2

Characteristics of an Analysis Method

Overview. This chapter lists and describes the functionalities that the user usually expects from a method of analyzing high-dimensional data. Next, more mathematical details are given, such as the way data are modeled, the criterion to optimize, and the algorithm that implements them. Finally, the chapter ends by introducing a typical method, namely the principal component analysis (PCA). The method is briefly assessed by applying it to simple examples. Afterwards, and because of the limitations of linear methods like principal component analysis, this chapter promotes the use of methods that can reduce the dimensionality in a nonlinear way. In contrast with PCA, these more complex methods assume that data are generated according to a nonlinear model. The last section of this chapter attempts to list some important characteristics that allow us to classify the methods in various categories. These characteristics are, among others, the way data are modeled, the criterion to be optimized, the way it is optimized, and the kind of algorithm that implements the method.

2.1 Purpose

This chapter aims at gathering all features or properties that characterize a method of analyzing high-dimensional data. The first section lists some functionalities that the user usually expects. The next sections present more technical characterictics like the mathematical or statistical model that underlies the method, the type of algorithm that identifies the model parameters, and, last but not least, the criterion optimized by the method. Although the criterion ends the list, it often has a great influence on other characteristics. Depending on the criterion, indeed, some functionalities are available or not; similarly, the optimization of a given criterion is achieved more easily with some type of algorithm and may be more difficult with another one.

2.2 Expected functionalities

As mentioned in the previous chapter, the analysis of high-dimensional data amounts to identifying and eliminating the redundancies among the observed variables. This requires three main functionalities: an ideal method should indeed be able to

1. Estimate the number of latent variables.
2. Embed data in order to reduce their dimensionality.
3. Embed data in order to recover the latent variable.

Before detailing them, it is noteworthy that these three functionalities are not always available together in most methods. Very often, methods are able to either reduce the dimensionality or separate latent variables, but can rarely perform both. Furthermore, only a few methods include an estimator of the intrinsic dimensionality: most of them take the dimensionality as an external hyperparameter and need an additional algorithm to evaluate it.

In some cases, two or even three methods can be combined in order to achieve the three tasks: a first method gives the number of latent variables, a second one yields a low-dimensional representation of data, and, if necessary, a third one further transforms data in order to retrieve the latent variables.

2.2.1 Estimation of the number of latent variables

The first necessary step to extract information from high-dimensional data consists of computing the number of latent variables. Sometimes latent variables are also called *degrees of freedom*. But how many are there? How do we estimate their number given only a few observations in a data set?

Detailed answers to those questions are given in Chapter 3. At this point, it is just useful to know that the number of latent variables is often computed from a topological point of view, by estimating the *intrinsic dimension(ality)* of data. In contrast with a number of variables, which is necessarily an integer value, the intrinsic dimension hides a more generic concept and may take on real values. As the intrinsic dimension is the most common way to estimate the number of latent variables, the term "intrinsic dimension(ality)" is often used instead of "number of latent variables" further in this book.

The intrinsic dimension reveals the presence of a topological structure in data. When the intrinsic dimension P of data equals the dimension D of the embedding space, there is no structure: there are enough degrees of freedom so that an ϵ-ball centered on any data point can virtually be completely filled by other data points. On the contrary, when $P < D$, data points are often[1] constrained to lie in a well-delimited subspace. Consequently, a low intrinsic dimension indicates that a topological object or structure underlies the data

[1] Not always, due to the existence of fractal objects; see Chapter 3.

set. Figure 2.1 gives an example: a two-dimensional object (a surface or 2-manifold) is embedded in a three-dimensional Euclidean space. Intuitively, two parameters (or degrees of freedom or latent variables) suffice to fully describe the manifold. The intrinsic dimension estimated on a few points drawn from the surface confirms that intuition.

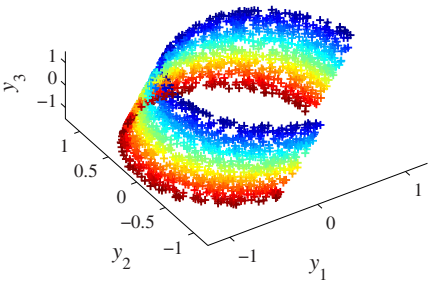

Fig. 2.1. A two-dimensional manifold embedded in a three-dimensional space. The data set contains only a finite number of points (or observations).

Without a good estimate of the intrinsic dimension, dimensionality reduction is no more than a risky bet since one does not known to what extent the dimensionality can be reduced.

2.2.2 Embedding for dimensionality reduction

The knowledge of the intrinsic dimension P indicates that data have some topological structure and do not completely fill the embedding space. Quite naturally, the following step would consist of re-embedding the data in a lower-dimensional space that would be better filled. The aims are both to get the most compact representation and to make any subsequent processing of data more easy. Typical applications include data compression and visualization.

More precisely, if the estimate of the intrinsic dimensionality P is reliable, then two assumptions can be made. First, data most probably hide a P-dimensional manifold.[2] Second, it is possible to re-embed the underlying P-dimensional manifold in a space having dimensionality between P and D, hopefully closer to P than D.

Intuitively, dimensionality reduction aims at re-embedding data in such way that the manifold structure is preserved. If this constraint is relaxed, then dimensionality reduction no longer makes sense. The main problem is,

[2] Of course, this is not necessarily true, as P is a *global* estimator and data may be a combination of several manifolds with various *local* dimensionalities.

of course, how to measure or characterize the structure of a manifold in order to preserve it.

Figure 2.2 shows a two-dimensional embedding of the manifold that was initially shown in a three-dimensional space in Fig. 2.1. After dimensionality reduction, the structure of the manifold is now completely unveiled: it is a rectangle. Obviously, for this toy example, that statement could have already been deduced by looking at the three-dimensional representation in Fig. 2.1. In this example, such a visual clue simply confirms that the dimensionality reduction worked properly and preserved the structure of the object. Actually, from the viewpoint of topology, the curvature of the rectangle in its three-dimensional embedding does not really matter, in contrast with the connectivity and local relationships between data points. More importantly, the dimensionality reduction establishes a one-to-one mapping between the three-dimensional points and the two-dimensional ones. This mapping allows us to go back to the initial embedding if necessary.

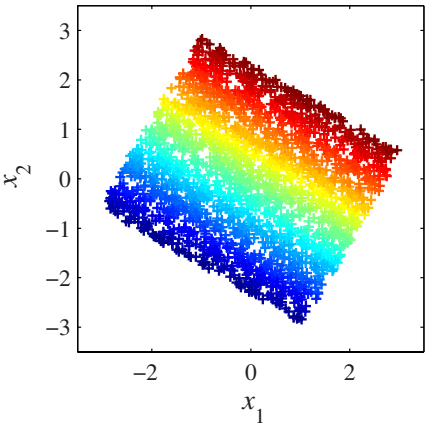

Fig. 2.2. Possible two-dimensional embedding for the object in Fig. 2.1. The dimensionality of the data set has been reduced from three to two.

2.2.3 Embedding for latent variable separation

Dimensionality reduction aims at decreasing the number of variables that describe data. In fact, most DR methods can only achieve that precise task. By comparison, the recovery of latent variables goes a step further than dimensionality reduction. Additional constraints are imposed on the desired low-dimensional representation.

These constraints are generally not related to topology. For example, it is often assumed that the latent variables that generated the data set are

(statistically) independent from each other. In this case, the low-dimensional representation must also satisfy this property in order to state that the latent variables have been retrieved.

The example of Fig. 2.1, which has been re-embedded in Fig. 2.2, can be further processed, as in Fig. 2.3. In the latter representation, the two parameters of the representation, corresponding to the axes of the coordinate system, have been made independent. Intuitively, it can be seen that knowing the abscissa of a point gives no clue about the ordinate of the same point, and vice versa. This was not the case in Fig. 2.2: each abscissa determines a different interval where the ordinate may lie. It is noteworthy that the

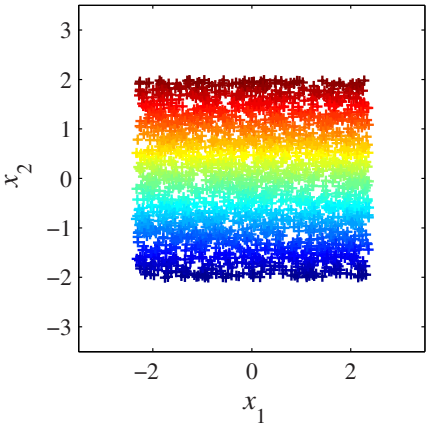

Fig. 2.3. Particular two-dimensional embedding for the object in Fig. 2.1. The latent variables, corresponding to the axes of the coordinate system, are independent from each other.

latent variable separation, whose result is illustrated in Fig. 2.3, has not been obtained directly from the three-dimensional data set in Fig. 2.1. Instead, it has been determined by modifying the low-dimensional representation of Fig. 2.2. And actually, most methods of latent variable separation are not able to reduce the dimensionality by themselves: they need another method or some kind of preprocessing to achieve it. Moreover, the additional constraints imposed on the desired representation, like statistical independence, mean that the methods are restricted to very simple data models. For example, observed variables are most often modeled as linear combinations of the latent ones, in order to preserve some of their statistical properties.

2.3 Internal characteristics

Behind the expected functionalities of an analysis method, less visible characteristics are hidden, though they play a key role. These characteristics are

- The model that data are assumed to follow.
- The type of algorithm that identifies the model parameters.
- The criterion to be optimized, which guides the algorithm.

2.3.1 Underlying model

All methods of analysis rely on the assumption that the data sets they are fed with have been generated according to a well-defined model. In colorful terms, the food must be compatible with the stomach: no vegetarian eats meat!

For example, principal component analysis (see Section 2.4) assumes that the dependencies between the variables are linear. Of course, the user should be aware of such a hypothesis, since the type of model determines the power and/or limitations of the method. Consequently for this model choice, PCA often delivers poor results when trying to project data lying on a nonlinear subspace. This is illustrated in Fig. 2.4, where PCA has been applied to the data set displayed in Fig. 2.1.

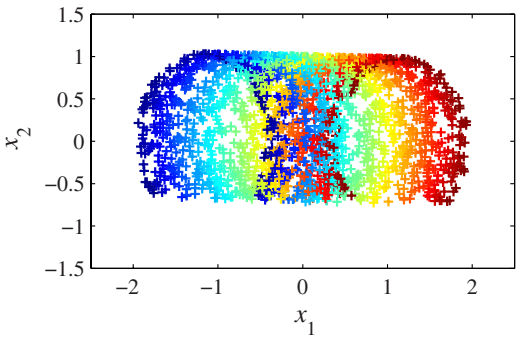

Fig. 2.4. Dimensionality reduction by PCA from 3 to 2 for the data set of Fig. 2.1. Obviously, data do not fit the model of PCA, and the initial rectangular distribution cannot be retrieved.

Hence, even for the relatively simple toy example in Fig. 2.1, methods based on a nonlinear data model seem to be preferable. The embedding of Fig. 2.2 is obtained by such a nonlinear method: the result is visually much more convincing.

The distinction between linear and nonlinear models is not the only one. For example, methods may have a continuous model or a discrete one. In the

first case, the model parameters completely define a continuous function (or mapping) between the high- and low-dimensional spaces. In the second case, the model parameters determine only a few values of such a function.

2.3.2 Algorithm

For the same model, several algorithms can implement the desired method of analysis. For example, in the case of PCA, the model parameters are computed in closed form by using general-purpose algebraic procedures. Most often, these procedures work quickly, without any external hyperparameter to tune, and are guaranteed to find the best possible solution (depending on the criterion, see ahead). Nevertheless, in spite of many advantages, one of their major drawbacks lies in the fact that they are so-called batch methods: they cannot start working until the whole set of data is available.

When data samples arrive one by one, other types of algorithms exist. For example, PCA can also be implemented by so-called online or adaptative algorithms (see Subsection 2.4.4). Each time a new datum is available, on-line algorithms handle it independently from the previous ones and then 'forget' it. Unfortunately, such algorithms do not show the same desirable properties as algebraic procedures:

- By construction, they work iteratively (with a stochastic gradient descent for example).
- They can fall in a local optimum of the criterion, i.e., find a solution that is not exactly the best, but only an approximation.
- They often require a careful adjustment of several hyperparameters (e.g., learning rates) to fasten the convergence and avoid the above-mentioned local minima.

Although PCA can be implemented by several types of algorithms, such versatility does not hold for all methods. Actually, the more complex a model is, the more difficult it is to compute its parameters in closed form. Along with the data model, the criterion to be optimized also strongly influences the algorithm.

2.3.3 Criterion

Despite the criterion is the last item in this list of the method characteristics, it probably plays the most important role. The choice of the criterion often determines which functionalities the method will offer, intervenes in the data model, and always orients the implementation to a particular type of algorithm.

Typically, the criterion to be optimized is written as a mathematical formula. For example, a well-known criterion for dimensionality reduction is the mean square error. In order to compute this criterion, the dimensionality is first reduced and then expanded back, provided that the data model could

be reversed. Most often the loss of information or deterioration of the data structure occurs solely in the first step, but the second is necessary in order to have a comparison reference. Mathematically, this reconstruction error can be written as

$$E_{\text{codec}} = E_{\mathbf{y}}\{\|\mathbf{y} - \text{dec}(\text{cod}(\mathbf{y}))\|_2^2\} \ , \tag{2.1}$$

where $E\{\}$ is the expectation operator; the dimensionality reduction and expansion are respectively denoted by the coding and decoding functions cod and dec:

$$\text{cod} : \mathbb{R}^D \to \mathbb{R}^P, \quad \mathbf{y} \mapsto \mathbf{x} = \text{cod}(\mathbf{y}) \ , \tag{2.2}$$

$$\text{dec} : \mathbb{R}^P \to \mathbb{R}^D, \quad \mathbf{x} \mapsto \mathbf{y} = \text{dec}(\mathbf{x}) \ . \tag{2.3}$$

As explained in the next section, PCA can be derived from the reconstruction error. Of course, other criteria exist. For example, statisticians may wish to get a projection that preserves the variance initially observable in the raw data. From a more geometrical or topological point of view, the projection of the object should preserve its structure, for example, by preserving the pairwise distances measured between the observations in the data set.

If the aim is latent variable separation, then the criterion can be decorrelation. This criterion can be further enriched by making the estimated latent variables as independent as possible. The latter idea points toward independent component analysis (ICA), which is out of the scope of this book. The interested reader can find more details in [95, 34] and references therein.

As shown in the next section, several criterions described above, like minimizing the reconstruction error, maximizing the variance preservation, maximizing the distance preservation, or even decorrelating the observed variables, lead to PCA when one considers a simple linear model.

2.4 Example: Principal component analysis

Principal component analysis (PCA in short) is perhaps one of the oldest and best-known methods in multivariate analysis and data mining. PCA was introduced by Pearson [149], who used it in a biological framework. Next, PCA was further developed by Hotelling [92] in the field of psychometry. In the framework of stochastic processes, PCA was also discovered independently by Karhunen [102] and was subsequently generalized by Loève [128]. This explains why PCA is also known as the "Karhunen-Loève transform" (or expansion) in this field.

2.4.1 Data model of PCA

The model of PCA essentially assumes that the D observed variables, gathered in the random vector $\mathbf{y} = [y_1, \ldots, y_d, \ldots, y_D]^T$, result from a linear transformation $\mathbf{W}$ of P unknown latent variables, written as $\mathbf{x} = [x_1, \ldots, x_p, \ldots, x_P]^T$:

$$\mathbf{y} = \mathbf{W}\mathbf{x} \ . \tag{2.4}$$

All latent variables are assumed to have a Gaussian distribution (see Appendix B). Additionally, transformation $\mathbf{W}$ is constrained to be an axis change, meaning that the columns $\mathbf{w}_d$ of $\mathbf{W}$ are orthogonal to each other and of unit norm. In other words, the D-by-P matrix $\mathbf{W}$ is a matrix such that $\mathbf{W}^T\mathbf{W} = \mathbf{I}_P$ (but the permuted product $\mathbf{W}\mathbf{W}^T$ may differ from $\mathbf{I}_D$). A last important but not too restrictive hypothesis of PCA is that both the observed variables $\mathbf{y}$ and the latent ones $\mathbf{x}$ are centered, i.e., $E_{\mathbf{y}}\{\mathbf{y}\} = \mathbf{0}_D$ and $E_{\mathbf{x}}\{\mathbf{x}\} = \mathbf{0}_P$.

Starting from this model, how can the dimension P and the linear transformation $\mathbf{W}$ be identified starting from a finite sample of the observed variables? Usually, the sample is an unordered set of N observations (or realizations) of the random vector $\mathbf{y}$:

$$\mathcal{Y} = \{\mathbf{y}(1), \ldots, \mathbf{y}(n), \ldots, \mathbf{y}(N)\} \ , \tag{2.5}$$

but it is often more convenient to write it in matrix form:

$$\mathbf{Y} = [\mathbf{y}(1), \ldots, \mathbf{y}(n), \ldots, \mathbf{y}(N)] \ . \tag{2.6}$$

Preprocessing

Before determining P and $\mathbf{W}$, it must be checked that the observations are centered. If this is not the case, they can be centered by removing the expectation of $\mathbf{y}$ from each observation $\mathbf{y}(n)$:

$$\mathbf{y}(n) \leftarrow \mathbf{y}(n) - E_{\mathbf{y}}\{\mathbf{y}\} \ , \tag{2.7}$$

where the left arrow means that the variable on the left-hand side is assigned a new value indicated in the right-hand side. Of course, the exact expectation of $\mathbf{y}$ is often unknown and must be approximated by the sample mean:

$$E_{\mathbf{y}}\{\mathbf{y}\} \approx \frac{1}{N}\sum_{n=1}^{N}\mathbf{y}(n) = \frac{1}{N}\mathbf{Y}\mathbf{1}_N \ . \tag{2.8}$$

With the last expression of the sample mean in matrix form, the centering can be rewritten for the entire data set as

$$\mathbf{Y} \leftarrow \mathbf{Y} - \frac{1}{N}\mathbf{Y}\mathbf{1}_N\mathbf{1}_N^T \ . \tag{2.9}$$

Once data are centered, P and $\mathbf{W}$ can be identified by PCA.

Nevertheless, the data set may need to be further preprocessed. Indeed, the components y_d of the observed vector $\mathbf{y}$ may come from very different origins. For example, in multivariate data analysis, one variable could be a weight expressed in kilograms and another variable a length expressed in millimeters.

But the same variables could as well be written in other units, like grams and meters. In both situations, it is expected that PCA detects the same dependencies between the variables in order to yield the same results. A simple way to solve this indeterminacy consists of standardizing the variables, i.e., dividing each y_d by its standard deviation after centering. Does this mean that the observed variables should always be standardized? The answer is negative, and actually the standardization could even be dangerous when some variable has a low standard deviation. Two cases should be distinguished from the others:

- When a variable is zero, its standard deviation is also zero. Trivially, the division by zero must be avoided, and the variable should be discarded. Alternatively, PCA can detect and remove such a useless zero-variable in a natural way.
- When noise pollutes an observed variable having a small standard deviation, the contribution of the noise to the standard deviation may be proportionally large. This means that discovering the dependency between that variable and the other ones can be difficult. Therefore, that variable should intuitively be processed exactly as in the previous case, that is, either by discarding it or by avoiding the standardization. The latter could only amplify the noise. By definition, noise is independent from all other variables and, consequently, PCA will regard the standardized variable as an important one, while the same variable would have been a minor one without standardization.

These two simple cases demonstrate that standardization can be useful but may not be achieved blindly. Some knowledge about the data set is necessary.

After centering (and standardization if appropriate), the parameters P and $\mathbf{W}$ can be identified by PCA. Of course, the exact values of P and $\mathbf{W}$ depend on the criterion optimized by PCA.

2.4.2 Criteria leading to PCA

PCA can be derived from several criteria, all leading to the same method and/or results. Two criteria — minimal reconstruction error and maximal preserved variance — are developed in the following two subsections. A third criterion — distance preservation — is detailed further in Subsection 4.2.2, which describes metric multidimensional scaling.

Minimal reconstruction error

This approach is due to Pearson [149]. Starting from the linear model of PCA (Eq. (2.4)), the coding and decoding functions (resp., Eq. (2.2) and Eq. (2.3)) can be rewritten as

$$\text{cod} : \mathbb{R}^D \to \mathbb{R}^P, \quad \mathbf{y} \mapsto \mathbf{x} = \text{cod}(\mathbf{y}) = \mathbf{W}^+\mathbf{y} \ , \tag{2.10}$$

$$\text{dec} : \mathbb{R}^P \to \mathbb{R}^D, \quad \mathbf{x} \mapsto \mathbf{y} = \text{dec}(\mathbf{x}) = \mathbf{W}\,\mathbf{x} \ , \tag{2.11}$$

where $\mathbf{W}^+ = (\mathbf{W}^T\mathbf{W})^{-1}\mathbf{W}^T = \mathbf{W}^T$ is the (left) pseudo-inverse of $\mathbf{W}$. Then the reconstruction mean square error (Eq. (2.1)) becomes:

$$E_{\text{codec}} = E_{\mathbf{y}}\{\|\mathbf{y} - \mathbf{W}\mathbf{W}^T\mathbf{y}\|_2^2\} \ , \tag{2.12}$$

As already remarked above, the equality $\mathbf{W}^T\mathbf{W} = \mathbf{I}_P$ holds, but $\mathbf{W}\mathbf{W}^T = \mathbf{I}_D$ is not necessarily true. Therefore, no simplification may occur in the reconstruction error. However, in a perfect world, the observed vector $\mathbf{y}$ has been generated precisely according to the PCA model (Eq. (2.4)). In this case only, $\mathbf{y}$ can be perfectly retrieved. Indeed, if $\mathbf{y} = \mathbf{W}\mathbf{x}$, then

$$\mathbf{W}\mathbf{W}^T\mathbf{y} = \mathbf{W}\mathbf{W}^T\mathbf{W}\mathbf{x} = \mathbf{W}\mathbf{I}_P\mathbf{x} = \mathbf{y} \ ,$$

and the reconstruction error is zero. Unfortunately, in almost all real situations, the observed variables in $\mathbf{y}$ are polluted by some noise, or do not fully respect the linear PCA model, yielding a nonzero reconstruction error. As a direct consequence, $\mathbf{W}$ cannot be identified perfectly, and only an approximation can be computed.

The best approximation is determined by developing and minimizing the reconstruction error. According to the definition of the Euclidean norm (see Subsection 4.2.1), E_{codec} successively becomes

$$\begin{aligned} E_{\text{codec}} &= E_{\mathbf{y}}\{\|\mathbf{y} - \mathbf{W}\mathbf{W}^T\mathbf{y}\|_2^2\} \\ &= E_{\mathbf{y}}\{(\mathbf{y} - \mathbf{W}\mathbf{W}^T\mathbf{y})^T(\mathbf{y} - \mathbf{W}\mathbf{W}^T\mathbf{y})\} \\ &= E_{\mathbf{y}}\{\mathbf{y}^T\mathbf{y} - 2\mathbf{y}^T\mathbf{W}\mathbf{W}^T\mathbf{y} + \mathbf{y}^T\mathbf{W}\mathbf{W}^T\mathbf{W}\mathbf{W}^T\mathbf{y}\} \\ &= E_{\mathbf{y}}\{\mathbf{y}^T\mathbf{y} - 2\mathbf{y}^T\mathbf{W}\mathbf{W}^T\mathbf{y} + \mathbf{y}^T\mathbf{W}\mathbf{W}^T\mathbf{y}\} \\ &= E_{\mathbf{y}}\{\mathbf{y}^T\mathbf{y} - \mathbf{y}^T\mathbf{W}\mathbf{W}^T\mathbf{y}\} \\ &= E_{\mathbf{y}}\{\mathbf{y}^T\mathbf{y}\} - E_{\mathbf{y}}\{\mathbf{y}^T\mathbf{W}\mathbf{W}^T\mathbf{y}\} \ , \end{aligned} \tag{2.13}$$

where the first term is constant. Hence, minimizing E_{codec} turns out to maximize the term $E_{\mathbf{y}}\{\mathbf{y}^T\mathbf{W}\mathbf{W}^T\mathbf{y}\}$. As only a few observations $\mathbf{y}(n)$ are available, the latter expression is approximated by the sample mean:

$$E_{\mathbf{y}}\{\mathbf{y}^T\mathbf{W}\mathbf{W}^T\mathbf{y}\} \approx \frac{1}{N}\sum_{n=1}^{N}(\mathbf{y}(n))^T\mathbf{W}\mathbf{W}^T(\mathbf{y}(n)) \tag{2.14}$$

$$\approx \frac{1}{N}\text{tr}(\mathbf{Y}^T\mathbf{W}\mathbf{W}^T\mathbf{Y}) \ , \tag{2.15}$$

where $\text{tr}(\mathbf{M})$ denotes the trace of some matrix $\mathbf{M}$. To maximize this last expression, $\mathbf{Y}$ has to be factored by singular value decomposition (SVD; see Appendix A.1):

$$\mathbf{Y} = \mathbf{V}\mathbf{\Sigma}\mathbf{U}^T \ , \tag{2.16}$$

where $\mathbf{V}$, $\mathbf{U}$ are unitary matrices and where $\mathbf{\Sigma}$ is a matrix with the same size as $\mathbf{Y}$ but with at most D nonzero entries σ_d, called singular values and

located on the first diagonal of $\mathbf{\Sigma}$. The D singular values are usually sorted in descending order. Substituting in the approximation of the expectation leads to

$$E_{\mathbf{y}}\{\mathbf{Y}^T\mathbf{W}\mathbf{W}^T\mathbf{Y}\} \approx \frac{1}{N}\mathrm{tr}(\mathbf{U}\mathbf{\Sigma}^T\mathbf{V}^T\mathbf{W}\mathbf{W}^T\mathbf{V}\mathbf{\Sigma}\mathbf{U}) \ . \tag{2.17}$$

Since the columns of $\mathbf{V}$ and $\mathbf{U}$ are orthonormal vectors by construction, it is easy to see that

$$\arg\max_{\mathbf{W}} \mathrm{tr}(\mathbf{U}\mathbf{\Sigma}^T\mathbf{V}^T\mathbf{W}\mathbf{W}^T\mathbf{V}\mathbf{\Sigma}\mathbf{U}) = \mathbf{V}\mathbf{I}_{D\times P} \ , \tag{2.18}$$

for a given P ($\mathbf{I}_{D\times P}$ is a matrix made of the first P columns of the identity matrix $\mathbf{I}_D$). Indeed, the above expression reaches its maximum when the P columns of $\mathbf{W}$ are colinear with the columns of $\mathbf{V}$ that are associated with the P largest singular values in $\mathbf{\Sigma}$. Additionally, it can be trivially proved that $E_{\mathrm{codec}} = 0$ for $\mathbf{W} = \mathbf{V}$. In the same way, the contribution of a principal component $\mathbf{v}_d$ to the E_{codec} equals σ_d^2, i.e., the squared singular value associated with $\mathbf{v}_d$.

Finally, P-dimensional latent variables are approximated by computing the product

$$\hat{\mathbf{x}} = \mathbf{I}_{P\times D}\mathbf{V}^T\mathbf{y} \ . \tag{2.19}$$

Maximal preserved variance and decorrelation

This approach is due to Hotelling [92]. From a statistical point of view, it can be assumed that the latent variables in $\mathbf{x}$ are uncorrelated (no linear dependencies bind them). In practice, this means that the covariance matrix of $\mathbf{x}$, defined as

$$\mathbf{C}_{\mathbf{xx}} = E\{\mathbf{x}\mathbf{x}^T\} \ , \tag{2.20}$$

provided $\mathbf{x}$ is centered, is diagonal. However, after the axis change induced by $\mathbf{W}$, it is very likely that the observed variables in $\mathbf{y}$ are correlated, i.e., $\mathbf{C}_{\mathbf{yy}}$ is no longer diagonal. The goal of PCA is then to get back the P uncorrelated latent variables in $\mathbf{x}$. Assuming that the PCA model holds and the covariance of $\mathbf{y}$ is known, we find that

$$\begin{aligned}
\mathbf{C}_{\mathbf{yy}} &= E\{\mathbf{y}\mathbf{y}^T\} && (2.21)\\
&= E\{\mathbf{W}\mathbf{x}\mathbf{x}^T\mathbf{W}^T\} && (2.22)\\
&= \mathbf{W}E\{\mathbf{x}\mathbf{x}^T\}\mathbf{W}^T && (2.23)\\
&= \mathbf{W}\mathbf{C}_{\mathbf{xx}}\mathbf{W}^T \ . && (2.24)
\end{aligned}$$

Since $\mathbf{W}^T\mathbf{W} = \mathbf{I}$, left and right multiplications by, respectively, $\mathbf{W}^T$ and $\mathbf{W}$ lead to

$$\mathbf{C}_{\mathbf{xx}} = \mathbf{W}^T\mathbf{C}_{\mathbf{yy}}\mathbf{W} \ . \tag{2.25}$$

Next, the covariance matrix $\mathbf{C}_{\mathbf{yy}}$ can be factored by eigenvalue decomposition (EVD; see Appendix A.2):

$$\mathbf{C_{yy}} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T \ , \tag{2.26}$$

where $\mathbf{V}$ is a matrix of normed eigenvectors $\mathbf{v}_d$ and $\mathbf{\Lambda}$ a diagonal matrix containing their associated eigenvalues λ_d, in descending order. Because the covariance matrix is symmetric and semipositive definite, the eigenvectors are orthogonal and the eigenvalues are nonnegative real numbers. Substituting in Eq. (2.25) finally gives

$$\mathbf{C_{xx}} = \mathbf{W}^T\mathbf{V}\mathbf{\Lambda}\mathbf{V}^T\mathbf{W} \ . \tag{2.27}$$

This equality holds only when the P columns of $\mathbf{W}$ are taken colinear with P columns of $\mathbf{V}$, among D ones. If the PCA model is fully respected, then only the first P eigenvalues in $\mathbf{\Lambda}$ are strictly larger than zero; the other ones are zero. The eigenvectors associated with these P nonzero eigenvalues must be kept:

$$\mathbf{W} = \mathbf{V}\mathbf{I}_{D\times P} \ , \tag{2.28}$$

yielding

$$\mathbf{C_{xx}} = \mathbf{I}_{P\times D}\mathbf{\Lambda}\mathbf{I}_{D\times P} \ . \tag{2.29}$$

This shows that the eigenvalues in $\mathbf{\Lambda}$ correspond to the variances of the latent variables (the diagonal entries of $\mathbf{C_{xx}}$).

In real situations, some noise may corrupt the observed variables in $\mathbf{y}$. As a consequence, all eigenvalues of $\mathbf{C_{yy}}$ are larger than zero, and the choice of P columns in $\mathbf{V}$ becomes more difficult. Assuming that the latent variables have larger variances than the noise, it suffices to choose the eigenvectors associated with the largest eigenvalues. Hence, the same solution as in Eq. (2.28) remains valid, and the latent variables are estimated exactly as in Eq. (2.19).

If the global variance of $\mathbf{y}$ is defined as

$$\sigma^2_{\mathbf{y}} = \mathrm{tr}(\mathbf{C_{yy}}) = \sum_{d=1}^{D} c_{d,d} = \sum_{d=1}^{D} \lambda_d \ , \tag{2.30}$$

then the proposed solution is guaranteed to preserve a maximal fraction of the global variance. From a geometrical point of view, the columns of $\mathbf{V}$ indicates the directions in $\mathbb{R}^D$ that span the subspace of the latent variables. If these columns are called *components*, the choice of the columns associated with the largest variances justifies the name "principal component analysis".

To conclude, it must be emphasized that in real situations the true covariance of $\mathbf{y}$ is not known but can be approximated by the sample covariance:

$$\hat{\mathbf{C}}_{\mathbf{yy}} = \frac{1}{N}\mathbf{Y}\mathbf{Y}^T \ . \tag{2.31}$$

2.4.3 Functionalities of PCA

The success of PCA finds an explanation not only in its simplicity but also in the broad applicability of the method. Indeed, PCA easily makes available the three main functionalities that a user may expect in a method that analyzes high-dimensional data.

Intrinsic dimension estimation

If the model of PCA (Eq. (2.4)) is fully respected, then only the P largest eigenvalues of $\mathbf{C_{yy}}$ will depart from zero. Hence, the rank of the covariance matrix (the number of nonzero eigenvalues) indicates trivially the number of latent variables. However, when having only a finite sample, the covariance can only be approximated. Moreover, the data probably do not entirely respect the PCA model (presence of noise, etc.). For all those reasons, all D eigenvalues are often different from zero, making the estimation of the intrinsic dimension more difficult.

A first way to determine it consists of looking at the eigenvalues as such. Normally, if the model holds reasonably well, large (significant) eigenvalues correspond to the variances of the latent variables, while smaller (negligible) ones are due to noise and other imperfections. Ideally, a rather visible gap should separate the two kinds of eigenvalues. The gap can be visualized by plotting the eigenvalues in descending order: a sudden fall should appear right after the Pth eigenvalue. If the gap is not visible, plotting minus the logarithm of the normalized eigenvalues may help:

$$0 \leq -\log \frac{\lambda_d}{\lambda_1} \quad . \tag{2.32}$$

In this plot, the intrinsic dimension is indicated by a sudden ascent.

Unfortunately, when the data dimensionality D is high, there may also be numerous latent variables showing a wide spectrum of variances. In the extreme, the variances of latent variables can no longer be distinguished from the variances related to noise. In this case, the intrinsic dimension P is chosen in order to preserve an arbitrarily chosen fraction of the global variance (Eq. (2.30)). Then the dimension P is determined so as to preserve at least this fraction of variance. For example, if it is assumed that the latent variables bear 95% of the global variance, then P is the smaller integer such that the inequality

$$0.95 \leq \frac{\sum_{d=1}^{P} \lambda_d}{\sum_{d=1}^{D} \lambda_d} = \frac{\mathrm{tr}(\mathbf{I}_{P\times D}\mathbf{\Lambda}\mathbf{I}_{D\times P})}{\sigma_{\mathbf{y}}^2} \tag{2.33}$$

holds. Sometimes the threshold is set on individual variances instead of cumulated ones. For example, all components having a variance lower than 1% of $\sigma_{\mathbf{y}}^2$ are discarded. The best way to set the threshold consists of finding a threshold that separates the significant variances from the negligible ones. This turns out to be equivalent to the visual methods proposed in the previous paragraph.

More complex methods exist to set the frontier between the latent and noise subspaces, such as Akaike's information criterion [4, 126] (AIC), the Bayesian information criterion [168] (BIC), and the minimum description length [153, 126] (MDL). These methods determine the value of P on the basis of information-theoretic considerations. Their particularization to PCA, with a noisy model, is well described in [34].

Projection for dimensionality reduction

After the estimation of the intrinsic dimensionality, PCA reduces the dimension by projecting the observed variables onto the estimated latent subspace in a linear way. Equation (2.19) shows how to obtain P-dimensional coordinates from D-dimensional ones. In that equation, the dimensionality reduction is achieved by the factor $\mathbf{I}_{P\times D}$, which discards the eigenvectors of $\mathbf{V}$ associated with the $D-P$ smallest eigenvalues. On the other hand, the factor $\mathbf{V}^T$ ensures that the dimensionality reduction minimizes the loss of information. Intuitively, this is done by canceling the linear dependencies between the observed variables.

Projection for latent variable separation

Beyond reduction of data dimensionality, PCA can also separate latent variables under certain conditions. In Eq. (2.19), the separation is achieved by the factor $\mathbf{V}^T$. As clearly stated in the PCA model, the observed variables can only be a rotation of the latent ones, which have a Gaussian distribution. These are, of course, very restrictive conditions that can be somewhat relaxed.

For example, in Eq. (2.4), the columns of $\mathbf{W}$ can be solely orthogonal instead of orthonormal. In this case, the latent variables will be retrieved up to a permutation and a scaling factor.

Additionally, if all latent variables have a Gaussian distribution but $\mathbf{W}$ is any matrix, then PCA can still retrieve a set of variables along orthogonal directions. The explanation is that a set of any linear combinations of Gaussian distributions is always equivalent to a set of orthogonal combinations of Gaussian distributions (see Appendix B).

From a statistical point of view, PCA decorrelates the observed variables $\mathbf{y}$ by diagonalizing the (sample) covariance matrix. Therefore, without consideration of the true latent variables, PCA finds a reduced set of uncorrelated variables from the observed ones. Actually, PCA cancels the second-order crosscumulants, i.e., the off-diagonal entries of the covariance matrix.

Knowing that higher-order cumulants, like the skewness (third order) and the kurtosis (fourth order), are null for Gaussian variables, it is not difficult to see that decorrelating the observed variables suffices to obtain fully independent latent variables. If latent variables are no longer Gaussian, then higher-order cumulants must be taken into account. This is what is done in independent component analysis (ICA, [95, 34]), for which more complex algorithms than PCA are able to cancel higher-order cross-cumulants. This leads to latent variables that are statistically independent.

2.4.4 Algorithms

As already unveiled in Subsection 2.4.2, PCA is often implemented by general-purpose algebraic procedures. The developments of PCA from the two

different criteria described in that subsection show that PCA can work in two different ways:

- by SVD (singular value decomposition; see Appendix A.1) of the matrix $\mathbf{Y}$, containing the available sample.
- by EVD (eigenvalue decomposition; see Appendix A.2) of the sample covariance $\hat{\mathbf{C}}_{\mathbf{yy}}$.

Obviously, both techniques are equivalent, at least if the singular values and the eigenvalues are sorted in the same way:

$$\hat{\mathbf{C}}_{\mathbf{yy}} = \frac{1}{N}\mathbf{Y}\mathbf{Y}^T \tag{2.34}$$

$$= \frac{1}{N}(\mathbf{V}\mathbf{\Sigma}\mathbf{U}^T)(\mathbf{U}\mathbf{\Sigma}^T\mathbf{V}^T) \tag{2.35}$$

$$= \mathbf{V}(\frac{1}{N}\mathbf{\Sigma}\mathbf{\Sigma}^T)\mathbf{V}^T \tag{2.36}$$

$$= \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T \quad . \tag{2.37}$$

By the way, the last equality shows the relationship between the eigenvalues and the singular values: $\lambda_d = \sigma_d^2/N$. From a numerical point of view, the SVD of the sample is more robust because it works on the whole data set, whereas EVD works only on the summarized information contained in the covariance matrix. As a counterpart, from the computational point of view, SVD is more expensive and may be very slow for samples containing many observations.

The use of algebraic procedures makes PCA a batch algorithm: all observations have to be known before PCA starts. However, online or adaptive versions of PCA exist; several are described in [95, 34]. These implementations do not offer the same strong guarantees as the algebraic versions, but may be very useful in real-time application, where computation time and memory space are limited.

When estimating the latent variables, it must be pointed out that Eq. (2.19) is not very efficient. Instead, it is much better to directly remove the unnecessary columns in $\mathbf{V}$ and to multiply by $\mathbf{y}$ afterwards, without the factor $\mathbf{I}_{P\times D}$. And if PCA works by SVD of the sample, it is noteworthy that

$$\hat{\mathbf{X}} = \mathbf{I}_{P\times D}\mathbf{V}^T\mathbf{Y} \tag{2.38}$$

$$= \mathbf{I}_{P\times D}\mathbf{V}^T\mathbf{V}\mathbf{\Sigma}\mathbf{U}^T \tag{2.39}$$

$$= \mathbf{I}_{P\times D}\mathbf{\Sigma}\mathbf{U}^T \quad . \tag{2.40}$$

As $\mathbf{\Sigma}$ is diagonal, the cheapest way to compute $\hat{\mathbf{X}}$ consists of copying the first P columns of $\mathbf{U}$, multiplying them by the corresponding diagonal entry of $\mathbf{\Sigma}$, and, finally, transposing the result.

As already mentioned, PCA assumes that the observed variables are centered. Sometimes data are also standardized, meaning that each variable y_d is scaled in order to have unit variance. This is usually done when the observed variables come from various origins and have very different variances.

The standardization allows PCA not to consider observed variables with small variances as being noise and not to discard them in the dimensionality reduction. On the other hand, the standardization can sometimes amplify variables that are really negligible. User knowledge is very useful to decide if a scaling is necessary.

2.4.5 Examples and limitations of PCA

In order to illustrate the capabilities as well the limitations of PCA, toy examples may be artificially generated. For visualization's sake, only two latent variables are created; they are embedded in a three-dimensional space, i.e., three variables are observed. Three simple cases are studied here.

Gaussian variables and linear embedding

In this first case, the two latent variables, shown in the first plot of Fig. 2.5, have Gaussian distributions, with variances 1 and 4. The observed variables, displayed in the second plot of Fig. 2.5, are obtained by multiplying the latent ones by

$$\mathbf{W} = \begin{bmatrix} 0.2 & 0.8 \\ 0.4 & 0.5 \\ 0.7 & 0.3 \end{bmatrix} . \tag{2.41}$$

As the mixing process (i.e., the matrix $\mathbf{W}$) is linear, PCA can perfectly reduce the dimensionality. The eigenvalues of the sample covariance matrix are 0.89, 0.11, and 0.00. The number of latent variables is then clearly two, and PCA reduces the dimensionality without any loss: the observed variables could be perfectly reconstructed from the estimated latent variables shown in Fig. 2.6. However, the columns of the mixing matrix $\mathbf{W}$ are *neither* orthogonal *nor* normed. Consequently, PCA cannot retrieve exactly the true latent variables. Yet as the latter have Gaussian distributions, PCA finds a still satisfying result: in Fig. 2.6, the estimated latent variables have Gaussian distributions but are scaled and rotated. This is visible by looking at the schematic representations of the distributions, displayed as solid and dashed ellipses. The ellipses are almost identical, but the axes indicating the directions of the true and estimated latent variables are different.

Nonlinear embedding

In this second case, the two latent variables are the same as in the previous case, but this time the mixing process is nonlinear:

$$\mathbf{y} = \begin{bmatrix} 4\cos(\frac{1}{4}x_1) \\ 4\sin(\frac{1}{4}x_1) \\ x_1 + x_2 \end{bmatrix} . \tag{2.42}$$

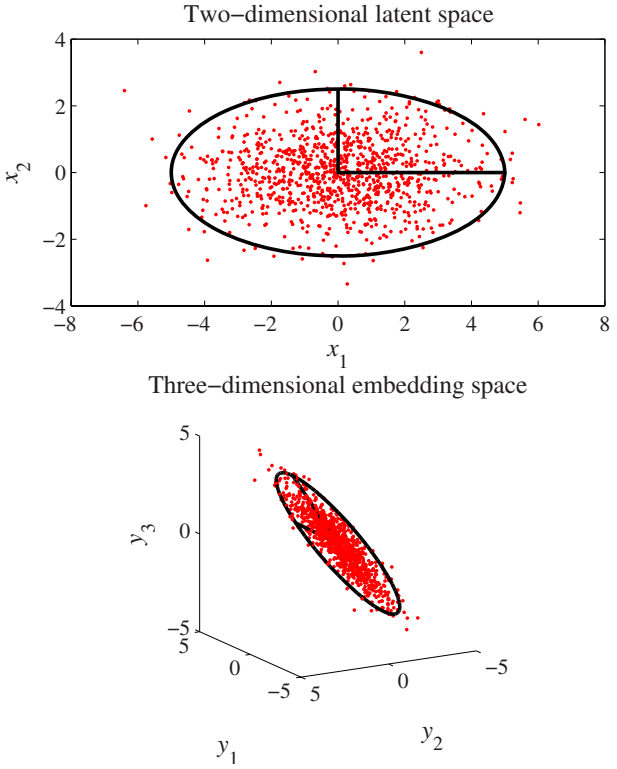

Fig. 2.5. Two Gaussian latent variables (1000 observations, displayed in the first plot) are embedded in a three-dimensional space by a linear mixing process (second plot). The ellipse schematically represents the joint distribution in both spaces.

The observed variables are shown in Fig. 2.7. Faced with nonlinear dependencies between observed variables, PCA does not detect that only two latent variables have generated them: the normalized eigenvalues of the sample covariance matrix are 0.90, 0.05 and 0.05 again. The projection onto the first two principal components is given in Fig. 2.8. PCA is unable to completely reconstruct the curved object displayed in Fig. 2.7 with these two principal components (the reconstruction would be strictly planar!).

Unfortunately, the result of the dimensionality reduction is not the only disappointing aspect. Indeed, the estimated latent variables are completely different from the true ones. The schematic representation of the distribution is totally deformed.

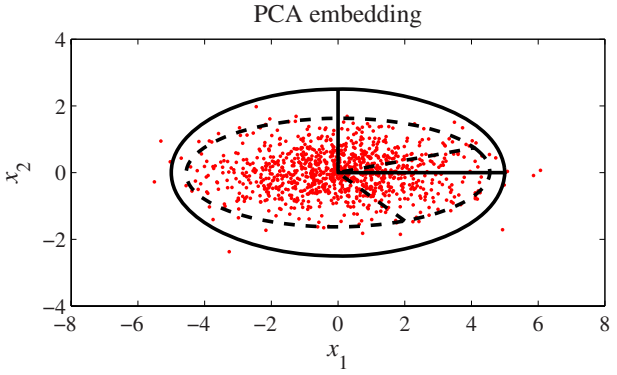

Fig. 2.6. Projection of the three-dimensional observations (second plot of Fig. 2.5) onto the two first principal components found by PCA. The solid line shows a schematic representation of the true latent distribution, whereas the dashed one corresponds to the estimated latent variables.

Non-Gaussian distributions

In this third and last case, the two latent variables are no longer Gaussian. As shown in the first plot of Fig. 2.9, their distribution is uniform. On the other hand, the mixing process is exactly the same as in the first case. Therefore, the eigenvalues of the sample covariance matrix are also identical (actually, very close, depending on the sample). The dimensionality reduction is performed without loss. However, the latent variable separation becomes really problematic. As in the first case, the estimated latent variables shown in Fig. 2.10 are rotated and scaled because the columns of $\mathbf{W}$ are not orthonormal. But because the latent variables have uniform distributions and not Gaussian ones, the scale factors and rotations make the estimated latent variables no longer uniformly distributed!

Concluding remarks about PCA

In the ideal situation, when its model is fully respected, PCA appears as a very polyvalent method to analyze data. It determines data dimensionality, builds an embedding accordingly, and retrieves the latent variables. In practice, however, the PCA model relies on assumptions that are much too restrictive, especially when it comes to latent variable separation. When only dimensionality reduction is sought, the sole remaining but still annoying assumption imposes that the dependencies between the observed variables are (not far from being) linear.

The three toy examples detailed earlier clearly demonstrate that PCA is not powerful enough to deal with complex data sets. This suggests designing other methods, maybe at the expense of PCA simplicity and polyvalence.

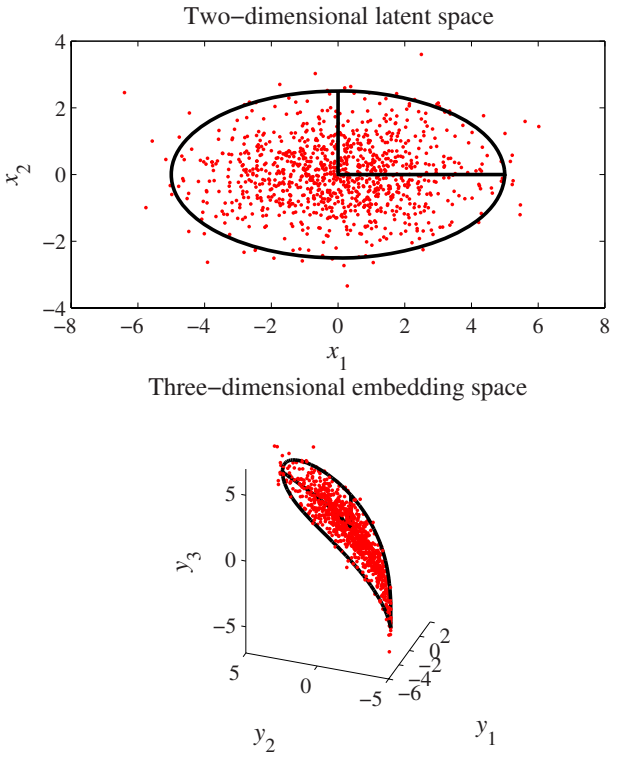

Fig. 2.7. Two Gaussian latent variables (1000 observations, displayed in the first plot) are embedded in a three-dimensional space by a nonlinear mixing process (second plot). The ellipse schematically represents the joint distribution in both spaces.

Two directions may be explored: true latent variable separation or merely dimensionality reduction. In the former case, requirements are high since an embedding has to be found that not only reduces the data dimensionality but also recovers the latent variables. If a linear model is kept, PCA can be refined in order to process non-Gaussian distributions; this leads to ICA for example. If the model cannot be assumed to be linear then latent variable separation becomes theoretically very difficult, at least when relying on statistical concepts like independence. In cases where a nonlinear model must be considered, the data analysis most often cannot go further than dimensionality reduction. Extending PCA to nonlinear models still remains an appealing challenge. Chapters 4 and 5 deal with pure dimensionality reduction. Without the necessicity of retrieving *exactly* the latent variables, more freedom is left and numerous models become possible. Nevertheless, most of them rely

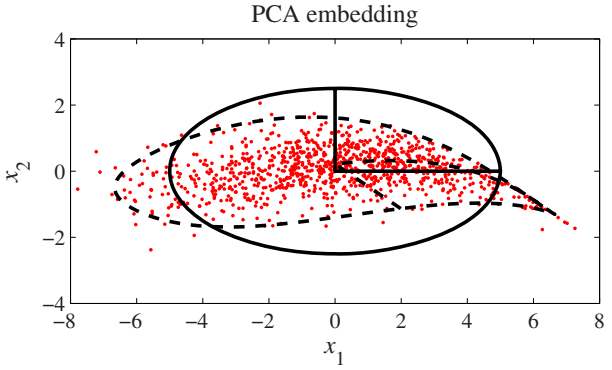

Fig. 2.8. Projection of the three-dimensional observations (second plot of Fig. 2.7) onto the first two principal components found by PCA. The solid line shows a schematic representation of the true latent distribution, whereas the dashed one corresponds to the estimated latent variables.

on geometrical considerations, ranging from simple distance measurements (Chapters 4) to more complex ideas from topology (Chapters 5).

2.5 Toward a categorization of DR methods

Before a detailed description of several DR methods, this section proposes a (nonexhaustive) list of several qualifications that will help to characterize and categorize the various methods. These qualifications mainly regard the purpose of the method, its underlying model, the mathematical criterion to be optimized and the algorithm that performs the optimization. Twelve possible qualifications are for instance

- hard vs. soft dimensionality reduction,
- traditional vs. generative model,
- linear vs. nonlinear model,
- continuous vs. discrete model,
- implicit vs. explicit mapping,
- integrated vs. external estimation of the dimensionality,
- layered vs. standalone embeddings,
- single vs. multiple coordinate systems,
- optional vs. mandatory vector quantization,
- batch vs. online algorithm,
- exact vs. approximate optimization,
- the type of criterion to be optimized.

The remainder of this section describes these features in detail.

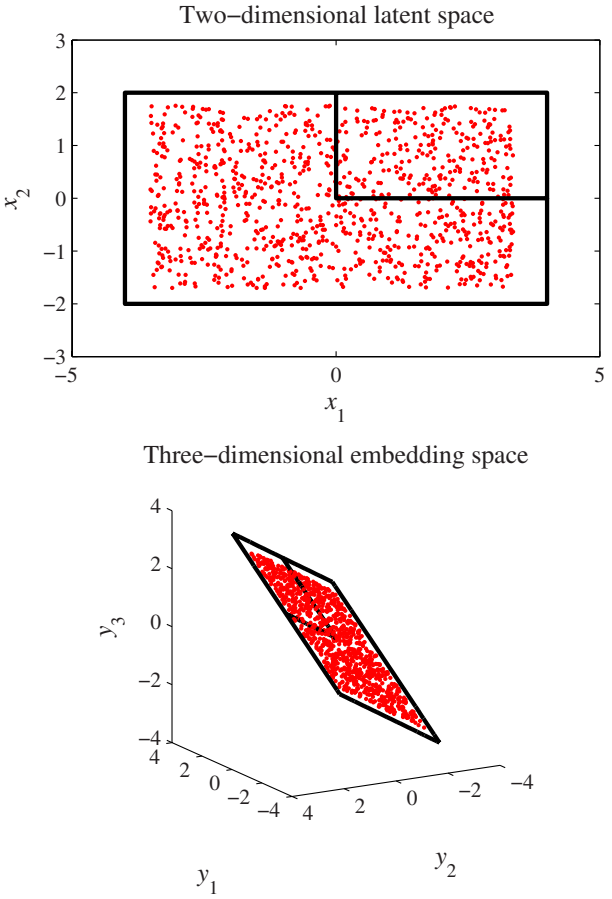

Fig. 2.9. Two uniform latent variables (1000 observations, displayed in the first plot) are embedded in a three-dimensional space by a linear mixing process (second plot). The rectangle schematically represents the joint distribution in both spaces.

2.5.1 Hard vs. soft dimensionality reduction

The problem of dimensionality reduction arises in many different applications. Depending on them, a first distinction between methods regards the ratio between the initial dimension of data and the desired dimension after re-embedding [33].

Hard dimensionality reduction is suited for problems in which the data have a dimension ranging from hundreds to maybe hundreds of thousands of variables. In such a case, a drastic dimensionality reduction is usually sought, possibly one of several orders of magnitude. The variables are often massively repeated measures of a certain phenomenon of interest in different points of

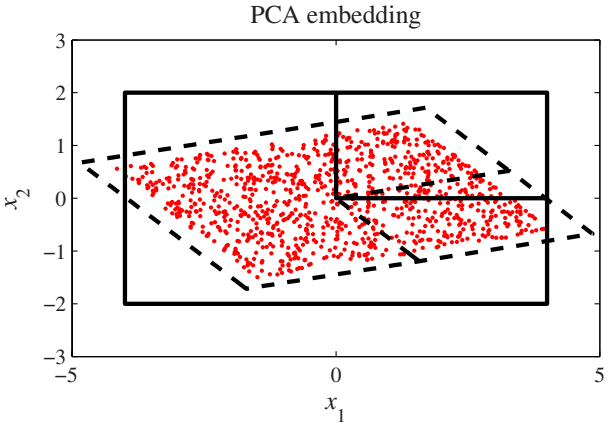

Fig. 2.10. Projection of the three-dimensional observations (second plot of Fig. 2.9) onto the first two principal components found by PCA. The solid line shows a schematic representation of the true latent distribution, whereas the dashed one corresponds to the estimated latent variables.

space (or at different instants over time). To this class belong classification and pattern recognition problems involving images or speech. Most often, the already difficult situation resulting from the huge number of variables is complicated by a low number of available samples. Methods with a simple model and few parameters like PCA are very effective for hard dimensionality reduction.

Soft dimensionality reduction is suited for problems in which the data are not too high-dimensional (less than a few tens of variables). Then no drastic dimensionality reduction is needed. Usually, the components are observed or measured values of different variables, which have a straightforward interpretation. Many statistical studies and opinion polls in domains like social sciences and psychology fall in this category. By comparison with hard problems described above, these applications usually deal with sufficiently large sample sizes. Typical methods include all the usual tools for multivariate analysis.

Finally, visualization problems lie somewhere in-between the two previous classes. The initial data dimensionality may equal any value. The sole constraint is to reduce it to one, two, or three dimensions.

2.5.2 Traditional vs. generative model

The *model* associated with a method actually refers to the way the method connects the latent variables with the observed ones. Almost each method makes different assumptions about this connection. It is noteworthy that this connection can go in both directions: from the latent to the observed variables

or from the observed to the latent variables. Most methods use the second solution, which is the simplest and most used one, since the method basically goes in the same direction: the goal is to obtain an estimation of the latent variables starting from the observed ones. More principled methods prefer the first solution: they model the observed variables as a function of the unknown latent variables. This more complex solution better corresponds to the real way data are generated but often implies that those methods must go back and forth between the latent and observed variables in order to determine the model parameters. Such *generative models* are seldom encountered in the field of dimensionality reduction.

2.5.3 Linear vs. nonlinear model

The distinction between methods based on a linear or a nonlinear model is probably the straightest way to classify them. This explains why methods using a linear (resp., nonlinear) model are simply called linear (resp., nonlinear) methods. Both linear and nonlinear dimensionality reduction are denoted by the acronyms LDR and NLDR, respectively, in the remainder of this book.

Nonlinear methods are often more powerful than linear ones, because the connection between the latent variables and the observed ones may be much richer than a simple matrix multiplication. On the other hand, their model often comprises many parameters, whose identification requires large amounts of data.

For example, if PCA projects D-dimensional vectors onto a P-dimensional plane, the model comprises $\mathcal{O}(PD)$ parameters, and already $P + 1$ points suffice to entirely determine them. For a nonlinear method like an SOM (see Subsection 5.2.1), the number of parameters grows much more quickly: $3^P D$ parameters hardly suffice to describe a basic nonlinear model.

2.5.4 Continuous vs. discrete model

A third distinction about the data model regards its continuity. For example, the model of PCA given in Section 2.4, is continuous: it is a linear transform of the variables. On the other hand, the model of an SOM is discrete: it consists of a finite set of interconnected points.

The continuity is a very desirable property when the dimensionality reduction must be generalized to other points than those used to determine the model parameters. When the model is continuous, the dimensionality reduction is often achieved by using a parameterized function or mapping between the initial and final spaces. In this case, applying the mapping to new points yields their coordinates in the embedding. With only a discrete model, new points cannot be so easily re-embedded: an interpolation procedure is indeed necessary to embed in-between points.

2.5.5 Implicit vs. explicit mapping

A fourth distinction about the model, which is closely related to the previous one, regards the way a method maps the high- and low-dimensional spaces. Mainly two classes of mappings exist: explicit and implicit.

An explicit mapping consists of directly associating a low-dimensional representation with each data point. Hence, using an explicit mapping clearly means that the data model is discrete and that the generalization to new points may be difficult. Sammon's nonlinear mapping (see Subsection 4.2.3) is a typical example of explicit mapping. Typically, the parameters of such a mapping are coordinates, and their number is proportional to the number of observations in the data set.

On the other hand, an implicit mapping is defined as a parameterized function. For example, the paramaters in the model of PCA define a hyperplane. Clearly, there is no direct connection between those parameters and the coordinates of the observations stored in the data set. Implicit mappings often originate from continuous models, and generalization to new points is usually straightforward.

A third intermediate class of mappings also exists. In this class may be gathered all models that define a mapping by associating a low-dimensional representation not to each data point, but to a subset of data points. In this case, the number of low-dimensional representations does not depend strictly on the number of data points, and the low-dimensional coordinates may be considered as generic parameters, although they have a straightforward geometric meaning. All DR methods, like SOMs, that involve some form of vector quantization (see Subection 2.5.9 ahead) belong to this class.

2.5.6 Integrated vs. external estimation of the dimensionality

When considering dimensionality reduction, a key point to discuss is the presence of an estimator of the intrinsic dimensionality. The case of PCA appears as an exception, as most other methods are deprived of an integrated estimator. Actually, they take the intrinsic dimensionality as an external hyperparameter to be given by the user. In that sense, this hyper-parameter is preferably called the *embedding dimension(ality)* rather than the intrinsic dimensionality of data. This is justified by the fact that the user may wish to visualize the data on a plane and thus force a two-dimensional embedding even if the intrinsic dimensionality is actually higher. On the contrary, methods that are not powerful enough to embed highly curved P-manifolds may need more than P dimensions to work properly.

Anyway, if a dimensionality estimator is not integrated in the dimensionality reduction itself, this functionality must be performed by an external procedure. Some typical techniques to estimate the intrinsic dimensionality of manifolds are described in Chapter 3.

2.5.7 Layered vs. standalone embeddings

When performing PCA on data, all embeddings of dimensionality ranging between 1 and D are computed at once. The different embeddings are obtained by removing the coordinates along one or several of the D eigenvectors. Since the eigenvectors are orthogonal by construction, the removal of an eigenvector obviously does not change the coordinates along the kept ones. In other words, PCA proves to be an *incremental* method that produces *layered* embeddings: adding or removing a dimension does not require any change of the coordinates along the other dimensions. For instance, computing a 2D embedding can be done by taking the leading eigenvector, which specifies coordinates along a first dimension, and then the second eigenvector, in decreasing order of eigenvalue magnitude. If a 3D embedding is needed, it suffices to retrieve the 2D one, take the third eigenvector, and append the corresponding coordinates.

All methods that translate the dimensionality reduction into an eigenproblem and assemble eigenvectors to form an embedding share this capability and are called *spectral* methods. To some extent, the embeddings of different dimensionality provided by spectral methods are not independent since coordinates are added or removed but never changed when the target dimensionality increases or decreases.

In contrast, methods relying on other principles (nonspectral methods) do not offer such a comfort. When they compute an embedding of a given dimensionality, only that specific embedding is determined. If the target dimensionality changes, then all coordinates of the new embedding must be computed again. This means that the method builds *standalone* embeddings for each specified dimensionality. Although such methods seem to waste computational power, the independence of the embeddings can also be an advantage: for each dimensionality, the embedding is specifically optimized.

2.5.8 Single vs. multiple coordinate systems

Strictly speaking, dimensionality reduction does not imply establishing a low-dimensional representation in a single system of coordinates. For example, a natural extension of PCA to a nonlinear model consists in dividing a nonlinear manifold into small pieces, like a (curved) jigsaw. If these pieces are small enough, they may be considered as linear and PCA may be applied to each of them. Obviously, each PCA is independent from the others, raising the difficulty of patching together the projections of each piece of the manifold in the low-dimensional space.

One of the rare applications for which multiple systems of coordinates raise no difficulties is data compression. But in this domain more specific methods exist that reach a better compression rate by considering the data as a binary flow instead of coordinates in a Cartesian space.

Most other applications of dimensionality reduction, like visualization, usually need a single system of coordinates. Indeed, if the data lie on a smooth

manifold, what would be the justification to cut it off into small pieces? One of the goals of dimensionality reduction is precisely to discover how the different parts of a manifold connect to each other. When using disconnected pieces, part of that information is lost, and furthermore it becomes impossible to visualize or to process the data as a whole.

The most widely known method using several coordinates systems is certainly the local PCA introduced by Kambathla and Leen [101]. They perform a simple vector quantization (see ahead or Appendix D for more details) on the data in order to obtain the tesselation of the manifold. More recent papers follow a similar approach but also propose very promising techniques to patch together the manifold pieces in the low-dimensional embedding space. For example, a nonparametric technique is given in [158, 166], whereas probabilistic ones are studied in [159, 189, 178, 29].

2.5.9 Optional vs. mandatory vector quantization

When the amount of available data is very large, the user may decide to work with a smaller set of representative observations. This operation can be done automatically by applying a method of vector quantization to the data set (see App. D for more details). Briefly, vector quantization replaces the original observations in the data set with a smaller set of so-called prototypes or centroids. The goal of vector quantization consists of reproducing as well as possible the shape of the initial (discrete) data distribution with the prototypes.

Unfortunately, the ideal case where data are overabondant seldom happens in real applications. Therefore, the user often skips the vector quantization and keeps the initial data set. However, some methods are designed in such a way that vector quantization is mandatory. For example, SOMs belong to this class (see Chapter 5).

2.5.10 Batch vs. online algorithm

Depending on the application, data observations may arrive consecutively or, alternatively, the whole data set may be available at once. In the first case, an online algorithm is welcome; in the second case, an offline algorithm suffices. More precisely, offline or "batch" algorithms cannot work until the whole set of observations is known. On the contrary, online algorithms typically work with no more than a single observation at a time.

For most methods, the choice of the model largely orients the implementation toward one or the other type of algorithm. Generally, the simpler the model is, the more freedom is left into the implementation. For example, as already seen in Section 2.4, the simple model of PCA naturally leads to a simple batch procedure, although online variants are possible as well. On the other hand, the more complex model of a SOM favors an online algorithm (a batch version also exists, but it is seldom used).

Actually, the behavior of true online algorithms is rather complex, especially when the data sequence does not fulfill certain conditions (like stationarity or ergodicity). For this reason, batch algorithms are usually preferred. Fortunately, most online algorithms can be made batch ones using the Robbins–Monro procedure [156]. The latter simulates a possibly infinite data sequence by repeating a finite set of observations; by construction, this method ensures some kind of stationarity. Each repetition is called an *epoch*; if the order of the available observations does not matter, then they are usually ramdomly permuted before each epoch, in order to avoid any influence on the algorithm convergence. The Robbins–Monro procedure guarantees the convergence on a solution if the algorithm satisfies certain conditions as epochs go by. These conditions typically regard parameters like step sizes, learning rates, and other time-varying parameters of online algorithms.

In this book, most algorithms that are said to be online are, in fact, implemented with the Robbins–Monro procedure for the various examples and comparisons. More details can be found in Appendix C.2.

2.5.11 Exact vs. approximate optimization

The kind of optimization that an algorithm can achieve is closely related to its offline or online character. Most often, batch algorithms result from some analytical or algebraic developments that give the solution in closed form, like PCA. Given a finite data set, which is known in advance, a batch algorithm like PCA can be proved to compute the optimal solution.

On the opposite, online or adaptive algorithms are often associated with generic optimization procedures like stochastic gradient descent (see App. C.2). Such procedures do not offer strong guarantees about the result: the convergence may fail. Nonetheless, on-line algorithms transformed into batch ones by the Robbins–Monro procedure are guaranteed to converge on a solution, after a certain number of epochs.[3] Unfortunately, this solution may be a local optimum.

Although they are slow and not optimal, generic optimization procedures have a major advantage: they are able to optimize rather complicated objective functions. On the other hand, solutions in closed form are only available for very simple objective functions; the latter ones are typically required to be not only differentiable but also concave (or convex).

2.5.12 The type of criterion to be optimized

Last but not least, the criterion that guides the dimensionality reduction is probably the most important characteristic of a DR method, even before the model specification. Actually, the data model and the algorithm are often fitted in order to satisfy the constraints imposed by the chosen criterion.

[3] An infinite number, according to the theory; see [156].

As already mentioned, the domain of dimensionality reduction is mainly motivated by geometrical considerations. From this point of view, data are interpreted as a cloud of points in a geometrical space; these points are often assumed to lie on a smooth manifold. A way to characterize a manifold independently from any coordinate system consists of discovering and making explicit the relationships between the points of the manifold. Human perception naturally focuses on proximity relations: when looking at a point in the manifold, the eyes immediately distinguish neighboring points from those lying farther away. Therefore, a good embedding of the manifold should reproduce those proximity relationships as well as possible.

Formally, this means that a criterion for a good DR method should efficiently measure the proximities observed in data and quantify their preservation in the embedding. There are two main ways to measure proximities between points, as detailed below.

The straightest one consists of computing pairwise distances between the points. As an advantage, distances are scalar values that can be easily compared to each other. Thus, a possible criterion for dimensionality reduction is distance preservation: the pairwise distances measured between the embedded points should be as close as possible to the ones measured between the initial data points. The next chapter explores this direction and shows how distance preservation can be translated in various objective functions.

On the other hand, a less intuitive but more satisfying way to measure proximities would be qualitative only. In this case, the exact value of the distances does not matter: for example, one just knows that, "From point **a**, point **b** is closer than point **c**." The translation of such qualitative concepts into an objective function proves more difficult than for distances. Chapter 5 presents solutions to this problem.