Principal component analysis

Given a collection of points in two, three, or higher dimensional space, a "best fitting" line can be defined as one that minimizes the average squared distance from a point to the line. The next best-fitting line can be similarly chosen from directions perpendicular to the first. Repeating this process yields an orthogonal basis in which different individual dimensions of the data are uncorrelated. These basis vectors are called **principal components**, and several related procedures **principal component analysis** (**PCA**).

PCA is mostly used as a tool in exploratory data analysis and for making predictive models. It is often used to visualize genetic distance and relatedness between populations. PCA is either done in the following 2 steps:

PCA of a multivariate Gaussian distribution centered at (1,3) with a standard deviation of 3 in roughly the (0.866, 0.5) direction and of 1 in the orthogonal direction. The vectors shown are the eigenvectors of the covariance matrix scaled by the square root of the corresponding eigenvalue, and shifted so their tails are at the mean.
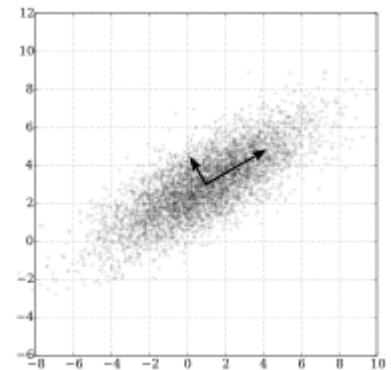
1. calculating the data covariance (or correlation) matrix of the original data
2. performing eigenvalue decomposition on the covariance matrix

or by singular value decomposition of a design matrix. Usually the original data is normalized before performing the PCA. The normalization of each attribute consists of *mean centering* – subtracting each data value from its variable's measured mean so that its empirical mean (average) is zero. Some fields, in addition to normalizing the mean, do so for each variable's variance (to make it equal to 1); see z-scores.[1] The results of a PCA are usually discussed in terms of *component scores*, sometimes called *factor scores* (the transformed variable values corresponding to a particular data point), and *loadings* (the weight by which each standardized original variable should be multiplied to get the component score).[2] If component scores are standardized to unit variance, loadings must contain the data variance in them (and that is the magnitude of eigenvalues). If component scores are not standardized (therefore they contain the data variance) then loadings must be unit-scaled, ("normalized") and these weights are called eigenvectors; they are the cosines of orthogonal rotation of variables into principal components or back.

PCA is the simplest of the true eigenvector-based multivariate analyses. Often, its operation can be thought of as revealing the internal structure of the data in a way that best explains the variance in the data. If a multivariate dataset is visualised as a set of coordinates in a high-dimensional data space (1 axis per variable), PCA can supply the user with a lower-dimensional picture, a projection of this object when viewed from its most informative viewpoint. This is done by using only the first few principal components so that the dimensionality of the transformed data is reduced.

PCA is closely related to factor analysis. Factor analysis typically incorporates more domain specific assumptions about the underlying structure and solves eigenvectors of a slightly different matrix.

PCA is also related to canonical correlation analysis (CCA). CCA defines coordinate systems that optimally describe the cross-covariance between two datasets while PCA defines a new orthogonal coordinate system that optimally describes variance in a single dataset.[3][4][5][6]

Robust and L1-norm-based variants of standard PCA have also been proposed.[7][8][6]

Contents

History

PCA was invented in 1901 by Karl Pearson,[9] as an analogue of the principal axis theorem in mechanics; it was later independently developed and named by Harold Hotelling in the 1930s.[10] Depending on the field of application, it is also named the discrete Karhunen–Loève transform (KLT) in signal processing, the Hotelling transform in multivariate quality control, proper orthogonal decomposition (POD) in mechanical engineering, singular value decomposition (SVD) of $\mathbf{X}$ (Golub and Van Loan, 1983), eigenvalue decomposition (EVD) of $\mathbf{X}^T\mathbf{X}$ in linear algebra, factor analysis (for a discussion of the differences between PCA and factor analysis see Ch. 7 of Jolliffe's *Principal Component Analysis*),[11] Eckart–Young theorem (Harman, 1960), or empirical orthogonal functions (EOF) in meteorological science, empirical eigenfunction decomposition (Sirovich, 1987), empirical component analysis (Lorenz, 1956), quasiharmonic modes (Brooks et al., 1988), spectral decomposition in noise and vibration, and empirical modal analysis in structural dynamics.

Intuition

PCA can be thought of as fitting a *p*-dimensional ellipsoid to the data, where each axis of the ellipsoid represents a principal component. If some axis of the ellipsoid is small, then the variance along that axis is also small, and by omitting that axis and its corresponding principal component from our representation of the dataset, we lose only an equally small amount of information.

To find the axes of the ellipsoid, we must first subtract the mean of each variable from the dataset to center the data around the origin. Then, we compute the covariance matrix of the data and calculate the eigenvalues and corresponding eigenvectors of this covariance matrix. Then we must normalize each of the orthogonal eigenvectors to become unit vectors. Once this is done, each of the mutually orthogonal, unit eigenvectors can be interpreted as an axis of the ellipsoid fitted to the data. This choice of basis will transform our covariance matrix into a diagonalised form with the diagonal elements representing the variance of each axis. The proportion of the variance that each eigenvector represents can be calculated by dividing the eigenvalue corresponding to that eigenvector by the sum of all eigenvalues.

This procedure is sensitive to the scaling of the data, and there is no consensus as to how to best scale the data to obtain optimal results.

Details

PCA is defined as an <u>orthogonal</u> <u>linear transformation</u> that transforms the data to a new <u>coordinate system</u> such that the greatest variance by some scalar projection of the data comes to lie on the first coordinate (called the first principal component), the second greatest variance on the second coordinate, and so on.[11]

Consider an $n \times p$ data <u>matrix</u>, $\mathbf{X}$, with column-wise zero <u>empirical mean</u> (the sample mean of each column has been shifted to zero), where each of the n rows represents a different repetition of the experiment, and each of the p columns gives a particular kind of feature (say, the results from a particular sensor).

Mathematically, the transformation is defined by a set of size l of p-dimensional vectors of weights or coefficients $\mathbf{w}_{(k)} = (w_1, \ldots, w_p)_{(k)}$ that map each row vector $\mathbf{x}_{(i)}$ of $\mathbf{X}$ to a new vector of principal component *scores* $\mathbf{t}_{(i)} = (t_1, \ldots, t_l)_{(i)}$, given by

$$t_{k(i)} = \mathbf{x}_{(i)} \cdot \mathbf{w}_{(k)} \qquad \text{for} \qquad i = 1, \ldots, n \qquad k = 1, \ldots, l$$

in such a way that the individual variables $t_1, \ldots, t_l$ of $\mathbf{t}$ considered over the data set successively inherit the maximum possible variance from $\mathbf{X}$, with each coefficient vector $\mathbf{w}$ constrained to be a <u>unit vector</u> (where l is usually selected to be less than p to reduce dimensionality).

First component

In order to maximize variance, the first weight vector $\mathbf{w}_{(1)}$ thus has to satisfy

$$\mathbf{w}_{(1)} = \underset{\|\mathbf{w}\|=1}{\arg\max} \left\{ \sum_i (t_1)^2_{(i)} \right\} = \underset{\|\mathbf{w}\|=1}{\arg\max} \left\{ \sum_i \left(\mathbf{x}_{(i)} \cdot \mathbf{w} \right)^2 \right\}$$

Equivalently, writing this in matrix form gives

$$\mathbf{w}_{(1)} = \underset{\|\mathbf{w}\|=1}{\arg\max} \left\{ \|\mathbf{X}\mathbf{w}\|^2 \right\} = \underset{\|\mathbf{w}\|=1}{\arg\max} \left\{ \mathbf{w}^T \mathbf{X}^T \mathbf{X} \mathbf{w} \right\}$$

Since $\mathbf{w}_{(1)}$ has been defined to be a unit vector, it equivalently also satisfies

$$\mathbf{w}_{(1)} = \arg\max \left\{ \frac{\mathbf{w}^T \mathbf{X}^T \mathbf{X} \mathbf{w}}{\mathbf{w}^T \mathbf{w}} \right\}$$

The quantity to be maximised can be recognised as a <u>Rayleigh quotient</u>. A standard result for a <u>positive semidefinite matrix</u> such as $\mathbf{X}^T\mathbf{X}$ is that the quotient's maximum possible value is the largest <u>eigenvalue</u> of the matrix, which occurs when w is the corresponding <u>eigenvector</u>.

With $\mathbf{w}_{(1)}$ found, the first principal component of a data vector $\mathbf{x}_{(i)}$ can then be given as a score $t_{1(i)} = \mathbf{x}_{(i)} \cdot \mathbf{w}_{(1)}$ in the transformed co-ordinates, or as the corresponding vector in the original variables, $\{\mathbf{x}_{(i)} \cdot \mathbf{w}_{(1)}\} \mathbf{w}_{(1)}$.

Further components

The kth component can be found by subtracting the first $k - 1$ principal components from $\mathbf{X}$:

$$\hat{\mathbf{X}}_k = \mathbf{X} - \sum_{s=1}^{k-1} \mathbf{X} \mathbf{w}_{(s)} \mathbf{w}_{(s)}^{\mathrm{T}}$$

and then finding the weight vector which extracts the maximum variance from this new data matrix

$$\mathbf{w}_{(k)} = \arg\max_{\|\mathbf{w}\|=1} \left\{ \|\hat{\mathbf{X}}_k \mathbf{w}\|^2 \right\} = \arg\max \left\{ \frac{\mathbf{w}^T \hat{\mathbf{X}}_k^T \hat{\mathbf{X}}_k \mathbf{w}}{\mathbf{w}^T \mathbf{w}} \right\}$$

It turns out that this gives the remaining eigenvectors of $\mathbf{X}^{\mathrm{T}}\mathbf{X}$, with the maximum values for the quantity in brackets given by their corresponding eigenvalues. Thus the weight vectors are eigenvectors of $\mathbf{X}^{\mathrm{T}}\mathbf{X}$.

The kth principal component of a data vector $\mathbf{x}_{(i)}$ can therefore be given as a score $t_{k(i)} = \mathbf{x}_{(i)} \cdot \mathbf{w}_{(k)}$ in the transformed co-ordinates, or as the corresponding vector in the space of the original variables, $\{\mathbf{x}_{(i)} \cdot \mathbf{w}_{(k)}\}$ $\mathbf{w}_{(k)}$, where $\mathbf{w}_{(k)}$ is the kth eigenvector of $\mathbf{X}^{\mathrm{T}}\mathbf{X}$.

The full principal components decomposition of $\mathbf{X}$ can therefore be given as

$$\mathbf{T} = \mathbf{X}\mathbf{W}$$

where $\mathbf{W}$ is a p-by-p matrix of weights whose columns are the eigenvectors of $\mathbf{X}^{\mathrm{T}}\mathbf{X}$. The transpose of $\mathbf{W}$ is sometimes called the <u>whitening or sphering transformation</u>. Columns of $\mathbf{W}$ multiplied by the square root of corresponding eigenvalues, that is, eigenvectors scaled up by the variances, are called *loadings* in PCA or in Factor analysis.

Covariances

$\mathbf{X}^{\mathrm{T}}\mathbf{X}$ itself can be recognised as proportional to the empirical sample <u>covariance matrix</u> of the dataset $\mathbf{X}^{\mathbf{T}}$.

The sample covariance Q between two of the different principal components over the dataset is given by:

$$\begin{aligned}
Q(\mathrm{PC}_{(j)}, \mathrm{PC}_{(k)}) &\propto (\mathbf{X}\mathbf{w}_{(j)})^T (\mathbf{X}\mathbf{w}_{(k)}) \\
&= \mathbf{w}_{(j)}^T \mathbf{X}^T \mathbf{X} \mathbf{w}_{(k)} \\
&= \mathbf{w}_{(j)}^T \lambda_{(k)} \mathbf{w}_{(k)} \\
&= \lambda_{(k)} \mathbf{w}_{(j)}^T \mathbf{w}_{(k)}
\end{aligned}$$

where the eigenvalue property of $\mathbf{w}_{(k)}$ has been used to move from line 2 to line 3. However eigenvectors $\mathbf{w}_{(j)}$ and $\mathbf{w}_{(k)}$ corresponding to eigenvalues of a symmetric matrix are orthogonal (if the eigenvalues are different), or can be orthogonalised (if the vectors happen to share an equal repeated value). The product in the final line is therefore zero; there is no sample covariance between different principal components over the dataset.

Another way to characterise the principal components transformation is therefore as the transformation to coordinates which diagonalise the empirical sample covariance matrix.

In matrix form, the empirical covariance matrix for the original variables can be written

$$\mathbf{Q} \propto \mathbf{X}^T \mathbf{X} = \mathbf{W}\mathbf{\Lambda}\mathbf{W}^T$$

The empirical covariance matrix between the principal components becomes

$$\mathbf{W}^T\mathbf{Q}\mathbf{W} \propto \mathbf{W}^T\mathbf{W}\,\mathbf{\Lambda}\,\mathbf{W}^T\mathbf{W} = \mathbf{\Lambda}$$

where $\mathbf{\Lambda}$ is the diagonal matrix of eigenvalues $\lambda_{(k)}$ of $\mathbf{X}^T\mathbf{X}$. $\lambda_{(k)}$ is equal to the sum of the squares over the dataset associated with each component k, that is, $\lambda_{(k)} = \Sigma_i\, t_k{}^2{}_{(i)} = \Sigma_i\, (\mathbf{x}_{(i)} \cdot \mathbf{w}_{(k)})^2$.

Dimensionality reduction

The transformation $\mathbf{T} = \mathbf{X}\,\mathbf{W}$ maps a data vector $\mathbf{x}_{(i)}$ from an original space of p variables to a new space of p variables which are uncorrelated over the dataset. However, not all the principal components need to be kept. Keeping only the first L principal components, produced by using only the first L eigenvectors, gives the truncated transformation
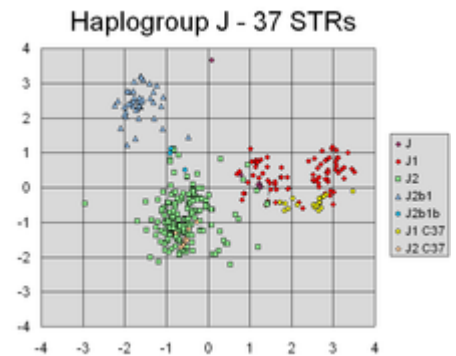
$$\mathbf{T}_L = \mathbf{X}\mathbf{W}_L$$

where the matrix $\mathbf{T}_L$ now has n rows but only L columns. In other words, PCA learns a linear transformation $t = W^T x,\, x \in R^p,\, t \in R^L,$ where the columns of $p \times L$ matrix W form an orthogonal basis for the L features (the components of representation t) that are decorrelated.[12] By construction, of all the transformed data matrices with only L columns, this score matrix maximises the variance in the original data that has been preserved, while minimising the total squared reconstruction error $\|\mathbf{T}\mathbf{W}^T - \mathbf{T}_L\mathbf{W}_L^T\|_2^2$ or $\|\mathbf{X} - \mathbf{X}_L\|_2^2$.

Such dimensionality reduction can be a very useful step for visualising and processing high-dimensional datasets, while still retaining as much of the variance in the dataset as possible. For example, selecting $L = 2$ and keeping only the first two principal components finds the two-dimensional plane through the high-dimensional dataset in which the data is most spread out, so if the data contains clusters these too may be most spread out, and therefore most visible to be plotted out in a two-dimensional diagram; whereas if two directions through the data (or two of the original variables) are chosen at random, the clusters may be much less spread apart from each other, and may in fact be much more likely to substantially overlay each other, making them indistinguishable.

Similarly, in regression analysis, the larger the number of explanatory variables allowed, the greater is the chance of overfitting the model, producing conclusions that fail to generalise to other datasets. One approach, especially when there are strong correlations between different possible explanatory variables, is to reduce them to a few principal components and then run the regression against them, a method called principal component regression.

A principal components analysis scatterplot of Y-STR haplotypes calculated from repeat-count values for 37 Y-chromosomal STR markers from 354 individuals.
PCA has successfully found linear combinations of the different markers, that separate out different clusters corresponding to different lines of individuals' Y-chromosomal genetic descent.

Dimensionality reduction may also be appropriate when the variables in a dataset are noisy. If each column of the dataset contains independent identically distributed Gaussian noise, then the columns of $\mathbf{T}$ will also contain similarly identically distributed Gaussian noise (such a distribution is invariant under the effects of the matrix $\mathbf{W}$, which can be thought of as a high-dimensional rotation of the co-ordinate axes). However, with more of the total variance concentrated in the first few principal components compared to the same noise variance, the proportionate effect of the noise is less—the first few components achieve a higher signal-to-noise ratio. PCA thus can have the effect of

concentrating much of the signal into the first few principal components, which can usefully be captured by dimensionality reduction; while the later principal components may be dominated by noise, and so disposed of without great loss.

Singular value decomposition

The principal components transformation can also be associated with another matrix factorization, the singular value decomposition (SVD) of $\mathbf{X}$,

$$\mathbf{X} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{W}^T$$

Here $\boldsymbol{\Sigma}$ is an n-by-p rectangular diagonal matrix of positive numbers $\sigma_{(k)}$, called the singular values of $\mathbf{X}$; $\mathbf{U}$ is an n-by-n matrix, the columns of which are orthogonal unit vectors of length n called the left singular vectors of $\mathbf{X}$; and $\mathbf{W}$ is a p-by-p whose columns are orthogonal unit vectors of length p and called the right singular vectors of $\mathbf{X}$.

In terms of this factorization, the matrix $\mathbf{X}^T\mathbf{X}$ can be written

$$\begin{aligned}\mathbf{X}^T\mathbf{X} &= \mathbf{W}\boldsymbol{\Sigma}^T\mathbf{U}^T\mathbf{U}\boldsymbol{\Sigma}\mathbf{W}^T \\ &= \mathbf{W}\boldsymbol{\Sigma}^T\boldsymbol{\Sigma}\mathbf{W}^T \\ &= \mathbf{W}\hat{\boldsymbol{\Sigma}}^2\mathbf{W}^T\end{aligned}$$

where $\hat{\boldsymbol{\Sigma}}$ is the square diagonal matrix with the singular values of $\mathbf{X}$ and the excess zeros chopped off that satisfies $\hat{\boldsymbol{\Sigma}}^2 = \boldsymbol{\Sigma}^T\boldsymbol{\Sigma}$. Comparison with the eigenvector factorization of $\mathbf{X}^T\mathbf{X}$ establishes that the right singular vectors $\mathbf{W}$ of $\mathbf{X}$ are equivalent to the eigenvectors of $\mathbf{X}^T\mathbf{X}$, while the singular values $\sigma_{(k)}$ of $\mathbf{X}$ are equal to the square-root of the eigenvalues $\lambda_{(k)}$ of $\mathbf{X}^T\mathbf{X}$.

Using the singular value decomposition the score matrix $\mathbf{T}$ can be written

$$\begin{aligned}\mathbf{T} &= \mathbf{X}\mathbf{W} \\ &= \mathbf{U}\boldsymbol{\Sigma}\mathbf{W}^T\mathbf{W} \\ &= \mathbf{U}\boldsymbol{\Sigma}\end{aligned}$$

so each column of $\mathbf{T}$ is given by one of the left singular vectors of $\mathbf{X}$ multiplied by the corresponding singular value. This form is also the polar decomposition of $\mathbf{T}$.

Efficient algorithms exist to calculate the SVD of $\mathbf{X}$ without having to form the matrix $\mathbf{X}^T\mathbf{X}$, so computing the SVD is now the standard way to calculate a principal components analysis from a data matrix, unless only a handful of components are required.

As with the eigen-decomposition, a truncated $n \times L$ score matrix $\mathbf{T}_L$ can be obtained by considering only the first L largest singular values and their singular vectors:

$$\mathbf{T}_L = \mathbf{U}_L\boldsymbol{\Sigma}_L = \mathbf{X}\mathbf{W}_L$$

The truncation of a matrix $\mathbf{M}$ or $\mathbf{T}$ using a truncated singular value decomposition in this way produces a truncated matrix that is the nearest possible matrix of rank L to the original matrix, in the sense of the difference between the two having the smallest possible Frobenius norm, a result known as the Eckart–Young theorem [1936].

Further considerations

Given a set of points in Euclidean space, the first principal component corresponds to a line that passes through the multidimensional mean and minimizes the sum of squares of the distances of the points from the line. The second principal component corresponds to the same concept after all correlation with the first principal component has been subtracted from the points. The singular values (in Σ) are the square roots of the eigenvalues of the matrix $\mathbf{X}^T\mathbf{X}$. Each eigenvalue is proportional to the portion of the "variance" (more correctly of the sum of the squared distances of the points from their multidimensional mean) that is associated with each eigenvector. The sum of all the eigenvalues is equal to the sum of the squared distances of the points from their multidimensional mean. PCA essentially rotates the set of points around their mean in order to align with the principal components. This moves as much of the variance as possible (using an orthogonal transformation) into the first few dimensions. The values in the remaining dimensions, therefore, tend to be small and may be dropped with minimal loss of information (see below). PCA is often used in this manner for dimensionality reduction. PCA has the distinction of being the optimal orthogonal transformation for keeping the subspace that has largest "variance" (as defined above). This advantage, however, comes at the price of greater computational requirements if compared, for example, and when applicable, to the discrete cosine transform, and in particular to the DCT-II which is simply known as the "DCT". Nonlinear dimensionality reduction techniques tend to be more computationally demanding than PCA.

PCA is sensitive to the scaling of the variables. If we have just two variables and they have the same sample variance and are positively correlated, then the PCA will entail a rotation by 45° and the "weights" (they are the cosines of rotation) for the two variables with respect to the principal component will be equal. But if we multiply all values of the first variable by 100, then the first principal component will be almost the same as that variable, with a small contribution from the other variable, whereas the second component will be almost aligned with the second original variable. This means that whenever the different variables have different units (like temperature and mass), PCA is a somewhat arbitrary method of analysis. (Different results would be obtained if one used Fahrenheit rather than Celsius for example.) Pearson's original paper was entitled "On Lines and Planes of Closest Fit to Systems of Points in Space" – "in space" implies physical Euclidean space where such concerns do not arise. One way of making the PCA less arbitrary is to use variables scaled so as to have unit variance, by standardizing the data and hence use the autocorrelation matrix instead of the autocovariance matrix as a basis for PCA. However, this compresses (or expands) the fluctuations in all dimensions of the signal space to unit variance.

Mean subtraction (a.k.a. "mean centering") is necessary for performing classical PCA to ensure that the first principal component describes the direction of maximum variance. If mean subtraction is not performed, the first principal component might instead correspond more or less to the mean of the data. A mean of zero is needed for finding a basis that minimizes the mean square error of the approximation of the data.[13]

Mean-centering is unnecessary if performing a principal components analysis on a correlation matrix, as the data are already centered after calculating correlations. Correlations are derived from the cross-product of two standard scores (Z-scores) or statistical moments (hence the name: *Pearson Product-Moment Correlation*). Also see the article by Kromrey & Foster-Johnson (1998) on *"Mean-centering in Moderated Regression: Much Ado About Nothing"*.

PCA is a popular primary technique in pattern recognition. It is not, however, optimized for class separability.[14] However, it has been used to quantify the distance between two or more classes by calculating center of mass for each class in principal component space and reporting Euclidean distance between center of mass of two or more classes.[15] The linear discriminant analysis is an alternative which is optimized for class separability.

Table of symbols and abbreviations

Symbol	Meaning	Dimensions	Indices
$\mathbf{X} = \{X_{ij}\}$	data matrix, consisting of the set of all data vectors, one vector per row	$n \times p$	$i = 1 \ldots n$ $j = 1 \ldots p$
n	the number of row vectors in the data set	1×1	*scalar*
p	the number of elements in each row vector (dimension)	1×1	*scalar*
L	the number of dimensions in the dimensionally reduced subspace, $1 \leq L \leq p$	1×1	*scalar*
$\mathbf{u} = \{u_j\}$	vector of empirical means, one mean for each column *j* of the data matrix	$p \times 1$	$j = 1 \ldots p$
$\mathbf{s} = \{s_j\}$	vector of empirical standard deviations, one standard deviation for each column *j* of the data matrix	$p \times 1$	$j = 1 \ldots p$
$\mathbf{h} = \{h_i\}$	vector of all 1's	$1 \times n$	$i = 1 \ldots n$
$\mathbf{B} = \{B_{ij}\}$	deviations from the mean of each column *j* of the data matrix	$n \times p$	$i = 1 \ldots n$ $j = 1 \ldots p$
$\mathbf{Z} = \{Z_{ij}\}$	z-scores, computed using the mean and standard deviation for each row *m* of the data matrix	$n \times p$	$i = 1 \ldots n$ $j = 1 \ldots p$
$\mathbf{C} = \{C_{jj'}\}$	covariance matrix	$p \times p$	$j = 1 \ldots p$ $j' = 1 \ldots p$
$\mathbf{R} = \{R_{jj'}\}$	correlation matrix	$p \times p$	$j = 1 \ldots p$ $j' = 1 \ldots p$
$\mathbf{V} = \{V_{jj'}\}$	matrix consisting of the set of all eigenvectors of **C**, one eigenvector per column	$p \times p$	$j = 1 \ldots p$ $j' = 1 \ldots p$
$\mathbf{D} = \{D_{jj'}\}$	diagonal matrix consisting of the set of all eigenvalues of **C** along its principal diagonal, and 0 for all other elements	$p \times p$	$j = 1 \ldots p$ $j' = 1 \ldots p$
$\mathbf{W} = \{W_{jl}\}$	matrix of basis vectors, one vector per column, where each basis vector is one of the eigenvectors of **C**, and where the vectors in **W** are a subset of those in **V**	$p \times L$	$j = 1 \ldots p$ $l = 1 \ldots L$
$\mathbf{T} = \{T_{il}\}$	matrix consisting of *n* row vectors, where each vector is the projection of the corresponding data vector from matrix **X** onto the basis vectors contained in the columns of matrix **W**.	$n \times L$	$i = 1 \ldots n$ $l = 1 \ldots L$

Properties and limitations of PCA

Properties

Some properties of PCA include:[16]

> ***Property 1:*** For any integer *q, 1 ≤ q ≤ p,* consider the orthogonal linear transformation
>
> $$y = \mathbf{B}'x$$
>
> where y is a *q-element* vector and $\mathbf{B}'$ is a *(q × p)* matrix, and let $\boldsymbol{\Sigma}_y = \mathbf{B}'\boldsymbol{\Sigma}\mathbf{B}$ be the variance-covariance matrix for y. Then the trace of $\boldsymbol{\Sigma}_y$, denoted $\mathrm{tr}(\boldsymbol{\Sigma}_y)$, is maximized by taking $\mathbf{B} = \mathbf{A}_q$, where $\mathbf{A}_q$ consists of the first *q* columns of $\mathbf{A}$ ($\mathbf{B}'$ is the transposition of $\mathbf{B}$).

Property 2: Consider again the <u>orthonormal transformation</u>

$$y = \mathbf{B}'x$$

with $x, \mathbf{B}, \mathbf{A}$ and $\mathbf{\Sigma}_y$ defined as before. Then $\mathrm{tr}(\mathbf{\Sigma}_y)$ is minimized by taking $\mathbf{B} = \mathbf{A}_q^*$, where $\mathbf{A}_q^*$ consists of the last q columns of $\mathbf{A}$.

The statistical implication of this property is that the last few PCs are not simply unstructured left-overs after removing the important PCs. Because these last PCs have variances as small as possible they are useful in their own right. They can help to detect unsuspected near-constant linear relationships between the elements of x, and they may also be useful in <u>regression</u>, in selecting a subset of variables from x, and in outlier detection.

Property 3: (Spectral Decomposition of $\mathbf{\Sigma}$)

$$\mathbf{\Sigma} = \lambda_1\alpha_1\alpha_1' + \cdots + \lambda_p\alpha_p\alpha_p'$$

Before we look at its usage, we first look at <u>diagonal</u> elements,

$$\mathrm{Var}(x_j) = \sum_{k=1}^{P} \lambda_k\alpha_{kj}^2$$

Then, perhaps the main statistical implication of the result is that not only can we decompose the combined variances of all the elements of x into decreasing contributions due to each PC, but we can also decompose the whole <u>covariance matrix</u> into contributions $\lambda_k\alpha_k\alpha_k'$ from each PC. Although not strictly decreasing, the elements of $\lambda_k\alpha_k\alpha_k'$ will tend to become smaller as k increases, as $\lambda_k\alpha_k\alpha_k'$ is nonincreasing for increasing k, whereas the elements of α_k tend to stay about the same size because of the normalization constraints: $\alpha_k'\alpha_k = 1, k = 1, \cdots, p.$

Limitations

As noted above, the results of PCA depend on the scaling of the variables. This can be cured by scaling each feature by its standard deviation, so that one ends up with dimensionless features with unital variance.[17]

The applicability of PCA as described above is limited by certain (tacit) assumptions[18] made in its derivation. In particular, PCA can capture linear correlations between the features but fails when this assumption is violated (see Figure 6a in the reference). In some cases, coordinate transformations can restore the linearity assumption and PCA can then be applied (see <u>kernel PCA</u>).

Another limitation is the mean-removal process before constructing the covariance matrix for PCA. In fields such as astronomy, all the signals are non-negative, and the mean-removal process will force the mean of some astrophysical exposures to be zero, which consequently creates unphysical negative fluxes,[19] and forward modeling has to be performed to recover the true magnitude of the signals.[20] As an alternative method, <u>non-negative matrix factorization</u> focusing only on the non-negative elements in the matrices, which is well-suited for astrophysical observations.[21][22][23] See more at <u>Relation between PCA and Non-negative Matrix Factorization</u>.

PCA and information theory

Dimensionality reduction loses information, in general. PCA-based dimensionality reduction tends to minimize that information loss, under certain signal and noise models.

Under the assumption that

$$\mathbf{x} = \mathbf{s} + \mathbf{n},$$

that is, that the data vector $\mathbf{x}$ is the sum of the desired information-bearing signal $\mathbf{s}$ and a noise signal $\mathbf{n}$ one can show that PCA can be optimal for dimensionality reduction, from an information-theoretic point-of-view.

In particular, Linsker showed that if $\mathbf{s}$ is Gaussian and $\mathbf{n}$ is Gaussian noise with a covariance matrix proportional to the identity matrix, the PCA maximizes the mutual information $I(\mathbf{y}; \mathbf{s})$ between the desired information $\mathbf{s}$ and the dimensionality-reduced output $\mathbf{y} = \mathbf{W}_L^T \mathbf{x}$.[24]

If the noise is still Gaussian and has a covariance matrix proportional to the identity matrix (that is, the components of the vector $\mathbf{n}$ are iid), but the information-bearing signal $\mathbf{s}$ is non-Gaussian (which is a common scenario), PCA at least minimizes an upper bound on the *information loss*, which is defined as[25][26]

$$I(\mathbf{x}; \mathbf{s}) - I(\mathbf{y}; \mathbf{s}).$$

The optimality of PCA is also preserved if the noise $\mathbf{n}$ is iid and at least more Gaussian (in terms of the Kullback–Leibler divergence) than the information-bearing signal $\mathbf{s}$.[27] In general, even if the above signal model holds, PCA loses its information-theoretic optimality as soon as the noise $\mathbf{n}$ becomes dependent.

Computing PCA using the covariance method

The following is a detailed description of PCA using the covariance method (see also here (http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf)) as opposed to the correlation method.[28]

The goal is to transform a given data set $\mathbf{X}$ of dimension p to an alternative data set $\mathbf{Y}$ of smaller dimension L. Equivalently, we are seeking to find the matrix $\mathbf{Y}$, where $\mathbf{Y}$ is the Karhunen–Loève transform (KLT) of matrix $\mathbf{X}$:

$$\mathbf{Y} = \mathbb{KLT}\{\mathbf{X}\}$$

Organize the data set

Suppose you have data comprising a set of observations of p variables, and you want to reduce the data so that each observation can be described with only L variables, $L < p$. Suppose further, that the data are arranged as a set of n data vectors $\mathbf{x}_1 \dots \mathbf{x}_n$ with each $\mathbf{x}_i$ representing a single grouped observation of the p variables.

- Write $\mathbf{x}_1 \dots \mathbf{x}_n$ as row vectors, each of which has p columns.
- Place the row vectors into a single matrix $\mathbf{X}$ of dimensions $n \times p$.

Calculate the empirical mean

- Find the empirical mean along each column $j = 1, \dots, p$.
- Place the calculated mean values into an empirical mean vector $\mathbf{u}$ of dimensions $p \times 1$.

$$u_j = \frac{1}{n} \sum_{i=1}^{n} X_{ij}$$

Calculate the deviations from the mean

Mean subtraction is an integral part of the solution towards finding a principal component basis that minimizes the mean square error of approximating the data.[29] Hence we proceed by centering the data as follows:

- Subtract the empirical mean vector $\mathbf{u}^T$ from each row of the data matrix **X**.
- Store mean-subtracted data in the $n \times p$ matrix **B**.

  $$\mathbf{B} = \mathbf{X} - \mathbf{h}\mathbf{u}^T$$
  where **h** is an $n \times 1$ column vector of all 1s:

  $$h_i = 1 \qquad \text{for } i = 1, \ldots, n$$

Find the covariance matrix

- Find the $p \times p$ empirical covariance matrix **C** from matrix **B**:

  $$\mathbf{C} = \frac{1}{n-1}\mathbf{B}^*\mathbf{B}$$

  where $*$ is the conjugate transpose operator. If **B** consists entirely of real numbers, which is the case in many applications, the "conjugate transpose" is the same as the regular transpose.

- The reasoning behind using $n - 1$ instead of n to calculate the covariance is Bessel's correction.

Find the eigenvectors and eigenvalues of the covariance matrix

- Compute the matrix **V** of eigenvectors which diagonalizes the covariance matrix **C**:

  $$\mathbf{V}^{-1}\mathbf{C}\mathbf{V} = \mathbf{D}$$

  where **D** is the diagonal matrix of eigenvalues of **C**. This step will typically involve the use of a computer-based algorithm for computing eigenvectors and eigenvalues. These algorithms are readily available as sub-components of most matrix algebra systems, such as SAS,[30] R, MATLAB,[31][32] Mathematica,[33] SciPy, IDL (Interactive Data Language), or GNU Octave as well as OpenCV.

- Matrix **D** will take the form of an $p \times p$ diagonal matrix, where

  $$D_{kl} = \lambda_k \qquad \text{for } k = l$$

  is the jth eigenvalue of the covariance matrix **C**, and

  $$D_{kl} = 0 \qquad \text{for } k \neq l.$$

- Matrix **V**, also of dimension $p \times p$, contains p column vectors, each of length p, which represent the p eigenvectors of the covariance matrix **C**.
- The eigenvalues and eigenvectors are ordered and paired. The jth eigenvalue corresponds to the jth eigenvector.
- Matrix **V** denotes the matrix of *right* eigenvectors (as opposed to *left* eigenvectors). In general, the matrix of right eigenvectors need *not* be the (conjugate) transpose of the matrix of left eigenvectors.

Rearrange the eigenvectors and eigenvalues

- Sort the columns of the eigenvector matrix **V** and eigenvalue matrix **D** in order of *decreasing* eigenvalue.
- Make sure to maintain the correct pairings between the columns in each matrix.

Compute the cumulative energy content for each eigenvector

- The eigenvalues represent the distribution of the source data's energy among each of the eigenvectors, where the eigenvectors form a <u>basis</u> for the data. The cumulative energy content g for the jth eigenvector is the sum of the energy content across all of the eigenvalues from 1 through j:

$$g_j = \sum_{k=1}^{j} D_{kk} \qquad \text{for} \qquad j = 1, \ldots, p$$

Select a subset of the eigenvectors as basis vectors

- Save the first L columns of **V** as the $p \times L$ matrix **W**:

$$W_{kl} = V_{kl} \qquad \text{for} \qquad k = 1, \ldots, p \qquad l = 1, \ldots, L$$

where

$$1 \leq L \leq p.$$

- Use the vector **g** as a guide in choosing an appropriate value for L. The goal is to choose a value of L as small as possible while achieving a reasonably high value of g on a percentage basis. For example, you may want to choose L so that the cumulative energy g is above a certain threshold, like 90 percent. In this case, choose the smallest value of L such that

$$\frac{g_L}{g_p} \geq 0.9$$

Project the z-scores of the data onto the new basis

- The projected vectors are the columns of the matrix

$$\mathbf{T} = \mathbf{Z} \cdot \mathbf{W} = \mathbb{KLT}\{\mathbf{X}\}.$$

- The rows of matrix **T** represent the <u>Kosambi-Karhunen–Loève</u> transforms (KLT) of the data vectors in the rows of matrix **X**.

Derivation of PCA using the covariance method

Let **X** be a *d*-dimensional random vector expressed as column vector. Without loss of generality, assume **X** has zero mean.

We want to find $(*)$ a $d \times d$ <u>orthonormal transformation matrix</u> **P** so that **PX** has a diagonal covariance matrix (that is, **PX** is a random vector with all its distinct components pairwise uncorrelated).

A quick computation assuming P were unitary yields:

$$
\begin{aligned}
\text{cov}(PX) &= \mathbb{E}[PX\,(PX)^*] \\
&= \mathbb{E}[PX\,X^*P^*] \\
&= P\,\mathbb{E}[XX^*]P^* \\
&= P\,\text{cov}(X)P^{-1}
\end{aligned}
$$

Hence $(*)$ holds if and only if $\text{cov}(X)$ were diagonalisable by P.

This is very constructive, as cov(**X**) is guaranteed to be a non-negative definite matrix and thus is guaranteed to be diagonalisable by some unitary matrix.

Covariance-free computation

In practical implementations, especially with <u>high dimensional data</u> (large p), the naive covariance method is rarely used because it is not efficient due to high computational and memory costs of explicitly determining the covariance matrix. The covariance-free approach avoids the np^2 operations of explicitly calculating and storing the covariance matrix $\mathbf{X^TX}$, instead utilizing one of <u>matrix-free methods</u>, for example, based on the function evaluating the product $\mathbf{X^T(X\,r)}$ at the cost of *2np* operations.

Iterative computation

One way to compute the first principal component efficiently[34] is shown in the following pseudo-code, for a data matrix **X** with zero mean, without ever computing its covariance matrix.

```
r = a random vector of length p
r = r/|r|
do c times:
      s = 0 (a vector of length p)
      for each row x ∈ X
            s = s + (x · r)x
      eigenvalue = rᵀs
      error = |eigenvalue · r − s|
      r = s/|s|
      exit if error < tolerance
return eigenvalue, r
```

This <u>power iteration</u> algorithm simply calculates the vector $\mathbf{X^T(X\,r)}$, normalizes, and places the result back in **r**. The eigenvalue is approximated by $\mathbf{r^T\,(X^TX)\,r}$, which is the <u>Rayleigh quotient</u> on the unit vector **r** for the covariance matrix $\mathbf{X^TX}$. If the largest singular value is well separated from the next largest one, the vector **r** gets close to the first principal component of **X** within the number of iterations c, which is small

relative to p, at the total cost $2cnp$. The power iteration convergence can be accelerated without noticeably sacrificing the small cost per iteration using more advanced matrix-free methods, such as the Lanczos algorithm or the Locally Optimal Block Preconditioned Conjugate Gradient (LOBPCG) method.

Subsequent principal components can be computed one-by-one via deflation or simultaneously as a block. In the former approach, imprecisions in already computed approximate principal components additively affect the accuracy of the subsequently computed principal components, thus increasing the error with every new computation. The latter approach in the block power method replaces single-vectors $\mathbf{r}$ and $\mathbf{s}$ with block-vectors, matrices $\mathbf{R}$ and $\mathbf{S}$. Every column of $\mathbf{R}$ approximates one of the leading principal components, while all columns are iterated simultaneously. The main calculation is evaluation of the product $\mathbf{X^T(X\ R)}$. Implemented, for example., in LOBPCG, efficient blocking eliminates the accumulation of the errors, allows using high-level BLAS matrix-matrix product functions, and typically leads to faster convergence, compared to the single-vector one-by-one technique.

The NIPALS method

Non-linear iterative partial least squares (NIPALS) is a variant the classical power iteration with matrix deflation by subtraction implemented for computing the first few components in a principal component or partial least squares analysis. For very-high-dimensional datasets, such as those generated in the *omics sciences (for example, genomics, metabolomics) it is usually only necessary to compute the first few PCs. The non-linear iterative partial least squares (NIPALS) algorithm updates iterative approximations to the leading scores and loadings $\mathbf{t_1}$ and $\mathbf{r_1}^T$ by the power iteration multiplying on every iteration by $\mathbf{X}$ on the left and on the right, that is, calculation of the covariance matrix is avoided, just as in the matrix-free implementation of the power iterations to $\mathbf{X^T X}$, based on the function evaluating the product $\mathbf{X^T(X\ r)} = \mathbf{((X\ r)^T X)^T}$.

The matrix deflation by subtraction is performed by subtracting the outer product, $\mathbf{t_1 r_1}^T$ from $\mathbf{X}$ leaving the deflated residual matrix used to calculate the subsequent leading PCs.[35] For large data matrices, or matrices that have a high degree of column collinearity, NIPALS suffers from loss of orthogonality of PCs due to machine precision round-off errors accumulated in each iteration and matrix deflation by subtraction.[36] A Gram–Schmidt re-orthogonalization algorithm is applied to both the scores and the loadings at each iteration step to eliminate this loss of orthogonality.[37] NIPALS reliance on single-vector multiplications cannot take advantage of high-level BLAS and results in slow convergence for clustered leading singular values—both these deficiencies are resolved in more sophisticated matrix-free block solvers, such as the Locally Optimal Block Preconditioned Conjugate Gradient (LOBPCG) method.

Online/sequential estimation

In an "online" or "streaming" situation with data arriving piece by piece rather than being stored in a single batch, it is useful to make an estimate of the PCA projection that can be updated sequentially. This can be done efficiently, but requires different algorithms.[38]

PCA and qualitative variables

In PCA, it is common that we want to introduce qualitative variables as supplementary elements. For example, many quantitative variables have been measured on plants. For these plants, some qualitative variables are available as, for example, the species to which the plant belongs. These data were subjected to PCA for quantitative variables. When analyzing the results, it is natural to connect the principal components to the qualitative variable *species*. For this, the following results are produced.

- Identification, on the factorial planes, of the different species, for example, using different colors.
- Representation, on the factorial planes, of the centers of gravity of plants belonging to the same species.
- For each center of gravity and each axis, p-value to judge the significance of the difference between the center of gravity and origin.

These results are what is called *introducing a qualitative variable as supplementary element*. This procedure is detailed in and Husson, Lê & Pagès 2009 and Pagès 2013. Few software offer this option in an "automatic" way. This is the case of SPAD (http://www.coheris.com/produits/analytics/logiciel-data-mining/) that historically, following the work of Ludovic Lebart, was the first to propose this option, and the R package FactoMineR (http://factominer.free.fr/).

Applications

Quantitative finance

In quantitative finance, principal component analysis can be directly applied to the risk management of interest rate derivative portfolios.[39] Trading multiple swap instruments which are usually a function of 30-500 other market quotable swap instruments is sought to be reduced to usually 3 or 4 principal components, representing the path of interest rates on a macro basis. Converting risks to be represented as those to factor loadings (or multipliers) provides assessments and understanding beyond that available to simply collectively viewing risks to individual 30-500 buckets.

PCA has also been applied to equity portfolios in a similar fashion,[40] both to portfolio risk and to risk return. One application is to reduce portfolio risk, where allocation strategies are applied to the "principal portfolios" instead of the underlying stocks.[41] A second is to enhance portfolio return, using the principal components to select stocks with upside potential.

Neuroscience

A variant of principal components analysis is used in neuroscience to identify the specific properties of a stimulus that increase a neuron's probability of generating an action potential.[42] This technique is known as spike-triggered covariance analysis. In a typical application an experimenter presents a white noise process as a stimulus (usually either as a sensory input to a test subject, or as a current injected directly into the neuron) and records a train of action potentials, or spikes, produced by the neuron as a result. Presumably, certain features of the stimulus make the neuron more likely to spike. In order to extract these features, the experimenter calculates the covariance matrix of the *spike-triggered ensemble*, the set of all stimuli (defined and discretized over a finite time window, typically on the order of 100 ms) that immediately preceded a spike. The eigenvectors of the difference between the spike-triggered covariance matrix and the covariance matrix of the *prior stimulus ensemble* (the set of all stimuli, defined over the same length time window) then indicate the directions in the space of stimuli along which the variance of the spike-triggered ensemble differed the most from that of the prior stimulus ensemble. Specifically, the eigenvectors with the largest positive eigenvalues correspond to the directions along which the variance of the spike-triggered ensemble showed the largest positive change compared to the variance of the prior. Since these were the directions in which varying the stimulus led to a spike, they are often good approximations of the sought after relevant stimulus features.

In neuroscience, PCA is also used to discern the identity of a neuron from the shape of its action potential. Spike sorting is an important procedure because extracellular recording techniques often pick up signals from more than one neuron. In spike sorting, one first uses PCA to reduce the dimensionality of the space of action potential waveforms, and then performs clustering analysis to associate specific action potentials with individual neurons.

PCA as a dimension reduction technique is particularly suited to detect coordinated activities of large neuronal ensembles. It has been used in determining collective variables, that is, order parameters, during phase transitions in the brain.[43]

Relation with other methods

Correspondence analysis

Correspondence analysis (CA) was developed by Jean-Paul Benzécri[44] and is conceptually similar to PCA, but scales the data (which should be non-negative) so that rows and columns are treated equivalently. It is traditionally applied to contingency tables. CA decomposes the chi-squared statistic associated to this table into orthogonal factors.[45] Because CA is a descriptive technique, it can be applied to tables for which the chi-squared statistic is appropriate or not. Several variants of CA are available including detrended correspondence analysis and canonical correspondence analysis. One special extension is multiple correspondence analysis, which may be seen as the counterpart of principal component analysis for categorical data.[46]

Factor analysis

Principal component analysis creates variables that are linear combinations of the original variables. The new variables have the property that the variables are all orthogonal. The PCA transformation can be helpful as a pre-processing step before clustering. PCA is a variance-focused approach seeking to reproduce the total variable variance, in which components reflect both common and unique variance of the variable. PCA is generally preferred for purposes of data reduction (that is, translating variable space into optimal factor space) but not when the goal is to detect the latent construct or factors.

Factor analysis is similar to principal component analysis, in that factor analysis also involves linear combinations of variables. Different from PCA, factor analysis is a correlation-focused approach seeking to reproduce the inter-correlations among variables, in which the factors "represent the common variance of variables, excluding unique variance".[47] In terms of the correlation matrix, this corresponds with focusing on explaining the off-diagonal terms (that is, shared co-variance), while PCA focuses on explaining the terms that sit on the diagonal. However, as a side result, when trying to reproduce the on-diagonal terms, PCA also tends to fit relatively well the off-diagonal correlations.[48] Results given by PCA and factor analysis are very similar in most situations, but this is not always the case, and there are some problems where the results are significantly different. Factor analysis is generally used when the research purpose is detecting data structure (that is, latent constructs or factors) or causal modeling. If the factor model is incorrectly formulated or the assumptions are not met, then factor analysis will give erroneous results.[49]

K-means clustering

It was asserted in [50][51] that the relaxed solution of k-means clustering, specified by the cluster indicators, is given by the principal components, and the PCA subspace spanned by the principal directions is identical to the cluster centroid subspace. However, that PCA is a useful relaxation of k-means clustering was not a

new result (see, for example,[52]), and it is straightforward to uncover counterexamples to the statement that the cluster centroid subspace is spanned by the principal directions.[53]

Non-negative matrix factorization

Non-negative matrix factorization (NMF) is a dimension reduction method where only non-negative elements in the matrices are used, which is therefore a promising method in astronomy,[21][22][23] in the sense that astrophysical signals are non-negative. The PCA components are orthogonal to each other, while the NMF components are all non-negative and therefore constructs a non-orthogonal basis.
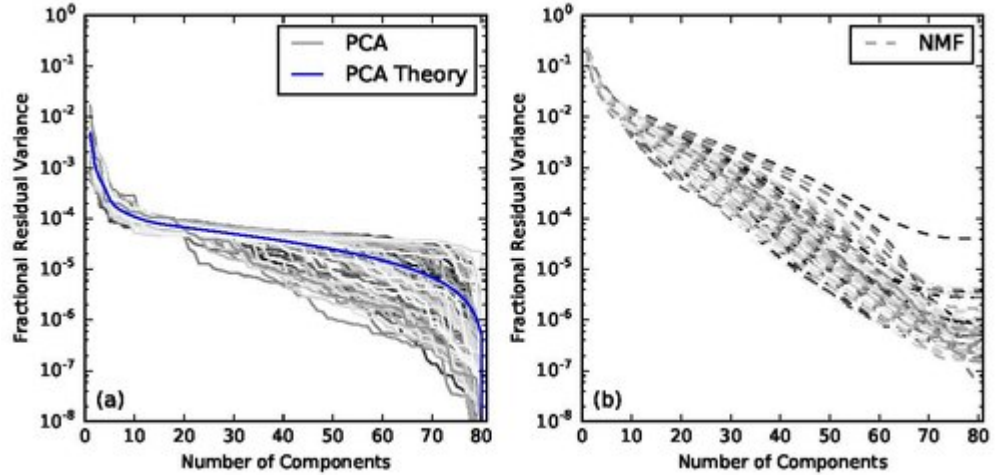
Fractional residual variance (FRV) plots for PCA and NMF;[23] for PCA, the theoretical values are the contribution from the residual eigenvalues. In comparison, the FRV curves for PCA reaches a flat plateau where no signal are captured effectively; while the NMF FRV curves are declining continuously, indicating a better ability to capture signal. The FRV curves for NMF also converges to higher levels than PCA, indicating the less-overfitting property of NMF.

In PCA, the contribution of each component is ranked based on the magnitude of its corresponding eigenvalue, which is equivalent to the fractional residual variance (FRV) in analyzing empirical data.[19] For NMF, its components are ranked based only on the empirical FRV curves.[23] The residual fractional eigenvalue plots, that is,

$$1 - \sum_{i=1}^{k} \lambda_i / \sum_{k=1}^{n} \lambda_k$$ as a function of component number k given a total of n components, for PCA has a flat

plateau, where no data is captured to remove the quasi-static noise, then the curves dropped quickly as an indication of over-fitting and captures random noise.[19] The FRV curves for NMF is decreasing continuously [23] when the NMF components are constructed sequentially,[22] indicating the continuous capturing of quasi-static noise; then converge to higher levels than PCA,[23] indicating the less over-fitting property of NMF.

Generalizations

Sparse PCA

A particular disadvantage of PCA is that the principal components are usually linear combinations of all input variables. Sparse PCA overcomes this disadvantage by finding linear combinations that contain just a few input variables. It extends the classic method of principal component analysis (PCA) for the reduction of dimensionality of data by adding sparsity constraint on the input variables. Several approaches have been proposed, including

- a regression framework,[54]
- a convex relaxation/semidefinite programming framework,[55]

- a generalized power method framework[56]
- an alternating maximization framework[57]
- forward-backward greedy search and exact methods using branch-and-bound techniques,[58]
- Bayesian formulation framework.[59]

The methodological and theoretical developments of Sparse PCA as well as its applications in scientific studies are recently reviewed in a survey paper.[60]
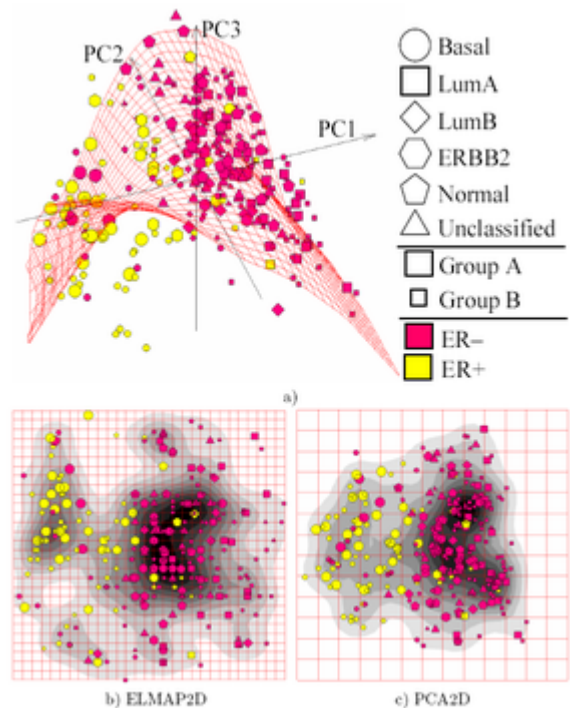
Nonlinear PCA

Most of the modern methods for nonlinear dimensionality reduction find their theoretical and algorithmic roots in PCA or K-means. Pearson's original idea was to take a straight line (or plane) which will be "the best fit" to a set of data points. **Principal curves and manifolds**[64] give the natural geometric framework for PCA generalization and extend the geometric interpretation of PCA by explicitly constructing an embedded manifold for data approximation, and by encoding using standard geometric projection onto the manifold, as it is illustrated by Fig. See also the elastic map algorithm and principal geodesic analysis. Another popular generalization is kernel PCA, which corresponds to PCA performed in a reproducing kernel Hilbert space associated with a positive definite kernel.

In multilinear subspace learning,[65] PCA is generalized to multilinear PCA (MPCA) that extracts features directly from tensor representations. MPCA is solved by performing PCA in each mode of the tensor iteratively. MPCA has been applied to face recognition, gait recognition, etc. MPCA is further extended to uncorrelated MPCA, non-negative MPCA and robust MPCA.

N-way principal component analysis may be performed with models such as Tucker decomposition, PARAFAC, multiple factor analysis, co-inertia analysis, STATIS, and DISTATIS.

Linear PCA versus nonlinear Principal Manifolds[61] for visualization of breast cancer microarray data: a) Configuration of nodes and 2D Principal Surface in the 3D PCA linear manifold. The dataset is curved and cannot be mapped adequately on a 2D principal plane; b) The distribution in the internal 2D non-linear principal surface coordinates (ELMap2D) together with an estimation of the density of points; c) The same as b), but for the linear 2D PCA manifold (PCA2D). The "basal" breast cancer subtype is visualized more adequately with ELMap2D and some features of the distribution become better resolved in comparison to PCA2D. Principal manifolds are produced by the elastic maps algorithm. Data are available for public competition.[62] Software is available for free non-commercial use.[63]

Robust PCA

While PCA finds the mathematically optimal method (as in minimizing the squared error), it is still sensitive to outliers in the data that produce large errors, something that the method tries to avoid in the first place. It is therefore common practice to remove outliers before computing PCA. However, in some contexts, outliers can be difficult to identify. For example, in data

mining algorithms like correlation clustering, the assignment of points to clusters and outliers is not known beforehand. A recently proposed generalization of PCA[66] based on a weighted PCA increases robustness by assigning different weights to data objects based on their estimated relevancy.

Outlier-resistant variants of PCA have also been proposed, based on L1-norm formulations (L1-PCA).[7][5]

Robust principal component analysis (RPCA) via decomposition in low-rank and sparse matrices is a modification of PCA that works well with respect to grossly corrupted observations.[67][68][69]

Similar techniques

Independent component analysis

Independent component analysis (ICA) is directed to similar problems as principal component analysis, but finds additively separable components rather than successive approximations.

Network component analysis

Given a matrix E, it tries to decompose it into two matrices such that $E = AP$. A key difference from techniques such as PCA and ICA is that some of the entries of A are constrained to be 0. Here P is termed the regulatory layer. While in general such a decomposition can have multiple solutions, they prove that if the following conditions are satisfied :

1. A has full column rank
2. Each column of A must have at least $L - 1$ zeroes where L is the number of columns of A (or alternatively the number of rows of P). The justification for this criterion is that if a node is removed from the regulatory layer along with all the output nodes connected to it, the result must still be characterized by a connectivity matrix with full column rank.
3. P must have full row rank.

then the decomposition is unique up to multiplication by a scalar.[70]

Software/source code

- ALGLIB - a C++ and C# library that implements PCA and truncated PCA
- Analytica – The built-in EigenDecomp function computes principal components.
- ELKI – includes PCA for projection, including robust variants of PCA, as well as PCA-based clustering algorithms.
- Gretl – principal component analysis can be performed either via the `pca` command or via the `princomp()` function.
- Julia – Supports PCA with the `pca` function in the MultivariateStats package
- KNIME – A java based nodal arranging software for Analysis, in this the nodes called PCA, PCA compute, PCA Apply, PCA inverse make it easily.
- Mathematica – Implements principal component analysis with the PrincipalComponents command using both covariance and correlation methods.
- MathPHP (https://github.com/markrogoyski/math-php) - PHP mathematics library with support for PCA.

- MATLAB Statistics Toolbox – The functions `princomp` and `pca` (R2012b) give the principal components, while the function `pcares` gives the residuals and reconstructed matrix for a low-rank PCA approximation.
- Matplotlib – Python library have a PCA package in the .mlab module.
- mlpack – Provides an implementation of principal component analysis in C++.
- NAG Library – Principal components analysis is implemented via the `g03aa` routine (available in both the Fortran versions of the Library).
- NMath – Proprietary numerical library containing PCA for the .NET Framework.
- GNU Octave – Free software computational environment mostly compatible with MATLAB, the function `princomp` gives the principal component.
- OpenCV
- Oracle Database 12c – Implemented via `DBMS_DATA_MINING.SVDS_SCORING_MODE` by specifying setting value `SVDS_SCORING_PCA`
- Orange (software) – Integrates PCA in its visual programming environment. PCA displays a scree plot (degree of explained variance) where user can interactively select the number of principal components.
- Origin – Contains PCA in its Pro version.
- Qlucore – Commercial software for analyzing multivariate data with instant response using PCA.
- R – Free statistical package, the functions `princomp` and `prcomp` can be used for principal component analysis; `prcomp` uses singular value decomposition which generally gives better numerical accuracy. Some packages that implement PCA in R, include, but are not limited to: `ade4`, `vegan`, `ExPosition`, `dimRed`, and `FactoMineR`.
- SAS - Proprietary software; for example, see [71]
- Scikit-learn – Python library for machine learning which contains PCA, Probabilistic PCA, Kernel PCA, Sparse PCA and other techniques in the decomposition module.
- Weka – Java library for machine learning which contains modules for computing principal components.

See also

- Correspondence analysis (for contingency tables)
- Multiple correspondence analysis (for qualitative variables)
- Factor analysis of mixed data (for quantitative **and** qualitative variables)
- Canonical correlation
- CUR matrix approximation (can replace of low-rank SVD approximation)
- Detrended correspondence analysis
- Dynamic mode decomposition
- Eigenface
- Exploratory factor analysis (Wikiversity)
- Factorial code
- Functional principal component analysis
- Geometric data analysis
- Independent component analysis
- Kernel PCA
- L1-norm principal component analysis
- Low-rank approximation

- Matrix decomposition
- Non-negative matrix factorization
- Nonlinear dimensionality reduction
- Oja's rule
- Point distribution model (PCA applied to morphometry and computer vision)
- Principal component analysis (Wikibooks)
- Principal component regression
- Singular spectrum analysis
- Singular value decomposition
- Sparse PCA
- Transform coding
- Weighted least squares

References

1. Abdi. H. & Williams, L.J. (2010). "Principal component analysis". *Wiley Interdisciplinary Reviews: Computational Statistics*. **2** (4): 433–459. arXiv:1108.4372 (https://arxiv.org/abs/1108.4372). doi:10.1002/wics.101 (https://doi.org/10.1002%2Fwics.101).

2. Shaw P.J.A. (2003) *Multivariate statistics for the Environmental Sciences*, Hodder-Arnold. ISBN 0-340-80763-6.

3. Barnett, T. P. & R. Preisendorfer. (1987). "Origins and levels of monthly and seasonal forecast skill for United States surface air temperatures determined by canonical correlation analysis". *Monthly Weather Review*. **115** (9): 1825. Bibcode:1987MWRv..115.1825B (https://ui.adsabs.harvard.edu/abs/1987MWRv..115.1825B). doi:10.1175/1520-0493(1987)115<1825:oaloma>2.0.co;2 (https://doi.org/10.1175%2F1520-0493%281987%29115%3C1825%3Aoaloma%3E2.0.co%3B2).

4. Hsu, Daniel; Kakade, Sham M.; Zhang, Tong (2008). *A spectral algorithm for learning hidden markov models*. arXiv:0811.4413 (https://arxiv.org/abs/0811.4413). Bibcode:2008arXiv0811.4413H (https://ui.adsabs.harvard.edu/abs/2008arXiv0811.4413H).

5. Markopoulos, Panos P.; Kundu, Sandipan; Chamadia, Shubham; Pados, Dimitris A. (15 August 2017). "Efficient L1-Norm Principal-Component Analysis via Bit Flipping". *IEEE Transactions on Signal Processing*. **65** (16): 4252–4264. arXiv:1610.01959 (https://arxiv.org/abs/1610.01959). Bibcode:2017ITSP...65.4252M (https://ui.adsabs.harvard.edu/abs/2017ITSP...65.4252M). doi:10.1109/TSP.2017.2708023 (https://doi.org/10.1109%2FTSP.2017.2708023).

6. Chachlakis, Dimitris G.; Prater-Bennette, Ashley; Markopoulos, Panos P. (22 November 2019). "L1-norm Tucker Tensor Decomposition". *IEEE Access*. **7**: 178454–178465. doi:10.1109/ACCESS.2019.2955134 (https://doi.org/10.1109%2FACCESS.2019.2955134).

7. Markopoulos, Panos P.; Karystinos, George N.; Pados, Dimitris A. (October 2014). "Optimal Algorithms for L1-subspace Signal Processing". *IEEE Transactions on Signal Processing*. **62** (19): 5046–5058. arXiv:1405.6785 (https://arxiv.org/abs/1405.6785). Bibcode:2014ITSP...62.5046M (https://ui.adsabs.harvard.edu/abs/2014ITSP...62.5046M). doi:10.1109/TSP.2014.2338077 (https://doi.org/10.1109%2FTSP.2014.2338077).

8. Kanade, T.; Ke, Qifa (June 2005). *Robust L1 Norm Factorization in the Presence of Outliers and Missing Data by Alternative Convex Programming. 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. **1**. IEEE. p. 739. CiteSeerX 10.1.1.63.4605 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.63.4605). doi:10.1109/CVPR.2005.309 (https://doi.org/10.1109%2FCVPR.2005.309). ISBN 978-0-7695-2372-9.

9. Pearson, K. (1901). "On Lines and Planes of Closest Fit to Systems of Points in Space" (http s://zenodo.org/record/1430636). *Philosophical Magazine*. **2** (11): 559–572. doi:10.1080/14786440109462720 (https://doi.org/10.1080%2F14786440109462720).

10. Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, **24**, 417–441, and 498–520.
    Hotelling, H (1936). "Relations between two sets of variates". *Biometrika*. **28** (3/4): 321–377. doi:10.2307/2333955 (https://doi.org/10.2307%2F2333955). JSTOR 2333955 (https://www.jsto r.org/stable/2333955).

11. Jolliffe I.T. *Principal Component Analysis*, Series: Springer Series in Statistics, 2nd ed., Springer, NY, 2002, XXIX, 487 p. 28 illus. ISBN 978-0-387-95442-4

12. Bengio, Y.; et al. (2013). "Representation Learning: A Review and New Perspectives". *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **35** (8): 1798–1828. arXiv:1206.5538 (https://arxiv.org/abs/1206.5538). doi:10.1109/TPAMI.2013.50 (https://doi.org/ 10.1109%2FTPAMI.2013.50). PMID 23787338 (https://pubmed.ncbi.nlm.nih.gov/23787338).

13. A. A. Miranda, Y. A. Le Borgne, and G. Bontempi. New Routes from Minimal Approximation Error to Principal Components (http://www.ulb.ac.be/di/map/yleborgn/pub/NPL_PCA_07.pdf), Volume 27, Number 3 / June, 2008, Neural Processing Letters, Springer

14. Fukunaga, Keinosuke (1990). *Introduction to Statistical Pattern Recognition* (https://books.goo gle.com/books?visbn=0-12-269851-7). Elsevier. ISBN 978-0-12-269851-4.

15. Alizadeh, Elaheh; Lyons, Samanthe M; Castle, Jordan M; Prasad, Ashok (2016). "Measuring systematic changes in invasive cancer cell shape using Zernike moments" (http://pubs.rsc.org/ en/Content/ArticleLanding/2016/IB/C6IB00100A#!divAbstract). *Integrative Biology*. **8** (11): 1183–1193. doi:10.1039/C6IB00100A (https://doi.org/10.1039%2FC6IB00100A). PMID 27735002 (https://pubmed.ncbi.nlm.nih.gov/27735002).

16. Jolliffe, I. T. (2002). *Principal Component Analysis,* second edition Springer-Verlag. ISBN 978-0-387-95442-4.

17. Leznik, M; Tofallis, C. 2005 Estimating Invariant Principal Components Using Diagonal Regression. (https://uhra.herts.ac.uk/bitstream/handle/2299/715/S56.pdf)

18. Jonathon Shlens, A Tutorial on Principal Component Analysis. (https://arxiv.org/abs/1404.110 0)

19. Soummer, Rémi; Pueyo, Laurent; Larkin, James (2012). "Detection and Characterization of Exoplanets and Disks Using Projections on Karhunen-Loève Eigenimages". *The Astrophysical Journal Letters*. **755** (2): L28. arXiv:1207.4197 (https://arxiv.org/abs/1207.4197). Bibcode:2012ApJ...755L..28S (https://ui.adsabs.harvard.edu/abs/2012ApJ...755L..28S). doi:10.1088/2041-8205/755/2/L28 (https://doi.org/10.1088%2F2041-8205%2F755%2F2%2FL2 8).

20. Pueyo, Laurent (2016). "Detection and Characterization of Exoplanets using Projections on Karhunen Loeve Eigenimages: Forward Modeling". *The Astrophysical Journal*. **824** (2): 117. arXiv:1604.06097 (https://arxiv.org/abs/1604.06097). Bibcode:2016ApJ...824..117P (https://ui.a dsabs.harvard.edu/abs/2016ApJ...824..117P). doi:10.3847/0004-637X/824/2/117 (https://doi.or g/10.3847%2F0004-637X%2F824%2F2%2F117).

21. Blanton, Michael R.; Roweis, Sam (2007). "K-corrections and filter transformations in the ultraviolet, optical, and near infrared". *The Astronomical Journal*. **133** (2): 734–754. arXiv:astro-ph/0606170 (https://arxiv.org/abs/astro-ph/0606170). Bibcode:2007AJ....133..734B (https://ui.a dsabs.harvard.edu/abs/2007AJ....133..734B). doi:10.1086/510127 (https://doi.org/10.1086%2F 510127).

22. Zhu, Guangtun B. (2016-12-19). "Nonnegative Matrix Factorization (NMF) with Heteroscedastic Uncertainties and Missing data". arXiv:1612.06037 (https://arxiv.org/abs/1612. 06037) [astro-ph.IM (https://arxiv.org/archive/astro-ph.IM)].

23. Ren, Bin; Pueyo, Laurent; Zhu, Guangtun B.; Duchêne, Gaspard (2018). "Non-negative Matrix Factorization: Robust Extraction of Extended Structures". *The Astrophysical Journal*. **852** (2): 104. arXiv:1712.10317 (https://arxiv.org/abs/1712.10317). Bibcode:2018ApJ...852..104R (https://ui.adsabs.harvard.edu/abs/2018ApJ...852..104R). doi:10.3847/1538-4357/aaa1f2 (https://doi.org/10.3847%2F1538-4357%2Faaa1f2).

24. Linsker, Ralph (March 1988). "Self-organization in a perceptual network". *IEEE Computer*. **21** (3): 105–117. doi:10.1109/2.36 (https://doi.org/10.1109%2F2.36).

25. Deco & Obradovic (1996). *An Information-Theoretic Approach to Neural Computing* (https://books.google.com/books?id=z4XTBwAAQBAJ&printsec=frontcover#v=onepage&q&f=false). New York, NY: Springer. ISBN 9781461240167.

26. Plumbley, Mark (1991). *Information theory and unsupervised neural networks*.Tech Note

27. Geiger, Bernhard; Kubin, Gernot (January 2013). "Signal Enhancement as Minimization of Relevant Information Loss". *Proc. ITG Conf. On Systems, Communication and Coding*. arXiv:1205.6935 (https://arxiv.org/abs/1205.6935). Bibcode:2012arXiv1205.6935G (https://ui.adsabs.harvard.edu/abs/2012arXiv1205.6935G).

28. "Engineering Statistics Handbook Section 6.5.5.2" (http://www.itl.nist.gov/div898/handbook/pmc/section5/pmc552.htm). Retrieved 19 January 2015.

29. A.A. Miranda, Y.-A. Le Borgne, and G. Bontempi. New Routes from Minimal Approximation Error to Principal Components (http://www.ulb.ac.be/di/map/yleborgn/pub/NPL_PCA_07.pdf), Volume 27, Number 3 / June, 2008, Neural Processing Letters, Springer

30. "SAS/STAT(R) 9.3 User's Guide" (http://support.sas.com/documentation/cdl/en/statug/63962/HTML/default/viewer.htm#statug_princomp_sect001.htm).

31. eig function (http://www.mathworks.com/access/helpdesk/help/techdoc/ref/eig.html#998306) Matlab documentation

32. MATLAB PCA-based Face recognition software (http://www.mathworks.com/matlabcentral/fileexchange/24634)

33. Eigenvalues function (http://reference.wolfram.com/mathematica/ref/Eigenvalues.html) Mathematica documentation

34. Roweis, Sam. "EM Algorithms for PCA and SPCA." Advances in Neural Information Processing Systems. Ed. Michael I. Jordan, Michael J. Kearns, and Sara A. Solla The MIT Press, 1998.

35. Geladi, Paul; Kowalski, Bruce (1986). "Partial Least Squares Regression:A Tutorial". *Analytica Chimica Acta*. **185**: 1–17. doi:10.1016/0003-2670(86)80028-9 (https://doi.org/10.1016%2F0003-2670%2886%2980028-9).

36. Kramer, R. (1998). *Chemometric Techniques for Quantitative Analysis* (https://books.google.com/books?id=iBpOzwAOfHYC&printsec=frontcover#v=onepage&q&f=false). New York: CRC Press. ISBN 9780203909805.

37. Andrecut, M. (2009). "Parallel GPU Implementation of Iterative PCA Algorithms". *Journal of Computational Biology*. **16** (11): 1593–1599. arXiv:0811.1081 (https://arxiv.org/abs/0811.1081). doi:10.1089/cmb.2008.0221 (https://doi.org/10.1089%2Fcmb.2008.0221). PMID 19772385 (https://pubmed.ncbi.nlm.nih.gov/19772385).

38. Warmuth, M. K.; Kuzmin, D. (2008). "Randomized online PCA algorithms with regret bounds that are logarithmic in the dimension" (http://www.jmlr.org/papers/volume9/warmuth08a/warmuth08a.pdf) (PDF). *Journal of Machine Learning Research*. **9**: 2287–2320.

39. The Pricing and Hedging of Interest Rate Derivatives: A Practical Guide to Swaps (http://www.tradinginterestrates.com), J H M Darbyshire, 2016, ISBN 978-0995455511

40. Giorgia Pasini (2017); Principal Component Analysis for Stock Portfolio Management (https://ijpam.eu/contents/2017-115-1/12/12.pdf). *International Journal of Pure and Applied Mathematics*. Volume 115 No. 1 2017, 153-167

41. Libin Yang. *An Application of Principal Component Analysis to Stock Portfolio Management* (https://ir.canterbury.ac.nz/bitstream/handle/10092/10293/thesis.pdf?sequence=1). Department of Economics and Finance, University of Canterbury, January 2015.

42. Brenner, N., Bialek, W., & de Ruyter van Steveninck, R.R. (2000).

43. Jirsa, Victor; Friedrich, R; Haken, Herman; Kelso, Scott (1994). "A theoretical model of phase transitions in the human brain". *Biological Cybernetics*. **71** (1): 27–35. doi:10.1007/bf00198909 (https://doi.org/10.1007%2Fbf00198909). PMID 8054384 (https://pubmed.ncbi.nlm.nih.gov/8054384).

44. Benzécri, J.-P. (1973). *L'Analyse des Données. Volume II. L'Analyse des Correspondances*. Paris, France: Dunod.

45. Greenacre, Michael (1983). *Theory and Applications of Correspondence Analysis*. London: Academic Press. ISBN 978-0-12-299050-2.

46. Le Roux; Brigitte and Henry Rouanet (2004). *Geometric Data Analysis, From Correspondence Analysis to Structured Data Analysis* (https://books.google.com/books?id=a6bDBUF58XwC&printsec=frontcover#v=snippet&q=%22multiple%20correspondence%20analysis%22&f=false). Dordrecht: Kluwer. ISBN 9781402022357.

47. Timothy A. Brown. Confirmatory Factor Analysis for Applied Research Methodology in the social sciences (https://books.google.com/books?id=JDb3BQAAQBAJ&printsec=frontcover#v=onepage&q&f=false). Guilford Press, 2006

48. I.T. Jolliffe. Principal Component Analysis, Second Edition. Chapter 7. 2002

49. Meglen, R.R. (1991). "Examining Large Databases: A Chemometric Approach Using Principal Component Analysis". *Journal of Chemometrics*. **5** (3): 163–179. doi:10.1002/cem.1180050305 (https://doi.org/10.1002%2Fcem.1180050305).

50. H. Zha; C. Ding; M. Gu; X. He; H.D. Simon (Dec 2001). "Spectral Relaxation for K-means Clustering" (http://ranger.uta.edu/~chqding/papers/Zha-Kmeans.pdf) (PDF). *Neural Information Processing Systems Vol.14 (NIPS 2001)*: 1057–1064.

51. Chris Ding; Xiaofeng He (July 2004). "K-means Clustering via Principal Component Analysis" (http://ranger.uta.edu/~chqding/papers/KmeansPCA1.pdf) (PDF). *Proc. Of Int'l Conf. Machine Learning (ICML 2004)*: 225–232.

52. Drineas, P.; A. Frieze; R. Kannan; S. Vempala; V. Vinay (2004). "Clustering large graphs via the singular value decomposition" (http://www.cc.gatech.edu/~vempala/papers/dfkvv.pdf) (PDF). *Machine Learning*. **56** (1–3): 9–33. doi:10.1023/b:mach.0000033113.59016.96 (https://doi.org/10.1023%2Fb%3Amach.0000033113.59016.96). Retrieved 2012-08-02.

53. Cohen, M.; S. Elder; C. Musco; C. Musco; M. Persu (2014). *Dimensionality reduction for k-means clustering and low rank approximation (Appendix B)*. arXiv:1410.6801 (https://arxiv.org/abs/1410.6801). Bibcode:2014arXiv1410.6801C (https://ui.adsabs.harvard.edu/abs/2014arXiv1410.6801C).

54. Hui Zou; Trevor Hastie; Robert Tibshirani (2006). "Sparse principal component analysis" (http://www-stat.stanford.edu/~hastie/Papers/spc_jcgs.pdf) (PDF). *Journal of Computational and Graphical Statistics*. **15** (2): 262–286. CiteSeerX 10.1.1.62.580 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.62.580). doi:10.1198/106186006x113430 (https://doi.org/10.1198%2F106186006x113430).

55. Alexandre d'Aspremont; Laurent El Ghaoui; Michael I. Jordan; Gert R. G. Lanckriet (2007). "A Direct Formulation for Sparse PCA Using Semidefinite Programming" (http://www.cmap.polytechnique.fr/~aspremon/PDF/sparsesvd.pdf) (PDF). *SIAM Review*. **49** (3): 434–448. arXiv:cs/0406021 (https://arxiv.org/abs/cs/0406021). doi:10.1137/050645506 (https://doi.org/10.1137%2F050645506).

56. Michel Journee; Yurii Nesterov; Peter Richtarik; Rodolphe Sepulchre (2010). "Generalized Power Method for Sparse Principal Component Analysis" (http://jmlr.csail.mit.edu/papers/volume11/journee10a/journee10a.pdf) (PDF). *Journal of Machine Learning Research*. **11**: 517–553. arXiv:0811.4724 (https://arxiv.org/abs/0811.4724). Bibcode:2008arXiv0811.4724J (https://ui.adsabs.harvard.edu/abs/2008arXiv0811.4724J). CORE Discussion Paper 2008/70.

57. Peter Richtarik; Martin Takac; S. Damla Ahipasaoglu (2012). "Alternating Maximization: Unifying Framework for 8 Sparse PCA Formulations and Efficient Parallel Codes". arXiv:1212.4137 (https://arxiv.org/abs/1212.4137) [stat.ML (https://arxiv.org/archive/stat.ML)].

58. Baback Moghaddam; Yair Weiss; Shai Avidan (2005). "Spectral Bounds for Sparse PCA: Exact and Greedy Algorithms" (http://books.nips.cc/papers/files/nips18/NIPS2005_0643.pdf) (PDF). *Advances in Neural Information Processing Systems*. **18**. MIT Press.

59. Yue Guan; Jennifer Dy (2009). "Sparse Probabilistic Principal Component Analysis" (http://jmlr.csail.mit.edu/proceedings/papers/v5/guan09a/guan09a.pdf) (PDF). *Journal of Machine Learning Research Workshop and Conference Proceedings*. **5**: 185.

60. Hui Zou; Lingzhou Xue (2018). "A Selective Overview of Sparse Principal Component Analysis". *Proceedings of the IEEE*. **106** (8): 1311–1320. doi:10.1109/JPROC.2018.2846588 (https://doi.org/10.1109%2FJPROC.2018.2846588).

61. A. N. Gorban, A. Y. Zinovyev, Principal Graphs and Manifolds (https://arxiv.org/abs/0809.0490), In: Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods and Techniques, Olivas E.S. et al Eds. Information Science Reference, IGI Global: Hershey, PA, USA, 2009. 28–59.

62. Wang, Y.; Klijn, J. G.; Zhang, Y.; Sieuwerts, A. M.; Look, M. P.; Yang, F.; Talantov, D.; Timmermans, M.; Meijer-van Gelder, M. E.; Yu, J.; et al. (2005). "Gene expression profiles to predict distant metastasis of lymph-node-negative primary breast cancer". *The Lancet*. **365** (9460): 671–679. doi:10.1016/S0140-6736(05)17947-1 (https://doi.org/10.1016%2FS0140-6736%2805%2917947-1). PMID 15721472 (https://pubmed.ncbi.nlm.nih.gov/15721472). Data online (http://www.ihes.fr/~zinovyev/princmanif2006/)

63. Zinovyev, A. "ViDaExpert – Multidimensional Data Visualization Tool" (http://bioinfo-out.curie.fr/projects/vidaexpert/). *Institut Curie*. Paris. (free for non-commercial use)

64. A.N. Gorban, B. Kegl, D.C. Wunsch, A. Zinovyev (Eds.), Principal Manifolds for Data Visualisation and Dimension Reduction (https://www.researchgate.net/publication/271642170_Principal_Manifolds_for_Data_Visualisation_and_Dimension_Reduction_LNCSE_58), LNCSE 58, Springer, Berlin – Heidelberg – New York, 2007. ISBN 978-3-540-73749-0

65. Lu, Haiping; Plataniotis, K.N.; Venetsanopoulos, A.N. (2011). "A Survey of Multilinear Subspace Learning for Tensor Data" (http://www.dsp.utoronto.ca/~haiping/Publication/SurveyMSL_PR2011.pdf) (PDF). *Pattern Recognition*. **44** (7): 1540–1551. doi:10.1016/j.patcog.2011.01.004 (https://doi.org/10.1016%2Fj.patcog.2011.01.004).

66. Kriegel, H. P.; Kröger, P.; Schubert, E.; Zimek, A. (2008). *A General Framework for Increasing the Robustness of PCA-Based Correlation Clustering Algorithms*. *Scientific and Statistical Database Management*. Lecture Notes in Computer Science. **5069**. pp. 418–435. CiteSeerX 10.1.1.144.4864 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.144.4864). doi:10.1007/978-3-540-69497-7_27 (https://doi.org/10.1007%2F978-3-540-69497-7_27). ISBN 978-3-540-69476-2.

67. Emmanuel J. Candes; Xiaodong Li; Yi Ma; John Wright (2011). "Robust Principal Component Analysis?". *Journal of the ACM*. **58** (3): 11. arXiv:0912.3599 (https://arxiv.org/abs/0912.3599). doi:10.1145/1970392.1970395 (https://doi.org/10.1145%2F1970392.1970395).

68. T. Bouwmans; E. Zahzah (2014). "Robust PCA via Principal Component Pursuit: A Review for a Comparative Evaluation in Video Surveillance". *Computer Vision and Image Understanding*. **122**: 22–34. doi:10.1016/j.cviu.2013.11.009 (https://doi.org/10.1016%2Fj.cviu.2013.11.009).

69. T. Bouwmans; A. Sobral; S. Javed; S. Jung; E. Zahzah (2015). "Decomposition into Low-rank plus Additive Matrices for Background/Foreground Separation: A Review for a Comparative Evaluation with a Large-Scale Dataset". *Computer Science Review*. **23**: 1–71. arXiv:1511.01245 (https://arxiv.org/abs/1511.01245). Bibcode:2015arXiv151101245B (https://ui.adsabs.harvard.edu/abs/2015arXiv151101245B). doi:10.1016/j.cosrev.2016.11.001 (https://doi.org/10.1016%2Fj.cosrev.2016.11.001).

70. Liao, J. C.; Boscolo, R.; Yang, Y.-L.; Tran, L. M.; Sabatti, C.; Roychowdhury, V. P. (2003). "Network component analysis: Reconstruction of regulatory signals in biological systems" (http://www.pnas.org/content/100/26/15522.full.pdf) (PDF). *Proceedings of the National Academy of Sciences*. **100** (26): 15522–15527. Bibcode:2003PNAS..10015522L (https://ui.adsabs.harvard.edu/abs/2003PNAS..10015522L). doi:10.1073/pnas.2136632100 (https://doi.org/10.1073%2Fpnas.2136632100). PMC 307600 (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC307600). PMID 14673099 (https://pubmed.ncbi.nlm.nih.gov/14673099). Retrieved February 10, 2015.
71. "Principal Components Analysis" (https://stats.idre.ucla.edu/sas/output/principal-components-analysis/). *Institute for Digital Research and Education*. UCLA. Retrieved 29 May 2018.

Further reading

- Jackson, J.E. (1991). *A User's Guide to Principal Components* (Wiley).
- Jolliffe, I. T. (1986). *Principal Component Analysis* (https://archive.org/details/principalcompone00joll_0/page/487). Springer Series in Statistics. Springer-Verlag. pp. 487 (https://archive.org/details/principalcompone00joll_0/page/487). CiteSeerX 10.1.1.149.8828 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.149.8828). doi:10.1007/b98835 (https://doi.org/10.1007%2Fb98835). ISBN 978-0-387-95442-4.
- Jolliffe, I.T. (2002). *Principal Component Analysis,* second edition (Springer).
- Husson François, Lê Sébastien & Pagès Jérôme (2009). *Exploratory Multivariate Analysis by Example Using R*. Chapman & Hall/CRC The R Series, London. 224p. ISBN 978-2-7535-0938-2
- Pagès Jérôme (2014). *Multiple Factor Analysis by Example Using R (https://books.google.com/books?id=_RIeBQAAQBAJ&printsec=frontcover#v=snippet&q=%22principal%20component%20analysis%22&f=false)*. Chapman & Hall/CRC The R Series London 272 p

External links

- University of Copenhagen video by Rasmus Bro (https://www.youtube.com/watch?v=UUxIXU_Ob6E) on YouTube
- Stanford University video by Andrew Ng (https://www.youtube.com/watch?v=ey2PE5xi9-A#t=2385) on YouTube
- A Tutorial on Principal Component Analysis (https://arxiv.org/abs/1404.1100)
- A layman's introduction to principal component analysis (https://www.youtube.com/watch?v=BfTMmoDFXyE) on YouTube (a video of less than 100 seconds.)
- StatQuest: Principal Component Analysis (PCA) clearly explained (https://www.youtube.com/watch?v=_UVHneBUBW0) on YouTube
- See also the list of Software implementations