensional principal subspace be represented by a solid rod. Now attach each data point to the rod via a spring obeying Hooke's law (force is proportional to the length of the spring and therefore stored energy is proportional to the square of the spring's length). In the E step, we keep the rod fixed and allow the attachment points to slide up and down the rod so as to minimize the energy. This causes each attachment point (independently) to position itself at the orthogonal projection of the corresponding data point onto the rod. In the M step, we keep the attachment points fixed and then release the rod and allow it to move to the minimum energy position. The E step and M step are then repeated until a suitable convergence criterion is satisfied, as is illustrated in Figure 16.10.

16.3.3 EM for factor analysis

Section 16.2.4

We can determine the parameters $\boldsymbol{\mu}$, $\mathbf{W}$, and $\boldsymbol{\Psi}$ in a factor analysis model by maximum likelihood. The solution for $\boldsymbol{\mu}$ is again given by the sample mean. However, unlike probabilistic PCA, there is no longer a closed-form maximum likelihood solution for $\mathbf{W}$, which must therefore be found iteratively. Because factor analysis is a latent-variable model, this can be done using an EM algorithm (Rubin and Thayer, 1982) that is analogous to the one used for probabilistic PCA. Specifically, the E-step equations are given by

Exercise 16.24

$$\mathbb{E}[\mathbf{z}_n] = \mathbf{G}\mathbf{W}^{\mathrm{T}}\boldsymbol{\Psi}^{-1}(\mathbf{x}_n - \overline{\mathbf{x}}) \tag{16.72}$$

$$\mathbb{E}[\mathbf{z}_n\mathbf{z}_n^{\mathrm{T}}] = \mathbf{G} + \mathbb{E}[\mathbf{z}_n]\mathbb{E}[\mathbf{z}_n]^{\mathrm{T}} \tag{16.73}$$

where we have defined

$$\mathbf{G} = (\mathbf{I} + \mathbf{W}^{\mathrm{T}}\boldsymbol{\Psi}^{-1}\mathbf{W})^{-1}. \tag{16.74}$$

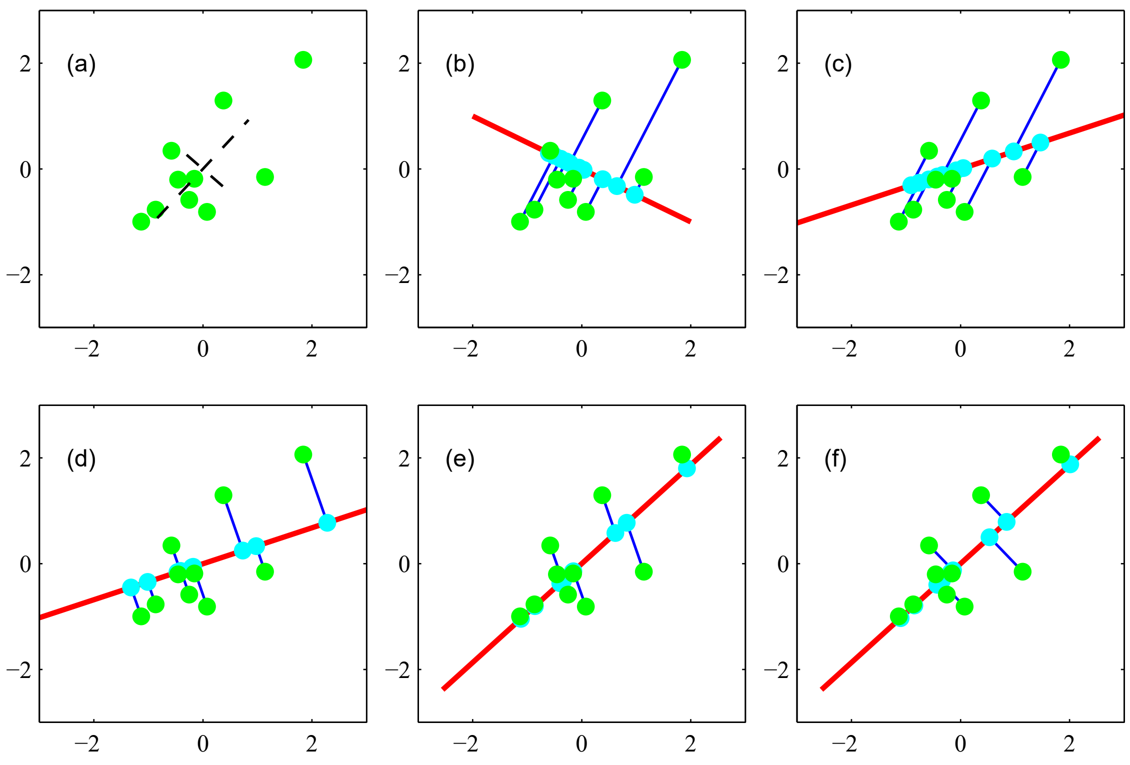


Figure 16.10 Synthetic data illustrating the EM algorithm for PCA defined by (16.70) and (16.71). (a) A set of data points shown in green, together with the true principal components (shown as eigenvectors scaled by the square roots of the eigenvalues). (b) Initial configuration of the principal subspace defined by $\mathbf{W}$, shown in red, together with the projections of the latent points $\mathbf{Z}$ into the data space, given by $\mathbf{Z}\mathbf{W}^{\mathrm{T}}$, shown in cyan. (c) After one M step, $\mathbf{W}$ has been updated with $\mathbf{Z}$ held fixed. (d) After the successive E step, the values of $\mathbf{Z}$ have been updated, giving orthogonal projections, with $\mathbf{W}$ held fixed. (e) After the second M step. (f) The converged solution.

Note that this is expressed in a form that involves inversion of matrices of size $M \times M$ rather than $D \times D$ (except for the $D \times D$ diagonal matrix $\boldsymbol{\Psi}$ whose inverse is trivial to compute in $\mathcal{O}(D)$ steps), which is convenient because often $M \ll D$. Similarly, the M-step equations take the form

Exercise 16.25

$$\mathbf{W}_{\text{new}} = \left[\sum_{n=1}^{N}(\mathbf{x}_n - \overline{\mathbf{x}})\mathbb{E}[\mathbf{z}_n]^{\mathrm{T}}\right]\left[\sum_{n=1}^{N}\mathbb{E}[\mathbf{z}_n\mathbf{z}_n^{\mathrm{T}}]\right]^{-1} \tag{16.75}$$

$$\boldsymbol{\Psi}_{\text{new}} = \operatorname{diag}\left\{\mathbf{S} - \mathbf{W}_{\text{new}}\frac{1}{N}\sum_{n=1}^{N}\mathbb{E}[\mathbf{z}_n](\mathbf{x}_n - \overline{\mathbf{x}})^{\mathrm{T}}\right\} \tag{16.76}$$

where the diag operator sets all the non-diagonal elements of a matrix to zero.

16.4. Nonlinear Latent Variable Models

So far in this chapter we have focused on latent variable models based on linear transformations from the latent space to the data space. It is natural to ask whether we can use the flexibility of deep neural networks to represent more complex transformations, while exploiting the learning ability of deep networks to allow the resulting distribution to be fitted to a data set. Consider a simple distribution over a vector variable $\mathbf{z}$, for example a Gaussian of the form

$$p_{\mathbf{z}}(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I}). \tag{16.77}$$

Now suppose we transform $\mathbf{z}$ using a function $\mathbf{x} = \mathbf{g}(\mathbf{z}, \mathbf{w})$ given by a deep neural network, where $\mathbf{w}$ represents the weights and biases. The combination of the distribution over $\mathbf{z}$ together with the neural network defines a distribution over $\mathbf{x}$. Sampling from such a model is straightforward because we can generate samples from $p_{\mathbf{z}}(\mathbf{z})$ and then transform each of them using the neural network function to give corresponding samples of $\mathbf{x}$. This is an efficient process since it does not involve iteration.

To learn $\mathbf{g}(\mathbf{z}, \mathbf{w})$ from data, consider how to evaluate the likelihood function $p(\mathbf{x}|\mathbf{w})$. The distribution of $\mathbf{x}$ is given by the change of variables formula for densities:

Section 2.4

$$p_{\mathbf{x}}(\mathbf{x}) = p_{\mathbf{z}}(\mathbf{z}(\mathbf{x}))\,|\det \mathbf{J}(\mathbf{x})| \tag{16.78}$$

where $\mathbf{J}$ is the Jacobian matrix of partial derivatives whose elements are given by

$$J_{ij}(\mathbf{x}) = \frac{\partial z_i}{\partial x_j}. \tag{16.79}$$

To evaluate the distribution $p_{\mathbf{z}}(\mathbf{z}(\mathbf{x}))$ on the right-hand side of (16.78) for a given data vector $\mathbf{x}$ and to evaluate the Jacobian matrix in (16.79) for that same value of $\mathbf{x}$, we need the inverse $\mathbf{z} = \mathbf{g}^{-1}(\mathbf{x}, \mathbf{w})$ of the neural network function. For most neural networks this inverse will not be well defined. For example, the network may represent a many-to-one function in which multiple different input values map to the same output value, in which case the change of variable formula does not give a well-defined density. Moreover, if the dimensionality of the latent space is different from that of the data space then the transformation will not be invertible.

One approach is to restrict our attention to functions $\mathbf{g}(\mathbf{z}, \mathbf{w})$ that are invertible, which requires that $\mathbf{z}$ and $\mathbf{x}$ have the same dimensionality. We will explore this approach in more detail when we introduce the technique of *normalizing flows*.

Chapter 18

16.4.1 Nonlinear manifolds

Requiring that the latent and data spaces have the same number of dimensions is a significant limitation. Consider the situation in which $\mathbf{z}$ has dimensionality M and $\mathbf{x}$ has dimensionality D, where $M < D$. In this case the distribution over $\mathbf{x}$ is confined to a *manifold*, or subspace, of dimensionality M, as illustrated in Figure 16.11. Low-dimensional manifolds arise in many machine learning applications,

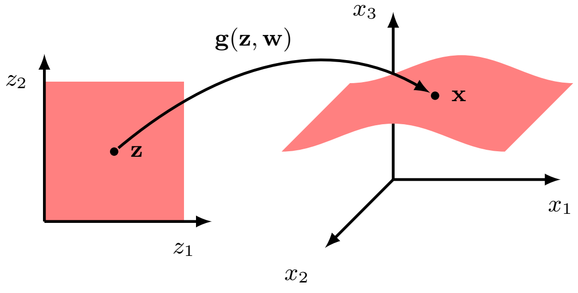


Figure 16.11 Illustration of a mapping from a two-dimensional latent space $\mathbf{z} = (z_1, z_2)$ to a three-dimensional data space $\mathbf{x} = (x_1, x_2, x_3)$ using a nonlinear function $\mathbf{x} = \mathbf{g}(\mathbf{z}, \mathbf{w})$ represented by a neural network with parameter vector $\mathbf{w}$.

Section 6.1.4

for example when modelling the distribution of natural images. Nonlinear latent-variable models can be very useful in modelling such data because they express the strong inductive bias that the data does not 'fill' the data space but is confined to a manifold, although the shape and dimensionality of this manifold are typically not known in advance.

However, one problem with this framework is that it assigns zero probability density to any data vector that does not lie *exactly* on the manifold, which is a problem for gradient-based learning since the likelihood function will be zero at each of the data points and constant for small changes in $\mathbf{w}$, for any realistic data set. To address this, we follow the approach used previously with regression and classification problems and define a conditional distribution across the entire data space, whose parameters are given by the output of the neural network. If, for example, $\mathbf{x}$ comprises a vector of continuous variables then we can choose the conditional distribution to be a Gaussian:

$$p(\mathbf{x}|\mathbf{z}, \mathbf{w}) = \mathcal{N}(\mathbf{x}|\mathbf{g}(\mathbf{z}, \mathbf{w}), \sigma^2\mathbf{I}) \tag{16.80}$$

in which the neural network $\mathbf{g}(\mathbf{z}, \mathbf{w})$ has linear output-unit activation functions, and $\mathbf{g} \in \mathbb{R}^D$. The generative model is specified by the latent distribution over $\mathbf{z}$ together with the conditional distribution over $\mathbf{x}$, and can be represented by the simple graphical model shown in Figure 16.12.

Note that it is straightforward, and computationally efficient, to draw independent samples from this distribution. We first draw a sample from the Gaussian distribution (16.77) using standard methods. Next, we use this value as input to the neural network, giving an output value $\mathbf{g}(\mathbf{z}, \mathbf{w})$. Finally, we draw a sample from a Gaussian distribution with mean $\mathbf{g}(\mathbf{z}, \mathbf{w})$ and covariance $\sigma^2\mathbf{I}$, as defined by (16.80). This three-step process can then be repeated to generate multiple independent samples.

Section 14.1.2

The combination of a latent-variable distribution $p(\mathbf{z})$ and a conditional distri-

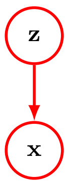
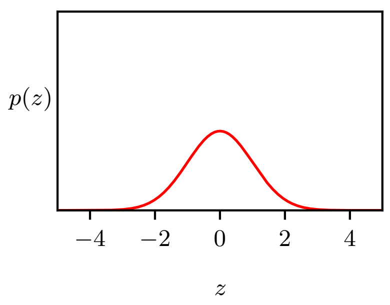
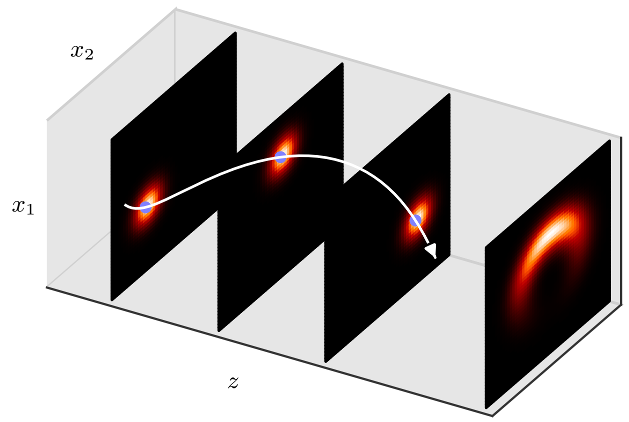


Figure 16.12 Graphical model representing the distribution given by (16.77) and (16.80), which together define a joint distribution $p(\mathbf{x}, \mathbf{z}) = p(\mathbf{x}|\mathbf{z})p(\mathbf{z})$.



(a)



(b)

Figure 16.13 Illustration of a nonlinear latent-variable model for a one-dimensional latent space and a two-dimensional data space. (a) The prior distribution in latent space is given by a zero-mean unit-variance Gaussian distribution. (b) The three left-most plots show examples of the Gaussian conditional distribution $p(\mathbf{x}|z)$ for different values of z, whereas the right-most plot shows the marginal distribution $p(\mathbf{x})$. The nonlinear function $\mathbf{g}(z)$, which defines the mean of the conditional distribution, is given by $g_1(z) = \sin(z)$, $g_2(z) = \cos(z)$, and, therefore, traces out a circle in data space. The standard deviation of the conditional distribution is given by $\sigma = 0.3$. [Based on Prince (2020) with permission.]

bution $p(\mathbf{x}|\mathbf{z})$ defines a marginal distribution over the data space given by

$$p(\mathbf{x}) = \int p(\mathbf{z})p(\mathbf{x}|\mathbf{z})\,\mathrm{d}\mathbf{z}. \tag{16.81}$$

We illustrate this using a simple example involving a one-dimensional latent space and a two-dimensional data space in Figure 16.13.

16.4.2 Likelihood function

We have seen that it is easy to draw samples from this nonlinear latent-variable model. Now suppose we wish to fit the model to an observed data set by maximizing the likelihood function. The likelihood is obtained from the product and sum rules of probability by integrating over $\mathbf{z}$:

$$\begin{aligned} p(\mathbf{x}|\mathbf{w}) &= \int p(\mathbf{x}|\mathbf{z},\mathbf{w})p(\mathbf{z})\,\mathrm{d}\mathbf{z} \\ &= \int \mathcal{N}(\mathbf{x}|\mathbf{g}(\mathbf{z},\mathbf{w}),\sigma^2\mathbf{I})\mathcal{N}(\mathbf{z}|\mathbf{0},\mathbf{I})\,\mathrm{d}\mathbf{z}. \end{aligned} \tag{16.82}$$

Although both distributions inside the integral are Gaussian, the integral is analytically intractable due to the highly nonlinear function $\mathbf{g}(\mathbf{z},\mathbf{w})$ defined by the neural network.

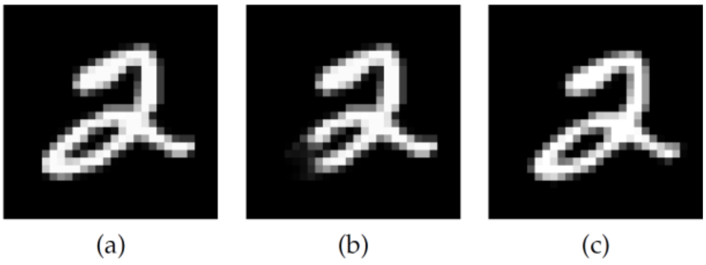


Figure 16.14 Three example images of handwritten digits, illustrating why sampling from the latent space to evaluate the likelihood function requires large numbers of samples. (a) shows the original image, (b) shows a corrupted image with part of the stroke removed, and (c) shows the original image shifted by half a pixel down and half a pixel to the right. Image (b) is closer to (a) in terms of likelihood, even though image (c) is much closer to (a) in appearance. [From Doersch (2016) with permission.]

One approach for evaluating the likelihood function would be to draw samples from the latent space distribution and use these to approximate (16.82) by

$$p(\mathbf{x}|\mathbf{w}) \simeq \frac{1}{K}\sum_{i=1}^{K} p(\mathbf{x}|\mathbf{z}_i, \mathbf{w}) \tag{16.83}$$

where $\mathbf{z}_i \sim p(\mathbf{z})$. This expresses the distribution over $\mathbf{z}$ as a mixture of Gaussians with fixed mixing coefficients given by $1/K$, and in the limit of an infinite number of samples, this gives the true likelihood function. However, the value of K needed for effective training will typically be far too high to be practical. To see why, consider the three images of handwritten digits shown in Figure 16.14, and suppose that image (a) represents the vector $\mathbf{x}$ for which we wish to evaluate the likelihood function. If a trained model generated image (b), we would consider this a poor model as this image is not a good representation of a digit '2', and so this should be assigned a much lower likelihood. Conversely, image (c), which was obtained by shifting the digit in (a) down and to the right by half a pixel, is a good example of a digit '2' and should therefore have a high likelihood. Since the distribution (16.80) is Gaussian, the likelihood function is proportional to the exponential of the negative squared distance between the output of the network and the data vector $\mathbf{x}$. However, the squared distance between (a) and (b) is 0.0387 whereas the squared distance between (a) and (c) is 0.2693. So if the variance parameter σ^2 is set to a sufficiently small value that image (b) has low likelihood, then image (c) will have an even lower likelihood. Even if the model is good at generating digits, we would have to consider extremely large numbers of samples for $\mathbf{z}$ before seeing a digit that is sufficiently close to (a). We therefore seek more sophisticated techniques for training nonlinear latent variable models that can be used in practical applications. Before outlining such methods, we first discuss briefly some considerations regarding discrete data spaces.

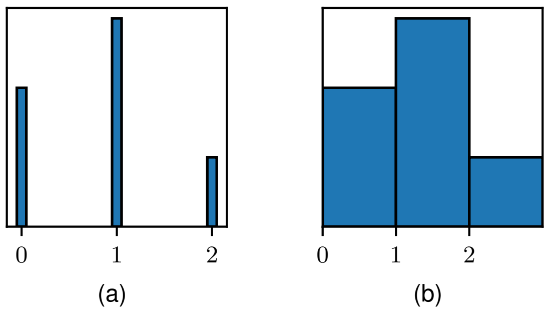


Figure 16.15 Schematic illustration of dequantization, showing (a) a discrete distribution over a single variable and (b) an associated dequantized continuous distribution.

16.4.3 Discrete data

If the observed data set comprises independent binary variables then we can use a conditional distribution of the form

$$p(\mathbf{x}|\mathbf{z}, \mathbf{w}) = \prod_{i=1}^{D} g_i(\mathbf{z}, \mathbf{w})^{x_i} \left(1 - g_i(\mathbf{z}, \mathbf{w})\right)^{1-x_i} \tag{16.84}$$

where $g_i(\mathbf{z}, \mathbf{w}) = \sigma(a_i(\mathbf{z}, \mathbf{w}))$ represents the activation of output unit i, the activation function $\sigma(\cdot)$ is given by the logistic sigmoid, and $a_i(\mathbf{z}, \mathbf{w})$ is the pre-activation for output unit i. Similarly, for one-hot encoded categorical variables, we can use a multinomial distribution:

$$p(\mathbf{x}|\mathbf{z}, \mathbf{w}) = \prod_{i=1}^{D} g_i(\mathbf{z}, \mathbf{w})^{x_i} \tag{16.85}$$

where

$$g_i(\mathbf{z}, \mathbf{w}) = \frac{\exp(a_i(\mathbf{z}, \mathbf{w}))}{\sum_j \exp(a_j(\mathbf{z}, \mathbf{w}))} \tag{16.86}$$

is the softmax activation function. We can also consider combinations of discrete and continuous variables by forming the product of the associated conditional distributions.

In practice, continuous variables are represented with discrete values, for example in images, the red, green, and blue channel intensities might be expressed using 8-bit numbers representing the values $\{0, \ldots, 255\}$. This can cause problems when we employ highly flexible models based on deep neural networks, as the likelihood function can go to zero if the density collapses onto one or more of the discrete values. The problem can be resolved using a technique called *dequantization*, which involves adding noise to the variables, typically drawn from a uniform distribution over the region between successive discrete values, as shown in Figure 16.15. A training set is dequantized by replacing each observed value with a sample drawn randomly from the associated continuous distribution associated with that discrete value, and this makes it less likely that the model will discover a pathological solution.

16.4.4 Four approaches to generative modelling

We have seen that nonlinear latent-variable models based on deep neural networks offer a highly flexible framework for building generative models. Due to the universality of the neural network transformation, such models are capable, in principle, of approximating essentially any desired distribution to high accuracy. Moreover, such models offer the potential, once trained, to generate samples from the distribution in using an efficient, non-iterative process. However, we have also identified some challenges associated with training such models that force us to develop more sophisticated techniques than those needed for linear models. Many such methods have been proposed, each having their own strengths and limitations. These can be broadly grouped into four approaches, as follows.

Chapter 17

With *generative adversarial networks*, or GANs, we relax the requirement for the network mapping to be invertible, thereby allowing the latent space to have a lower dimensionality than the data space. We also abandon the concept of a likelihood function and instead introduce a second neural network whose function is to provide a training signal for the generative network. Due to the absence of a well-defined likelihood function, the training procedure may be brittle, but once trained it is straightforward to generate samples from the model, and the results can be of high quality.

Chapter 19

The framework of *variational autoencoders*, or VAEs, also uses a second neural network whose role is to approximate the posterior distribution over the latent variables, thereby allowing an approximation to the likelihood function to be evaluated. Training is more robust than with GANs, and sampling from the trained model is straightforward, although it can be harder to obtain the highest quality results.

Chapter 18

In *normalizing flows*, we set the dimensionality of the latent space to be equal to that of the data space and then modify the generative neural network so that it becomes invertible. The requirement that the network is invertible restricts its functional form but it allows the likelihood function to be evaluated without approximation and it also allows for efficient sampling.

Chapter 20

Finally, *diffusion models* use a network that learns to transform a sample from the prior distribution into a sample from the data distribution through a sequence of denoising steps. This leads to state-of-the-art performance in many applications, although the cost of sampling can be high due to the multiple denoising passes through the network.

We explore these approaches in detail in the final four chapters of this book.

Exercises

16.1 ($\star\star$) In this exercise, we use proof by induction to show that the linear projection onto an M-dimensional subspace that maximizes the variance of the projected data is defined by the M eigenvectors of the data covariance matrix $\mathbf{S}$, given by (16.3), corresponding to the M largest eigenvalues. In Section 16.1, this result was proven for $M = 1$. Now suppose the result holds for some general value of M and show that it consequently holds for dimensionality $M + 1$. To do this, first set the derivative of the variance of the projected data with respect to a vector $\mathbf{u}_{M+1}$ defining the new