A survey of dimensionality reduction techniques

C.O.S. Sorzano[1, †], J. Vargas[1], A. Pascual-Montano[1]

[1]*Natl. Centre for Biotechnology (CSIC)*

C/Darwin, 3. Campus Univ. Autónoma, 28049 Cantoblanco, Madrid, Spain

{coss,jvargas,pascual}@cnb.csic.es

[†]*Corresponding author*

Abstract—**Experimental life sciences like biology or chemistry have seen in the recent decades an explosion of the data available from experiments. Laboratory instruments become more and more complex and report hundreds or thousands measurements for a single experiment and therefore the statistical methods face challenging tasks when dealing with such high-dimensional data. However, much of the data is highly redundant and can be efficiently brought down to a much smaller number of variables without a significant loss of information. The mathematical procedures making possible this reduction are called dimensionality reduction techniques; they have widely been developed by fields like Statistics or Machine Learning, and are currently a hot research topic. In this review we categorize the plethora of dimension reduction techniques available and give the mathematical insight behind them.**

Keywords: **Dimensionality Reduction, Data Mining, Machine Learning, Statistics**

1. Introduction

During the last decade life sciences have undergone a tremendous revolution with the accelerated development of high technologies and laboratory instrumentations. A good example is the biomedical domain that has experienced a drastic advance since the advent of complete genome sequences. This post-genomics era has leaded to the development of new high-throughput techniques that are generating enormous amounts of data, which have implied the exponential growth of many biological databases. In many cases, these datasets have much more variables than observations. For example, standard microarray datasets usually are composed by thousands of variables (genes) in dozens of samples. This situation is not exclusive of biomedical research and many other scientific fields have also seen an explosion of the number of variables measured for a single experiment. This is the case of image processing, mass spectrometry, time series analysis, internet search engines, and automatic text analysis among others.

Statistical and machine reasoning methods face a formidable problem when dealing with such high-dimensional data, and normally the number of input variables is reduced before a data mining algorithm can be successfully applied. The dimensionality reduction can be made in two different ways: by only keeping the most relevant variables from the original dataset (this technique is called *feature selection*) or by exploiting the redundancy of the input data and by finding a smaller set of new variables, each being a combination of the input variables, containing basically the same information as the input variables (this technique is called *dimensionality reduction*).

This situation is not new in Statistics. In fact one of the most widely used dimensionality reduction techniques, Principal Component Analysis (PCA), dates back to Karl Pearson in 1901 [Pearson1901]. The key idea is to find a new coordinate system in which the input data can be expressed with many less variables without a significant error. This new basis can be global or local and can fulfill very different properties. The recent explosion of data available together with the evermore powerful computational resources have attracted the attention of many researchers in Statistics, Computer Science and Applied Mathematics who have developed a wide range of computational techniques dealing with the dimensionality reduction problem (for reviews see [Carreira1997, Fodor2002,Mateen2009]).

In this review we provide an up-to-date overview of the mathematical properties and foundations of the different dimensionality reduction techniques. For feature selection, the reader is referred to the reviews of [Dash1997], [Guyon2003] and [Saeys2007].

There are several dimensionality reduction techniques specifically designed for time series. These methods specifically exploit the frequential content of the signal and its usual sparseness in the frequency space. The most popular methods are those based on wavelets [Rioul1991, Graps1995], followed at a large distance by the Empirical Mode Decomposition [Huang1998, Rilling2003] (the reader is referred to the references above for further details). We do not cover these techniques here since they are not usually applied for the general purpose dimensionality reduction of data. From a general point of view, we may say that wavelets project the input time series onto a fixed dictionary (see Section 3). This dictionary has the property of making the projection sparse (only a few coefficients are sufficiently large), and the dimensionality reduction is obtained by setting most coefficients (the small ones) to zero. The empirical mode decomposition, instead, constructs a dictionary specially adapted to each input signal.

To keep the consistency of this review, we do not cover neither those dimensionality reduction techniques that take into account the class of observations, i.e., there are observations from a class A of objects, observations from a class B, … and the dimensionality reduction technique should keep as well as possible the separability of the original classes. Fisher's Linear Discriminant Analysis (LDA) was one of the first techniques to address this issue [Fisher1936, Rao1948]. Many other works followed since then, for the most recent works and for a bibliographical review see [Bian2011, Cai2011, Kim2011, Lin2011, Batmanghelich2012].

In the following we will refer to the observations as input vectors $\mathbf{x}$, whose dimension is M. We will assume that we have N observations and we will refer to the n-th observation as $\mathbf{x}_n$. The whole dataset of observations will be X, while X will be a $M \times N$ matrix with all the observations as columns. Note that small, bold letters represent vectors ($\mathbf{x}$), while capital, non-bold letters (X) represent matrices. The goal of the dimensionality reduction is to find another representation $\boldsymbol{\chi}$ of a smaller dimension m such that as much information as possible is retained from the original set of observations $\mathbf{x}$. This involves some transformation operator from the original vectors onto the new vectors, $\boldsymbol{\chi} = T(\mathbf{x})$. These projected vectors are sometimes called feature vectors, and the projection of $\mathbf{x}_n$ will be noted as $\boldsymbol{\chi}_n$. There might not be an inverse for this projection, but there must be a way of recovering an approximate value to the original vector, $\hat{\mathbf{x}} = R(\boldsymbol{\chi})$, such that $\hat{\mathbf{x}} \approx \mathbf{x}$.

An interesting property of any dimensionality reduction technique is to consider its stability. In this context, a technique is said to be ε-stable, if for any two input data points, $\mathbf{x}_1$ and $\mathbf{x}_2$, the following inequation holds [Baraniuk2010]: $(1-\varepsilon)\|\mathbf{x}_1 - \mathbf{x}_2\|_2^2 \leq \|\boldsymbol{\chi}_1 - \boldsymbol{\chi}_2\|_2^2 \leq (1+\varepsilon)\|\mathbf{x}_1 - \mathbf{x}_2\|_2^2$. Intuitively, this equation implies that Euclidean distances in the original input space are relatively conserved in the output feature space.

2. Methods based on Statistics and Information Theory

This family of methods reduces the input data according to some statistical or information theory criterion. Somehow, the methods based on information theory can be seen as a generalization of the ones based on statistics in the sense that they can capture non-linear relationships between variables, can handle interval and categorical variables at the same time, and many of them are invariant to monotonic transformations of the input variables.

2.1 Vector Quantization and Mixture models

Probably the simplest way of reducing dimensionality is by assigning a class (among a total of K classes) to each one of the observations $\mathbf{x}_n$. This can be seen as an extreme case of dimensionality reduction in which we go from M dimensions to 1 (the discrete class label χ). Each class, χ, has a representative $\overline{\mathbf{x}}_\chi$ which is the average of all

the observations assigned to that class. If a vector $\mathbf{x}_n$ has been assigned to the χ_n-th class, then its approximation after the dimensionality reduction is simply $\hat{\mathbf{x}}_n = \overline{\mathbf{x}}_{\chi_n}$ (see Fig. 1).

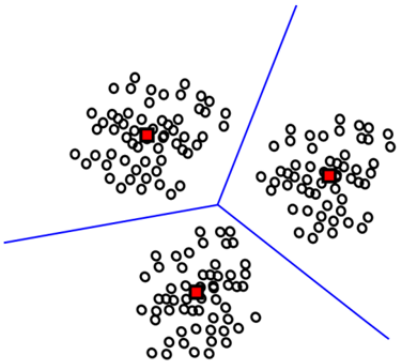

Figure 1.Example of the use of a vector quantization. Black circles represent the input data, $\mathbf{x}_n$; red squares represent class representatives, $\overline{\mathbf{x}}_\chi$.

The goal is thus to find the representatives $\overline{\mathbf{x}}_\chi$ and class assignments $u_\chi(\mathbf{x})$ ($u_\chi(\mathbf{x})$ is equal to 1 if the observation $\mathbf{x}$ is assigned to the χ-th class, and is 0 otherwise) such that $J_{VQ} = E\left\{\sum_{\chi=1}^{K} u_\chi(\mathbf{x})\left\|\mathbf{x}-\overline{\mathbf{x}}_\chi\right\|^2\right\}$ is minimized. This problem is known as vector quantization or K-means [Hartigan1979]. The optimization of this goal function is a combinatorial problem although there are heuristics to cut down its cost [Gray1984,Gersho1992]. An alternative formulation of the K-means objective function is $J_{VQ} = \left\|X - WU\right\|_F^2$ subject to $U^t U = I$ and $u_{ij} \in \{0,1\}$ (i.e., that each input vector is assigned to one and only one class) [Batmanghelich2012]. In this expression, W is a $M \times m$ matrix with all representatives as column vectors, U is a $m \times N$ matrix whose ij-th entry is 1 if the j-th input vector is assigned to the i-th class, and $\left\|\cdot\right\|_F^2$ denotes the Frobenius norm of a matrix.

This intuitive goal function can be put in a probabilistic framework. Let us assume we have a generative model of how the data is produced. Let us assume that the observed data are noisy versions of K vectors $\mathbf{x}_\chi$ which are equally likely *a priori*. Let us assume that the observation noise is normally distributed with a spherical covariance matrix $\Sigma = \sigma^2 I$. The likelihood of observing $\mathbf{x}_n$ having produced $\mathbf{x}_\chi$ is $l(\mathbf{x}_n \mid \mathbf{x}_\chi, \sigma^2) = \frac{1}{(2\pi)^{\frac{M}{2}}\sigma}\exp\left(-\frac{1}{2}\frac{\left\|\mathbf{x}_n - \mathbf{x}_\chi\right\|^2}{\sigma^2}\right)$. With our previous definition of $u_\chi(\mathbf{x})$ we can express it as $l(\mathbf{x}_n \mid \mathbf{x}_\chi, \sigma^2) = \frac{1}{(2\pi)^{\frac{M}{2}}\sigma}\exp\left(-\frac{1}{2}\frac{\sum_{\chi=1}^{K} u_\chi(\mathbf{x}_n)\left\|\mathbf{x}_n - \mathbf{x}_\chi\right\|^2}{\sigma^2}\right)$. The log likelihood of observing the whole dataset $\mathbf{x}_n$ ($n = 1, 2, \ldots, N$) after removing all constants is $L(X \mid \mathbf{x}_\chi) = \sum_{n=1}^{N}\sum_{\chi=1}^{K} u_\chi(\mathbf{x}_n)\left\|\mathbf{x}_n - \mathbf{x}_\chi\right\|^2$. We, thus, see that the goal function of vector quantization J_{VQ} produces the maximum likelihood estimates of the underlying $\mathbf{x}_\chi$ vectors.

Under this generative model, the probability density function of the observations is the convolution of a Gaussian function and a set of delta functions located at the $\mathbf{x}_\chi$ vectors, i.e., a set of Gaussians located at the $\mathbf{x}_\chi$ vectors. The vector quantization then is an attempt to find the centers of the Gaussians forming the probability

density function of the input data. This idea has been further pursued by Mixture Models [Bailey1994] that are a generalization of vector quantization in which, instead of looking only for the means of the Gaussians associated to each class, we also allow each class to have a different covariance matrix Σ_χ, and different *a priori* probability π_χ. The algorithm looks for estimates of all these parameters by Expectation-Maximization, and at the end produces for each input observation $\mathbf{x}_n$, the label χ of the Gaussian that has the maximum likelihood of having generated that observation.

We can extend this concept and, instead of making a hard class assignment, we can make a fuzzy class assignment by allowing $0 \le u_\chi(\mathbf{x}) \le 1$ and requiring $\sum_{\chi=1}^{I} u_\chi(\mathbf{x}) = 1$ for all $\mathbf{x}$. This is another famous vector quantization algorithm called fuzzy K-means [Bezdek1981, Bezdek1984].

The K-means algorithm is based on a quadratic objective function, which is known to be strongly affected by outliers. This drawback can be alleviated by taking the l_1 norm of the approximation errors and modifying the problem to $J_{K-medians} = \|X - WU\|_1^2$ subject to $U^t U = I$ and $u_{ij} \in \{0,1\}$ [Arora1998, Batmanghelich2012]. [Iglesias2007] proposed a different approach to find data representatives less affected by outliers which we may call robust Vector Quantization, $J_{RVQ} = E\left\{\sum_{\chi=1}^{K} u_\chi(\mathbf{x})\Phi\left(\left\|\mathbf{x} - \bar{\mathbf{x}}_\chi\right\|^2\right)\right\}$ where $\Phi(x)$ is a function less sensitive to outliers than $\Phi(x) = x$, for instance [Iglesias2007] proposes $\Phi(x) = x^\alpha$ with α about 0.5.

Some authors [Girolami2002, Dhillon2004, Yu2012] have proposed to use a non-linear embedding of the input vectors $\mathbf{\Phi}(\mathbf{x})$ into a higher dimensional space (dimensionality expansion, instead of reduction), and then perform the vector quantization in this higher dimensional space (this kind of algorithms are called Kernel algorithms and are further explained below with Kernel PCA). The reason for performing this non-linear mapping is that the topological spherical balls induced by the distance $\left\|\mathbf{\Phi}(\mathbf{x}) - \mathbf{\Phi}(\bar{\mathbf{x}}_\chi)\right\|^2$ in the higher-dimensional space, correspond to non-spherical neighborhoods in the original input space, thus allowing for a richer family of distance functions.

Although vector quantization has all the ingredients to be considered a dimensionality reduction (mapping from the high dimensional space to the low dimensional space by assigning a class label χ, and back to high dimensional space through an approximation), this algorithm has a serious drawback. The problem is that the distances in the feature space (χ runs from 1 to K) do not correspond to distances in the original space. For example, if $\mathbf{x}_n$ is assigned label 0, $\mathbf{x}_{n+1}$ label 1, and $\mathbf{x}_{n+2}$ label 2, it does not mean that $\mathbf{x}_n$ is closer to $\mathbf{x}_{n+1}$ than to $\mathbf{x}_{n+2}$ in the input M dimensional space. Labels are arbitrary and do not allow to conclude anything about the relative organization of the input data other than knowing that all vectors assigned to the same label are closer to the representative of that label than to the representative of any other label (this fact creates a Voronoi partition of the input space) [Gray1984,Gersho1992].

The algorithms presented from now on do not suffer from this problem. Moreover, the problem can be further attenuated by imposing a neighborhood structure on the feature space. This is done by Self-Organizing Maps and Generative Topographic Mappings, which are presented below.

2.2 PCA

Principal Component Analysis (PCA) is by far one of the most popular algorithms for dimensionality reduction [Pearson1901, Wold1987, Dunteman1989, Jollife2002]. Given a set of observations $\mathbf{x}$, with dimension M (they lie in $\mathbb{R}^M$), PCA is the standard technique for finding the single best (in the sense of least-square error) subspace of a given dimension, m. Without loss of generality, we may assume the data is zero-mean and the subspace to fit is a linear subspace (passing through the origin).

This algorithm is based on the search of orthogonal directions explaining as much variance of the data as possible. In terms of dimensionality reduction it can be formulated [Hyvarinen2001] as the problem of finding the m orthonormal directions $\mathbf{w}_i$ minimizing the representation error $J_{PCA} = E\left\{\left\|\mathbf{x} - \sum_{i=1}^{m} \langle \mathbf{w}_i, \mathbf{x} \rangle \mathbf{w}_i\right\|^2\right\}$. In this objective function, the reduced vectors are the projections $\boldsymbol{\chi} = \left(\langle \mathbf{w}_1, \mathbf{x} \rangle, ..., \langle \mathbf{w}_m, \mathbf{x} \rangle\right)^t$. This can be much more compactly written as $\boldsymbol{\chi} = W^t \mathbf{x}$, where W is a $M \times m$ matrix whose columns are the orthonormal directions $\mathbf{w}_i$ (or equivalently $W^t W = I$). The approximation to the original vectors is given by $\hat{\mathbf{x}} = \sum_{i=1}^{m} \langle \mathbf{w}_i, \mathbf{x} \rangle \mathbf{w}_i$, or what is the same, $\hat{\mathbf{x}} = W\boldsymbol{\chi}$. In Figure 2, we show a graphical representation of a PCA transformation in only two dimensions ($\mathbf{x} \in \mathbb{R}^2$). As can be seen from Figure 2, the variance of the data in the original data space is best captured in the rotated space given by vectors $\boldsymbol{\chi} = W^t \mathbf{x}$.

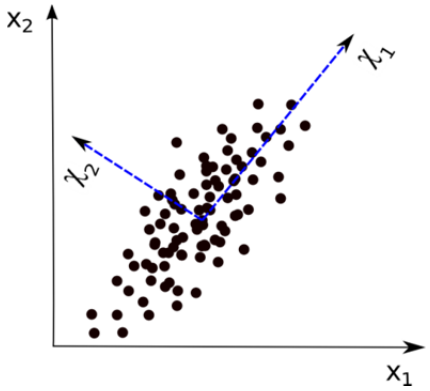

Figure 2. Graphical representation of a PCA transformation in only two dimensions.

χ_1 is the first principal component and it goes in the direction of most variance, χ_2 is the second principal component, it is orthogonal to the first and it goes in the second direction with most variance (in $\mathbb{R}^2$ there is not much choice, but in the general case, $\mathbb{R}^M$, there is). Observe that without loss of generality the data is centred about the origin of the output space.

We can rewrite the objective function as $J_{PCA} = E\left\{\left\|\mathbf{x} - W\boldsymbol{\chi}\right\|^2\right\} = E\left\{\left\|\mathbf{x} - WW^t\mathbf{x}\right\|^2\right\} \propto \left\|X - WW^t X\right\|_F^2$. Note that the class membership matrix (U in vector quantization) has been substituted in this case by $W^t X$, which in general can take any positive or negative value. It, thus, has lost its membership meaning and simply constitutes the weights of the linear combination of the column vectors of W that better approximate each input $\mathbf{x}$. Finally, PCA objective function can also be written as $J_{PCA} = \mathrm{Tr}\left\{W^t \Sigma_X W\right\}$ [He2011], where $\Sigma_X = \frac{1}{N}\sum_i (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^t$ is the covariance matrix of the observed data. The PCA formulation has also been extended to complex-valued input vectors [Li2011], the method is called non-circular PCA.

The matrix projection of the input vectors onto a lower dimensional space ($\boldsymbol{\chi} = W^t \mathbf{x}$) is a wide-spread technique in dimensionality reduction as will be shown in this article. The elements involved in this projection have an interesting interpretation as explained in the following example. Let us assume that we are analyzing scientific articles related to a specific domain. Each article will be represented by a vector $\mathbf{x}$ of word frequencies, i.e., we choose a set of M words representative of our scientific area, and we annotate how many times each word appears in each article. Each vector $\mathbf{x}$ is then orthogonally projected onto the new subspace defined by the

vectors $\mathbf{w}_i$. Each vector $\mathbf{w}_i$ has dimension M and it can be understood as a "topic" (i.e., a topic is characterized by the relative frequencies of the M different words; two different topics will differ in the relative frequencies of the M words). The projection of $\mathbf{x}$ onto each $\mathbf{w}_i$ gives an idea of how important is topic $\mathbf{w}_i$ to represent the article $\mathbf{x}$. Important topics have large projection values and, therefore, large values in the corresponding component of $\boldsymbol{\chi}$ (see Fig. 3).

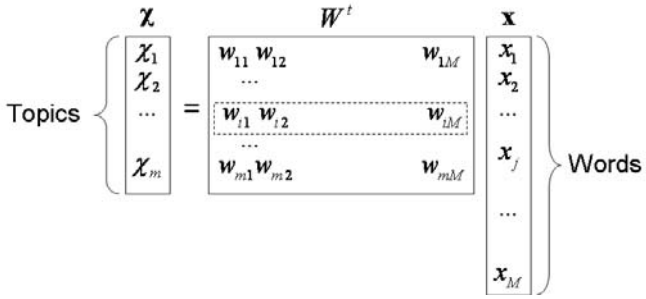

Figure 3. Projection of the vector $\mathbf{x}$ onto the subspace spanned by the vectors $\mathbf{w}_i$ (rows of W^t). The components of $\mathbf{x}$ represent the frequency of each word in a given scientific article. The vectors $\mathbf{w}_i$ represent the word composition of a given topic. Each component of the projected vector $\boldsymbol{\chi}$ represents how important is that topic for the article being treated.

It can be shown [Hyvarinen2001,Jenssen2010] that when the input vectors, $\mathbf{x}$, are zero-mean (if they are not, we can transform the input data simply by subtracting the sample average vector), then the solution of the minimization of J_{PCA} is given by the m eigenvectors associated to the largest m eigenvalues of the covariance matrix of $\mathbf{x}$ ($C_{\mathbf{x}} = \frac{1}{N} XX^t$, note that the covariance matrix of $\mathbf{x}$ is a $M \times M$ matrix with M eigenvalues). If the eigenvalue decomposition of the input covariance matrix is $C_{\mathbf{x}} = W_M \Lambda_M W_M^t$ (since $C_{\mathbf{x}}$ is a real-symmetric matrix), then the feature vectors are constructed as $\boldsymbol{\chi} = \Lambda_m^{-\frac{1}{2}} W_m^t \mathbf{x}$, where Λ_m is a diagonal matrix with the m largest eigenvalues of the matrix Λ_M and W_m are the corresponding m columns from the eigenvectors matrix W_M. We could have constructed all the feature vectors at the same time by projecting the whole matrix X, $U = \Lambda_m^{-\frac{1}{2}} W_m^t X$. Note that the i-th feature is the projection of the input vector $\mathbf{x}$ onto the i-th eigenvector, $\chi_i = \lambda_i^{-\frac{1}{2}} \mathbf{w}_i^t \mathbf{x}$. The so-computed feature vectors have identity covariance matrix, $C_{\boldsymbol{\chi}} = I$, meaning that the different features are decorrelated.

Univariate variance is a second-order statistical measure of the departure of the input observations with respect to the sample mean. A generalization of the univariate variance to multivariate variables is the trace of the input covariance matrix. By choosing the m largest eigenvalues of the covariance matrix $C_{\mathbf{x}}$, we guarantee that we are making a representation in the feature space explaining as much variance of the input space as possible with only m variables. In fact, $\mathbf{w}_1$ is the direction in which the data has the largest variability, $\mathbf{w}_2$ is the direction with largest variability once the variability along $\mathbf{w}_1$ has been removed, $\mathbf{w}_3$ is the direction with largest variability once the variability along $\mathbf{w}_1$ and $\mathbf{w}_2$ has been removed, etc. Thanks to the orthogonality of the $\mathbf{w}_i$ vectors, and the subsequent decorrelation of the feature vectors, the total variance explained by PCA decomposition can be conveniently measured as the sum of the variances of each feature, $\sigma_{PCA}^2 = \sum_{i=1}^{m} \lambda_i = \sum_{i=1}^{m} \mathrm{Var}\{\chi_i\}$.

2.2.1 Incremental, stream or online PCA

Let us assume that we have observed a number of input vectors $\mathbf{x}$ and we have already performed their dimensionality reduction with PCA. Let us assume that we are given new observations and we want to refine our directions $\mathbf{w}_i$ to accommodate the new vectors. In the standard PCA approach we would have to re-estimate the covariance matrix of $\mathbf{x}$ (now using all the vectors, old and new), and to recompute its eigenvalue decomposition. Incremental methods (such as Incremental PCA [Artac2002]) provide clever ways of updating our estimates of the best directions $\mathbf{w}_i$ based on the old estimates of the best directions and the new data available. In this way, we can efficiently process new data as they arrive. For this reason, incremental methods are also known as stream methods or online methods. This idea can be applied to many of the methods discussed in this review and will not be further commented.

2.2.2 Relationship of PCA and SVD

Another approach to the PCA problem, resulting in the same projection directions $\mathbf{w}_i$ and feature vectors $\boldsymbol{\chi}$ uses Singular Value Decomposition (SVD, [Golub1970, Klema1980, Wall2003]) for the calculations. Let us consider the matrix X whose columns are the different observations of the vector $\mathbf{x}$. SVD decomposes this matrix as the product of three other matrices $X = WDU$ (W is of size $M \times M$, D is a diagonal matrix of size $M \times N$, and U is of size $N \times N$) [Abdi2007]. The columns of W are the eigenvectors of the covariance matrix of $\mathbf{x}$ and D_{ii} is the square root of its associated eigenvalue. The columns of U are the feature vectors. So far we have not performed any dimensionality reduction yet. As in PCA the dimensionality reduction is achieved by discarding those components associated to the lowest eigenvalues. If we keep the m directions with largest singular values, we are approximating the data matrix by $\hat{X} = W_m D_m U_m$ (W_m is of size $M \times m$, D_m is of size $m \times m$, and U_m is of size $m \times N$). It has been shown [Johnson1963] that $\hat{X}$ is the matrix better approximating X in the Frobenius norm sense (i.e., exactly the same as required by J_{PCA}).

Another interesting property highlighted by this decomposition is that the eigenvectors $\mathbf{w}_i$ (columns of the W matrix) are an orthonormal basis of the subspace spanned by the observations $\mathbf{x}$. That means that for each element in this basis we can find a linear combination of input vectors such that $\mathbf{w}_i = \sum_{n=1}^{N} \alpha_{in} \mathbf{x}_n$. This fact will be further exploited by Sparse PCA and Kernel PCA. This digression on the SVD approach to PCA helps us to understand a common situation in some experimental settings. For instance, in microarray experiments, we have about 50 samples and 1000 variables. As shown by the SVD decomposition, the rank of the covariance matrix is the minimum between $M = 1000$ and $N = 50$, therefore, we cannot compute more than 50 principal components.

An interesting remark on the SVD decomposition is that among all possible matrix decompositions of the form $X = WDU$, SVD is the only family of decompositions (SVD decomposition is not unique) yielding diagonal matrices in D. In other words, the matrices W and U can differ significantly from the SVD decomposition as long as D is not a diagonal matrix. This fact is further exploited by Sparse Tensor SVD (see dictionary-based methods below).

2.2.3 Nonlinear PCA

PCA can be extended to non-linear projections very easily conceptually although its implementation is more involved. Projections can be replaced by $\mathbf{f}(W^t \mathbf{x})$, being $\mathbf{f}(\boldsymbol{\chi}) : \mathbb{R}^m \rightarrow \mathbb{R}^m$ a non-linear function chosen by the user. The goal function is then $J_{NLPCA} = E\left\{ \left\| \mathbf{x} - W\mathbf{f}(W^t \mathbf{x}) \right\|^2 \right\}$ which is minimized subject to $W^t W = I$ [Girolami1997b].

2.2.4 PCA rotations and Sparse PCA

A drawback of PCA is that the eigenvectors ($\mathbf{w}_i$) have usually contributions from all input variables (all their components are significantly different from zero). This makes their interpretation more difficult since a large feature value cannot be attributed to a few (ideally a single) input values. Possible solutions are rotating the eigenvectors using PCA rotation methods (Varimax, Quartimax, etc.) or forcing many of the components of $\mathbf{w}_i$ to be zero, i.e., to have a Sparse PCA.

An easy way of forcing the interpretability of the principal components $\mathbf{w}_i$ is by rotating them once they have been computed. The subspace spanned by the rotated vectors is the same as the one spanned by the unrotated vectors. Therefore, the approximation $\hat{\mathbf{x}}$ is still the same, although the feature vector must be modified to account for the rotation. This is the approach followed by Varimax [Kaiser958]. It looks for the rotation that maximizes the variance of the feature vector after rotation (the idea is that maximizing this variance implies that each feature vector uses only a few eigenvectors). Quartimax looks for a rotation minimizing the number of factors different from zero in the feature vector. Equimax and Parsimax are compromises between Varimax and Quartimax. All these criteria are generally known as Orthomax and they have been unified under a single rotation criterion [Crawford1970]. Among these criteria, Varimax is by far the most popular. If the orthogonality condition of the vectors $\mathbf{w}_i$ is removed, then the new principal components are free to move and the rotation matrix is called an "oblique" rotation matrix. Promax [Abdi2003] is an example of such an oblique rotation technique. However, these oblique rotations have been superseded by Generalized PCA.

Recently, there have been some papers imposing the sparseness of the feature vectors by directly regularizing a functional solving the PCA problem (formulated as a regression problem). This kind of methods is commonly called Sparse PCA. One of the most popular algorithms of this kind is the one of Zou [Zou2006]. We have already seen that the PCA problem can be seen as regression problem whose objective function is $J_{PCA} = E\left\{\left\|\mathbf{x} - WW^t\mathbf{x}\right\|^2\right\}$. Normally the optimization is performed with the constraint $W^tW = I$ (i.e., the directions $\mathbf{w}_i$ have unit module). We can generalize this problem, and instead of using the same matrix to build the feature vectors (W^t) and reconstruct the original samples (W), we can make them different. We can use ridge regression (a Tikhonov regularization to avoid possible instabilities caused by the eventual ill-conditioning of the regression, the most common one is simply the l_2 norm), and a regularization based on some norm promoting sparseness (like the l_1 norm): $J_{SPCA} = E\left\{\left\|\mathbf{x} - W\tilde{W}^t\mathbf{x}\right\|^2\right\} + \lambda_1\left\|\tilde{W}\right\|_{l_1} + \lambda_2\left\|\tilde{W}\right\|_{l_2}^2$. In the previous formula the l_p norms of the matrices are computed as $\left\|W\right\|_{l_p} = \left(\sum_{i,j}\left|w_{ij}\right|^p\right)^{\frac{1}{p}}$ and the objective function is optimized with respect to W and $\tilde{W}$ that are $M \times m$ matrices. It has been shown [Zou2006] that promoting the sparseness of $\tilde{W}$ promotes the sparseness of the feature vectors which is, after all, the final goal of this algorithm.

The previous sparse approaches tried to find sparse projection directions by zeroing some of the elements in the projection directions. However, we may prefer absolutely removing some of the input features (so that their contribution to all projection directions is zero). [Ulfarsson2011] proposed the sparse variable PCA (svPCA). svPCA is based on noisy PCA (nPCA), which is a special case of Factor Analysis (see below). The observed data is supposed to have been generated as $\mathbf{x} = W\boldsymbol{\chi} + \mathbf{n}$. Assuming that the covariance of the noise is $\sigma^2 I$, the goal of nPCA is to maximize the log-likelihood of observing the data matrix X under the nPCA model, that is $J_{nPCA} = -\frac{1}{2}\mathrm{Tr}\left\{\Sigma_X\Omega^{-1}\right\} - \frac{1}{2}\log\left|\Omega\right|$ where $\Omega = WW^t + \sigma^2 I$. svPCA objective function is $J_{svPCA} = J_{nPCA} - \frac{N}{2\sigma^2}\sum_{i=1}^{m}\left\|\mathbf{w}_i\right\|_0$.

2.2.5 Localized PCA and Subspace segmentation

As we have explained above, given a set of data samples in $\mathbb{R}^M$, the standard technique for finding the single best (in the sense of least-square error) subspace of a given dimension is the Principal Component Analysis (PCA). However, in practice, a single linear model has severe limitations for modelling real data. Because the data can be heterogeneous and contain multiple subsets each of which is best fitted with a different linear model. Suppose that we have a high dimensional dataset, that can be conveniently decomposed into different small datasets or clusters, which can be approximated well by different linear subspaces of small dimension by means of a principal component analysis. This situation is presented in Fig. 4. Note from Fig. 4 that the dataset composed by the red and blue points can be decomposed in two sets, one formed by the red and the other by the blue ones and each of this groups can effectively be approximated using a two dimensional linear subspaces.

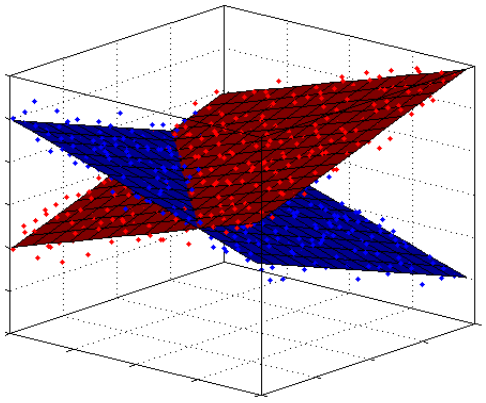

Figure 4. Mixed dataset composed by the red and blue points that can be decomposed in two smaller datasets, which can effectively be described using two dimensional linear subspaces.

The term Localized PCA has been used several times through literature to refer to different algorithms. Here we will refer to the most successful ones. [Fukunaga1971] proposed an extension of the K-means algorithm which we will refer to as Cluster-PCA. In K-means, a cluster is represented by its centroid. In Cluster-PCA, a cluster is represented by a centroid plus an orthogonal basis defining a subspace that embeds locally the cluster. An observation $\mathbf{x}$ is assigned to a cluster if the projection of $\mathbf{x}$ onto the cluster subspace ($\hat{\mathbf{x}} = WW^t\mathbf{x}$) is the closest one (the selection of the closest subspace must be done with care so that extrapolation of the cluster is avoided). Once all observations have been assigned to their corresponding clusters, the cluster centroid is updated as in K-means and the cluster subspace is recalculated by using PCA on the observations belonging to the cluster. As with K-means, a severe drawback of the algorithm is its dependence with the initialization, and several hierarchically divisive algorithms have been provided (Recursive Local PCA) [Liu2003b]. For a review on this kind of algorithms see [Einbeck2008].

Subspace segmentation extends the idea of locally embedding the input points into linear subspaces. The assumption is that the data has been generated using several subspaces that may not be orthogonal. The goal is to identify all these subspaces. Generalized PCA [Vidal2005] is a representative of this family of algorithms. Interestingly, the subspaces to be identified are represented as polynomials whose degree is the amount of subspaces to identify and whose derivatives at a data point give normal vectors to the subspace passing through the point.

2.3 Principal curves, surfaces and manifolds

PCA is the perfect tool to reduce data that in their original M -dimensional space lie in some linear manifold. However, there are situations at which the data follow some curved structure (e.g., a slightly bent line). In this case, approximating the curve by a straight line will not perform a good approximation of the original data. We can observe a situation of this type in Fig. 5.

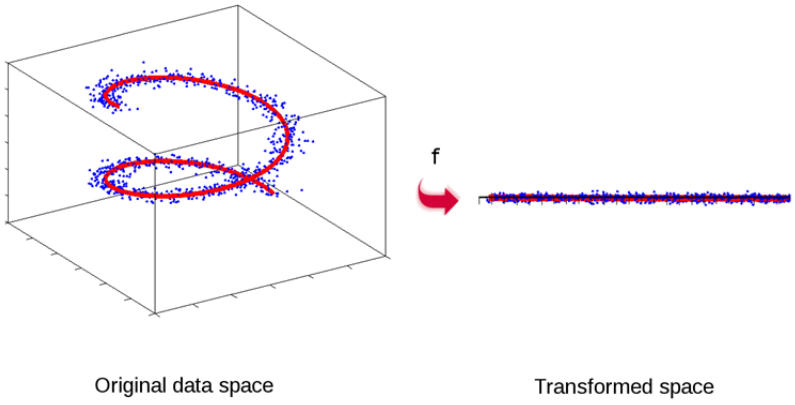

Figure 5. Dataset that lie in a curved structure (left) and transformed dataset (right)

In Fig. 5 we show a dataset following a curved structured and therefore this dataset will not be conveniently described using the PCA method. Note that in the case of the data shown in Fig. 5, it will be needed at least three principal components to describe the data precisely. In Fig. 5 we show the same dataset after transforming it. Note that this data does no longer follow a curved a structured and in this case, it follows a linear one. Therefore, the data shown on the right of Fig. 5 can be conveniently described using PCA approach and using only one principal component.

Before introducing principal curves, surfaces and manifolds in depth, let us review PCA from a different perspective. Given a set of observations of the input vectors $\mathbf{x}$ with zero average (if the original data is not zero-average, we can simply subtract the average from all data points), we can look for the line passing through the origin and with direction $\mathbf{w}_1$ (whose equation is $\mathbf{f}(\chi)=\mathbf{w}_1\chi$) that better fits this dataset, i.e., that minimizes $J_{line}=E\left\{\inf_{\chi}\|\mathbf{x}-\mathbf{f}(\chi)\|^2\right\}$. The infimum in the previous objective function implies that for each observation $\mathbf{x}_n$ we have to look for the point in the line (defined by its parameter χ_n) that is closest to it. The point $\mathbf{f}(\chi_n)$ is the orthogonal projection of the observation onto the line. It can be proved that the solution of this problem is the direction with the largest data variance, that is, the same solution as in PCA. Once we have found the first principal line, we can look for the second simply by constructing a new dataset in which we have subtracted the line previously computed ($\mathbf{x}_n^{'}=\mathbf{x}_n-\mathbf{f}(\chi_n)$). Then, we apply the same procedure $m-1$ times to detect the subsequent most important principal lines. The dimensionality reduction is achieved simply by substituting the observation $\mathbf{x}_n$ by the collection of parameters χ_n needed for its projection onto the different principal lines.

The objective function J_{line} is merely a linear regression on the input data. For detecting curves instead of lines, one possibility would be to fix the family of curves sought (parabolic, hyperbolic, exponential, polynomial …) and optimize its parameters (as was done with the line). This is exactly non-linear regression and several methods based on Neural Networks (as sophisticated non-linear regressors) have been proposed (Non-linear PCA, [Kramer1991, Scholz2008], autoencoder neural networks [Baldi1989,DeMers1993,Kambhatla1997]).

Alternatively, we can look for the best curve (not in a parametric family) [Hastie1989]. In Statistics, it is well known that the best regression function is given by the conditional expectation $\mathbf{f}(\chi)=E\left\{\mathbf{x}\mid\chi_{\mathbf{f}}(\mathbf{x})\right\}$, where $\chi_{\mathbf{f}}(\mathbf{x})$ represents the curve parameter χ needed to project $\mathbf{x}$ onto $\mathbf{f}$. In other words, the best curve is the one that assigns for each χ the average of all observed values projected onto χ. The parameterization of the curve $\mathbf{f}$ must be such that it has unit speed (i.e., $\left\|\frac{d\mathbf{f}}{d\chi}\right\|=1$ for all χ), otherwise we could not uniquely determine this function. There are two warnings on this approach. The first one is that it might be locally biased if the noise of the observations is larger than the local curvature of the function. The second one is that if we only have a finite set of

observations $\mathbf{x}$, we will have to use some approximation of the expectation so that we make the curve continuous. The two most common choices to make the curve continuous are kernel estimates of the expectation and the use of splines. In fact, the goal function of the classical smoothing spline is $J_{spline} = E\left\{\inf_{\chi} \|\mathbf{x}-\mathbf{f}(\chi)\|^2\right\} + \lambda \int \left\|\frac{d\mathbf{f}(\chi)}{d\chi}\right\|^2 d\chi$, which regularizes the curve fitting problem with the curvature of the curve. An advantage of the use of splines is their efficiency (the algorithm runs as $O(N)$ as compared to the $O(N^2)$ of the kernel estimates). However, it is difficult to choose the regularization weight, λ.

Principal Curves can be combined with the idea of Localized PCA (constructing local approximations to data). This has been done by several authors: Principal Curves of Oriented Points (PCOP) [Delicado2001], and Local Principal Curves (LPC) [Einbeck2005].

The Principal Curves idea can be extended to more dimensions (see Fig. 6). Principal surfaces are the functions minimizing $J_{surface} = E\left\{\inf_{\chi_1,\chi_2} \|\mathbf{x}-\mathbf{f}(\chi_1,\chi_2)\|^2\right\}$. The solution is again, $\mathbf{f}(\chi_1,\chi_2) = E\{\mathbf{x} \mid \boldsymbol{\chi}_{\mathbf{f}}(\mathbf{x})\}$, where $\boldsymbol{\chi}_{\mathbf{f}}(\mathbf{x})$ returns the parameters of the surface needed for the projection of $\mathbf{x}$. Intuitively, the principal surface at the point (χ_1,χ_2) is the average of all observations whose orthogonal projection is at $\mathbf{f}(\chi_1,\chi_2)$. The extension to manifolds is straightforward [Smola1999], $J_{manifold} = E\left\{\inf_{\chi} \|\mathbf{x}-\mathbf{f}(\boldsymbol{\chi})\|^2\right\} + \lambda \|P\mathbf{f}\|^2$ that is a Tikhonov regularized version of the non-linear regression problem. P is a homogenous invariant scalar operator penalizing unsmooth functions. The fact that the regularization is homogeneous invariant implies that all surfaces which can be transformed into each other by rotations are equally penalized. A feasible way of defining the function $\mathbf{f}(\boldsymbol{\chi})$ is by choosing a number of locations $\boldsymbol{\chi}_i$ (normally distributed on a regular grid although the method is not restricted to this choice) and expanding this function as a weighted sum of a kernel function $k(\boldsymbol{\chi}-\boldsymbol{\chi}_i)$ at those locations, $\mathbf{f}(\boldsymbol{\chi}) = \sum_{i=1}^{K} \boldsymbol{\alpha}_i k(\boldsymbol{\chi}-\boldsymbol{\chi}_i)$. The number of locations, K, controls the complexity of the manifold and the vectors $\boldsymbol{\alpha}_i$ (which are in the space of $\mathbf{x}$ and, thus, have dimension M) control its shape. However, the dimensionality reduction is still controlled by the dimension of the vector $\boldsymbol{\chi}$. Radial basis functions such as the Gaussian are common kernels ($k(\boldsymbol{\chi},\boldsymbol{\chi}_i) = k\left(\|\boldsymbol{\chi}-\boldsymbol{\chi}_i\|\right)$). With this expansion, the regularization term becomes [Smola1999] $\|P\mathbf{f}\|^2 = \sum_{i,j=1}^{I} \langle \boldsymbol{\alpha}_i, \boldsymbol{\alpha}_j \rangle k(\boldsymbol{\chi}_i,\boldsymbol{\chi}_j)$. An interesting feature of this approach is that by using periodical kernels, one can learn circular manifolds.

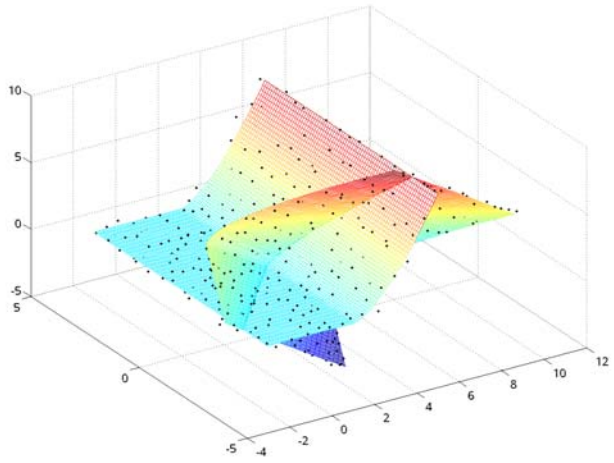

Figure 6. Example of data distributed along two principal surfaces.

2.4 Generative Topographic Mapping

A different but related approach to reduce the dimensionality by learning a manifold is the Generative Topographic Mapping (GTM) [Bishop1998]. This method is "generative" because it assumes that the observations $\mathbf{x}$ have been generated by noisy observations of a mapping of the low dimensional vectors $\boldsymbol{\chi}$ onto a higher dimension, $\mathbf{f}(\boldsymbol{\chi})$ (see Fig. 7, in fact the form of this non-linear mapping is exactly the same as in the principal manifolds of the previous section, $\mathbf{f}(\boldsymbol{\chi}) = \sum_{i=1}^{K} \boldsymbol{\alpha}_i k(\boldsymbol{\chi} - \boldsymbol{\chi}_i)$). In this method, it is presumed that the possible $\boldsymbol{\chi}$ vectors lie on a discrete grid with K points, and that the *a priori* probability of each one of the points of the grid is the same (uniform distribution). If the noise is supposed to be Gaussian (or any other spherical distribution), the maximum likelihood estimates of the vectors $\boldsymbol{\alpha}_i$ boils down to the minimization of $J_{GTM} = E\left\{\inf_{\boldsymbol{\chi}} \left\|\mathbf{x} - \mathbf{f}(\boldsymbol{\chi})\right\|^2\right\}$. This objective function can be regularized by $\sum_{i=1}^{K} \left\|\boldsymbol{\alpha}_i\right\|^2$ (instead of $\left\|P\mathbf{f}\right\|^2$) which is the result of estimating the Maximum *a Posteriori* under the assumption that the $\boldsymbol{\alpha}_i$ are normally distributed with 0 mean.

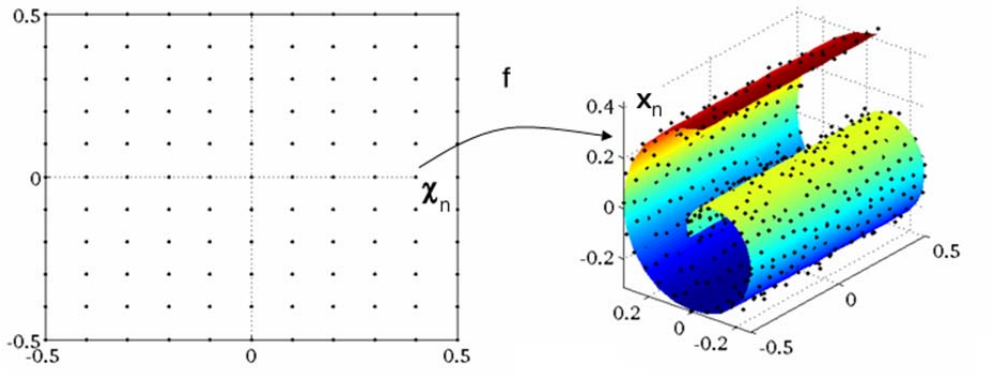

Figure 7. Example of Generative Topographic Mapping. The observed data (right) is assumed to be generated by mapping points in a lower dimensional space.

2.5 Self-Organizing Maps

In fact, GTM has been proposed as a generalization of Self-Organizing Maps, which in their turn are generalizations of the vector quantization approaches presented at the beginning. Self-Organizing Maps (SOM) work as in Vector Quantization by assigning to each input vector a label χ_n corresponding to the class closest to its representative vector. The reconstruction of $\mathbf{x}_n$ is still $\hat{\mathbf{x}}_n = \overline{\mathbf{x}}_{\chi_n}$, i.e., the class representative of class χ_n. However, class labels are forced to lie in a manifold at which a topological neighborhood is defined (see Fig. 8). In this way, classes that are close to each other in the feature space are also close to each other in the input space.

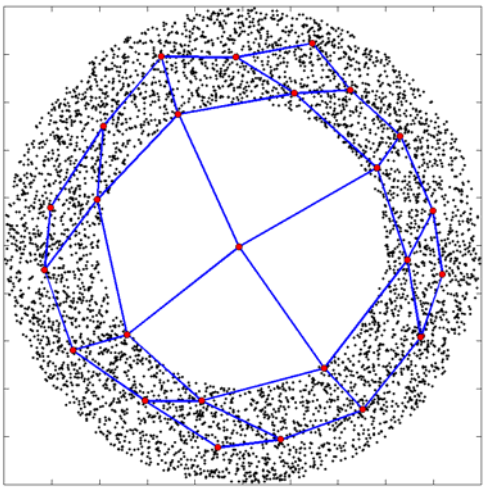

Figure 8. Original data lies in a ring, vector representatives calculated by SOM are represented as red points. The output map topology has been represented by linking each representative vector to its neighbors with a blue edge. Note that because of the topological constraint there might be representative vectors that are not actually representing any input point (e.g., the vector in the center of the ring).

Kohonen's SOMs [Kohonen1990, Kohonen1993, Kohonen2001] are the most famous SOMs. They work pretty well in most contexts, they are very simple to understand and implement, but they lack from a solid mathematical framework (they are not optimizing any functional and they cannot be put in a statistical framework). They start by creating a set of labels on a given manifold (usually a plane). Labels are distributed in a regular grid and the topological neighbourhood is defined as the neighbours in the plane of each point of the grid (note that this idea can be easily extended to higher dimensions). For initialization we assign to each label a class representative at random. Each input observation $\mathbf{x}_n$ is compared to all class representatives and it is assigned to the closest class whose label we will refer to as χ_n. In its batch version, once all the observations have been assigned, the class representatives are then updated according to $\overline{\mathbf{x}}_\chi = \frac{\sum_{n=1}^{N} k(\chi,\chi_n)\mathbf{x}_n}{\sum_{n=1}^{N} k(\chi,\chi_n)}$. The function $k(\chi,\chi_n)$ is a kernel that gives more weight to pairs of classes that are closer in the manifold. The effect of this is that when an input vector $\mathbf{x}_n$ is assigned to a given class, the classes surrounding this class will also be updated with that input vector (although with less weight than the winning class). Classes far from the winning class receive updates with a weight very close to 0. This process is iterated until convergence.

GTM generalizes Kohonen's SOM because the class representatives $\overline{\mathbf{x}}_\chi$ in SOMs can be assimilated to the $\mathbf{f}(\chi_i)$ elements of GTM, and the function $\mathbf{f}(\chi)$ of GTM can be directly be computed from the kernel $k(\chi,\chi_n)$ in SOM [Bishop1998]. However, GTM has the advantage over SOMs that they are clearly defined in a statistical framework and the function $\mathbf{f}(\chi)$ is maximizing the likelihood of observing the given data. There is another difference of practical consequences, while SOM makes the dimensionality reduction by assigning one of the points of the grid in the manifold (i.e., it produces a discrete dimensionality reduction), GTM is capable of producing a continuous dimensionality reduction by choosing the parameters χ_n such that $\|\mathbf{x}_n - \mathbf{f}(\chi_n)\|$ is minimized.

Other generalizations of SOMs in the same direction are the Fuzzy SOM [Pascual-Marqui2001] and the Kernel Density SOM (KenDerSOM) [Pascual-Montano2001]. These generalizations rely on the regularization of the objective functions of Vector Quantization and the Mixture Models, respectively, by the term $\sum_{\chi,\chi'=1}^{K} \|\overline{\mathbf{x}}_\chi - \overline{\mathbf{x}}_{\chi'}\|^2 k(\chi,\chi')$, that is, class representatives corresponding to labels close to each other in the manifold

should have smaller differences. For a review on SOM and its relationships to Non-Linear PCA and Manifold learning see [Yin2008].

Neural Gas networks [Martinetz1991, Martinetz1993, Fritzke1995] is an approach similar to the standard SOM, only that the neighborhood topology is automatically learnt from the data. Edges appear and disappear following an aging strategy. This automatic topology learning allows adapting to complex manifolds with locally different intrinsic dimensionality [Pettis1979, Kegl2002, Costa2004] (see Fig. 9).

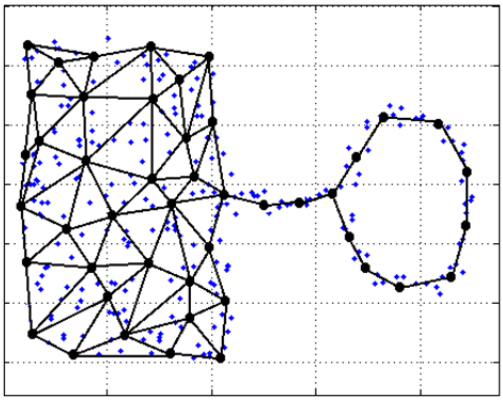

Figure 9. Example of Neural Gas network. Note that the network has been able to learn the 2D topology present at the left points, and the 1D topology of the right points.

2.6 Elastic maps, nets, principal graphs and principal trees

Elastic maps and nets [Gorban2004, Gorban2007] are half-way between SOMs and GTM. Like these two methods, the elastic map can be thought of as a net of nodes in a low-dimensional space. For each node, there is a mapping between the low-dimensional node and the high-dimensional space (like in SOM), i.e., for each node in the net χ there is a corresponding vector in the input space $\overline{\mathbf{x}}_\chi$. If an input vector $\mathbf{x}_n$ is assigned to a node χ_n, then its representative is $\hat{\mathbf{x}}_n = \overline{\mathbf{x}}_{\chi_n}$ (as in SOM). The goal function combines similarity to the input data with regularity and similarity within the net: $J_{EN} = \sum_{n=1}^{N} \left\| \mathbf{x}_n - \overline{\mathbf{x}}_{\chi_n} \right\|^2 + \sum_{\chi,\chi'=1}^{K} \left(\lambda_\chi \left\| \overline{\mathbf{x}}_\chi - \overline{\mathbf{x}}_{\chi'} \right\|^2 g(\chi,\chi') \right) + \sum_{\chi}^{K} \mu_\chi \left\| \overline{\mathbf{x}}_\chi - \frac{\sum_{\chi'=1}^{K} g(\chi,\chi')\overline{\mathbf{x}}_{\chi'}}{\sum_{\chi'=1}^{K} g(\chi,\chi')} \right\|^2$. The first term accounts for the fidelity of the data representation. In the second and third terms, $g(\chi,\chi')$ define a neighborhood (is equal to 1 if the two nodes are neighbors, and equal to 0 otherwise). The second term reinforces smoothness of the manifold by favoring similarity between neighbors; the third term imposes smoothness in a different way: a node representative must be close to the average of its neighbors. As opposed to SOM, elastic nets can delete or add nodes adaptively, creating nets that are well adapted to the manifold topology. For this reason, this method is also known as Principal Graphs. Interestingly, the rules to create and delete nodes, can force the graph to be a tree.

2.7 Kernel PCA and Multidimensional Scaling

Kernel PCA [Scholkopf1997, Scholkopf1999] is another approach trying to capture non-linear relationships. It can be well understood if Multidimensional Scaling (MDS), another linear method, is presented first. We have already seen that PCA can be computed from the eigenvalue decomposition of the input covariance matrix, $C_{\mathbf{x}} = \frac{1}{N} XX^t$. However, we could have built the inner product matrix (Gram matrix) $G_{\mathbf{x}} = X^t X$, i.e., the i, j-th component of

this matrix has the inner product of $\mathbf{x}_i$ with $\mathbf{x}_j$. The eigendecomposition of the Gram matrix yields $G_{\mathbf{x}} = W_N \Lambda_N W_N^t$ (since $G_{\mathbf{x}}$ is a real, symmetric matrix). MDS is a classical statistical technique [Kruskal1964a, Kruskal1964b, Schiffman1981, Kruskal1986, Cox2000, Borg2005] that builds with the m largest eigenvalues a feature matrix given by $U = \Lambda^{-\frac{1}{2}} W^t$. This feature matrix is the one best preserving the inner products of the input vectors, i.e., it minimizes the Frobenius norm of the difference $G_{\mathbf{x}} - G_{\chi}$. It can be proven [Jenssen2010] that the eigenvalues of $G_{\mathbf{x}}$ and $C_{\mathbf{x}}$ are the same, that $\eta = \text{rank}(G_{\mathbf{x}}) = \text{rank}(C_{\mathbf{x}}) \leq M$, and that for any $m \leq \eta$, the space spanned by MDS and the space spanned by PCA are identical, that means that one could find a rotation such that $U_{MDS} = U_{PCA}$. Additionally, MDS can also be computed even if the data matrix X is unknown, all we need is the Gram matrix, or alternatively a distance matrix. This is a situation rather common in some kind of data analysis [Cox2000].

Kernel PCA [Scholkopf1997, Scholkopf1999] is another approach trying to capture non-linear relationships. It uses a function $\mathbf{\Phi}$ transforming the input vector $\mathbf{x}$ onto a new vector $\mathbf{\Phi}(\mathbf{x})$ whose dimension is larger than M (a kind of dimensionality "expansion"). However, if we choose $\mathbf{\Phi}$ well enough, the data in this new space may become more linear (e.g., following a straight line instead of a curve). In this new space we can perform the standard PCA and obtain the feature vector χ. In Fig. 10, we show an example of the use of this multidimensional reduction method. In Fig. 10(a) it is shown a dataset (red circles) following a curved structured that cannot be described conveniently using linear PCA as the black dashed line does not describe conveniently the dataset. Therefore, to correctly describing this dataset by the standard PCA method we will need at least two principal components. In Fig. 10(b) we show the dataset after transforming it by $\mathbf{\Phi}$. As can be seen in this transformed and expanded space we can describe accurately the dataset using only one principal component as the dataset follows a linear relationship.

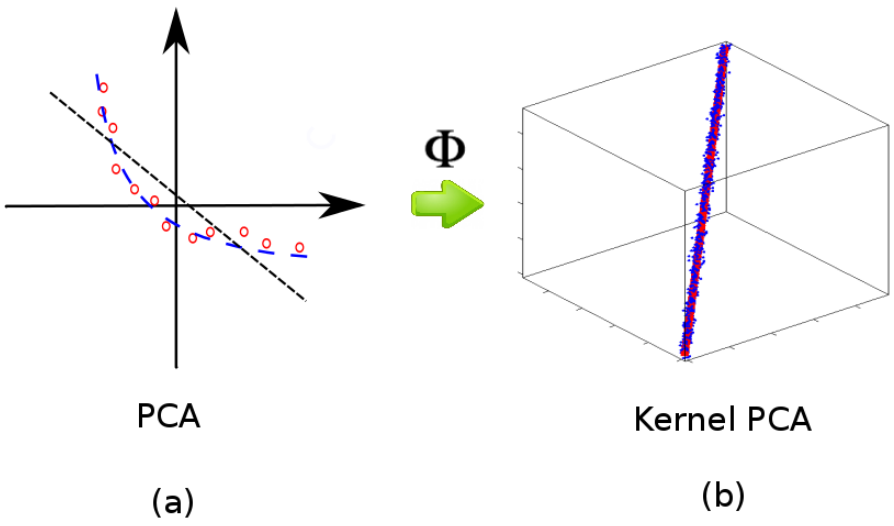

Figure 10. Input dataset (red circles) lying in a curved structure and its corresponding first principal component using standard PCA method (black dashed line) (a). Transformed dataset by function $\mathbf{\Phi}$ (blue points) and its corresponding first principal component of the expanded dataset (red line) (b)

Making use of the relationship between MDS and PCA, we do not need to compute the covariance of the $\mathbf{\Phi}$ vectors, but we can compute their inner products instead. We will do so through a Mercer kernel which defines a valid inner product in the space of $\mathbf{\Phi}$ making use of the input vectors, $\langle \mathbf{\Phi}(\mathbf{x}), \mathbf{\Phi}(\mathbf{y}) \rangle = k(\mathbf{x}, \mathbf{y})$. Common kernels

are $k(\mathbf{x},\mathbf{y}) = \langle \mathbf{x},\mathbf{y} \rangle^{\beta}$, $k(\mathbf{x},\mathbf{y}) = \exp\left(-\frac{1}{2}\left\|\frac{\mathbf{x}-\mathbf{y}}{\beta}\right\|^2\right)$, and $k(\mathbf{x},\mathbf{y}) = \tanh\left(\beta_1 \langle \mathbf{x},\mathbf{y} \rangle + \beta_2\right)$ (where the β parameters define the kernel shape). PCA vectors $\mathbf{w}_i$ are the eigenvectors of the covariance matrix of the transformed vectors $\mathbf{\Phi}(\mathbf{x})$; but these vectors are never explicitly built neither their covariance matrix. Instead, the orthogonal directions $\mathbf{w}_i$ are computed as a linear combination of the observed data, $\mathbf{w}_i = \sum_{n=1}^{N} \alpha_{in} \mathbf{x}_n = X\boldsymbol{\alpha}_i$. The $\boldsymbol{\alpha}_i$ vectors are computed as the eigenvectors of a matrix $G_{\mathbf{\Phi}}$ whose ij -th entry is the inner product $\langle \mathbf{\Phi}(\mathbf{x}_i), \mathbf{\Phi}(\mathbf{x}_j) \rangle$. The feature vectors can finally be computed as $\boldsymbol{\chi}_i = \sum_{n=1}^{N} \alpha_{in} \langle \mathbf{\Phi}(\mathbf{x}_n), \mathbf{\Phi}(\mathbf{x}) \rangle$. Obtaining the approximation of the original vector $\hat{\mathbf{x}}$ is not as straight-forward. In practice it is done by looking for a vector $\hat{\mathbf{x}}$ that minimizes $\left\|\mathbf{\Phi}(\mathbf{x}) - \mathbf{\Phi}(\hat{\mathbf{x}})\right\|^2$. The minimization is performed numerically starting from a solution specifically derived for each kernel. Again thanks to the kernel "magic", only the dot product of the transformed vectors are needed during the minimization.

All these techniques together with Locally Linear Embedding and ISOMAP (see below) are called spectral dimensionality reduction techniques because they are based on the eigenvalue decomposition of some matrix. [Bengio2006] provides an excellent review of them.

2.8 Kernel Entropy Component Analysis

We have already seen that PCA looks for directions that maximize the input variance explained by the feature vectors. Variance is a statistical second order measurement reflecting the amount of information contained by some variables (if there is no variability of the input vectors, there is no information in them). However, variance is a limited measure of information. Renyi's quadratic entropy is a more appropriate measure of the input information. Renyi's quadratic entropy is defined as $H(\mathbf{x}) = -\log E\{p(\mathbf{x})\}$, where $p(\mathbf{x})$ is the multivariate probability density function of the input vectors $\mathbf{x}$. Since the logarithm is a monotonic function, we may concentrate on its argument: the expectation of $p(\mathbf{x})$. However, the true underlying probability density function is unknown. We can approximate it through a kernel estimator $\hat{p}(\mathbf{x}) = \frac{1}{N}\sum_{n=1}^{N} k(\mathbf{x},\mathbf{x}_n)$, where $k(\mathbf{x},\mathbf{y})$ is a Parzen window (it is also required to be a Mercer kernel). If we now estimate Renyi's quadratic entropy as the sample average of the Parzen estimator, we get $E\{p(\mathbf{x})\} \approx \frac{1}{N}\sum_{n=1}^{N} \hat{p}(\mathbf{x}_n) = \frac{1}{N^2}\sum_{n_1=1}^{N}\sum_{n_2=1}^{N} k(\mathbf{x}_{n_1},\mathbf{x}_{n_2})$, which in the light of our previous discussion on Kernel PCA can be rewritten in terms of the Gram matrix of the $\mathbf{\Phi}$ vectors $E\{p(\mathbf{x})\} \approx \frac{1}{N^2}\mathbf{1}^t G_{\mathbf{\Phi}} \mathbf{1}$ (being $\mathbf{1}$ a vector of ones with dimension N). The eigendecomposition of the Gram matrix yields $E\{p(\mathbf{x})\} \approx \frac{1}{N^2}\mathbf{1}^t W_N \Lambda_N W_N^t \mathbf{1} = \frac{1}{N^2}\sum_{n=1}^{N} \lambda_n \langle \mathbf{1}, \mathbf{w}_n \rangle^2$, this means that for maximizing the information carried by the feature vectors is not enough choosing the eigenvectors with the m largest eigenvalues, but the eigenvectors with the m largest contributions to the entropy, $\lambda_n \langle \mathbf{1}, \mathbf{w}_n \rangle^2$. This is the method called Kernel Entropy Component Analysis [Jenssen2010] which can be seen to be an information-theoretic generalization of the PCA.

2.9 Robust PCA

There are many situations in which the input vectors $\mathbf{x}$ are contaminated by outliers. If this is the case, the outliers may dominate the estimation of the mean and covariance of the input data resulting in very poor performance of the PCA (see Fig. 11). There are approaches to have a robust PCA which basically amounts to have robust estimates of the mean and covariance.

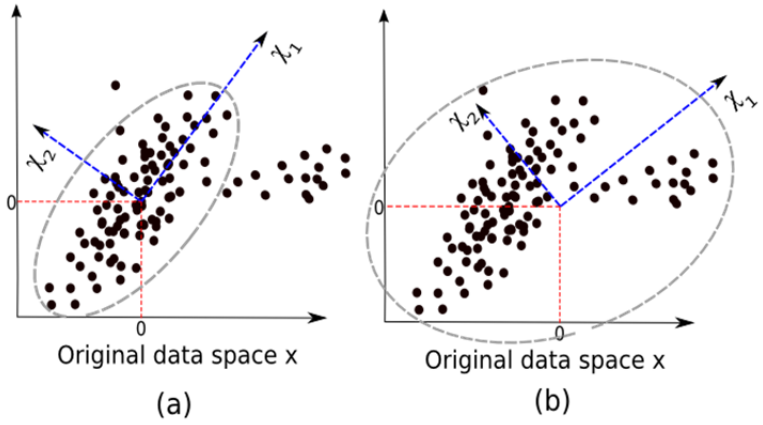

Figure 11. Comparison between Robust PCA (a) and Standard PCA (b).

An obvious modification to deal with univariate outliers is to change l_2-norm of the PCA objective function, $J_{PCA} = E\left\{\left\|\mathbf{x} - W\boldsymbol{\chi}\right\|^2\right\}$, by a l_1-norm which is known to be more robust to outliers, $J_{RPCA} = E\left\{\left\|\mathbf{x} - W\boldsymbol{\chi}\right\|_1\right\}$ [Baccini1996, Ke2005]. However, these modifications are not invariant to rotations of the input features [Ding2006]. [Ding2006] solved this problem by using the R_1 norm of a matrix that is defined as $\left\|E\right\|_{R_1} = \sum_{n=1}^{N}\left(\sum_{i=1}^{M} e_{in}^2\right)^{\frac{1}{2}}$, and constructing the objective function $J_{RPCA} = \left\|X - WW^tX\right\|_{R_1}$. Another possibility is to substitute the norm by a kernel as is done in robust statistics. This can be done at the level of individual variables $J_{RPCA} = E\left\{\sum_{i=1}^{M} k\left(\mathbf{x}_i - (W\boldsymbol{\chi})_i\right)\right\}$ (for instance [He2011] used a kernel based on the correntropy function) or at the level of the multivariate distance $J_{RPCA} = E\left\{k\left(\left\|\mathbf{x} - W\boldsymbol{\chi}\right\|^2\right)\right\}$ ([Iglesias2007] proposed to use the function $k(x) = x^{\alpha}$ with α about 0.5, they refer to this method as α-PCA; [Subbarao2006] proposed to use M-estimators, they refer to their method as pbM (projection based M-estimators)).

[Kwak2008] proposes a different approach, instead of minimizing the L_1 norm of the reconstruction error as before $J_{RPCA} = E\left\{\left\|\mathbf{x} - W\boldsymbol{\chi}\right\|_1\right\}$, he maximizes the L_1 norm of the projections $J_{RPCA} = \left\|W^tX\right\|_1$ subject to $W^tW = I$ trying to maximize the dispersion of the new features.

The approach of De la Torre [Delatorre2003] can deal with univariate outliers. It modifies the PCA objective function to explicitly account for individual components of the observations that can be regarded as outliers and to account for the possible differences in the variance of each variable. The Robust PCA goal function is then $J_{RPCA} = \sum_{n=1}^{N}\sum_{i=1}^{M} O_{ni}\rho_i(\mathbf{x}_{ni} - (\boldsymbol{\mu}_i + \mathbf{w}_i\boldsymbol{\chi}_n)) + P(O_{ni})$, where O_{ni} is a variable between 0 and 1 stating whether the i-th component of the n-th observation is an outlier (low values) or not (high values). If it is an outlier, then its error will not be counted, but the algorithm will be penalized by $P(O_{ni}) = \left(\sqrt{O_{ni}} - 1\right)^2$ to avoid the trivial solution of

considering all samples to be an outlier. This algorithm does not assume that the input data is zero valued and estimates the mean, $\boldsymbol{\mu}$, from the non-outlier samples and components. For each component, the function ρ_i robustly measures the error committed. This is done by $\rho_i(e) = \frac{e^2}{e^2 + \sigma_i^2}$, i.e., the squared error is "modulated" by the variance of that variable σ_i^2.

The approach of Hubert [Hubert2004] addresses the problem of multivariate outliers. It distinguishes the case when we have more observations than variables ($N > M$) and when we do not ($N < M$). In the first case ($N > M$), we have to specify the number h of outliers we want the algorithm to be resistant to (it has to be $h < \frac{1}{2}(N - M - 1)$). Then, we look for the subset of $N - h$ input vectors such that the determinant of its covariance matrix is minimum (this determinant is a generalization of the variance to multivariate variables: when the determinant is large it means that the corresponding dataset has a large "variance"). We compute the average and covariance of this subset and make some adjustments to account for the finite-sample effect. These estimates are called the MCD (Minimum Covariance Determinant) estimates. Finally, PCA is performed as usual on this covariance estimate. In the second case ($N < M$), we cannot proceed as before since the determinant of the covariance matrix of any subset will be zero (remind our discussion when talking about the SVD decomposition of the covariance matrix). So we first perform a dimensionality reduction without loss of information using SVD and keeping $N - 1$ variables. Next, we identify outliers by choosing a large number of random directions, and projecting the input data onto each direction. For each direction, we compute MCD estimates (robust to h outliers) of the mean ($\hat{\mu}_{MCD,\mathbf{w}}$) and standard deviation ($s_{MCD,\mathbf{w}}$) of the projections (note that these are univariate estimates). The outlyingness of each input observation is computed as $outl(\mathbf{x}_n) = \max_{\mathbf{w}} \frac{\left| \langle \mathbf{x}_n, \mathbf{w} \rangle - \hat{\mu}_{MCD,\mathbf{w}} \right|}{s_{MCD,\mathbf{w}}}$. The h points with the highest outlyingness are removed from the dataset and, finally, PCA is performed normally on the remaining points.

[Pinto2011] proposes a totally different approach. It is well known that rank-statistics is more robust to noise and outliers than the standard statistical analysis. For this reason, they propose to substitute the original observations by their ranks (the i-th component of the n-th individual, $\mathbf{x}_{ni}$, is ranked among the i-th component of all individuals, then the observation $\mathbf{x}_{ni}$ is substituted by its rank that we will refer to as $\mathbf{r}_{ni}$, and the corresponding data matrix will be referred to as R). Looking at the PCA objective function, $J_{PCA} = \mathrm{Tr}\left\{W^t \Sigma_X W\right\}$, they recognize that the covariance matrix in that equation, Σ_X, is very much related to the correlation coefficient among the different input features, the ij-th entry of this matrix is the covariance between the i-th and j-th variable. Therefore, they propose to substitute this covariance matrix by a new correlation measure better suited to handle rank data. They refer to the approach as Weighted PCA. The objective function is finally $J_{PCA} = \mathrm{Tr}\left\{W^t \Sigma_R W\right\}$ subject to $W^t W = I$.

2.10 Factor Analysis

Factor analysis (FA) [Spearman1904, Thurstone1947, Kaiser1960, Lawley1971, Mulaik1971, Harman1976] is another statistical technique intimately related to PCA and dimensionality reduction. FA is a generative model that assumes that the observed data has been produced from a set of latent, unobserved variables (called factors) through the equation $\mathbf{x} = W\boldsymbol{\chi} + \mathbf{n}$ (if there is no noise, this model is the same as in PCA, although PCA is not a generative model in its conception). In this technique all the variances of the factors are absorbed into W such that the covariance of $\boldsymbol{\chi}$ is the identity matrix. Factors are assumed to follow a multivariate normal distribution, and to be uncorrelated to noise. Under these conditions, the covariance of the observed variables is $C_{\mathbf{x}} = WW^t + C_{\mathbf{n}}$ where $C_{\mathbf{n}}$ is the covariance matrix of the noise and has to be estimated from the data. Matrix W

is solved by factorization of the matrix $WW^t = C_{\mathbf{x}} - C_{\mathbf{n}}$. This factorization is not unique since any orthogonal rotation of W results in the same decomposition of $C_{\mathbf{x}} - C_{\mathbf{n}}$ [Kaiser1958]. This fact, rather than a drawback, is exploited to produce simpler factors in the same way as the PCA was rotated (actually the rotation methods for FA are the same as the ones for PCA).

2.11 Independent Component Analysis

Independent Component Analysis (ICA) [Common1994,Hyvarinen1999,Hyvarinen2000,Hyvarinen2001] is an example of information-theory based algorithm. It is also a generative model with equation $\mathbf{x} = W\boldsymbol{\chi}$. While PCA looks for uncorrelated factors (i.e., a constraint on the second-order statistics), ICA looks for independent factors (i.e., a constraint on all their moments, not only second-order). This is an advantage if the factors are truly independent (for two Gaussian variables decorrelation implies independence but this is not generally true for variables with any other distribution). In the language of ICA, the $\boldsymbol{\chi}$ vectors are called the sources, while the $\mathbf{x}$ are called the observations. Matrix W is called the mixing matrix and the problem is formulated as one of source separation, i.e., finding an unmixing matrix $\tilde{W}$ such that the components of $\hat{\boldsymbol{\chi}} = \tilde{W}^t\mathbf{x}$ are as independent as possible (see Fig. 12). The sources can be recovered up to a permutation (the sources are recovered in different order) and a scale change (the sources are recovered with different scale). A limitation of the technique is that usually sources are supposed to be non-Gaussian since the linear combination of two Gaussian variables is also Gaussian making the separation an ill-posed problem.

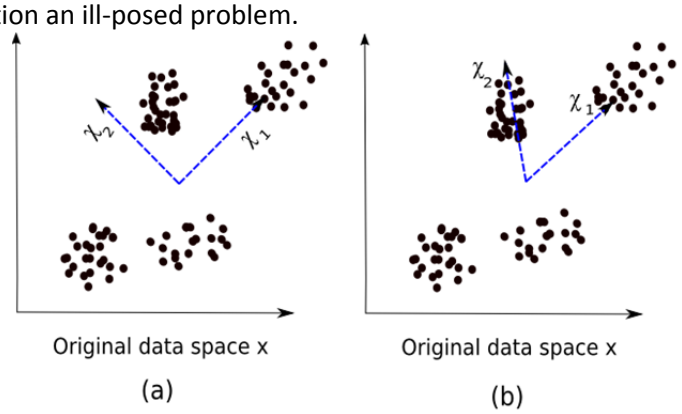

Figure 12. a) Example of PCA results for a given input distribution. b) ICA results for the same distribution.

Different ICA methods differ in the way they measure the independence of the estimates of the source variables, resulting in different estimates of the mixing matrix W and source variables $\hat{\boldsymbol{\chi}}$. The following are different options commonly adopted:

- Non-Gaussianity: The central limit theorem states that the distribution of the weighted sum of the sources tends to be normally distributed whichever the distributions of the original sources. Thus, a possible way to achieve the separation of the sources is by looking for transformations $\tilde{W}$ that maximize the kurtosis of each of the components of the vector $\hat{\boldsymbol{\chi}}$ [Hyvarinen2001] (see Fig. 13). The kurtosis is related to the third order moment of the distribution. The kurtosis of the Gaussian is zero and, thus, maximizing the kurtosis, minimizes the Gaussianity of the output variables. In fact, maximizing the kurtosis can be seen as a Non-linear PCA problem (see above) with the non-linear function for each feature vector $f_i(\chi_i) = \chi_i + \mathrm{sgn}(\chi_i)\chi_i^2$. FastICA is an algorithm based on this goal function. The problem of kurtosis is that it can be very sensitive to outliers. Alternatively, we can measure the non-Gaussianity by negentropy which is the Kullack-Leibler divergence between the multivariate distribution of the estimated sources $\hat{\boldsymbol{\chi}}$, and the distribution of a multivariate variable $\boldsymbol{\chi}_G$ of the same mean and covariance matrix as $\hat{\boldsymbol{\chi}}$. Projection pursuit

[Friedman1974, Friedman1987] is an exploratory data analysis algorithm looking for projection directions of maximum kurtosis (as in our first ICA algorithm) while Exploratory Projection Pursuit [Girolami1997] uses the maximum negentropy to look for the projection directions. Non-Gaussian Component Analysis [Theis2011], instead of looking for a single direction like in projection pursuit, looks for an entire linear subspace where the projected data is as non-Gaussian as possible.

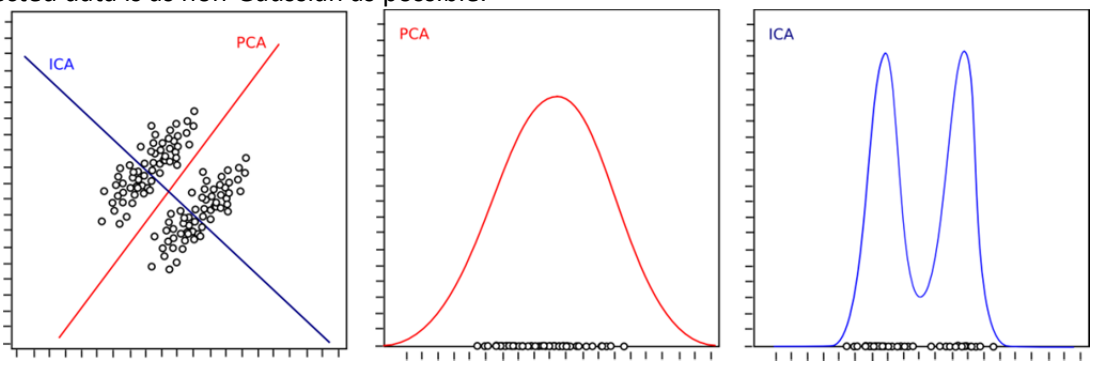

Figure 13. a) Sample distribution in which the first PCA component maximizes the explained variance but projections onto this direction are normally distributed (b). ICA first component is also shown on the sample data, data projection onto this direction clearly shows a non-Gaussian distribution.

- Maximum-likelihood (ML): Let us assume that we know the *a priori* distribution of each of the components $\hat{\chi}$. Then, we could find the matrix $\tilde{W}$ simply by maximizing the likelihood of all the observations $l(\mathbf{x}) = p_{\mathbf{x}}(\mathbf{x}) = \left|\det \tilde{W}\right| p_{\chi}(\hat{\chi}) = \left|\det \tilde{W}\right| \prod_{i=1}^{m} p_{\chi_i}(\hat{\chi}_i)$. Taking logarithms and the expected value over all input vectors we obtain $L(X) = \log\left|\det \tilde{W}\right| + E\left\{\sum_{i=1}^{m} p_{\chi_i}(\hat{\chi}_i))\right\}$ which is the objective function to maximize with respect to $\tilde{W}$ (the Bell-Sejnowski and the natural gradient algorithms [Hyvarinen2001] are typical algorithms for performing this optimization). If the distribution of the features is not known *a priori*, it can be shown [Hyvarinen2001] that "reasonable" errors in the estimation of the p_{χ_i} distributions result into locally consistent ML estimators as long as for all i $E\left\{\chi_i g_i(\chi_i) - g_i'(\chi_i)\right\} > 0$, where $g_i(\chi_i) = \frac{\tilde{p}'_{\chi_i}(\chi_i)}{\tilde{p}_{\chi_i}(\chi_i)}$ and $\tilde{p}_{\chi_i}$ is our estimate of the truly underlying distribution p_{χ_i}. A common approach is to choose for each i between a super-Gaussian and a sub-Gaussian distribution. This choice is done by checking which one of the two fulfills $E\left\{\chi_i g_i(\chi_i) - g_i'(\chi_i)\right\} > 0$. Common choices are $g_i(\chi_i) = \chi_i + \tanh(\chi_i)$ for the super-Gaussian and $g_i(\chi_i) = \chi_i - \tanh(\chi_i)$ for the sub-Gaussian. This criterion is strongly related to the Infomax (Information Maximization) in neural networks. There, the problem is to look for the network weights such that the joint entropy of the output variables, $H(\chi_1, \chi_2, ..., \chi_m)$, is maximized. It has been shown [Lee2000] that the Infomax criterion is equivalent to the maximum likelihood one when $g_i(\chi_i)$ is the non-linear function of each output neuron of the neural network (usually a sigmoid). Equivalently, instead of maximizing the joint entropy of the feature variables, we could have minimized their mutual information. Mutual information is a measure of the dependency among a set of variables. So, it can be easily seen how all these criteria is maximizing the independence of the feature vectors.
- Nonlinear decorrelation: two variables χ_1 and χ_2 are independent if for all continuous functions f_1 and f_2 with finite support we have $E\{f_1(\chi_1) f_2(\chi_2)\} = E\{f_1(\chi_1)\} E\{f_2(\chi_2)\}$. The two variables are non-linearly decorrelated if $E\{f_1(\chi_1) f_2(\chi_2)\} = 0$. In fact, PCA looks for output variables that are second-order decorrelated, $E\{\chi_1 \chi_2\} = 0$, although they may not be independent because of their higher-order

moments. Making the Taylor expansion of $f_1(\chi_1) f_2(\chi_2)$ we arrive to $E\{f_1(\chi_1) f_2(\chi_2)\} = \sum_{i,j=0}^{\infty} c_{ij} E\{\chi_1^i \chi_2^j\} = 0$. For which we need that all high-order correlations are zero, $E\{\chi_1^i \chi_2^j\} = 0$. Hérault-Jutten and Cichocki-Unbehauen algorithms look for independent components by making specific choices of the non-linear functions f_1 and f_2 [Hyvarinen2001].

3. Methods based on Dictionaries

Another family of methods is based on the decomposition of a matrix formed by all input data as columns, X. The input data matrix using the input variables is transformed into a new data matrix using the new variables. The transformation is nothing but a linear change of basis between the two variable sets. In the field of dimensionality reduction, the matrix expressing the change of basis is known as a dictionary (dictionary elements are called atoms) and there are several ways of producing such a dictionary [Rubinstein2010]. We have already seen Singular Value Decomposition (SVD) and its strong relationship to PCA. Vector Quantization (K-means) can also be considered an extreme case of dictionary based algorithm (input vectors are represented by a single atom, instead of as a combination of several atoms). Under this section we will explore other dictionary based algorithms.

3.1 Non-negative Matrix Factorization

A drawback of many dimensionality reduction techniques is that they produce feature vectors with negative components. In some applications like text analysis, it is natural to think of the observed vectors as the weighted sum of some underlying "factors" with no subtraction involved (for instance, it is natural to think that if a scientific article is about two related topics, the word frequencies of the article will be a weighted sum (with positive weights) of the word frequencies of the two topics). Let us consider the standard decomposition dictionary $X = WU$ where W is the dictionary (of size $M \times m$, its columns are called "atoms" of the dictionary) and U (of size $m \times N$) is the expression of the observations in the subspace spanned by the dictionary atoms (see Fig. 14). Not considering subtractions imply constraining the feature vectors to be $U > 0$.

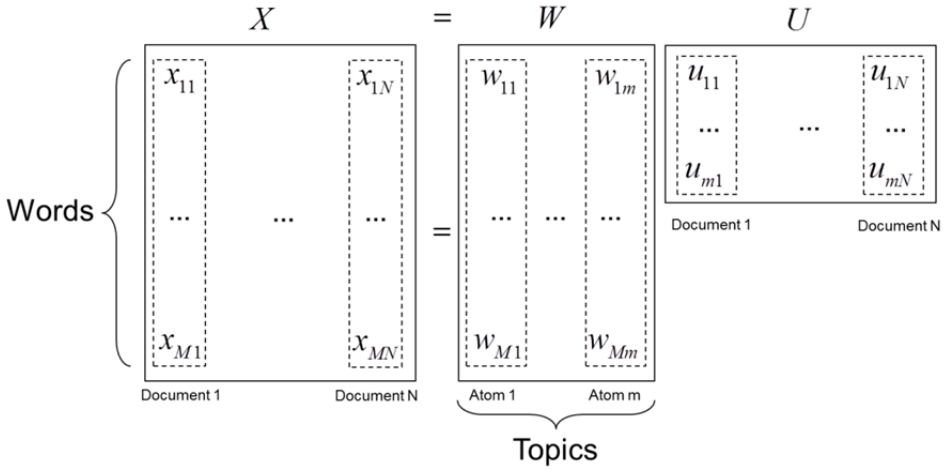

Figure 14. Dictionary decomposition of a set of documents (see Fig. 3). Each document is decomposed as the linear combination given by the weights in U of the topics (atoms) contained in W.

If X is made only of positive values, it might be interesting to constrain the atoms to be positive as well ($W > 0$). This is the problem solved by Non-negative Matrix Factorization [Lee1999, Lee2001]. The goal is to minimize

$\|X - WU\|_F^2$ or $D(X\|WU)$ (defined as $D(A\|B) = \sum_{i,j}\left(A_{ij}\log\frac{A_{ij}}{B_{ij}} - A_{ij} + B_{ij}\right)$; this is the Kullback-Leibler divergence if A and B are normalized so that they can be regarded as probability distributions) subject to $W,U > 0$. The advantage of this decomposition is that, if the application is naturally defined with positive values, the dictionary atoms are much more understandable and related to the problem than the standard dimensionality reduction methods. [Sandler2011] proposed to minimize the Earth Mover's Distance between the matrices X and WU with the aim of making the method more robust, especially to small samples. The Earth's Mover Distance, also called Wasserstein metric, is a way of measuring distances between two probability distributions (for a review on how to measure distances between probability distributions see [Rubner2000]). It is defined as the minimum cost of turning one probability distribution into the other and it is computed through a transportation problem. This distance was extended by [Sandler2011] to measure the distance between matrices by applying the distance to each column (feature) of the matrices and then summing all distances.

In the recent years there is much interest in the construction of sparse feature vectors, sparse dictionaries or both. The underlying idea is to produce feature vectors with as many zeroes as possible, or what is the same, approximating the observations with as few dictionary atoms as possible. This has obvious advantages when trying to explain the "atomic composition" of a given observation. In the following paragraphs we will review some of the approaches already proposed for NMF in this direction.

Local NMF [Feng2002] enforces sparsity by adding to the NMF goal function the term $\|W^tW\|_1$ (which promotes the orthogonality of the atoms, i.e., minimizes the overlapping between atoms) and $-\|U\|_2^2$ (which maximizes the variance of the feature vectors, i.e., it favours the existence of large components). Non-negative sparse coding [Hoyer2002] and Sparse NMF [Liu2003] add the term $\|U\|_1$ in order to minimize the number of features with significant values. Pauca *et al.* [Pauca2006] regularize by adding $\|U\|_2^2$ and $\|W\|_2^2$ (this is especially suited for noisy data). NMF with Sparseness Constraints [Hoyer2004] performs the NMF with the constraint that the sparseness of each column of W is a given constant S_w (i.e., it promotes the sparseness of the dictionary atoms) and the sparseness of each row of U is another constant S_U (i.e., it promotes that each atom is used in as few feature vectors as possible). In [Hoyer2004], the sparseness of a vector is defined as $Sparseness(\mathbf{x}) = \frac{1}{\sqrt{n}-1}\left(\sqrt{n} - \frac{\|\mathbf{x}\|_1}{\|\mathbf{x}\|_2}\right)$ that measures how much energy of the vector is packed in as few components as possible. This function evaluates to 1 if there is a single non-zero component, and evaluates to 0 if all the elements are identical. Non-smooth NMF [Pascual-Montano2006] modifies the NMF factorization to $X = WS_\lambda U$. The matrix S_λ controls the sparseness of the solution through the parameter λ. It is defined as $S_\lambda = (1-\lambda)I + \lambda\frac{1}{m}\mathbf{1}\mathbf{1}^t$. For $\lambda = 0$ it is the standard NMF. For $\lambda = 1$, we can think of the algorithm as using "effective" feature vectors defined by $U_\lambda = S_\lambda U$ that substitute each feature vector $\boldsymbol{\chi}$ by a vector of the same dimensionality whose all components are equal to the mean of $\boldsymbol{\chi}$. This is just imposing non-sparseness on the feature vectors, and this will promote sparseness on the dictionary atoms. On the other hand, we could have also thought of the algorithm as using the "effective" atoms given by $W_\lambda = WS_\lambda$ that substitute each atom by the average of all atoms. In this case, the non-sparseness of the dictionary atoms will promote sparseness of the feature vectors. Non-smooth NMF is used with typical λ values about 0.5.

Another flavor of NMF enforces learning the local manifold structure of the input data [Cai2011b], Graph-regularized NMF (GNMF). Assuming that the input vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ are close in the original space, one might like that the reduced representations, $\boldsymbol{\chi}_i$ and $\boldsymbol{\chi}_j$, are also close. For doing so, the algorithm constructs a graph G encoding the neighbors of the input observations. Observations are represented by nodes in the graph, and two nodes are connected by an edge if their distance is smaller than a given threshold and they are in the K-neighbors

list of each other. The weight of each edge is 1, or if we prefer we can assign a different weight to each edge depending on the distance between the two points (for instance, $e^{-\frac{1}{2}\left\|\mathbf{x}_i-\mathbf{x}_j\right\|^2}$). We build the diagonal matrix D whose elements are the row sums of G. The Laplacian of this graph is defined as $L=D-G$. The sum of the Euclidean distances of the reduced representations corresponding to all neighbor pairs can be computed as $\mathrm{Tr}\left\{ULU^t\right\}$ [Cai2011b]. In this way, the GNMF objective function becomes $\left\|X-WU\right\|_F^2+\lambda\,\mathrm{Tr}\left\{ULU^t\right\}$. This algorithm has the corresponding version in case that the Kullback-Leibler divergence is preferred over Euclidean distances [Cai2011b].

3.2 Principal Tensor Analysis and Non-negative Tensor Factorization

In some situations the data is better represented by a multidimensional array rather than by a matrix. For instance, we might have a table representing the gene expression level for a number of drugs. It is naturally represented by a (drug, gene) matrix and all the previous methods to factorize matrices are applicable. However, we might have a (drug, gene, time) table that specifies the gene expression for each combination of gene, drug and time. This three-way table (and in general multiway tables) is a tensor (strictly speaking a multiway table of dimension d is a tensor if and only if it can be decomposed as the outer product of d vectors; however, in the literature multiway tables are usually referred to as tensors and we will also adhere here to this loose definition). Tensors can be flattened into matrices and then all the previous techniques would be available. However, the locality imposed by some variables (like time or spatial location) would be lost. Non-negative tensor factorization [Cichocki2009] is an extension of NMF to multiway tables. The objective is, as usual, minimizing the representation error $\left\|X-\sum_{i=1}^{m}\mathbf{w}_i^1\otimes\mathbf{w}_i^2\otimes\ldots\otimes\mathbf{w}_i^d\right\|_F^2=\left\|X-\sum_{i=1}^{m}\bigotimes_{j=1}^{d}\mathbf{w}_i^j\right\|_F^2$ subject to $\mathbf{w}_i^j\geq 0$ for all i and j. In the previous expression X is a tensor of dimension d (in our three-way table example, $d=3$), $\otimes$ represents the outer product, m is a parameter controlling the dimensionality reduction. For each dimension j (drug, gene or time, in our example), there will be m associated vectors $\mathbf{w}_i^j$. The length of each vector depends on the dimension it is associated to (see Fig. 15). If there are N_j elements in the j-th dimension (N_{drugs}, N_{genes} and N_{time} entries in our example), the length of the vectors $\mathbf{w}_i^j$ is N_j. The approximation after dimensionality reduction is $\hat{X}=\sum_{i=1}^{m}\bigotimes_{j=1}^{d}\mathbf{w}_i^j$. For a three-way table, the pqr element of the tensor is given by $\hat{X}_{pqr}=\sum_{i=1}^{m}\mathbf{w}_{ip}^1\mathbf{w}_{iq}^2\mathbf{w}_{ir}^3$, where $\mathbf{w}_{ip}^1$ represents the p-th component of the vector $\mathbf{w}_i^1$ (analogously with $\mathbf{w}_{iq}^2$ and $\mathbf{w}_{ir}^3$).

NTF is a particular case of a family of algorithms decomposing tensors. PARAFAC [Harshman1970, Bro1997] may be one of the first algorithms doing so and can be regarded as a generalization of SVD to tensors. The SVD approximation $\hat{X}=W_mD_mU_m$ can be rewritten as $\hat{X}=\sum_{i=1}^{m}d_i\mathbf{w}_i\mathbf{u}_i^t=\sum_{i=1}^{m}d_i\mathbf{w}_i\otimes\mathbf{u}_i$. PARAFAC model is minimizing the approximation error $J_{PARAFAC}=\left\|X-\sum_{i=1}^{m}d_i\mathbf{w}_i^1\otimes\mathbf{w}_i^2\otimes\ldots\otimes\mathbf{w}_i^d\right\|_F^2$. We can enrich the model to consider more outer products as in $J_{Tucker}=\left\|X-\sum_{i_1=1}^{m_1}\sum_{i_2=1}^{m_2}\ldots\sum_{i_d=1}^{m_d}d_{i_1i_2\ldots i_d}\mathbf{w}_{i_1}^1\otimes\mathbf{w}_{i_2}^2\otimes\ldots\otimes\mathbf{w}_{i_d}^d\right\|_F^2$. This is called the Tucker model [Tucker1966].

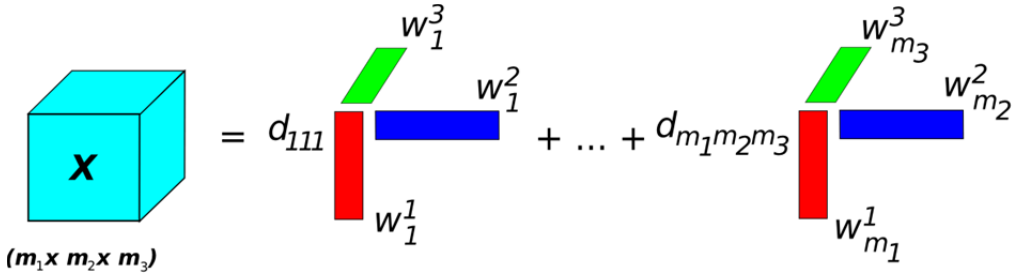

Figure 15. Tensor decomposition as the sum of a series of outer vector products. The figure corresponds to the more general Tucker model, which is simplified to PARAFAC or NTF.

3.3 Generalized SVD

According to the SVD decomposition, $X = WDU$, the columns of W are the eigenvectors of the matrix XX^t, while the rows of U are the eigenvectors of the matrix X^tX. Eigenvectors are orthogonal and therefore $W^tW = I$ and $UU^t = I$. After performing the dimensionality reduction, the approximation $\hat{X} = W_m D_m U_m$ is the matrix minimizing the representation error $\varepsilon = \left\| X - \hat{X} \right\|^2 = trace\left\{ (X - \hat{X})(X - \hat{X})^t \right\}$.

Generalized SVD [Paige1981] performs a similar decomposition but relaxes, or modifies, the orthogonality conditions according to two constrain matrices, C_W and C_U, such that $W^tC_WW = I$ and $UC_UU^t = I$ [Abdi2007]. After the dimensionality reduction, the approximation matrix is the one minimizing $\varepsilon = trace\left\{ C_W(X - \hat{X})C_U(X - \hat{X})^t \right\}$.

Generalized SVD is a very versatile tool since under the appropriate choices of the constrain matrices it can be particularized to correspondence analysis (a generalization of factor analysis for categorical variables, C_W is the relative frequency of the rows of the data matrix, X, and C_U is the relative frequencies of its rows), discriminant analysis (a technique relating a set of continuous variables to a categorical variable), and canonical correlation analysis (a technique analyzing two groups of continuous variables and performing simultaneously two dimensionality reductions so that the two new sets of features have maximum cross correlation) [Abdi2007].

3.4 Sparse representations and overcomplete dictionaries

A different approach to dimensionality reduction is by cutting the input signal into small pieces, and performing a dimensionality reduction of them. For instance, we can divide an image into small 8x8 pieces (vectors of dimension 64). Then we try to express each piece as a linear combination of a few atoms from a large dictionary (of size larger than 64, that is why it is called overcomplete). At the level of pieces, the dictionary acts as a dimensionality expansion, although overall, there is a dimensionality reduction since each piece can be represented with just a few atoms instead of the 64 values needed originally. This approach can be applied to any domain where the original vectors $\mathbf{x}$ can be decomposed into pieces of similar nature: images and time series are good examples. Let us call $\hat{\mathbf{x}}$ to each one of these pieces. The idea is that for each piece we solve the problem $\min \left\| \hat{\boldsymbol{\chi}} \right\|_0$ subject to $\left\| \hat{\mathbf{x}} - \widehat{W}\hat{\boldsymbol{\chi}} \right\|_2 \le \varepsilon$ (by setting $\varepsilon = 0$ we require exact representation of the original vector). The columns of $\widehat{W}$ are the atoms, and the feature vector $\hat{\boldsymbol{\chi}}$ define the specific linear combination of atoms used to represent the corresponding piece. The sparseness of the feature vector is measured simply by counting the number of elements different from zero (this is usually referred to as the l_0 norm, although actually it is not a

norm since it is not positive homogeneous). The problem of the l_0 norm is that it yields non-convex optimization problems whose solution is NP-complete. For tackling this problem some authors have replaced the l_0 by the l_p norm (for $0 < p < 1$ the problem is still non-convex, although there are efficient algorithms; $p = 1$ is a very popular norm to promote sparseness). Related problems are $\min \left\| \hat{\mathbf{x}} - \hat{W}\hat{\boldsymbol{\chi}} \right\|_2 + \lambda \left\| \hat{\boldsymbol{\chi}} \right\|_p$ (the Least Absolute Shrinkage and Selector Operator (LASSO) is such a problem with $p = 1$, and ridge regression is also this problem with $p = 2$) and $\min \left\| \hat{\mathbf{x}} - \hat{W}\hat{\boldsymbol{\chi}} \right\|_2$ subject to $\left\| \hat{\boldsymbol{\chi}} \right\|_p \leq t$ where t is a user-defined value (for $p = 0$ it restricts the feature vector to use at most t atoms). The previous problems are called the sparse coding step and many algorithms have been devised for its solution. The most popular ones are basis pursuit [Chen1994, Chen2001], matching pursuit [Mallat1993], orthogonal matching pursuit [Pati1993, Tropp2007], orthogonal least squares [Chen1991], focal underdetermined system solver (FOCUSS) [Gorodnitsky1997], gradient pursuit [Blumensath2008] and conjugate gradient pursuit [Blumensath2008]. For a review on these techniques, please, see [Bruckstein2009].

The other problem is how to learn the overcomplete dictionary $\hat{W}$ from the input pieces $\hat{\mathbf{x}}$. In a way, this can be considered as an extension of the vector quantization problem. In vector quantization we look for a set of class averages minimizing the representation error when all except one of the feature vector components are zero (the value of the non-null χ_i component is 1). Now, we have relaxed this condition and we allow the feature components to be real valued (instead of 0 or 1) and we represent our data by a weighted sum of a few atoms. Nearly all methods iteratively alternate between the estimation of the feature vectors and the estimation of the dictionary, and they differ in the goal function being optimized, which ultimately result into different update equations for the dictionary. The Method of Optimal Directions (MOD) [Engan2000] is a possible way of learning dictionaries. This method optimizes $\hat{W}^*, \hat{U}^* = \arg\min_{W,U} \left\| \hat{X} - \hat{W}\hat{U} \right\|_F^2$ and it has proven to be very efficient. Another possibility to learn the dictionary is by Maximum Likelihood [Lewicki2000]. Under a generative model it is assumed that the observations have been generated as noisy versions of a linear combination of atoms $\hat{\mathbf{x}} = \hat{W}\hat{\boldsymbol{\chi}} + \boldsymbol{\varepsilon}$. The observations are assumed to be produced independently, the noise to be white and Gaussian, and the *a priori* distribution of the feature vectors to be Cauchy or Laplacian. Under these assumptions the problem of maximizing the likelihood of observing all pieces, $\hat{\mathbf{x}}_n$, given $\hat{W}$ is maximized by $\hat{W}^* = \arg\min_{\hat{W}} \sum_n \min_{\boldsymbol{\chi}_n} \left\{ \left\| \hat{\mathbf{x}}_n - \hat{W}\hat{\boldsymbol{\chi}}_n \right\|_2^2 + \lambda \left\| \hat{\boldsymbol{\chi}}_n \right\|_1 \right\}$. If a prior distribution of the dictionary is available we can use a Bayesian approach (Maximum *a posteriori*) [Kreutz2003]. K-SVD [Aharon2006] solves the problem $\hat{W}^*, \hat{U}^* = \arg\min_{\hat{W},\hat{U}} \left\| \hat{X} - \hat{W}\hat{U} \right\|_F^2$ subject to $\left\| \hat{\boldsymbol{\chi}}_n \right\|_1 \leq t$ for all the pieces, n. It is conceived as generalization of the K-means algorithm, and in the update of the dictionary there is a Singular Value Decomposition (therefore, its name).

K-SVD can be integrated into a larger framework capable of optimally reconstructing the original vector from its pieces. The goal function is $\lambda \left\| \mathbf{x} - \hat{\mathbf{x}} \right\|_2^2 + \sum_n \left(\lambda_n \left\| \hat{\boldsymbol{\chi}}_n \right\|_0 + \left\| \hat{\mathbf{x}}_n - \hat{W}\hat{\boldsymbol{\chi}}_n \right\|_2^2 \right)$, where the λ_n variables are Lagrangian multipliers. Once all the patches have been approximated by their corresponding feature vectors, we can recover the original input vector by $\hat{\mathbf{x}} = \left(\lambda I + \sum_n P_n^t P_n \right)^{-1} \left(\lambda \mathbf{x} + \sum_n P_n^t \hat{W}\hat{\boldsymbol{\chi}}_n \right)$, where P_n is an operator extracting the n-th piece as a vector, and P_n^t is the operator putting it back in its original position.

Tensor representations can be coupled to sparse representations as shown by [Gurumoorthy2010] which we will refer to as Sparse Tensor SVD. In certain situations, the input data is better represented by a matrix or tensor than by a vector. For instance, image patches can be represented as a vector by lexicographically ordering the pixel values. However, this representation spoils the spatial correlation of nearby pixels. Let us then consider that we no

longer have input vectors, $\mathbf{x}_n$, but input matrices (we will generalize later to tensors). This method learns a dictionary of K SVD-like basis (W_i, U_i). Each input matrix, X_n, is sparsely represented in the i-th SVD-like basis, $X_n \approx W_i D_n U_i$. Sparsity is measured through the number of non-zero components of the representation $\|D_n\|_0$. Finally, a membership matrix, $\hat{u}_{in} \in (0,1)$, specifies between 0 (no membership) and 1 (full membership) whether the input matrix X_n is represented with the i-th basis or not. In this way, the goal of this algorithm is the minimization of the representation error given by $J_{SparseTensorSVD} = \sum_{n=1}^{N}\sum_{i=1}^{K} \hat{u}_{in} \|X_n - W_i D_n U_i\|_F^2$. This functional is minimized with respect to the SVD-like basis, the membership function and the sparse representations. The optimization is constrained by the orthogonality of the basis ($W_i^t W_i = I$, $U_i U_i^t = I$, for all i), the sparsity constraints ($\|D_n\|_0 \leq t$, t is a user-defined threshold), and that the columns of the membership matrix define a probability distribution ($\sum_{i=1}^{K} \hat{u}_{in} = 1$). The generalization of this approach to tensors is straightforward simply replacing the objective function by $J_{SparseTensorSVD} = \sum_{n=1}^{N}\sum_{i=1}^{K} \hat{u}_{in} \left\| X_n - U_n \otimes \left(\bigotimes_{j=1}^{d} \mathbf{w}_i^j \right) \right\|_F^2$ subject to the same orthogonality, sparsity and membership constraints.

4. Methods based on projections

A different family of algorithms poses the dimensionality reduction problem as one of projecting the original data onto a subspace with some interesting properties.

4.1 Projection onto interesting directions

Projection pursuit defines the output subspace by looking for "interesting" directions. What is "interesting" depends on the specific problem but usually directions in which the projected values are non-Gaussian are considered to be interesting. Projection pursuit looks for directions maximizing the kurtosis of the projected values as a measure of non-Gaussianity. This algorithm was visited during our review of Independent Component Analysis and presented as a special case of that family of techniques.

All the techniques presented so far are relatively costly in computational terms. Depending on the application it might be enough to reduce the dimensionality without optimizing any goal function but in a very fast way. Most techniques project the observations $\mathbf{x}$ onto the subspace spanned by a set of orthogonal vectors. However, choosing the best (in some sense) orthogonal vectors is what is computationally costly while the projection itself is rather quick. In certain application domains some "preconceived" directions are known. This is the case of the Discrete Cosine Transform (DCT) used in the image standard JPEG [Watson1993]. The "cosine" vectors usually yield good reduction results with low representation error for signals and images. Many other transform-based compression methods, like wavelets, also fall under this category. Alternatively, random mapping [Kaski1998, Dasgupta2000, Bingham2001] solves this problem by choosing zero-mean random vectors as the "interesting" directions onto which project the original observations (this amounts to simply taking a random matrix W whose columns are normalized to have unit module). Random vectors are orthogonal in theory ($E\left\{\left\langle \mathbf{w}_i, \mathbf{w}_j \right\rangle\right\} = 0$), and nearly orthogonal in practice. Therefore, the dot product between any pair of observations is nearly conserved in the feature space. This is a rather interesting property since in many applications the similarity between two observations is computed through the dot product of the corresponding vectors. In this way, these similarities are

nearly conserved in the feature space but at a computational cost that is only a small fraction of the cost of most dimensionality reduction techniques.

4.2 Projection onto manifolds

Instead of projecting the input data onto an "intelligent" set of directions, we might look for a manifold close to the data, project onto that manifold and unfold the manifold for representation. This family of algorithms is represented by a number of methods like Sammon projection, Multidimensional Scaling, Isomap, Laplacian eigenmaps, and Local linear embedding. All these algorithms can be shown to be special cases of Kernel PCA under certain circumstances [Williams2002, Bengio2004]. The reader is also referred to [Cayton2005] for an excellent review of manifold learning.

We already saw MDS as a technique preserving the inner product of the input vectors. Alternatively, it could be also seen as a technique preserving distances in the original space. From this point of view, Sammon projection [Sammon1969] and MDS [Kruskal1986] look for a projected set of points, $\boldsymbol{\chi}$, such that distances in the output subspace are as close as possible to the distances in the original space. Let $D_{n_1 n_2} = d(\mathbf{x}_{n_1}, \mathbf{x}_{n_2})$ be the distances between any pair of observations in the original space (for an extensive review of distances see [Ramanan2011]). Let $d_{n_1 n_2} = d(\boldsymbol{\chi}_{n_1}, \boldsymbol{\chi}_{n_2})$ be the distance between their feature vectors, and $\hat{D}_{n_1 n_2} = d(\hat{\mathbf{x}}_{n_1}, \hat{\mathbf{x}}_{n_2})$ the distance between their approximations after dimensionality reduction. Classical MDS look for the feature vectors minimizing $\sum_{n_1,n_2} \left(D_{n_1,n_2} - \hat{D}_{n_1,n_2} \right)^2$, while Sammon projection minimizes $\frac{\sum_{n_1,n_2} \omega_{n_1,n_2} \left(D_{n_1,n_2} - d_{n_1,n_2} \right)^2}{\sum_{n_1,n_2} D_{n_1,n_2}^2}$ with $\omega_{n_1,n_2} = \frac{1}{D_{n_1,n_2}}$ (this goal function is called the stress function and it is defined between 0 and 1) (see Fig. 16). Classical MDS solves the problem in a single step involving the computation of the eigenvalues of a Gram matrix involving the distances in the original space. In fact, it can be proved that classical MDS is equivalent to PCA [Williams2002]. Sammon projection uses a gradient descent algorithm and can be shown [Williams2002] to be equivalent to Kernel PCA. Metric MDS modifies the $D_{n_1 n_2}$ distances by an increasing, monotonic nonlinear function $f(D_{n_1 n_2})$. This modification can be part of the algorithm itself (some proposals are [Bengio2004] $f(D_{n_1 n_2}) = D_{n_1 n_2}^{\alpha}$ and $f(D_{n_1 n_2}) = \frac{1}{2}\left(D_{n_1 n_2} - D_{n_1 \cdot} - D_{\cdot n_2} + D_{\cdot\cdot} \right)$ where $D_{n_1 \cdot}$ and $D_{\cdot n_2}$ denote the average distance of observations n_1 and n_2 to the rest of observations, and $D_{\cdot\cdot}$ is the average distance between all observations) or as part of the data collection process (e.g. distances have not been directly observed but are asked to a number of human observers). If the distances $D_{n_1 n_2}$ are measured as geodesic distances (the geodesic distance between two points in a manifold is the one measured along the manifold itself; in practical terms it is computed as the shortest path in a neighborhood graph connecting each observation to its K-nearest neighbors, see Fig. 17), then the MDS method is called Isomap [Tenenbaum2000].

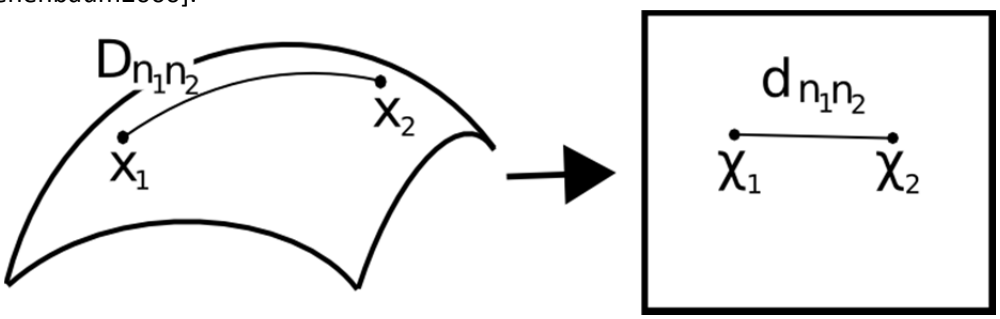

Fig.16. Distances in the original space are mapped onto the lowest dimensional space trying to find projection points that keep the set of distances as faithful as possible.

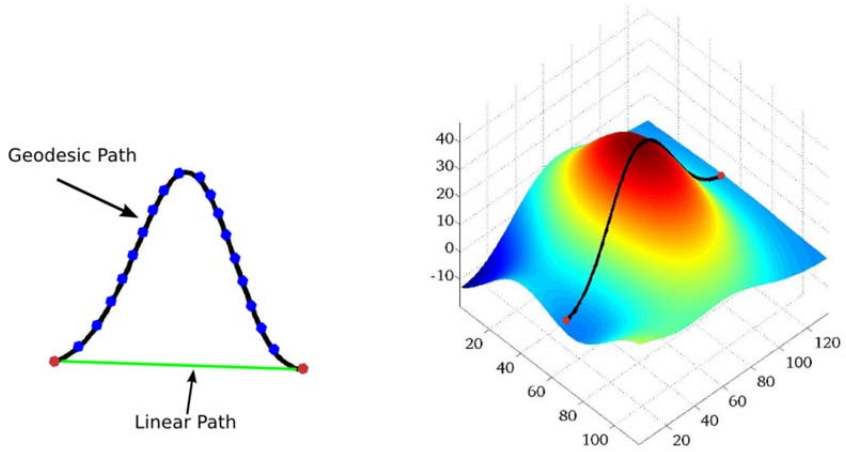

Figure 17. Geodesic versus Euclidean distance. The geodesic distance between two points is the length of the path belonging to a given manifold that joins the two points, while the Euclidean distance is the length of the linear path joining the two points.

Laplacian eigenmaps [Belkin2001, Belkin2002] start from an adjacency graph similar to that of Isomap for the computation of the geodesic distances. The neighbor similarity graph G is calculated as was done in GNMF. Then, the generalized eigenvalues and eigenvectors of the Laplacian of the graph G are computed, i.e., we solve the problem $(D-G)\mathbf{w}=\lambda D\mathbf{w}$. Finally, we keep the eigenvectors of the m smallest eigenvalues discarding the smallest one which is always 0. The dimensionality reduction is performed by $\boldsymbol{\chi}_n=(\mathbf{w}_{1n},\mathbf{w}_{2n},...,\mathbf{w}_{mn})$, i.e., by keeping the n-th component of the m eigenvectors. The interesting property of the Laplacian eigenmap is that the cost function, which measures the distance among the projected features, can be expressed in terms of the Graph Laplacian: $J_{LE}=\frac{1}{2}\sum_{i,j}G_{ij}\left\|\boldsymbol{\chi}_i-\boldsymbol{\chi}_j\right\|^2=\boldsymbol{\chi}_i^tL\boldsymbol{\chi}_j$. So the goal is to minimize J_{LE} subject to $\boldsymbol{\chi}_i^tD\boldsymbol{\chi}_i=1$ [Zhang2009]. Finally, it is worth mentioning that Laplacian Eigenmaps and Principal Component Analysis (and their kernel versions) have been found to be particular cases of a more general problem called Least Squares Weighted Kernel Reduced Ranked Regression (LS-WKRRR) [Delatorre2012] (in fact, this framework also generalizes Canonical Correlation Analysis, Linear Discriminant Analysis and Spectral Clustering, techniques that are out of the scope of this review). The objective function is to minimize $J_{LS-WKRRR}=\left\|W_{\boldsymbol{\chi}}(\Phi_{\boldsymbol{\chi}}-BA^t\Phi_{\mathbf{x}})W_{\mathbf{x}}\right\|_F^2$ subject to $\mathrm{rank}(BA^t)=m$. $W_{\boldsymbol{\chi}}$ is a diagonal weight matrix for the feature points, $W_{\mathbf{x}}$ is a diagonal weight matrix for the input data points, $\Phi_{\mathbf{x}}$ is a matrix of the expanded dimensionality (kernel algorithms) of the input data. The objective function is minimized with respect to the A and B matrices (they are considered to be regression matrices and decoupling the transformation BA^t in two matrices allows the generalization of techniques like Canonical Correlation Analysis). The rank constraint is set to promote robustness of the solution towards a rank deficient $\Phi_{\mathbf{x}}$ matrix.

Hessian eigenmaps [Donoho2003] work with the Hessian of the graph instead of its Laplacian. By doing so, they extend ISOMAP and Laplacian eigenmaps, and they remove the need to map the input data onto a convex subset of $\mathbb{R}^m$.

Locally linear embedding (LLE) [Roweis2000, Saul2003] is another technique used to learn manifolds close to the data and project them onto it. For each observation $\mathbf{x}_n$ we look for the K-nearest neighbors (noted as $\mathbf{x}_{n'}$) and

produce a set of weights for its approximation minimizing $\left\| \mathbf{x}_n - \sum_{n'} \beta_{nn'} \mathbf{x}_{n'} \right\|^2$ (see Fig. 18). This optimization is performed simultaneously for all observations, i.e. $J_{LLE} = \sum_n \left\| \mathbf{x}_n - \sum_{n'} \beta_{nn'} \mathbf{x}_{n'} \right\|^2 = \sum_n \left\| \mathbf{x}_n - X\boldsymbol{\beta}_n \right\|^2$, where $\boldsymbol{\beta}_n$ is a weight vector to be determined and whose value is non-zero only for the neighbors of $\mathbf{x}_n$. We can write this even more compactly by stacking all weight vectors as columns of a matrix $\boldsymbol{\beta}$, then $J_{LLE} = \left\| X - X\boldsymbol{\beta} \right\|_F^2$. The optimization is constrained so that the sum of $\boldsymbol{\beta}_n$ is 1 for all n. Once the weights have been determined, we look for lower dimension points, $\boldsymbol{\chi}_n$, such that $\sum_n \left\| \boldsymbol{\chi}_n - \sum_{n'} \beta_{nn'} \boldsymbol{\chi}_{n'} \right\|^2$, that is the new points have to be reconstructed from its neighbors in the same way (with the same weights) as the observations they represent. This latest problem is solved by solving an eigenvalue problem and also keeping the smallest eigenvalues. See Fig. 19 for a comparison of the results of LLE, Hessian LLE and ISOMAP in a particular case.

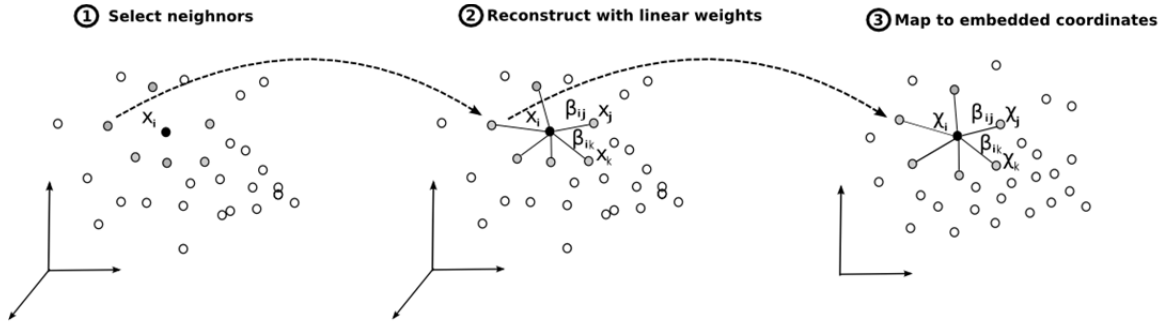

Figure 18. Schematic representation of the transformations involved in LTSA.

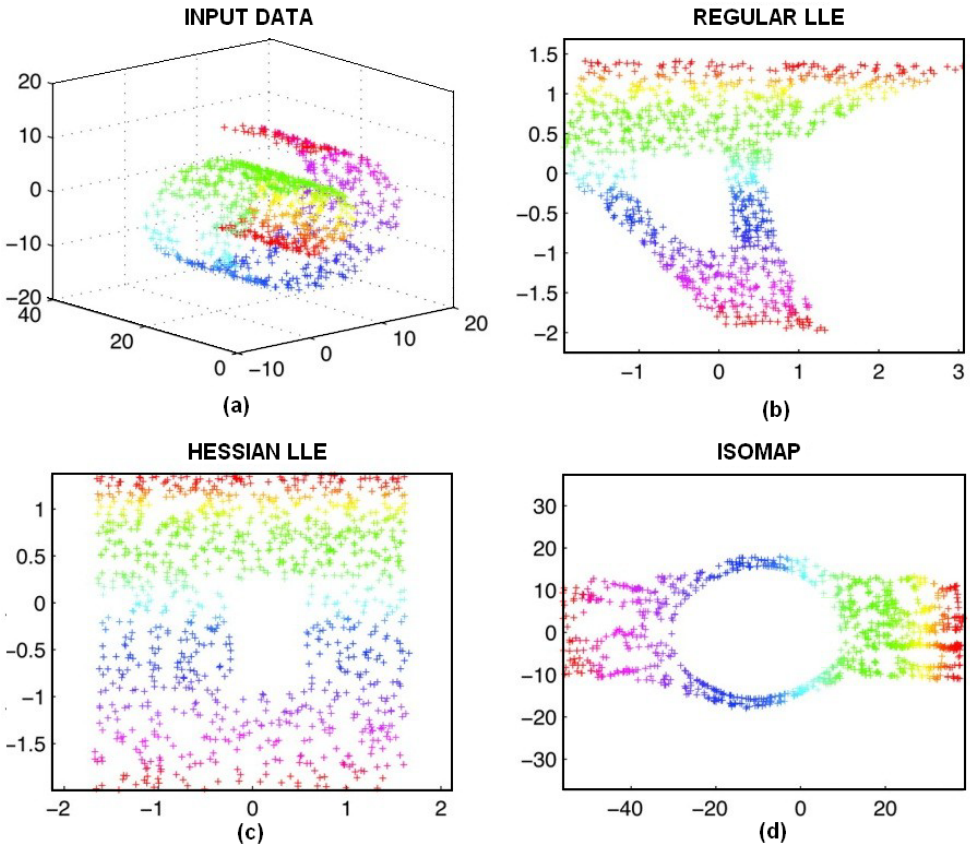

Figure 19. Comparison of the results of LLE, Hessian LLE and ISOMAP for the Swiss roll input data.

Latent Tangent Space Alignment (LTSA) [Zhang2004, Zhang2012] is another technique locally learning the manifold structure. As in LLE, we look for the local neighbors of a point. However, we now compute the local coordinates, $\boldsymbol{\chi}'_n$, of all the input points in the PCA subspace associated to this neighborhood. Next, we need to align all local coordinates. For each input point we compute the reconstruction error from its coordinates in the different neighborhoods where it participated $J_{LTSA_n} = \sum_{n'} \left\| \boldsymbol{\chi}_n - (c_{n'} + L_{n'} \boldsymbol{\chi}'_n) \right\|^2$ ($c_{n'}$ is a vector and $L_{n'}$ is a matrix, both to be determined for all neighborhoods; we may think of them as translation and shape parameters that properly locate the different neighborhoods in a common geometrical framework). The objective function of LTSA is $J_{LTSA} = \sum_{n=1}^{N} J_{LTSA_n}$ that has to be optimized with respect to $\boldsymbol{\chi}_n$, $c_{n'}$ and $L_{n'}$ (see Fig. 19).

One of the problems of the previous techniques (ISOMAP, Laplacian eigenmaps, Locally Linear Embedding, and Latent Tangent Space Alignment) is that they are only defined in a neighborhood of the training data, and they normally extrapolate very poorly. One of the reasons is because the mapping is not explicit, but implicit. Locality Preserving Projections (LPP) [He2004] tries to tackle this issue by constraining the projections to be a linear projection of the input vectors, $\boldsymbol{\chi}_n = A^t \mathbf{x}_n$. The goal function is the same as in Laplacian eigenmaps. The A matrix is of size $m \times M$ and it is the parameter with respect to which the J_{LE} objective function is minimized. An orthogonal version has been proposed [Kokiopoulo2007], with the constraint $A^t A = I$. A Kernel version of LPP also exists [He2004]. As in all kernel methods, the idea is to map the input vectors $\mathbf{x}_n$ onto a higher dimensional space with a nonlinear function, so that the linear constraint imposed by $\boldsymbol{\chi}_n = A^t \mathbf{x}_n$ becomes a nonlinear projection. A variation of this technique is called Neighborhood Preserving Embedding (NPE) [He2005] where

$J_{NPE} = \boldsymbol{\chi}_i^t M \boldsymbol{\chi}_j$ being $M = (I-G)^t(I-G)$. This matrix approximates the squared Laplacian of the weight graph G and it offers numerical advantages over the LPP algorithm. Orthogonal Neighborhood Preserving Projections (ONPP) [Kokiopoulou2007] extends the linear projection idea, $\boldsymbol{\chi}_n = A^t \mathbf{x}_n$, to the LLE algorithm.

The idea of constructing linear projections for the dimensionality reduction can be performed locally, instead of globally (as in LPP, OLPP, NPE and ONPP). This has been proposed by [Wang2011]. The manifold is divided in areas in which it can be well approximated by a linear subspace (a manifold is locally similar to a linear subspace if the geodesic distance between points is similar to the Euclidean distance among those points). The division is a disjoint partition of the input data points $\mathbf{x}_n$ such that the number of parts is minimized and each local linear subspace is as large as possible. Within each partition a linear PCA model is adjusted. Finally, all models are aligned following an alignment procedure similar to that of LTSA.

5. Trends and Conclusions

We have analyzed the number of citations that the most relevant papers in each section have received in the last decade (2003-2012). In Table I we show the number of citations summarized by large areas as well as their share (%) for the different years. At the sight of this table we can draw several conclusions:

- The interest in the field has grown by a factor 3 in the last decade as shown by the absolute number of citations.
- By far, the most applied techniques are those based on the search of components in its different brands (ICA, PCA, FA, MDS, …), although the tendency in the last decade is to loose importance in favor of those techniques using projections (especially, projections onto manifolds) or dictionaries. This is a response to the non-linear nature of experimental data in most fields.
- Dimensionality reduction techniques based on projections and dictionaries are growing very fast in the last decade: both, in the number of new methods and in the application of those methods to real problems.
- Interestingly, old methods based on vector quantization keep nearly a constant market share meaning that they are very well suited to a specific kind of problems. However, those methods that tried to preserve the input data topology while doing the vector quantization have lost impact, mostly because of the appearance of new methods capable of analyzing manifolds.

We can further subdivide these large areas into smaller subareas. Table II shows the subareas sorted by total number of citations. After analyzing this table we draw the following conclusions:

- The analysis on manifolds is the clear winner of the decade. The reason is its ability to analyze non-linearities and its capability of adapting to the local structure of the data. Among the different techniques, ISOMAP, Locally Linear Embedding, and Laplacian Eigenmaps are the most successful. This increase has been at the cost of the non-linear PCA versions (principal curves, principal surfaces and principal manifolds) and the Self-Organizing Maps since the new techniques can explore non-linear relationships in a much richer way.
- PCA in its different versions (standard PCA, robust PCA, sparse PCA, kernel PCA, …) is still one of the preferred techniques due to its simplicity and intuitiveness. The increase in the use of PCA contrasts with the decrease in the use of Factor Analysis, which is more constrained in its modeling capabilities.
- Independent Component Analysis reached its boom in the middle 2000's, but now it is declining. Probably, it will remain at a niche of applications related to signal processing for which it is particularly well suited. But it might not stand as a general purpose technique. It is possible that this decrease also responds to a diversification of the techniques falling under the umbrella of ICA.
- Non-negative Matrix Factorization has experienced an important raise, probably because of its ability of producing more interpretable bases and because they are well suited to many situations in which the sum of positive factors is the natural way of modeling the problem.

- The rest of the techniques have kept their market share. This is most likely explained by the fact that they have their own niche of applications, which they are very well suited to.

Overall, we can say that dimensionality reduction techniques are being applied in many scientific areas ranging from biomedical research to text mining and computer science. In this review we have covered different families of methodologies; each of them based on different criteria but all chasing the same goal: reduce the complexity of the data structure while at the same time delivering a more understandable representation of the same information. The field is still very active and ever more powerful methods are continuously appearing providing an excellent application test bed for applied mathematicians.

Acknowledgements

This work was supported by the Spanish Minister of Science and Innovation (BIO2010-17527) and Madrid government grant (P2010/BMD-2305). C.O.S. Sorzano was also supported by the Ramón y Cajal program.

Bibliography

1. [Abdi2003] Abdi, H. Lewis-Beck, M.; Bryman, A. & Futing, T. (ed.) Encyclopedia for research methods for the social sciences Factor rotations in factor analyses Sage, 2003, 792-795
2. [Abdi2007] Abdi, H. Salkind, N. (ed.) Encyclopedia of measurements and statistics Singular value decomposition (SVD) and Generalized Singular Value Decomposition (GSVD) Sage Publications, 2007, 907-912
3. [Aharon2006] Aharon, M.; Elad, M. & Bruckstein, A. K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation IEEE Trans. Signal Processing, 2006, 54, 4311-4322
4. [Arora1998] Arora, S.; Raghavan, P. & Rao, S. Approximation schemes for Euclidean k-medians and related problems Proc. 30th Annual ACM symposium on Theory of computing (STOC), 1998
5. [Artac2002] Artac, M.; Jogan, M. & Leonardis, A. Incremental PCA or On-Line Visual Learning and Recognition Proc.16th International Conference on Pattern Recognition (ICPR), 2002, 3, 30781
6. [Baccini1996] Baccini, A.; Besse, P. & Falguerolles, A L_1-norm PCA and a heuristic approach. A. Ordinal and Symbolic Data Analysis. Diday, E.; Lechevalier, Y. & Opitz, O. (Eds.) Springer, 1996, 359-368
7. [Bailey1994] Bailey, T. L. & Elkan, C. Fitting a mixture model by expectation maximization to discover motifs in biopolymers Univ. California San Diego, 1994. Tech. Report CS94-351.
8. [Baldi1989] Baldi, P. & Hornik, K. Neural networks and principal component analysis: Learning from examples without local minima Neural Networks, 1989, 2, 53-58
9. [Baraniuk2010] Baraniuk, R. G.; Cevher, V. & Wakin, M. B. Low-dimensional models for dimensionality reduction and signal recovery: a geometric perspective Proc. IEEE, 2012, 98, 959-971
10. [Batmanghelich2012] Batmanghelich, N. K.; Taskar, B. & Davatzikos, C. Generative-discriminative basis learning for medical imaging. IEEE Trans. Medical Imaging, 2012, 31, 51-69
11. [Belkin2001] Belkin, M. & Niyogi, P. Laplacian Eigenmaps and Spectral Techniques for Embedding and Clustering Advances in Neural Information Processing Systems, 2001, 14, 585-591
12. [Belkin2002] Belkin, M. & Niyogi, P. Laplacian Eigenmaps for Dimensionality Reduction and Data Representation Neural Computation, 2002, 15, 1373-1396
13. [Bengio2004] Bengio, Y.; Delalleau, O.; Le Roux, N.; Paiement, J. F.; Vincent, P. & Ouimet, M. Learning eigenfunctions links spectral embedding and Kernel PCA Neural Computation, 2004, 16, 2197-2219
14. [Bengio2006] Bengio, Y.; Delalleau, O.; Le Roux, N.; Paiement, J. F.; Vincent, P. & Ouimet, M. Spectral Dimensionality Reduction. Studies in Fuzziness and Soft Computing: Feature Extraction Springer, 2006, 519-550
15. [Bezdek1981] Bezdek, J. C. Pattern Recognition with Fuzzy Objective Function Algorithms Plenum, 1981
16. [Bezdek1984] Bezdek, J. C.; Ehrlich, R. & Full, W. FCM: The fuzzy c-means clustering algorithm Computers & Geosciences, 1984, 10, 191-203
17. [Bian2011] Bian, W. & Tao, D. Max-min distance analysis by using sequential SDP relaxation for dimension reduction. IEEE Trans. Pattern Analysis & Machine Intelligence, 2011, 33, 1037-1050
18. [Bingham2001] Bingham, E. & Mannila, H. Random projection in dimensionality reduction: applications to image and text data Proc. ACM Intl. Conf. Knowledge discovery and data mining, 2001
19. [Bishop1998] Bishop, C. M.; Svensén, M. & Williams, C. K. I. GTM: The Generative Topographic Mapping Neural Computation, 1998, 10, 215-234
20. [Blumensath2008] Blumensath, T. & Davies, M. E. Gradient pursuits IEEE Trans. Signal Processing, 2008, 56, 2370-2382
21. [Borg2005] Borg, I. & Groenen, P. F. Modern Multidimensional Scaling Springer, 2005
22. [Bro1997] Bro, R. PARAFAC. Tutorial and applications Chemometrics and intelligent laboratory systems, 1997, 38, 149-171
23. [Bruckstein2009] Bruckstein, A. M.; Donoho, D. L. & Elad, M. From Sparse Solutions of Systems of Equations to Sparse Modeling of Signals and Images SIAM Review, 2009, 51, 34-81
24. [Cai2011] Cai, H.; Mikolajczyk, K. & Matas, J. Learning linear discriminant projections for dimensionality reduction of image descriptors. IEEE Trans. Pattern Analysis & Machine Intelligence, 2011, 33, 338-352
25. [Cai2011b] Cai, D.; He, X.; Han, J. & Huang, T. S. Graph Regularized Non-Negative Matrix Factorization for Data Representation. IEEE Trans. Pattern Analysis & Machine Intelligence, 2010, 33, 1548-1560
26. [Carreira1997] Carreira-Perpiñán, M. A. A review of dimension reduction techniques Dept. Computer Science, Univ. Sheffield, 1997
27. [Cayton2005] Cayton, L. Algorithms for manifold learning University of California, San Diego, Tech. Rep. CS2008-0923, 2005

28. [Chen1991] Chen, S.; Cowan, C. F. N. & Grant, P. M. Orthogonal least squares learning algorithm for radial basis function networks IEEE Trans. Neural Networks, 1991, 2, 302-309
29. [Chen1994] Chen, S. S. & Donoho, D. L. Basis pursuit Proc. IEEE Conf. Signals, Systems and Computers, 1994
30. [Chen2001] Chen, S. S.; Donoho, D. L. & Saunders, M. A. Atomic decomposition by basis pursuit SIAM Review, 2001, 43, 129-159
31. [Cichocki2009] Cichocki, A.; Zdunek, R.; Phan, A. H. & Amari, S. Non-negative matrix and tensor factorizations Wiley, 2009
32. [Common1994] Common, P. Independent Component Analysis, a new concept? Signal Processing, 36, 287-314 (1994)
33. [Costa2004] Costa, J. A. & Hero, A. O. I. Geodesic Entropic Graphs for Dimension and Entropy Estimation in Manifold Learning IEEE Trans. Signal Processing, 2004, 52, 2210-2221
34. [Cox2000] Cox, T. F. & Cox, M. A. A. Multidimensional Scaling Chapman & Hall, 2000
35. [Crawford1970] Crawford, C. B. & Ferguson, G. A. A general rotation criterion and its use in orthogonal rotation Psychometrika, 1970, 35, 321-332
36. [Dasgupta2000] Dasgupta, S. Experiments with random projection Proc. Conf. Uncertainty in artificial intelligence, 2000
37. [Dash1997] Dash, M. & Liu, H. Feature selection for classification Intelligent Data Analysis, 1997, 1, 131-156
38. [Delatorre2003] De la Torre, F. & Black, M. J. A framework for robust subspace learning Intl. J. Computer Vision, 2003, 54, 117-142
39. [Delatorre2012] De la Torre, F. A least-squares framework for Component Analysis. IEEE Trans. Pattern Analysis & Machine Intelligence, 2012, 34, 1041-1055
40. [Delicado2001] Delicado, P. Another look at principal curves and surfaces J. Multivariate Analysis, 2001, 77, 84-116
41. [DeMers1993] DeMers, D. & Cottrell, G. Nonlinear dimensionality reduction Advances in Neural Information Processing Systems, 1993, 5, 580-587
42. [Dhillon2004] Dhillon, I. S.; Guan, Y. & Kulis, B. Kernel k-means: spectral clustering and normalized cuts Proc. ACM SIGKDD Intl. Conf. on Knowledge discovery and data mining, 2004, 551-554
43. [Ding2006] Ding, C.; Zhou, D.; He, X. & Zha, H. R_1-PCA: Rotational Invariant L_1-norm Principal Component Analysis for Robust Subspace Factorization Proc. Intl. Workshop Machine Learning, 2006
44. [Donoho2003] Donoho, D. L. & Grimes, C. Hessian eigenmaps: locally linear embedding techniques for high-dimensional data. Proc. Natl. Acad. Sci. USA, 2003, 100, 5591-5596
45. [Dunteman1989] Dunteman, G. H. Principal Component Analysis Sage Publications, 1989
46. [Einbeck2005] Einbeck, J.; Tutz, G. & Evers, L. Local principal curves Statistics and Computing, 2005, 15, 301-313
47. [Einbeck2008] Einbeck, J.; Evers, L. & Bailer-Jones, C. Representing Complex Data Using Localized Principal Components with Application to Astronomical Data Lecture Notes in Computational Science and Engineering, 2008, 58, 178-201
48. [Engan2000] Engan, K.; Aase, S. O. & Husoy, J. H. Multi-frame compression: theory and design EURASIP Signal Processing, 2000, 80, 2121-2140
49. [Feng2002] Feng, T.; Li, S. Z.; Shum, H. Y. & Zhang, H. Local nonnegative matrix factorization as a visual representation Proc. 2nd Intl. Conf. Development and Learning (ICDL), 2002
50. [Fisher1936] Fisher, R. A. The Use of Multiple Measurements in Taxonomic Problems Annals of Eugenics, 1936, 7, 179-188
51. [Fodor2002] Fodor, I. K. A survey of dimension reduction techniques Lawrence Livermore Natl. Laboratory, 2002
52. [Friedman1974] Friedman, J. H. & Tukey, J. W. A Projection Pursuit Algorithm for Exploratory Data Analysis IEEE Trans. Computers, 1974, C-23, 881-890
53. [Friedman1987] Friedman, J. H. Exploratory projection pursuit J. American Statistical Association, 1987, 82, 249-266
54. [Fritzke1995] Fritzke, A growing neural gas network learns topologies Advances in Neural Information Processing, B. Tesauro, G.; Touretzky, D. & Lean, T. K. (Eds.) MIT Press, 1995, 625-632
55. [Fukunaga1971] Fukunaga, K. & Olsen, D. R. An algorithm for finding intrinsic dimensionality of data IEEE Trans. Computers, 1971, C-20, 176-183
56. [Gersho1992] Gersho, A., Gray, R.M. Vector quantization and signal compression. Kluwer Academic Publishers, 1992.
57. [Girolami1997] Girolami, M. & Fyfe, C. Extraction of independent signal sources using a deflationary exploratory projection pursuit network with lateral inhibition IEE Proc. Vision, Image and Signal Processing Journal, 1997, 14, 299-306
58. [Girolami1997b] Girolami, M. & Fyfe, C. ICA Contrast Maximisation Using Oja's Nonlinear PCA Algorithm Intl. J. Neural Systems, 1997, 8, 661-678
59. [Girolami2002] Girolami, M. Mercer kernel-based clustering in feature space. IEEE Trans. Neural Networks, 2002, 13, 780-784
60. [Golub1970] Golub, G. H. & Reinsch, C. Singular value decomposition and least squares solutions Numerische Mathematik, 1970, 14, 403-420
61. [Gorban2004] Gorban, A. N.; Karlin, I. V. & Zinovyev, A. Y. Constructive methods of invariant manifolds for kinetic problems Constructive methods of invariant manifolds for kinetic problems, 2004, 396, 197-403
62. [Gorban2007] Gorban, A. N.; Kgl, B.; Wunsch, D. C. & Zinovyev, A. Principal Manifolds for Data Visualization and Dimension Reduction Springer, 2007
63. [Gorodnitsky1997] Gorodnitsky, I. F. & Rao, B. D. Sparse signal reconstruction from limited data using FOCUSS: re-weighted minimum norm algorithm IEEE Trans. Signal Processing, 1997, 3, 600-616
64. [Graps1995] Graps, A. An introduction to wavelets IEEE Computational Science & Engineering, 1995, 2, 50-61
65. [Gray1984] Gray, R. M. Vector quantization IEEE Acoustics, Speech and Signal Processing Magazine, 1984, 1, 4-29
66. [Gurumoorthy2010] Gurumoorthy, K. S.; Rajwade, A.; Banerjee, A. & Rangarajan, A. A method for compact image representation using sparse matrix and tensor projections onto exemplar orthonormal bases IEEE Trans. Image Processing, 2010, 19, 322-334
67. [Guyon2003] Guyon, I. & Eliseeff, A. An introduction to variable and feature selection J. Machine Learning Research, 2003, 3, 1157-1182
68. [Harman1976] Harman, H. H. Modern Factor Analysis Univ. Chicago Press, 1976
69. [Harshman1970] Harshman, R. A. Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multimodal factor analysis UCLA Working Papers in Phonetics, 1970, 16, 1-84
70. [Hartigan1979] Hartigan, J. A. & Wong, M. A. Algorithm AS 136: A K-Means Clustering Algorithm J. Royal Statistical Soc. C, 1979, 28, 100-108
71. [Hastie1989] Hastie, T. & Stuetzle, W. Principal curves J. American Statistical Association, 1989, 84, 502-516
72. [He2004] He, X. & Niyogi, Locality Preserving Projections. Advances In Neural Information Processing Systems P. Thrun, S.; Saul, L. K. & Schölkopf, B. (Eds.) MIT Press, 2004, 153-160
73. [He2005] He, X.; Cai, D.; Yan, S. & Zhang, H. J. Neighborhood Preserving Embedding Proc. IEEE Intl. Conf. Computer Vision, ICCV, 2005.
74. [He2011] He, R.; Hu, B.-G.; Zheng, W.-S. & Kong, X.-W. Robust principal component analysis based on maximum correntropy criterion. IEEE Trans. Image Processing, 2011, 20, 1485-1494
75. [Hoyer2002] Hoyer, P. O. Nonnegative sparse coding Proc. IEEE Workshop Neural Networks for Signal Processing, 2002
76. [Hoyer2004] Hoyer, P. O. Nonnegative matrix factorization with sparseness constraints J. Machine Learning Research, 2004, 5, 1457-1469
77. [Huang1998] Huang, H. E.; Shen, Z.; Long, S. R.; Wu, M. L.; Shih, H. H.; Zheng, Q.; Yen, N. C.; Tung, C. C. & Liu, H. H. The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis Proc. Roy. Soc. London A, 1998, 454, 903-995

78. [Hubert2004] Hubert, M. & Engelen, S. Robust PCA and classification in biosciences Bioinformatics, 2004, 20, 1728-1736
79. [Hyvarinen1999] Hyvärinen, A. Fast and robust fixed-point algorithms for independent component analysis. IEEE Trans. Neural Networks, 1999, 10, 626-634
80. [Hyvarinen2000] Hyvarinen, A., Oja. E. Independent Component Analysis: algorithms and applications. Neural networks, 2000, 13, 411-430
81. [Hyvarinen2001] Hÿvarinen, A., Karhunen, J., Oja, E. Independent Component Analysis. John Wiley & Sons, Inc. 2001
82. [Iglesias2007] Iglesias, J. E.; de Bruijne, M.; Loog, M.; Lauze, F. & Nielsen, M. A family of principal component analyses for dealing with outliers. Lecture Notes in Computer Science, 2007, 4792, 178-185
83. [Jenssen2010] Jenssen, R. Kernel entropy component analysis IEEE Trans. Pattern Analysis & Machine Intelligence, 2010, 32, 847-860
84. [Johnson1963] Johnson, R. M. On a theorem stated by Eckart and Young Psychometrika, 1963, 28, 259-263
85. [Jollife2002] Jollife, I. T. Principal Component Analysis Wiley, 2002
86. [Kaiser1958] Kaiser, H. F. The varimax criterion for analytic rotation in factor analysis. Psychometrika, 1958, 23, 187-200
87. [Kaiser1960] Kaiser, H. F. The application of electronic computers to factor analysis. Educational and psychological measurement, 1960, 20, 141-151
88. [Kambhatla1997] Kambhatla, N. & Leen, T. K. Dimension reduction by local Principal Component Analysis Neural Computation, 1997, 9, 1493-1516
89. [Kaski1998] Kaski, S. Dimensionality reduction by random mapping: fast similarity computation for clustering Proc. Intl. Joint Conf. Neural Networks (IJCNN), 1998
90. [Ke2005] Ke, Q., Kanade, T. Robust L_1 norm factorization in the presence of outliers and missing data by alternative convex programming Proc. Comput. Vis. Pattern Recogn. Conf., 2005
91. [Kegl2002] Kegl, B. Intrinsic Dimension Estimation Using Packing Numbers. Advances in Neural Information Processing Systems 2002
92. [Kim2011] Kim, M. & Pavlovic, V. Central subspace dimensionality reduction using covariance operators. IEEE Trans Pattern Anal Mach Intell, 2011, 33, 657-670
93. [Klema1980] Klema, V. & Laub, A. The singular value decomposition: Its computation and some applications IEEE Trans. Automatic Control, 1980, 25, 164-176
94. [Kohonen1990] Kohonen, T. The Self-Organizing Map Proc. IEEE, 1990, 78, 1464-1480
95. [Kohonen1993] Kohonen, T. Things you haven't heard about the self-organizing map Proc. IEEE Intl. Conf. Neural Networks, 1993
96. [Kohonen2001] Kohonen, T. Self-Organizing Maps. Springer, 2001.
97. [Kokiopoulou2007] Kokiopoulou, E. & Saad, Y. Orthogonal neighborhood preserving projections: a projection-based dimensionality reduction technique. IEEE Trans. Pattern Analysis & Machine Intelligence, 2007, 29, 2143-2156
98. [Kramer1991] Kramer, M. A. Nonlinear Principal Component Analysis Using Autoassociative Neural Networks AIChE Journal, 1991, 37, 233-243
99. [Kreutz2003] Kreutz-Delgado, K.; Murray, J. F.; Rao, B. D.; Engan, K.; Lee, T. & Sejnowski, T. J. Dictionary learning algorithms for sparse representation Neural Computation, 2003, 15, 349-396
100. [Kruskal1964a] Kruskal, J. B. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis Psychometrika, 1964, 29, 1-27
101. [Kruskal1964b] Kruskal, J. B. Nonmetric multidimensional scaling: a numerical method. Psychometrika, 1964, 29, 115-129
102. [Kruskal1986] Kruskal, J. B. & Wish, M. Multidimensional scaling Sage, 1986
103. [Kwak2008] Kwak, N. Principal component analysis based on l1-norm maximization. IEEE Trans Pattern Anal Mach Intell, 2008, 30, 1672-1680
104. [Lawley1971] Lawley, D. N. & Maxwell, A. E. Factor analysis as a statistical method Butterworths, 1971
105. [Lee1999] Lee, D. D. & Seung, S. Learning the parts of objects by non-negative matrix factorization Nature, 1999, 401, 788-791
106. [Lee2000] Lee, T. W.; Girolami, M. & Bell, A. J. Sejnowski, T. J. A unifying information-theoretic framework for independent component analysis Computers & Mathematics with Applications, 2000, 39, 1-21
107. [Lee2001] Lee, D. D. & Seung, H. S. Algorithms for non-negative matrix factorization Advances in Neural Information Processing Systems, 2001, 556-562
108. [Lewicki2000] Lewicki, M. S. & Sejnowski, T. J. Learning overcomplete representations Neural Computation, 2000, 12, 337-365
109. [Li2011] Li, X. L.; Adali, T. & Anderson, M. Noncircular Principal Component Analysis and Its Application to Model Selection IEEE Trans. Signal Processing, 2011, 59, 4516-4528
110. [Lin2011] Lin, Y.Y.; Liu, T.L. & Fuh, C.S. Multiple kernel learning for dimensionality reduction. IEEE Trans. Pattern Analysis & Machine Intelligence, 2011, 33, 1147-1160
111. [Liu2003] Liu, W.; Zheng, N. & Lu, X. Nonnegative matrix factorization for visual coding Proc. IEEE Intl. Conf. Acoustics, Speech and Signal Processing (ICASSP), 2003
112. [Liu2003b] Liu, Z. Y.; Chiu, K. C. & Xu, L. Improved system for object detection and star/galaxy classification via local subspace analysis Neural Networks, 2003, 16, 437-451
113. [Mallat1993] Mallat, S.G. & Zhang, Z. Matching pursuits with time-frequency dictionaries. IEEE Trans. Signal Processing, 1993, 41, 3397-3415
114. [Martinetz1991] Martinetz, T. & Schulten. A ``neural-gas'' network learns topologies. Artificial neural networks, K. Kohonen, T.; Makisara, K.; Simula, O. & Kangas, J. (Eds.) Elsevier, 1991, 397-402
115. [Martinetz1993] Martinetz, T. M.; Berkovich, S. G. & Schulten, K. J. Neural-gas network for vector quantization and its application to time-series prediction. IEEE Trans. Neural Networks, 1993, 4, 558-569
116. [Mateen2009] van der Mateen, L.; Postma, E. & van den Herik, J. Dimensionality Reduction: A Comparative Review Tilburg Centre for Creative Computing, Tilburg Univ., 2009
117. [Mulaik1971] Mulaik, S. A. The foundations of factor analysis Chapman & Hall, 1971
118. [Paige1981] Paige, C. C. & Saunders, M. A. Towards a Generalized Singular Value Decomposition SIAM Journal on Numerical Analysis, 1981, 18, 398-405
119. [Pascual-Marqui2001] Pascual-Marqui, R. D.; Pascual-Montano, A.; Kochi, K. & Carazo, J. M. Smoothly Distributed Fuzzy c-Means: A New Self-Organizing Map Pattern Recognition, 2001, 34, 2395-2402
120. [Pascual-Montano2001] Pascual-Montano, A.; Donate, L. E.; Valle, M.; Bárcena, M.; Pascual-Marqui, R. D. & Carazo, J. M. A novel neural network tecnique for analysis and classification of EM single-particle images J. Structural Biology, 2001, 133, 233-245
121. [Pascual-Montano2006] Pascual-Montano, A.; Carazo, J.; Kochi, K.; Lehmann, D. & Pascual-Marqui, R. Nonsmooth nonnegative matrix factorization (nsNMF) IEEE Trans. Pattern Analysis & Machine Intelligence, 2006, 28, 403-415
122. [Pati1993] Pati, Y.; Rezaiifar, R. & Krishnaprasad, P. Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition Proc. Conf. Record of The Twenty-Seventh Asilomar Conference on Signals, Systems and Computers, 1993, 40-44

123. [Pauca2006] Pauca, V. P.; Piper, J. & Plemmons, R. J. Nonnegative matrix factorization for spectral data analysis. Linear Algebra and its Applications, 2006, 416, 29-47
124. [Pearson1901] Pearson, K. On Lines and Planes of Closest Fit to Systems of Points in Space. Philosophical Magazine, 1901, 2, 559-572
125. [Pettis1979] Pettis, K. W.; Bailey, T. A.; Jain, A. K. & Dubes, R. C. An intrinsic dimensionality estimator from near-neighbor information. IEEE Trans. Pattern Analysis & Machine Intelligence, 1979, 1, 25-37
126. [Pinto2011] Pinto da Costa, J. F.; Alonso, H. & Roque, L. A weighted principal component analysis and its application to gene expression data. IEEE/ACM Trans. Computational Biology and Bioinformatics, 2011, 8, 246-252
127. [Ramanan2011] Ramanan, D. & Baker, S. Local distance functions: a taxonomy, new algorithms, and an evaluation. IEEE Trans. Pattern Analysis & Machine Intelligence, USA. dramanan@ics.uci.edu, 2011, 33, 794-806
128. [Rilling2003] Rilling, G.; Flandrin, P. & Goncalves, P. On empirical mode decomposition and its algorithms Proc. IEEE-EURASIP Workshop on Nonlinear Signal and Image Processing, 2003
129. [Rioul1991] Rioul, O. & Vetterli, M. Wavelets and signal processing. IEEE Signal Processing Magazine, 1991, 8, 14-38
130. [Roweis2000] Roweis, S. T. & Saul, L. K. Nonlinear Dimensionality Reduction by Locally Linear Embedding Science, 2000, 290, 2323-2326
131. [Rubinstein2010] Rubinstein, R.; Bruckstein, A. M. & Elad, M. Dictionaries for sparse representation modeling Proc. IEEE, 2010, 98, 1045-1057
132. [Rubner2000] Rubner, Y.; Tomasi, C. & Guibas, L. J. The Earth Mover's Distance as a Metric for Image Retrieval Intl. J. Computer Vision, 2000, 40, 99-121
133. [Saeys2007] Saeys, Y.; Inza, I. & Larrañaga, P. A review of feature selection techniques in bioinformatics Bioinformatics, 2007, 23, 2507-2517
134. [Sammon1969] Sammon, J. W. A nonlinear mapping for data structure analysis. IEEE Trans. Computers, 1969, 18, 401-409
135. [Sandler2011] Sandler, R. & Lindenbaum, M. Nonnegative Matrix Factorization with Earth Mover's Distance Metric for Image Analysis. IEEE Trans. Pattern Analysis & Machine Intelligence, Yahoo! Research in Haifa., 2011, 33, 1590-1602
136. [Saul2003] Saul, L. K. & Roweis, S. Think Globally, Fit Locally : Unsupervised Learning of Low Dimensional Manifolds Department of Computer & Information Science, Univ. Pennsylvania, 2003
137. [Schiffman1981] Schiffman, S. S.; Reynolds, M. L. & Young, F. W. Introduction to multidimensional scaling: Theory, methods, and applications Academic Press, 1981
138. [Scholkopf1997] Schölkopf, B.; Smola, A. & Müller, K. R. Kernel Principal Component Analysis Proc. of ICANN, 1997, 583-58
139. [Scholkopf1999] Schölkopf, B.; Smola, A. & Müller, K. R. Kernel Principal Component Analysis Advances in kernel methods-support vector learning, 1999
140. [Scholz2008] Scholz, M.; Fraunholz, M. & Selbig, J. Nonlinear Principal Component Analysis: Neural Network Models and Applications Lecture Notes in Computational Science and Engineering, 2008, 58, 44-67
141. [Smola1999] Smola, A. J.; Williamson, R. C.; Mika, S. & Schölkopf, B. Regularized principal manifolds Lecture Notes in Artificial Intelligence, 1999, 1572, 214-229
142. [Spearman1904] Spearman, C. "General Intelligence" Objectively Determined and Measured American J. Psychology, 1904, 15, 201-292
143. [Subbarao2006] Subbarao, R. & Meer, P. Subspace Estimation Using Projection Based M-Estimators over Grassmann Manifolds Lecture Notes in Computer Science, 2006, 3951, 301-312
144. [Tenenbaum2000] Tenenbaum, J. B.; de Silva, V. & Langford, J. C. A global geometric framework for nonlinear dimensionality reduction Science, 2000, 290, 2319-2323
145. [Theis2011] Theis, F. J.; Kawanabe, M. & Muller, K. R. Uniqueness of Non-Gaussianity-Based Dimension Reduction IEEE Trans. Signal Processing, 2011, 59, 4478-4482
146. [Thurstone1947] Thurstone, L. Multiple Factor Analysis Univ. Chicago Press, 1947
147. [Tropp2007] Tropp, J. A. & Gilbert, A. C. Signal Recovery From Random Measurements Via Orthogonal Matching Pursuit IEEE Trans. Information Theory, 2007, 53, 4655-4666
148. [Tucker1966] Tucker, L. R. Some mathematical notes on three-mode factor analysis Psychometrika, 1966, 31, 279-311
149. [Ulfarsson2011] Ulfarsson, M. O. & Solo, V. Vector l_0 sparse variable PCA IEEE Trans. Signal Processing, 2011, 59, 1949-1958
150. [Vidal2005] Vidal, R.; Ma, Y. & Sastry, S. Generalized principal component analysis (GPCA). IEEE Trans. Pattern Analysis & Machine Intelligence, 2005, 27, 1945-1959
151. [Wall2003] Wall, M.; Rechtsteiner, A. & Rocha, L. A practical approach to microarray data analysis Singular Value Decomposition and Principal Component Analysis Springer, 2003, 91-10
152. [Wang2011] Wang, R.; Shan, S.; Chen, X.; Chen, J. & Gao, W. Maximal Linear Embedding for Dimensionality Reduction. IEEE Trans. Pattern Analysis & Machine Intelligence, 2011, 33, 1776-1792
153. [Watson1993] Watson, A. B. DCT quantization matrices visually optimized for individual images Proc. SPIE Workshop on Augmented Visual Display (AVID) Research, 1993, 1913-1914, 363-377
154. [Williams2002] Williams, C. K. I. On a Connection between Kernel PCA and Metric Multidimensional Scaling Machine Learning, 2002, 46, 11-19
155. [Wold1987] Wold, S.; Esbensen, K. & Geladi, P. Principal component analysis Chemometrics and Intelligent Laboratory Systems, 1987, 2, 37-5
156. [Yin2008] Yin, H. Learning Nonlinear Principal Manifolds by Self-Organising Maps Lecture Notes in Computational Science and Engineering, 2008, 58, 68-95
157. [Yu2012] Yu, S.; Tranchevent, L.-C.; Liu, X.; Glänzel, W.; Suykens, J. A. K.; Moor, B. D. & Moreau, Y. Optimized data fusion for kernel k-means clustering. IEEE Trans. Pattern Analysis & Machine Intelligence, 2012, 34, 1031-1039
158. [Zhang2004] Zhang, Z. & Zha, H. Principal Manifolds and Nonlinear Dimensionality Reduction via Tangent Space Alignment SIAM J. Scientific Computing, 2004, 26, 313-338
159. [Zhang2009] Zhang, J.; Niyogi, P. & McPeek, M. S. Laplacian eigenfunctions learn population structure. PLoS One, 2009, 4, e7928
160. [Zhang2012] Zhang, Z.; Wang, J. & Zha, H. Adaptive manifold learning. IEEE Trans. Pattern Analysis & Machine Intelligence, 2012, 34, 253-265
161. [Zou2006] Zou, H.; Hastie, T. & Tibshirani, R. Sparse Principal Component Analysis J. Computational and Graphical Statistics, 2006, 15, 262-286