A survey of dimension reduction techniques : what comes after PCA ?

Mélanie Blazère [1]
IMT-EDF R&D

ETICS Summer School, Barcelonnette, June 6, 2016

1. melanie.blazere@math.univ-toulouse.fr

Introduction and outline

Introduction

- Advances in data collection and storage capabilities have led **to an information overload.**
  - ➠ Explosion of the information available from experiments.
- **Problem : curse of dimensionality.**
  - ➠ Traditional statistical methods break down.
- **Hope :**
  - Data highly redundant.
  - Most of the information in the initial variables can be summarized by **a small number of variables**.

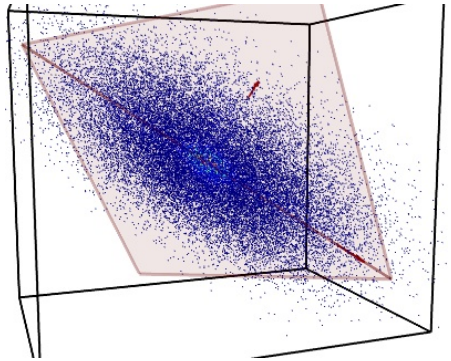


Figure : Concentration of 3D data (blue) around a plane (red)

How to reduce the number of variables ?

➤**Two different ways to reduce the number of variables :**

1. By keeping the most relevant variables from the original dataset.
   ➠ **Feature selection methods** : Backward-forward, Lasso, Elastic net...
2. By finding a smaller set of new variables, combination of the input variables, containing basically the same information as the input variables.
   ➠ **Dimension reduction methods** : ACP, ICA, PLS...

Notations

➤ **Here are some useful notations.**

- X denotes a scalar or a vector.
- X_i denotes the i^{th} coordinate of X.
- $\mathbf{X}$ denotes a matrix.
- X_{ij} denotes the value in the i^{th} row and j^{th} column of $\mathbf{X}$.
- $X^{\cdot i}$ denotes the vector corresponding to the i^{th} column of $\mathbf{X}$.
- $X^{i\cdot}$ denotes the vector corresponding to the i^{th} row of $\mathbf{X}$.

Dimension reduction methods

➤ **Inputs :** A multivariate random variable $X = (X_1, ..., X_p) \in \mathbb{R}^p$.
➤ **Goal :** To find a function

$$f : \begin{cases} \mathbb{R}^p & \rightarrow & \mathbb{R}^k \\ X & \mapsto & C := f(X) \end{cases}$$

with $k \ll p$, such that $C := (C_1, ..., C_k) \in \mathbb{R}^k$ best preserves :

- **The structure** of the original data.
- **The relevant information** in the data.

➤ The variables $C_1, ..., C_k$ are called **latent variables**.

✦ Can we represent each data point with fewer features, without losing much ?
➠ Yes, if there is **redundancy** in the data !

Linear case

- If the operator is linear, i.e. of the form

$$X \mapsto C := \mathbf{F}^T X,$$

where $\mathbf{F} \in \mathbb{R}^{p,k}$, the method is said to be **linear**.

- Latent variables are in this case **linear combinations of the input variables**, since, for all $j = 1, ..., k$, we have

$$C_j = \mathbf{F}^{\cdot j\,T} X = \sum_{i=1}^{p} F_{ij} X_i.$$

- **Most commonly used** method
- **Simpler, easier** to implement.
- Even if f is not invertible, one can find an operator g such that

$$X \approx g(C).$$

Different motivations for dimension reduction

- Data compression.
- Visualization.
- Latent variables construction to reduce the number of input variables in a statistical model (for regression, classification, predictive models...).
- Density estimation, simulation of physical processes.
- ...

Presentation outline

1. Dimension reduction methods through matrix factorization
2. Independent component analysis
3. Non linear methods and manifolds
4. Multilinear PCA
5. Conclusion

Dimension reduction methods through matrix factorization

Matrix factorization

- Equivalent to search an approximation of $X \in \mathbb{R}^p$ in the form of

$$X \approx \mathbf{F}C$$

where

- $X := (X_1, ..., X_p)^T$ are the p **input variables**.
- $C := (C_1, ..., C_k)^T$ contains the k latent variables, with $k \leq p$. The random variables $C_1, ..., C_k$ are also called **components**.
- $\mathbf{F} \in \mathbb{R}^{p,k}$ is the factor matrix. Its columns are called the **factors**.

- We have

$$X \approx \sum_{j=1}^{k} C_j F^{\cdot j}$$

and

$$X^{i\cdot} \approx \sum_{j=1}^{k} F_{ij} C_j, \ \ i = 1, ..., p.$$

Matrix factorization

- In other words, we attempt to decompose $X \in \mathbb{R}^p$ as

$$X = X_k + E,$$

where $X_k := \mathbf{F}C$, $\mathbf{F} \in \mathbb{R}^{p,k}$ and $C \in \mathbb{R}^k$, such that $L(X, X_k)$ is minimal.

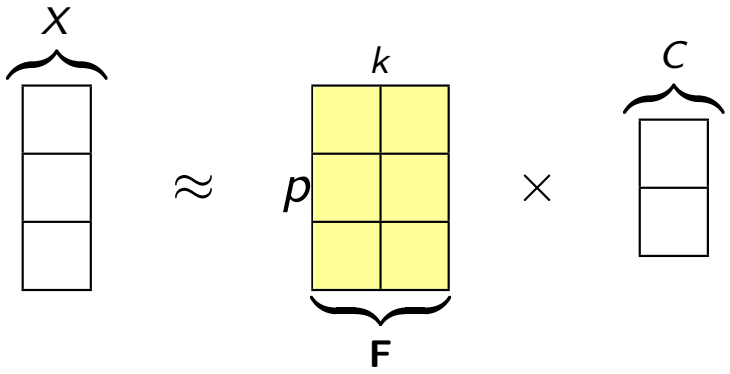
- L quantifies the quality of the approximation and may be
  - **a least square criterion**, i.e.

$$L(A, B) = \| A - B \|_F = \sum_{i,j} (a_{ij} - b_{ij})^2 ,$$

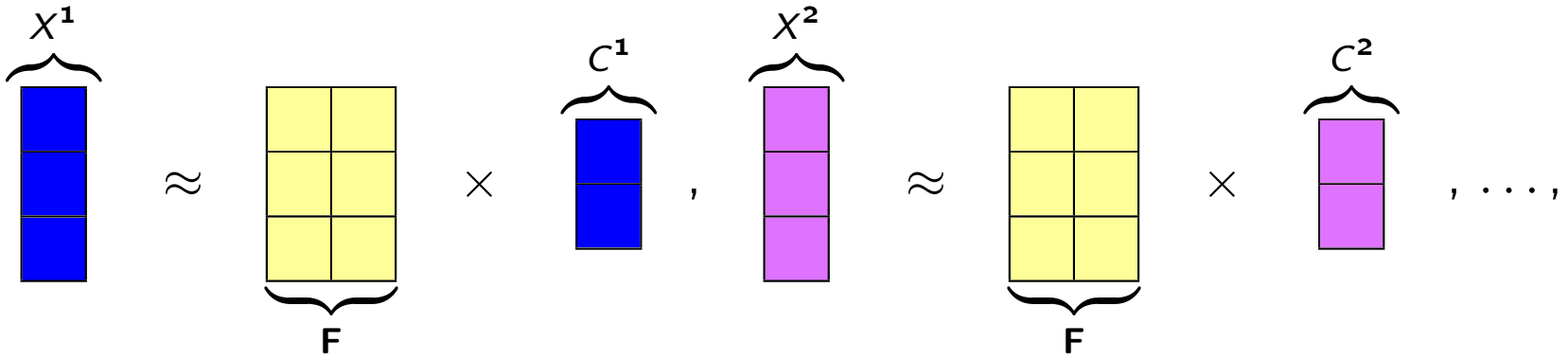
  - or the **Kullback-Leibler distance**, i.e.

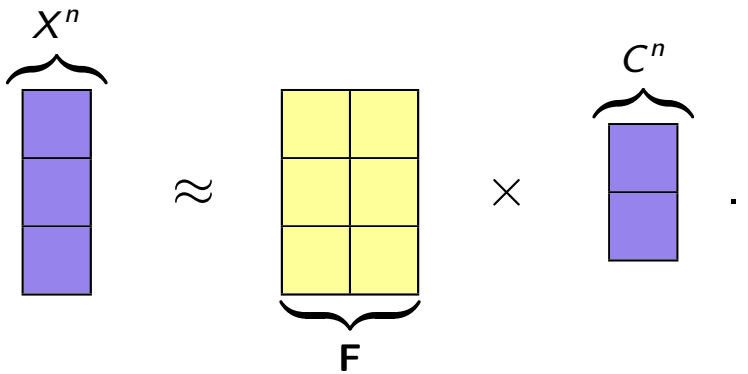
$$L(A, B) = KL(A \parallel B) = \sum_{i,j} a_{ij} \log \frac{a_{ij}}{b_{ij}} - a_{ij} + b_{ij}.$$

A visual point of view

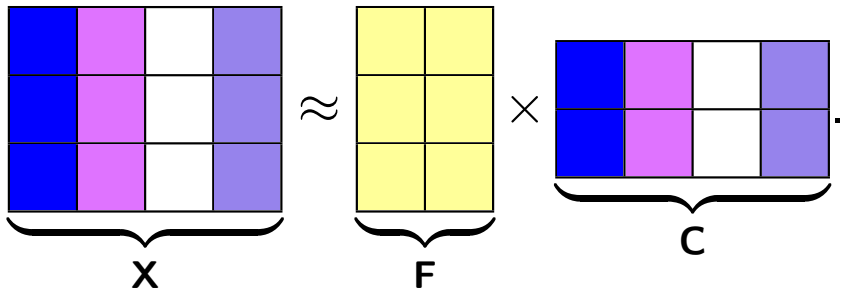


When we have a sample, this is equivalent to

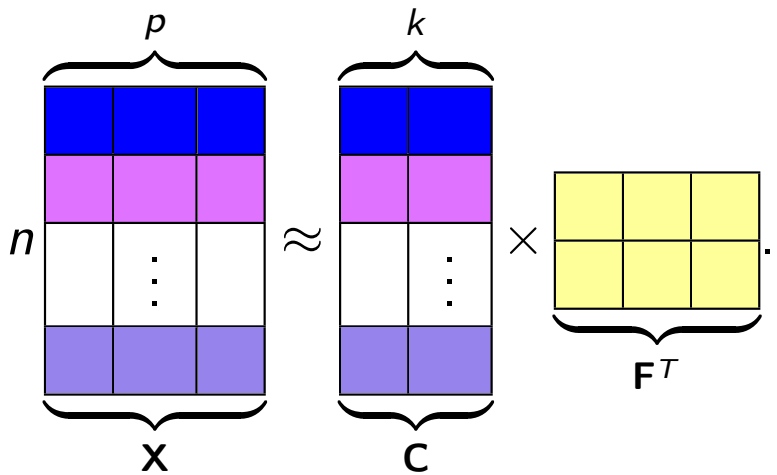




By concatening,



Finally, after transposing, we get



Empirical point of view

➤ Let $\mathbf{X} \in \mathbb{R}^{n,p}$ be the **design matrix**, where each row represents an observation of the multivariate random variable $X = (X_1, ..., X_p)$.

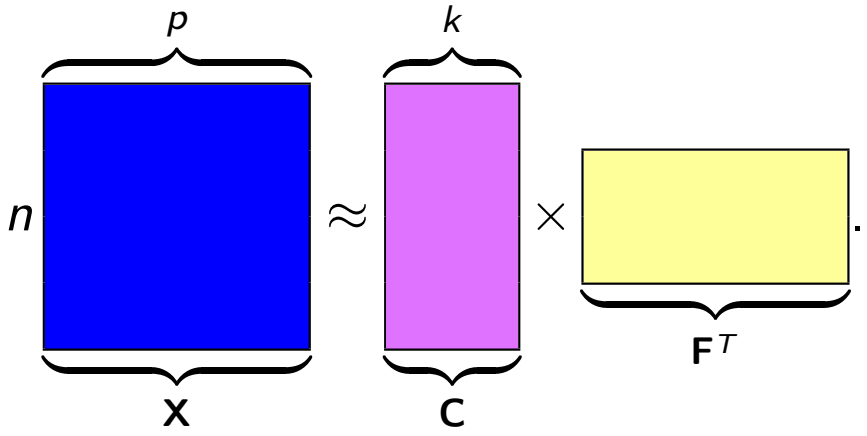
➤ In practice, we want to find matrices $\mathbf{C}$ and $\mathbf{F}$ such that

$$\mathbf{X} \approx \mathbf{C}\mathbf{F}^T := \widehat{\mathbf{X}}_k.$$

where

- $\mathbf{C} \in \mathbb{R}^{n,k}$ is the matrix of the **components**. Each row represents a realization of the k random latent variables $C_1, ..., C_k$,
- $\mathbf{F} \in \mathbb{R}^{p,k}$ is the **factor matrix**.

Low rank approximation ($k \ll p$)



- Why are low-rank approximations important ?
  - Intuitively, if the matrix **X** is low rank, then **the observations can be explained by linear combinations of few underlying variables**.
  - Allows to know which variables control the observations, contribute to the largest variations in the data.
- The different dimension reduction methods depend on the **properties fulfilled** by the components and by the factors.

Example 1 : PCA

- **Goal** : to build k **decorrelated latent variables** $C_1, ..., C_k$ where $C_i = X^T f_i$ (with $f_i \in \mathbb{R}^p$ and $\| f_i \| = 1$) satisfying

$$\mathrm{Var}(C_1) \geq \mathrm{Var}(C_2) \geq ... \geq \mathrm{Var}(C_k) > 0.$$

- **In practice**, this is equivalent to solve

$$\underset{\mathbf{F} \in \mathbb{R}^{p,k}}{\operatorname{argmin}} \ \| \mathbf{X} - \mathbf{X}\mathbf{F}\mathbf{F}^T \|^2 \quad s.t \quad \mathbf{F}^T\mathbf{F} = \mathbf{I},$$

  where $\mathbf{X}$ is centered, or equivalently

$$\underset{\mathbf{F},\ \mathbf{C}}{\operatorname{argmin}} \ \| \mathbf{X} - \mathbf{C}\mathbf{F}^T \|^2 \quad s.t \quad \mathbf{F}^T\mathbf{F} = \mathbf{I},\ \mathbf{C}^T\mathbf{1} = 0.$$

- **The solution** is given by
  - $\mathbf{C} := \mathbf{X}\mathbf{P}_k$,
  - $\mathbf{F} := \mathbf{P}_k$,

  where the columns of $\mathbf{P}_k$ form a basis of the space spanned by the eigenvectors of $\frac{1}{n}\mathbf{X}^T\mathbf{X}$ associated to the k highest eigenvalues.
- The **latent variables are decorrelated and the column factors $F^{\cdot j}$ of $\mathbf{F}$ are orthogonal** $\rightarrow$ **Principal component analysis** (PCA).

Example 2 : NMF [2]

- We often deal with "**non-negative data**" (pixels, concentration, counts...)
  ➠ **X** contains only non-negative elements.
- Non-negative data need special treatment because **negative valued factors can contradict reality**.
- PCA involves adding up some factors then substracting others.
  Substracting does not make sense in context of some applications.
- **What does NMF do ?**
  ✦ Like PCA, except that the coefficients in the linear combination and the factors cannot be negative.

$$\min_{\mathbf{F},C} \mathbb{E}\left(\| X - \mathbf{F}C \|_2^2\right) \quad s.t. \quad \mathbf{F} \geq 0,\ C \geq 0.$$

  Or, from an empirical point of view,

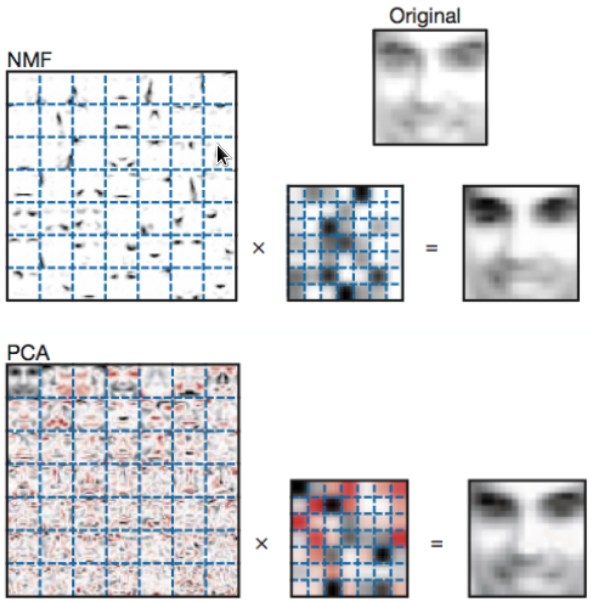
$$\min_{\mathbf{F},\mathbf{C}} \| \mathbf{X} - \mathbf{C}\mathbf{F}^T \|_F^2 \quad s.t. \quad \mathbf{F} \geq 0,\ \mathbf{C} \geq 0.$$

- The **latent variables and the factors contain only non-negative elements** → **Non-negative matrix factorization** (NMF).

2. D.D.Lee and H.S.Seung (1999). Learning the parts of objects by non-negative matrix factorization. Nature, 401(6755) : 788-791.

Example of NMF on faces [3]

➤ **Training set** : 2429 examples of centered face images of size 19×19.



- PCA factors are **not interpretable.**
- NMF factors **find parts** that are additive (noses, mouths, eyes...).

Figure : PCA and NMF components

3. Figure taken from : de D.D.Lee and H.S.Seung (1999). Learning the parts of objects by non-negative matrix factorization. Nature, 401(6755) : 788-791.

NMF summary

- NMF does **additive decomposition**.
- NMF is a better way to explain **structured data**, is more a part based representation
- The NMF algorithm is based on gradient descent. The decomposition is **not unique**.

Example 3 : ICA

➤ The **latent variables are independent** ➠ **Independent component analysis** (ICA)

$$\min_{\mathbf{F},C} KL(X \parallel \mathbf{F}C)$$

under the constraint that $(C_i)_{1\leq i\leq k}$ are as independent as possible.

➤ More details in the next few slides.

Summary

	F	**C**	L
PCA	Columns of **F** orthonormal. $X^i \approx$ linear combination of the factors.	Columns of **C** orthogonal. i.e. Latent variables C decorrelated.	$\| . \|_F$
NMF	Coefficients of $\mathbf{F} \geq 0$. $X^i \approx$ **positive** linear combination of the factors.	Coefficients of $\mathbf{C} \geq 0$ i.e. Latent variables C with positive coefficients.	$\| . \|_F$
ICA	Columns of **F** linearly independent but not necessarily orthogonal.	Rows of **C** are statistically independent as much as possible. i.e. Latent variables C are independent .	$KL(. \| .)$

Independent component analysis

Objective of the ICA method [4]

- **Goal :** to write $X := (X_1, ..., X_p)^T$ as a **mixture of independent variables** $C := (C_1, ..., C_k)^T$,

i.e.

$$X = \mathbf{F}C,$$

where

- $X := (X_1, ..., X_p)^T$ are the p **input variables**.
- $C := (C_1, ..., C_k)^T$ are k **independent** variables, called **components** or **sources**.
- $\mathbf{F} \in \mathbb{R}^{p,k}$ is the factor matrix, with the **factors** in column.
- $k \leq p$.

4. Hyvärinen, Aapo and Karhunen, Juha and Oja, Erkki (2004). Independent component analysis, 46, John Wiley & Sons.
http ://www. cis. hut. fi/projects/ica/

Pre-processing

- **Centering** of the variables
- **Whitening** (decorrelation and normalization), so that

$$\mathbb{E}(XX^T) = \mathbf{I}_p.$$

  - The data are whitened using PCA.
  - **Advantage** : restores the initial "shape" of the data, then ICA must only rotate the resulting matrix

$$\mathbf{I} = \mathbb{E}(XX^T) = \mathbf{F}\mathbb{E}(CC^T)\mathbf{F} = \mathbf{F}\mathbf{I}\mathbf{F}^T = \mathbf{F}\mathbf{F}^T.$$

Link ICA-PCA

✦ PCA build non correlated variables, but **decorrelation does not imply independence**[5].

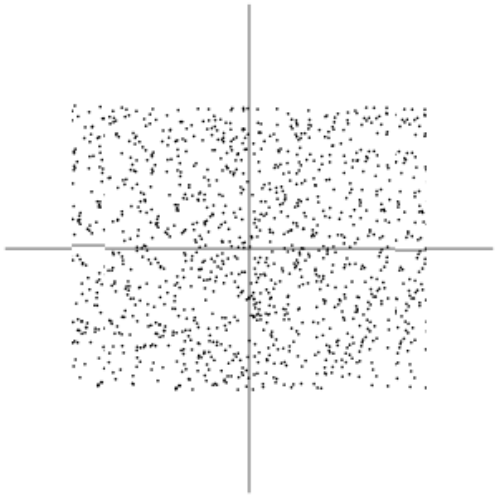


Figure : Two independent uniform r.v.

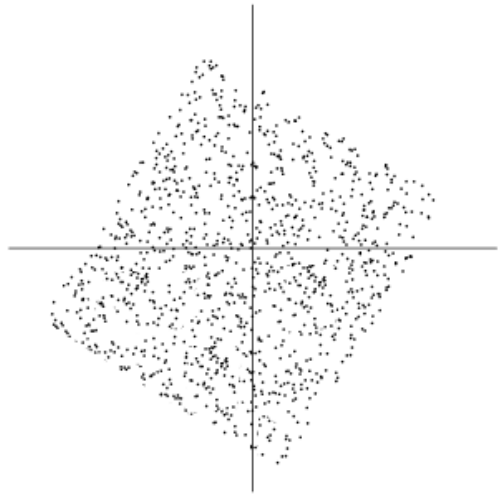


Figure : Mixture of two independent r.v.

✦ In the two cases, the r.v. are decorrelated but they are independent only in the left case.

5. **Decorrelation :** $\mathbb{E}(XY) = \mathbb{E}(X)\mathbb{E}(Y)$.
Independence : $\mathbb{E}(f(X)g(Y)) = \mathbb{E}(f[X])\mathbb{E}(g[Y])$, for all measurable functions f and g.

Link ICA-PCA

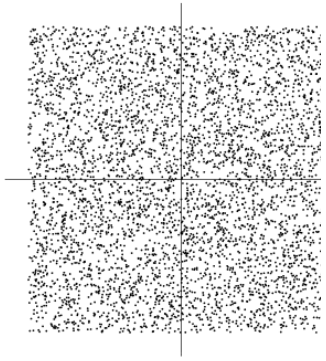


Figure : Inputs

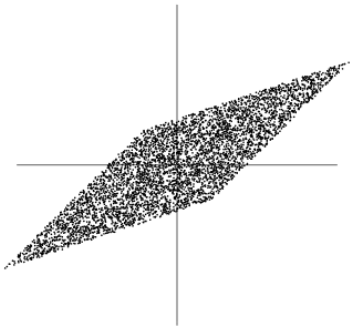


Figure : Mixed

Figure : Whitened (PCA)

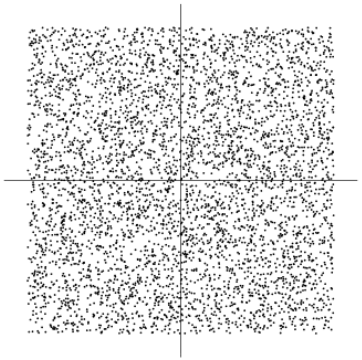


Figure : Rotated (ICA)

✦ PCA decorrelate but the resulting projection has **not produced independence**.

✦ ICA outputs are **independent**.

✦ Applying ICA means to **rotate** the whitened representation back to the original independent variables.

Conditions and constraints

- To ensure **identifiability**, the following assumptions are made.
  1. At most one source has a normal distribution, because the distribution of Gaussian whitened variables is spherically symmetric.

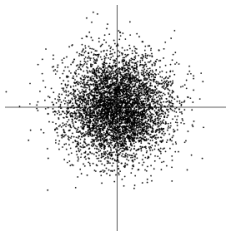


Figure : Gaussian distribution

  2. $\text{rang}(\mathbf{F}) = k = \min(k, p)$ (in other words $k \leq p$ and the matrix $\mathbf{F}$ is full rank).

- **Indetermination.**
  - The sign.
  - The order (invariance to output permutations, no sense of ordering of components).
  - The variance of the components.

Objective function

- In practice, ICA search for
  - k components $C_1, ..., C_k$ as independent as possible that are **linear combination** of the input variables $X_1, ..., X_p$, i.e

$$C = \mathbf{W}X,$$

    where $\mathbf{W} \in \mathbb{R}^{k,p}$
  - and a matrix $\mathbf{F} \in \mathbf{R}^{p,k}$

such that

$$X \approx \mathbf{F}C.$$

-

ICA = Linear dimension reduction method.

How to characterize independence ?

✦ The independence between random variables $C_1, ..., C_k$ (with a density) is characterized by

$$p_C = \prod_{j=1}^{k} p_{C_j}.$$

✦ A natural way of verifying if the components of X are independent is to measure

$$\delta(p_C, \prod_{j=1}^{k} p_{C_j}),$$

where δ is a divergence measure between densities.

✦ If δ= Kullback-Leibler divergence D

➠ **Mutual information I_C**

$$I_C = D\left(p_C \parallel \prod_{i=1}^{k} p_{C_i}\right).$$

Notion of independence

- **Mutual information**

$$I_C = D(p_C \parallel \prod_{i=1}^{k} p_{C_i}).$$

  - $I_C \geq 0$.
  - $I_C = 0$ iff $C_1, ..., C_k$ independent.

  ➠ **Problem :** p_C is not known.

- **Key= non-gaussianity**

Non-gaussian is independent

- We search an approximation Z_i of one of the $\{C_j\}$ as a **linear combination** of X

$$\Rightarrow Z_i = w_i^T X.$$

- If $X = \mathbf{F}C$ then $Z_i = w_i^T X = w_i^T \mathbf{F} C = y_i^T C = \sum_{j=1}^{k} y_{ij} C_j$, with $y_i = \mathbf{F}^T w$.

$$\Rightarrow Z_i \text{ is a } \textbf{linear combination} \text{ of } \{C_j\}.$$

- **CLT** : distribution of a sum of independent random variables tends toward a Gaussian distribution, under certain conditions.

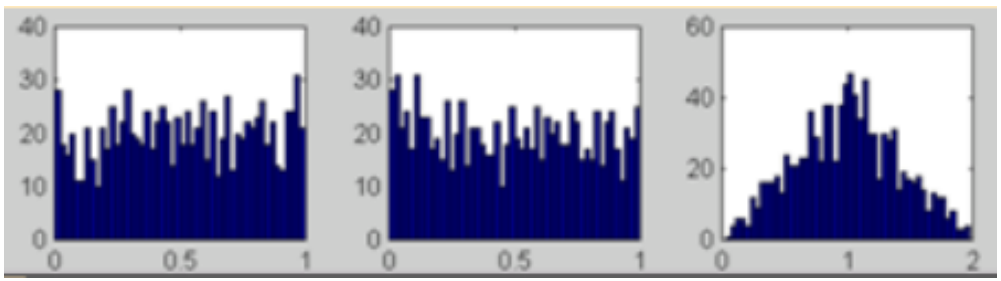


Figure : Sum of two independent r.v usually has a distribution that is closer to a Gaussian

➤ Maximizing the non-gaussianity of $w_i^T X$ to get independent components.

Non-gaussianity and independence

- Let us go back to

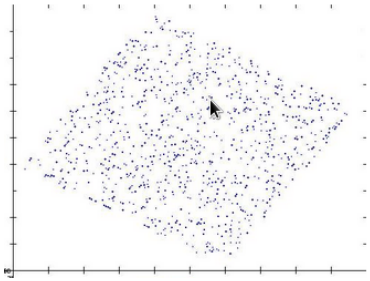


Figure : Mixture of two independent r.v.

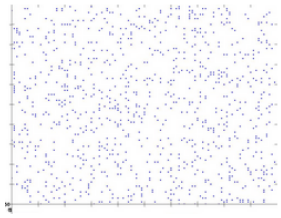


Figure : ICA components

- And look at the marginal distribution

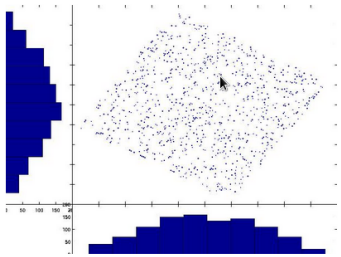


Figure : Almost gaussian

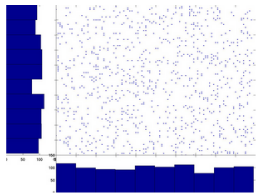


Figure : Far from gaussian

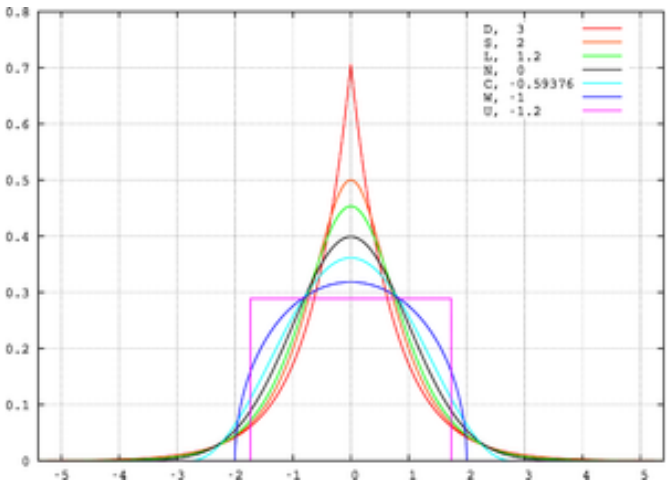
Non-gaussianity measures

✦ Some examples of measures.

- **Kurtosis.**

$$K_C := \mathbb{E}\left\{C^4\right\} - 3\left(\mathbb{E}(C^2)\right)^2.$$

  - $K_X = 0$ for standardized Gaussian r.v. X.
  - Very sensitive to outliers.



- **Entropy H_C.**
  A **Gaussian r.v. has maximum entropy** among all real-valued r.v with same mean and variance.

Non-gaussianity measures

✦ Some examples of measures.

- **Negentropy.**

$$J_C = H_G - H_C,$$

where $G \sim \mathcal{N}(\mathbb{E}(C), \mathbb{V}(C))$.

  - $J_C = 0$ iff C is Gaussian.
  - For the ICA model, we have

$$I_C = -\sum_{i=1}^{p} J_{C_i} + c,$$

where c is a constant.

Minimizing mutual information = Maximizing negentropy.

Approximation of the negentropy of a r.v. Y

- **Simple approximation.**

$$J_Y \approx \frac{1}{12}\mathbb{E}\left(Y^3\right)^2 + \frac{1}{48}\mathrm{Kurt}(\mathrm{Y})^2$$

where Y is centered and reduced.

- **Advanced approximations**.

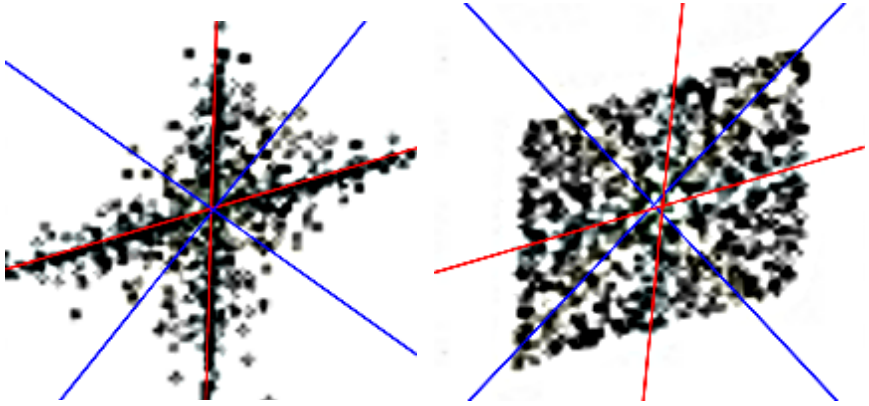
$$J_Y \approx \sum_{i=1}^{r} k_i \left[\mathbb{E}\left\{F_i(Y)\right\} - \mathbb{E}\left\{F_i(G)\right\}\right]^2,$$

where

  - r is the number of functions used in the approximation.
  - $\{F_i\}$ can be
    - $F(y) = y^4$.
    - $F(y) = -\exp(-y^2/2)$.
    - $F(y) = \frac{1}{a}\log(\cosh(ay))$ with $1 < a < 2$.
    - ...
  - $\{k_i\}$ are some positive weights.
  - G is a Gaussian variable of zero mean and unit variance.

Other difference between PCA and ICA

- ICA is a **generalization** of PCA.
- ICA components provide a **more compact representation** because the ones of PCA can still have statistical dependencies.
- PCA components are decorrelated and its factors are orthogonals.
- Only the ICA components are required to be independent, **no orthogonality condition on the ICA factors**.



(a) Exemple 1 (b) Exemple 2

Figure : ACP (bleu) and ICA (rouge) : the factors are not the same, ICA factors **not orthogonal**.

ICA and dimension reduction

➤ **If we need to perform dimensionality reduction**, we usually precede ICA with PCA :

1. Do PCA to reduce the dimension

$$\mathbf{C}^P = \mathbf{X}\mathbf{F}_k^{P\,T}.$$

2. Do ICA on the PCA components to produce independence

$$\mathbf{C}^I = \mathbf{C}^P\mathbf{F}_k^{I\,T}.$$

3. Finally,

$$\mathbf{C}^I = \mathbf{X}\left(\mathbf{F}_k^I\mathbf{F}_k^P\right)^T \Rightarrow \mathbf{F}_k = \mathbf{F}_k^I\mathbf{F}_k^P$$

and

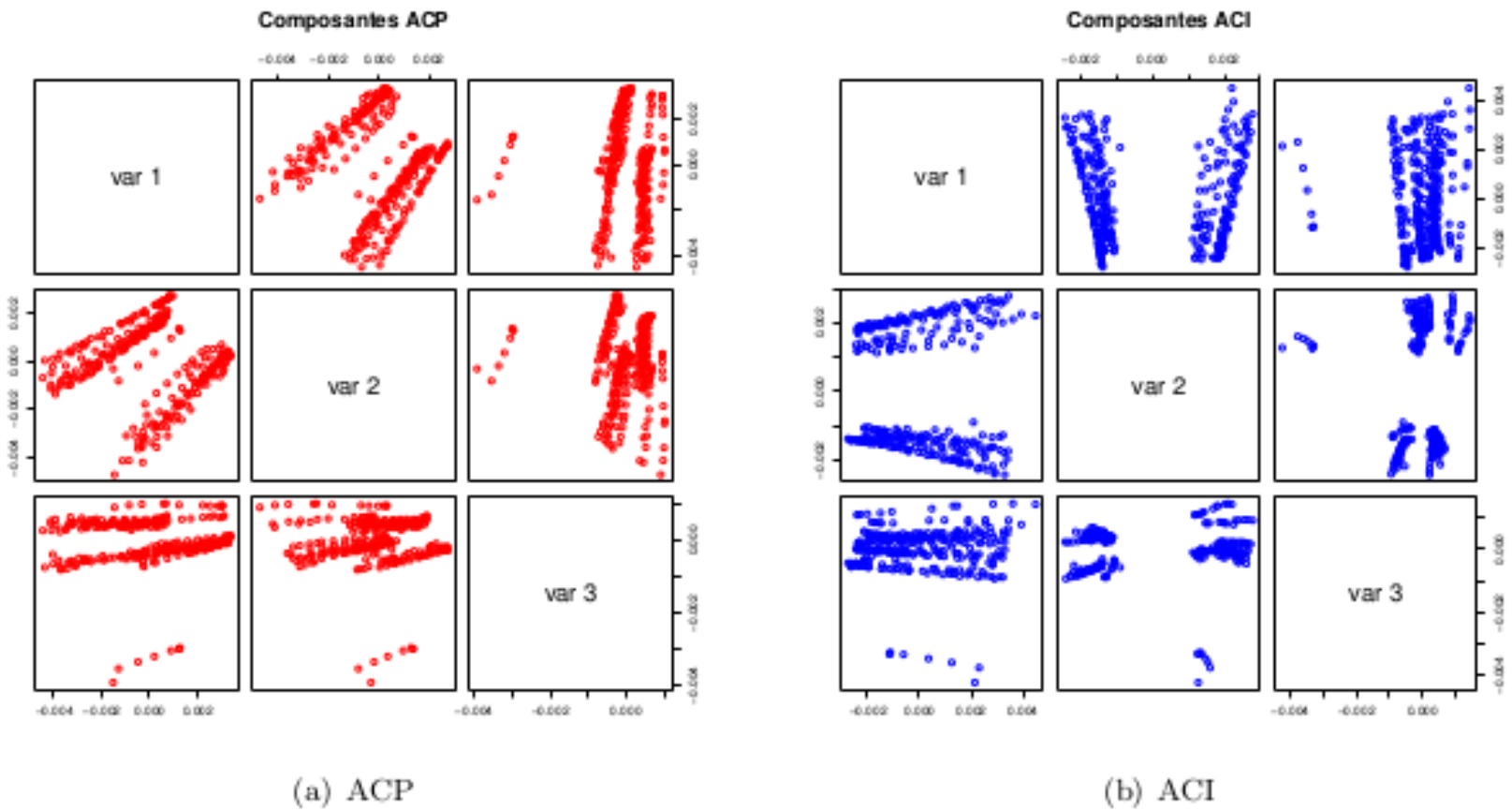
$$\widehat{\mathbf{X}}_k = \mathbf{C}^I\left(\mathbf{F}_k\right)^\dagger$$

➤ ICA can only separate **linear mixed sources**.

➤ There might not be independence. But, even if the sources are not independent, ICA finds a space where they are **maximally independents**.

ICA

Example



(a) ACP (b) ACI

Figure : ICA versus PCA components

Non linear dimension reduction methods

Non linear manifolds

- Many data **lies near or on a non-linear manifold**.
- ACP, ICA... are linear dimension reduction methods and linear projection **may not preserve distances and topology** along a non-linear structure.

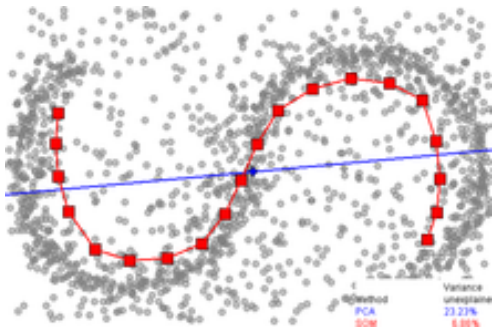


Figure : Figure taken from Wikipedia (article on self-organizing map)

- **Issue :** what can we do in this case to
  1. Produce a compact low-dimensional encoding of a given high-dimensional data set.
  2. Simplify, reduce and clean the data for visualization.
- **A possible solution :** : **Kernel PCA**.

Kernel PCA

➤ **Inputs :** $x_1, ..., x_n$ points in $\mathbb{R}^p$ (high dimension).

➤ **Objective :** a representation $y_1, ..., y_n$ of $x_1, ..., x_n$ in dimension k (low dimension) that **preserves the topology** of the observed point cloud.

➤ **Notation :** we denote by $\mathbf{Y}$ the matrix whose rows are the $\{y_i\}_{1 \leq i \leq n}$.

Its columns can be interpreted as realizations of k features ($\sim$ components) of the initial data.

➤ **Idea :** find **a non-linear function** ϕ that maps the data in $\mathbb{R}^p$ into a space $\mathcal{E}$ of higher dimension $m \geq p$, to introduce linearity

$$\phi : \begin{cases} \mathbb{R}^p & \to & \mathcal{E} \\ x & \mapsto & \phi(x) \end{cases} .$$

➤ Then, extract principal component in that space.
➤ The result will be non-linear in the original data space

➠ Kernel PCA extracts **nonlinear features**.

An example

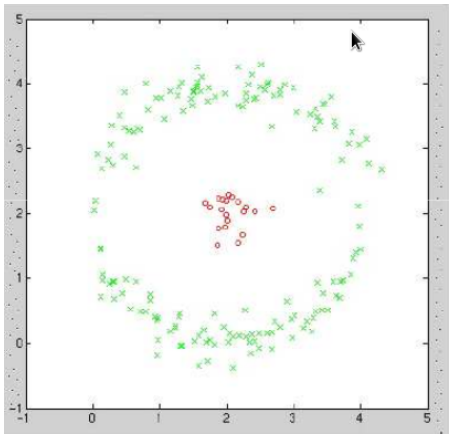


Figure : Initial observations

- This is not possible to separate the red and green points by a line.
  - These classes are **linearly inseparable** in the input space.
- But, if we consider the following simple mapping

$$\phi : \begin{cases} \mathbb{R}^2 & \mapsto & \mathbb{R}^3 \\ x = (x_1, x_2)^T & \to & x' = (x_1, x_2, x_1^2 + x_2^2) \end{cases},$$

then the points can be split up by a plane in this feature space.

An example

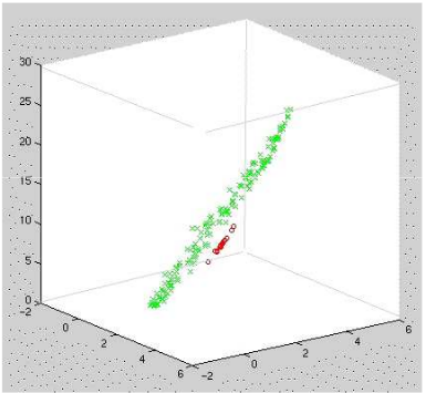


Figure : After mapping

- We can perform PCA in the space produced by the nonlinear mapping.

Kernel PCA

- We assume that

$$\frac{1}{n}\sum_{i=1}^{n}\phi(x_i) = 0.$$

- Performing PCA in $\mathcal{E}$ is equivalent to solve

$$\underset{\mathbf{W}\in\mathcal{C}}{\operatorname{argmin}}\left\{\frac{1}{n}\sum_{i=1}^{n}\parallel \phi(x_i) - \mathbf{W}\mathbf{W}^T\phi(x_i)\parallel\right\},$$

where $\mathcal{C} = \left\{\mathbf{W}\in\mathbb{R}^{m,k} : \mathbf{W}^T\mathbf{W} = \mathbf{I}_k\right\}$.

- **Solution :** the solution is given by the SVD of $\phi(\mathbf{X}) = \mathbf{U}\mathbf{D}\mathbf{V}^T$, where $\mathbf{X}$ is the design matrix whose rows are the $\{x_i\}_{1\leq i\leq n}$.

- **Problem :** $\phi(\mathbf{X})\in\mathbb{R}^{n,m}$ and if m is large, this makes computations quickly **very costly** and even impractical+ problem of the **choice of** ϕ.

Kernel trick

➤ **Solution to get around this problem.**

This is possible to **reduce the dependence on m and ϕ**, if we have a kernel K that allows to compute

$$K(x_i, x_j) = \phi(x_i)^T \phi(x_j).$$

➤ **Kernel trick :** PCA in the feature space can be **formulated entirely in terms of dot products** between data point, since it requires only the computation of the eigenvectors of the covariance matrix

$$\widehat{\mathbf{Z}} = \frac{1}{n} \sum_{i=1}^{n} \phi(x_i)\phi(x_i)^T,$$

i.e. to solve

$$\widehat{\mathbf{Z}} v = \lambda v.$$

➤ $\mathbf{K}$ does not depend on m and can be computed in a run time that depends only on n.

Kernel trick

- If $\lambda \neq 0$, we have

$$v = \frac{1}{n\lambda} \sum_{i=1}^{n} (\phi(x_i), v)\phi(x_i).$$

i.e v can be written as

$$v = \sum_{i=1}^{n} \alpha_i \phi(x_i).$$

- **Finding the eigenvectors is equivalent to finding the coefficients** $\alpha := (\alpha_1, ..., \alpha_n) \in \mathbb{R}^n$ that satisfies

$$\mathbf{K}\alpha = n\lambda\alpha,$$

where $\mathbf{K} = (K(x_i, x_j))_{ij} \in \mathbb{R}^{n,n}$ with

$$K(x_i, x_j) = \phi(x_i)^T \phi(x_j).$$

- The normalization condition $\| v \| = 1$ implies a normalization condition on α

$$\alpha^T \mathbf{K} \alpha = 1.$$

- Let $v_j = \sum_{i=1}^{n} \alpha_i^j \phi(x_i), j = 1, ..., k$ the resulting k principal directions.
- We sum up each of the x observation in the input space by k features $y = (y^1, ..., y^k)$, where

$$y^j := \phi(x)^T p_j = \sum_{i=1}^{n} \alpha_i^j \phi(x)^T \phi(x_i) = \sum_{i=1}^{n} \alpha_i^j K(x, x_i).$$

- Data can be reconstructed in feature space by

$$\widehat{\phi(x)} = \sum_{j=1}^{k} (\phi(x)^T p_j) p_j.$$

- **Drawback :**
  - Unfortunately, Kernel PCA does not inherit all the strength of PCA.
  - **Reconstruction in the initial space is not trivial**.
  - Finding $\hat{x}$ in the initial space such that $\widehat{\phi(x)} = \phi(\hat{x})$ is **difficult and often even impossible**.

Some popular and useful kernels

- Kernel PCA comes to select a Kernel **K** instead of a dot product in $\mathbb{R}^p$.
- A kernel commonly used is the **Gaussian** kernel

$$K(x_i, x_j) = \exp\left(-\gamma \parallel x_i - x_j \parallel_2^2\right)$$

where $\gamma = \frac{1}{2\sigma^2}$ is a parameter.

- Other possible kernels are the **polynomial** one

$$K(x_i, x_j) = (1 + x_i^T x_j)^d,$$

or the following kernel

$$K(x_i, x_j) = \tanh(x_i^T x_j + \delta),$$

where δ is a parameter.

Algorithm

1. Pick a Kernel $\mathbf{K}$.
2. Construct the normalized (if the data are not centered) kernel matrix of the data (dimension $n \times n$)

$$\tilde{\mathbf{K}} = \mathbf{K} - \frac{2}{n}\mathbf{1}\mathbf{K} + \frac{1}{n^2}\mathbf{1}\mathbf{K}\mathbf{1}$$

where $\mathbf{K} = (K_{ij}) \in \mathbb{R}^{n,n}$ and $K_{ij} = K(x_i, x_j)$. In other words, compute

$$\tilde{\mathbf{K}} = \mathbf{H}\mathbf{K}\mathbf{H}^T$$

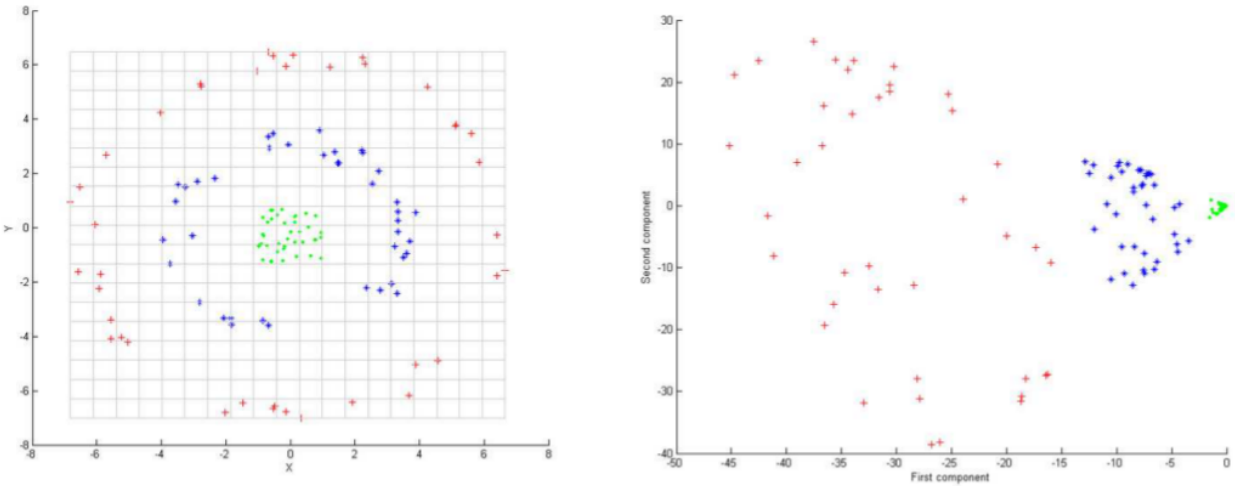
where $\mathbf{H} = \mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^T$.

3. For $i = 1, ..., k$, solve the eigenvalue problem

$$\tilde{\mathbf{K}}\alpha^i = \lambda_i \alpha^i,$$

where $\lambda_1, ..., \lambda_k$ are the k largest eigenvalues of $\tilde{\mathbf{K}}$.

Example



(a) Original space (b) Feature space

Figure : Kernel PCA

Not always suited to manifold learning

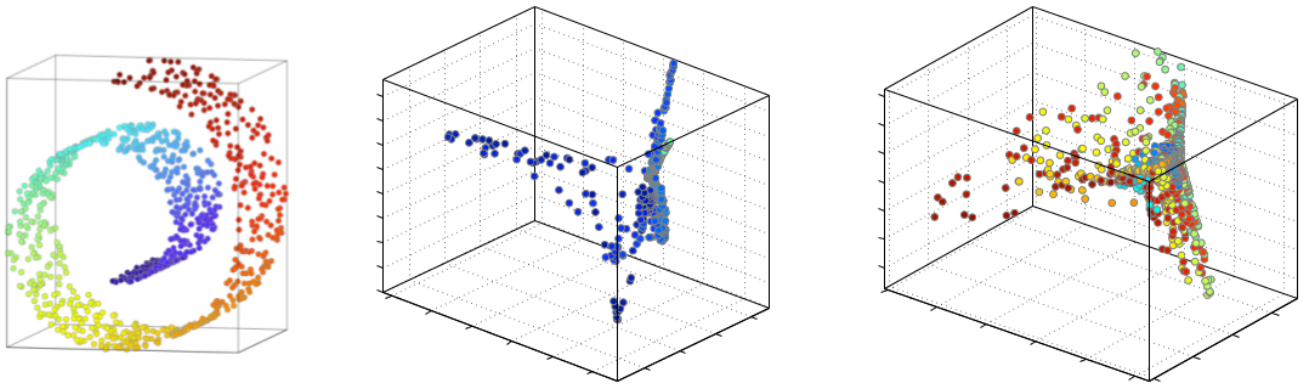


Figure : Results of kernel PCA with Gaussian and polynomial kernels. These kernels do not lead to low dimensional representations that unfold the Swiss roll.

MultiDimensional Scaling (MDS)

➤ **Goal :** To map the original high dimensional space to a lower dimensional space, in an attempt to **preserve pairwise distance**.

✦ Let $\mathbf{D}^X := (d_{ij}^X) \in \mathbb{R}^{n,n}$ be the distance matrix associated to the input space.

➤ **Objective :** Attempt to find n points $y_1, ..., y_n$ in $\mathbb{R}^k$ such that $\mathbf{D}^Y := (d_{ij}^Y = \| y_i - y_j \|^2) \in \mathbb{R}^{n,n}$ is similar to $\mathbf{D}^X$.

✦ To do so, we can consider MDS that minimizes

$$\underset{\mathbf{Y}}{\operatorname{argmin}} \sum_{i=1}^{n} \sum_{j=1}^{n} \left(d_{ij}^X - d_{ij}^Y \right)^2 .$$

MDS

✦ Let

$$\mathbf{K} = -\frac{1}{2}\mathbf{H}\mathbf{D}^X\mathbf{H},$$

where $\mathbf{H} = \mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^T$ is the centering matrix.
This is equivalent to solve

$$\underset{\mathbf{Y}}{\operatorname{argmin}} \operatorname{Tr}\left(\mathbf{K} - \mathbf{Y}^T\mathbf{Y}\right)^2. \tag{1}$$

➤ Let $\mathbf{K} = \mathbf{V}\mathbf{D}\mathbf{V}^T$ its spectral decomposition. The solution of Problem (1) is given by

$$\mathbf{Y} = \mathbf{D}_k^{1/2}\mathbf{V}_k^T,$$

where $\mathbf{D}_k$ corresponds to the top k eigenvalues of $\mathbf{K}$ and $\mathbf{V}_k$ to the associated eigenvectors.

➤ If $\mathbf{D}^X := (d_{ij}^X = \| x_i - x_j \|^2)$, then **MDS=PCA**.

Isomap [6]

➤ Isomap is a nonlinear generalization of classical MDS. The idea is to perform MDS, not in the input space $\mathbb{R}^p$, but in the geodesic space of the nonlinear data manifold to preserve **the geodesic distance**.

6. Tenenbaum, Joshua B and De Silva, Vin and Langford, John C (2000). A global geometric framework for nonlinear dimensionality reduction. Science, 290(5500) : 2319–2323. American Association for the Advancement of Science.

Isomap

1. **Graph construction :** Identify the nearest neighbours of each point. Represent the neighbourhood relations by a graph $\mathcal{G}$ with edges of weigh $\|x_i - x_j\|^2$ between neighbours and 0 otherwise.
2. **Approximation of the geodesic pairwise distance between all points** $d_{ij}^{\mathcal{G}}$ by adding the weights of the shortest path distance between x_i and x_j on $\mathcal{G}$.
3. **Embed the data via MDS** with $d_{ij}^X = d_{ij}^{\mathcal{G}}$ and $d_{ij}^Y = \|x_i - x_j\|^2$.

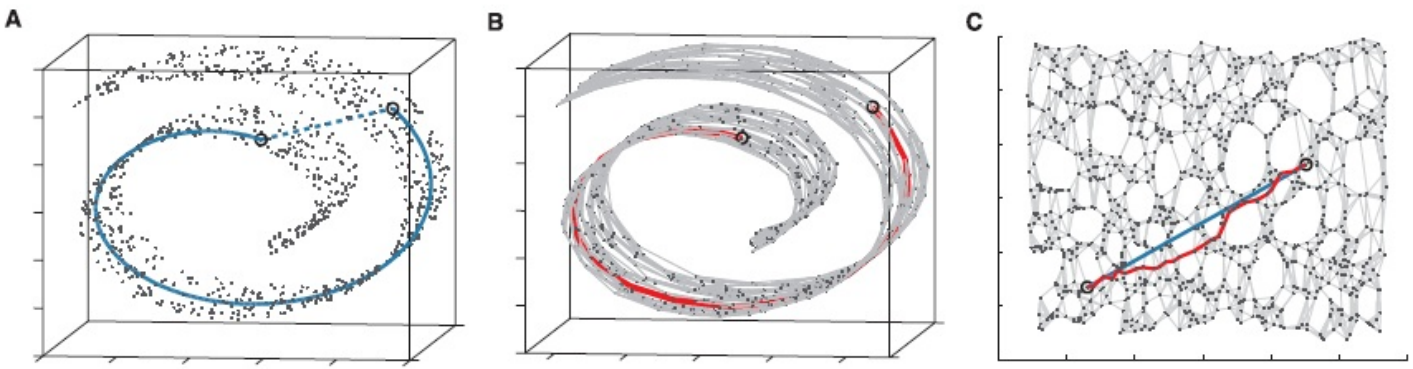


Figure : Isomap. *Figure taken from : Tenenbaum, Joshua B and De Silva, Vin and Langford, John C (2000). A global geometric framework for nonlinear dimensionality reduction. Science, 290(5500) : 2319–2323.*

Locally Linear Embedding (LLE) [7]

- The global nonlinear structure is recovered by **locally linear fits**.
- If the non-linear manifold is smooth, we can assume each data point and its neighbours lie on (or close to) a locally linear patch of the manifold.
- **Goal** : to map the high-dimensional data points to a single global coordinate system such that the **relationships between neighbouring points are preserved**.

7. Roweis, Sam T and Saul, Lawrence K (2000). Nonlinear dimensionality reduction by locally linear embedding. Science, 290(5500) : 2323–2326.

LLE in three steps

1. **Identify the neighbours** $N(i)$ of each data point x_i.
2. For all $x_i \in \mathbb{R}^p$, compute the **weights that best linearly reconstruct** x_i from its neighbours

$$x_i = \sum_{j \in N(i)} w_{ij} x_j + \varepsilon_i.$$

In other words, find $w^i = (w_{i1}, ..., w_{in})$ that minimizes

$$\| x_i - \sum_{j \in N(i)} w_{ij} x_j \|^2 .$$

From a global point of view, we look for a matrix $\mathbf{W} := (w_{ij})_{i,j} \in \mathbb{R}^{n,n}$ such that

$$\underset{\mathbf{W}}{\operatorname{argmin}} \left\{ \sum_{i=1}^{n} \| x_i - \sum_{j \in N(i)} w_{ij} x_j \|^2 \right\} \quad s.t. \quad \sum_{j \in N(i)} w_{ij} = 1, \forall i.$$

LLE last step

- Then, given $\mathbf{W}$, we attempt to find $y_1, ..., y_n \in \mathbb{R}^k$ solution of

$$\min_{\mathbf{W}} \left\{ \sum_{i=1}^{n} \| y_i - \sum_{j \in N(i)} w_{ij} y_j \|^2 \right\}.$$

with

$$\frac{1}{n} \mathbf{Y}^T \mathbf{Y} = \mathbf{I}_k.$$

➤ This is equivalent to solve

$$\underset{\mathbf{Y} \in \mathcal{C}}{\operatorname{argmin}} \left\{ \operatorname{Tr} \left(\mathbf{Y} \mathbf{L} \mathbf{Y}^T \right) \right\}$$

where $\mathbf{L} = (\mathbf{I} - \mathbf{W})^T (\mathbf{I} - \mathbf{W})$ and $\mathcal{C} = \left\{ \mathbf{Y} \in \mathbb{R}^{n,k} : \mathbf{Y}^T \mathbf{Y} = \mathbf{I}_k \right\}$.

➤ The solution is given by the k eigenvectors associated to the lowest k eigenvalues of $\mathbf{L}$.

LLE illustration

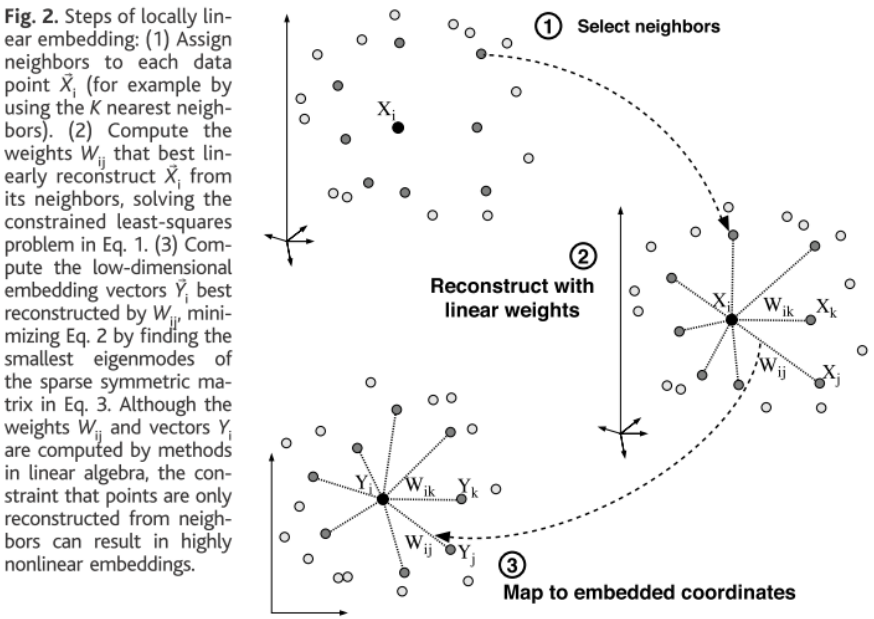


Figure : *Figure taken from : Roweis, Sam T and Saul, Lawrence K (2000). Nonlinear dimensionality reduction by locally linear embedding. Science, 290(5500) : 2323–2326.*

LLE example

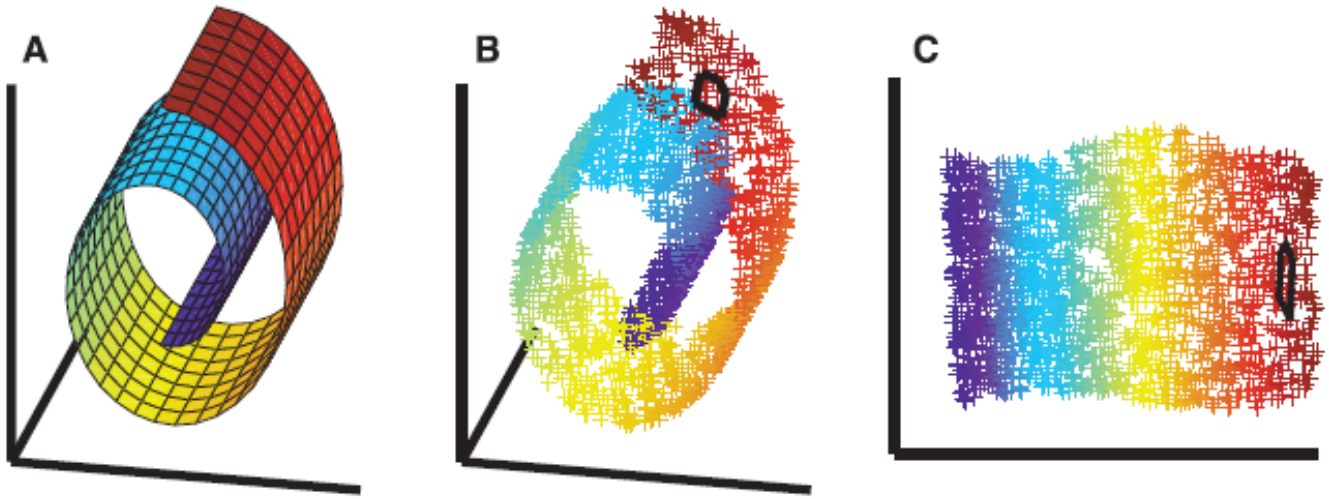


Fig. 1. The problem of nonlinear dimensionality reduction, as illustrated (*10*) for three-dimensional data (B) sampled from a two-dimensional manifold (**A**). An unsupervised learning algorithm must discover the global internal coordinates of the manifold without signals that explicitly indicate how the data should be embedded in two dimensions. The color coding illustrates the neighborhood-preserving mapping discovered by LLE; black outlines in (**B**) and (**C**) show the neighborhood of a single point. Unlike LLE, projections of the data by principal component analysis (PCA) (*28*) or classical MDS (*2*) map faraway data points to nearby points in the plane, failing to identify the underlying structure of the manifold. Note that mixture models for local dimensionality reduction (*29*), which cluster the data and perform PCA within each cluster, do not address the problem considered here: namely, how to map high-dimensional data into a single global coordinate system of lower dimensionality.

Figure : *Figure taken from : Roweis, Sam T and Saul, Lawrence K (2000). Nonlinear dimensionality reduction by locally linear embedding. Science, 290(5500) : 2323–2326.*

Unified framework

Method	Kernel
PCA	$\mathbf{K} = X^T Y$
LLE	$\mathbf{K} = \mathbf{L}^\dagger$ où $\mathbf{L} = (\mathbf{I} - \mathbf{W})^T (\mathbf{I} - \mathbf{W})$
LEM	$\mathbf{K} = \mathbf{L}^\dagger$ où $\mathbf{L} = \mathbf{R} - \mathbf{W}$
MDS	$\mathbf{K} = \frac{1}{2} \left(\mathbf{I} - \mathbf{1}\mathbf{1}^T\right) \mathbf{D} \left(\mathbf{I} - \mathbf{1}\mathbf{1}^T\right)$
ISOMAP	$\mathbf{K} = \frac{1}{2} \left(\mathbf{I} - \mathbf{1}\mathbf{1}^T\right) \mathbf{D}^{\mathcal{G}} \left(\mathbf{I} - \mathbf{1}\mathbf{1}^T\right)$

Multilinear principal component analysis

High-order PCA

➤ Objects of interest in many computer vision and pattern recognition applications, such as 2D or 3D images, are naturally described as **tensors** or multilinear arrays.

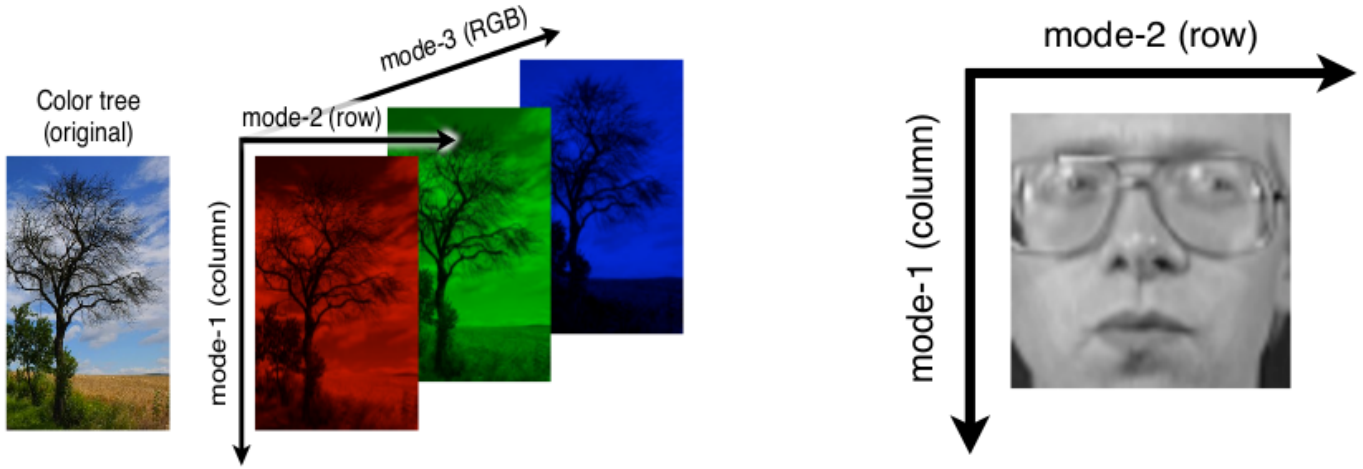


Figure : Color image : order-3 tensors, Black-white image : order-2 tensors [8]

8. Figure taken from : Hung Hung (2012). On multilinear principal component analysis of order-two tensors. Biometrika, 99(3), 569–583.

High-order PCA

- **Vectorization ?** Naive application of PCA to tensor objects requires to reshape the tensors into vectors with high dimensionality.
- **Problem :**
  - High processing cost in terms of increased computational and memory demands.
  - Loss of the **local character of some variables**.
  - Loss of potentially **more compact and useful representations**.

  ➠ We need more efficient dimension reduction tools.
- **Solution :** Multilinear Principal Component Analysis (**MPCA**)

  ➠ Multilinear projection that better captures the variations in the tensorial input space.

Best approximation for a given rank

➤ **PCA : Best rank-k** approximation of the design matrix $\mathbf{X}$ by the sum of the products of **two order-1 tensors**.
Indeed,

$$\mathbf{X} \approx \sum_{j=1}^{k} \lambda_j u_j^{(1)} \circ u_j^{(2)},$$

where $\left\{u_j^{(1)}\right\}_{1\leq j\leq k}$ and $\left\{u_j^{(2)}\right\}_{1\leq j\leq k}$ are the left and right eigenvectors of the SVD of $\mathbf{X}$.

➤ **Generalization :** Let $\mathcal{A}$ be an order-N tensor of size $(I_1, ..., I_N)$. Find the best rank-k tensor of the form

$$\sum_{j=1}^{k} \lambda_j u_j^{(1)} \circ ... \circ u_j^{(N)},$$

where $u_j^{(i)}$ is a vector in $\mathbb{R}^{I_i}$.

MPCA

With illustrations

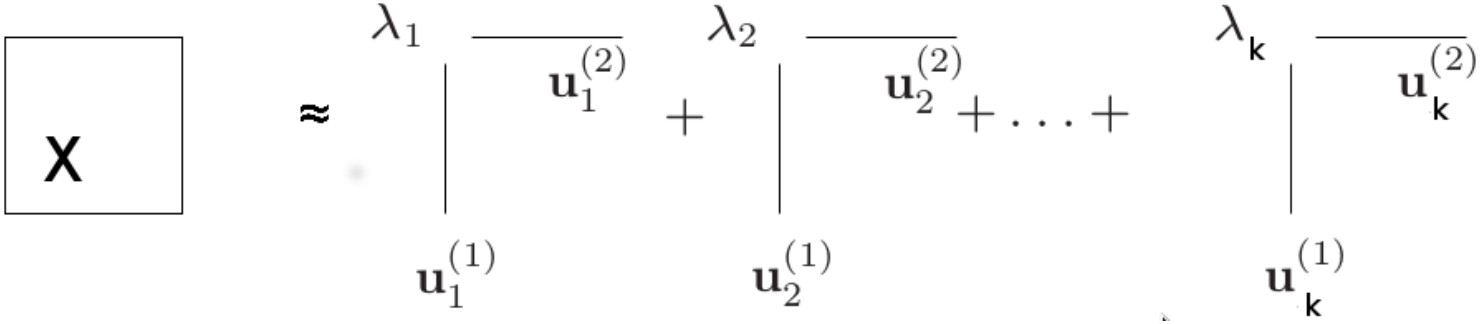


Figure : PCA rank approximation

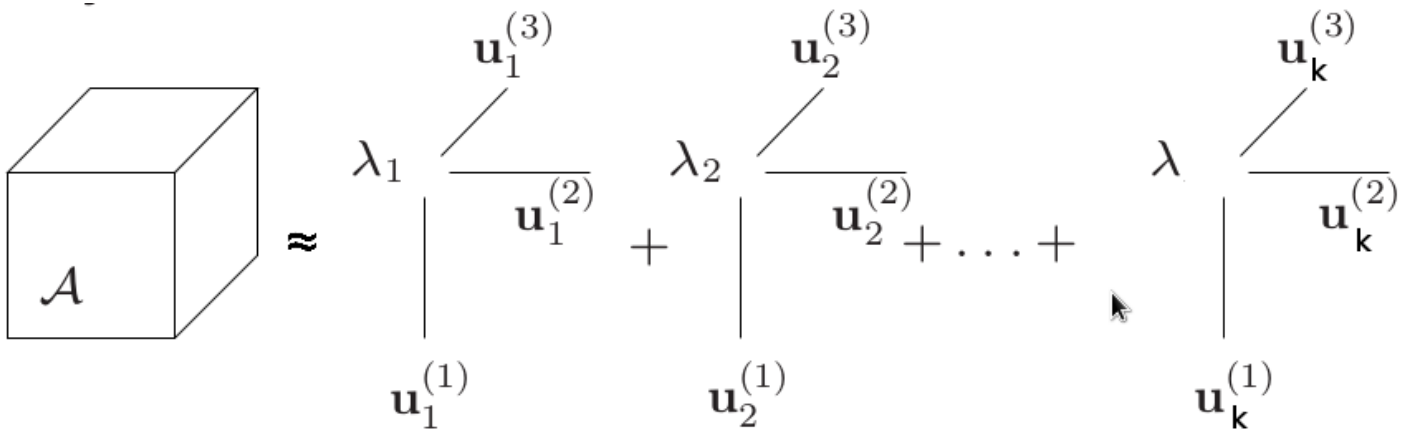


Figure : Tensor rank approximation

High-Order SVD (HO-SVD) [9]

Let $\mathcal{A} \in \mathcal{T}^{I_1 \times I_2 \times \ldots \times I_N}$ an order-N tensor. The HO-SVD of $\mathcal{A}$ is defined by

$$\mathcal{A} = \mathcal{S} \times_1 U^{(1)} \times_2 U^{(2)} \times \ldots \times_N U^{(N)}, \tag{2}$$

1. $U^{(n)}$ is a square orthogonal matrix of size I_n ,i.e

$$U^{(n)^T} U^{(n)} = U^{(n)} U^{(n)^T}) = I_{I_n}. \tag{3}$$

2. $\mathcal{S} \in \mathcal{T}^{I_1 \times I_2 \times \ldots \times I_N}$ and the sub-tensor $\mathcal{S}_{i_n=\alpha}$ (obtained by setting the n^{th} index of the tensor equal to α) satisfies
   - $\forall n \in \{1, ..., N\}, \alpha, \beta$ with $\alpha \neq \beta$

$$\left\langle \mathcal{S}_{i_n=\alpha}, \mathcal{S}_{i_n=\beta} \right\rangle = 0. \tag{4}$$

   - For all $n \in \{1, ..., N\}$,

$$\| \mathcal{S}_{i_n=1} \| \geq \| \mathcal{S}_{i_n=2} \| \geq \cdots \geq \| \mathcal{S}_{i_n=I_n} \| \geq 0. \tag{5}$$

     Notice that, for all $n = 1, ..., N$ and $j = 1, ..., I_n$, $\sigma_j^{(n)} := \| \mathcal{S}_{i_n=j} \|$.

9. De Lathauwer, L., De Moor, B. and Vandewalle, J. (2000). A multilinear singular value decomposition. SIAM journal on Matrix Analysis and Applications, 21(4) : 1253–1278.

HO-SVD

- **Remarks :**
  - For $N = 2$, this is the classical SVD.
  - $\mathcal{S} = \mathcal{A} \times_1 U^{(1)^T} \times_2 U^{(2)^T} \times ... \times_N U^{(N)^T}$.
  - For $N > 2$, $\mathcal{S}$ is not pseudo-diagonal : non-zero coefficients may be out of the indices $i_1 = i_2 = ... = i_N$.
- $\mathcal{A}$ is **combination of order-1 tensors that are mutually orthogonal**

$$\mathcal{A} = \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \dots \sum_{i_n=1}^{I_n} s_{i_1 i_2 \dots i_N} U_{i_1}^{(1)} \circ U_{i_2}^{(2)} \circ ... \circ U_{i_N}^{(N)},$$

where $U_{i_j}^{(l)}$ denotes the j^{th} column of $U^{(l)}$.

- Usually, the summation does not involves $\prod_{n=1}^{N} I_{i_n}$ terms but only $\prod_{n=1}^{N} R_n$ terms, in which R_n is the highest index for which $\| S_{i_n=R_n} \| > 0$.

MPCA

An equivalent representation

➤ **Mode-n unfolding of a tensor $\mathcal{A}$ (denoted by $A_{(n)}$)**

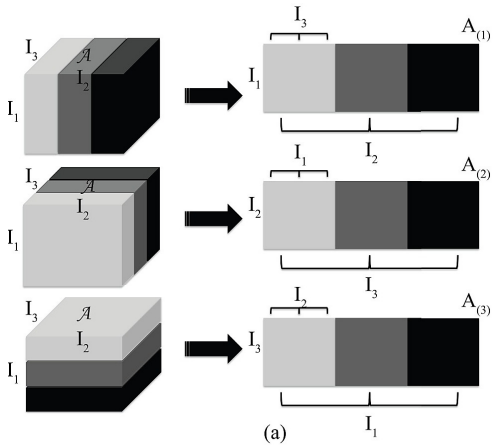


Figure : Unfoldings of an order-3 tensor

➤ The mode-n unfolding of (2) gives the following **matrix decomposition**

$$A_{(n)} = U^{(n)} S_{(n)} \left(U^{(n+1)} \otimes U^{(n)} \otimes \cdots \otimes U^{(N)} \otimes U^{(1)} \otimes U^{(2)} \otimes \cdots \otimes U^{(n-1)} \right)^T,$$

where $\otimes$ denotes the Kronecker product.

Link HO-SVD et SVD

➤ A **matrix representation** of the HO-SVD can be obtained using the unfoldings of $\mathcal{A}$ and $\mathcal{S}$

$$A_{(n)} = U^{(n)} \Sigma^{(n)} {V^{(n)}}^T$$

where

$$\Sigma^{(n)} = \mathrm{diag}\left(\sigma_1^{(n)}, \sigma_2^{(n)}, ..., \sigma_{I_n}^{(n)}\right)$$

and

$${V^{(n)}}^T = {\Sigma^{(n)}}^{-1} S_{(n)} \left(U^{(n+1)} \otimes U^{(n)} \otimes \cdots \otimes U^{(N)} \otimes U^{(1)} \otimes U^{(2)} \otimes \cdots \otimes U^{(n-1)}\right)$$

➤ $S_{(n)}$ has mutually orthogonal rows, having frobenius normal equal to $\sigma_1^{(n)}, \sigma_2^{(n)}, ..., \sigma_{I_n}^{(n)}$.

➤ $V^{(n)}$ is orthogonal, like $U^{(n)}$.

➤ Therefore, the $\left\{U^{(i)}\right\}$ are given by the **SVD on the unfolding of $\mathcal{A}$**.

HO-SVD and dimension reduction [10]

- Let $\hat{\mathcal{A}}$ be the tensor defined by

$$\hat{\mathcal{A}} = \sum_{i_1=1}^{J_1} \sum_{i_2=1}^{J_2} \dots \sum_{i_N=1}^{J_N} s_{i_1 i_2 \dots i_N} U_{i_1}^{(1)} \circ U_{i_2}^{(2)} \circ \dots \circ U_{i_N}^{(N)}$$

with $J_i < R_i$.

- Then, we have

$$\| \mathcal{A} - \hat{\mathcal{A}} \|^2 \leq \sum_{i_1=J_1+1}^{R_1} (\sigma_{i_1}^{(1)})^2 + \sum_{i_2=J_2+1}^{R_2} (\sigma_{i_2}^{(2)})^2 + \dots + \sum_{i_N=J_N+1}^{R_N} (\sigma_{i_N}^{(N)})^2.$$

10. MPCA : Multilinear principal component analysis of tensor objects, Lu, Haiping and Plataniotis, Konstantinos and Venetsanopoulos, Anastasios (2008). IEEE Transactions on Neural Networks, 19(1) : 18–39.

MPCA

With illustrations

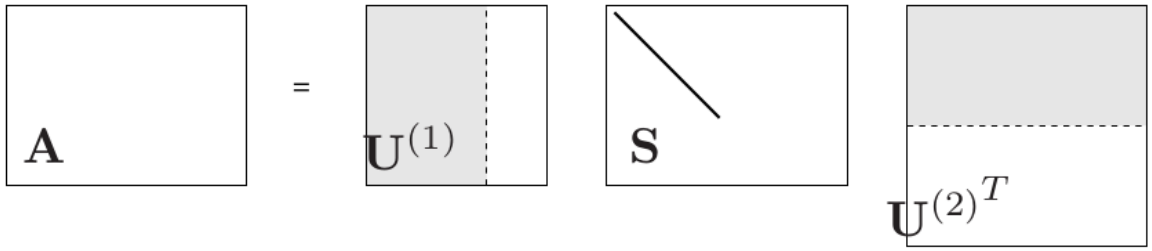


Figure : PCA

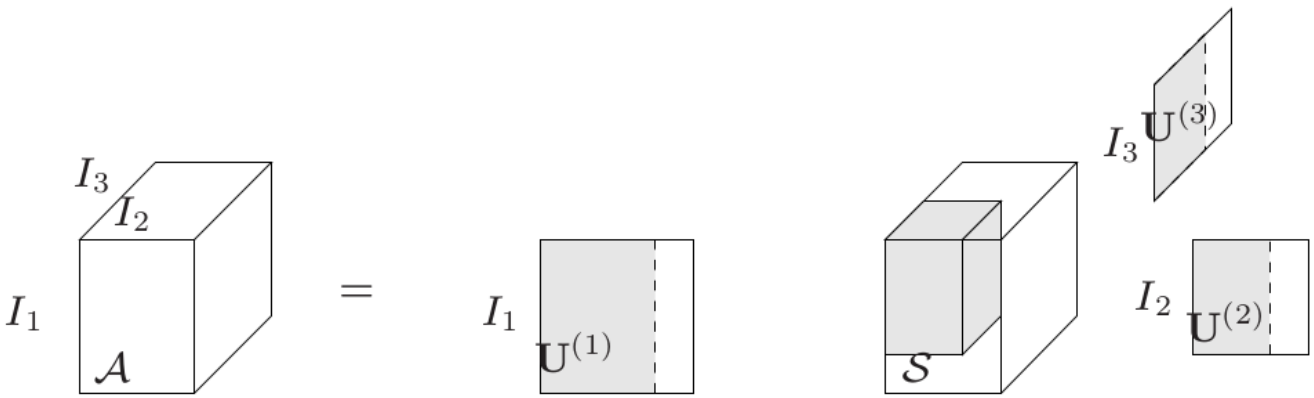


Figure : MPCA

Algorithm

1. For n=1,...,N
   - Given a tensor $\mathcal{A} \in \mathcal{T}^{I_1 \times \ldots \times I_N}$, construct the mode-$n$ unfolding $A_{(n)}$ of $\mathcal{A}$.
   - Compute the SVD of $A_{(n)} := U^{(n)} D^{(n)} {V^{(n)}}^T$, $n = 1, ..., N$ and strore the top k_n eigenvectors in a matrix $U^{(n)}_{k_n}$.
2. The core tensor $\mathcal{S}$ is then the projection of $\mathcal{A}$ onto the tensor basis formed by the factor matrixes $\left\{ U^{(n)}_{k_n} \right\}_{n=1}^{N}$ i.e $\mathcal{S} = \mathcal{A} \times_{n=1}^{N} {U^{(n)}_{k_n}}^T$.

Particular case : 2D-SVD

- Let $\{X_1, ..., X_n\}$ be n realizations of a matrix $X \in \mathbb{R}^{I,c}$.
- Define

$$F = \sum_{i=1}^{n} (X_i - \overline{X})(X_i - \overline{X})^T$$

$$G = \sum_{i=1}^{n} (X_i - \overline{X})^T (X_i - \overline{X}),$$

where $\overline{X} = \frac{1}{n} \sum_{i=1}^{n} X_i$.

- Let U_k the matrix containing the top k eigenvectors of F and V_s the one containing the top s eigenvectors of G, with $k \ll c$ and $s \ll I$.
- In the same spirit as SVD, X_i is approximated by

$$X_i \approx U_k M_i V_s^T,$$

where $M_i = U_k^T X_i V_s$.

An example : comparison MPCA and PCA

Figure : **20 test faces randomly drawn (rows 1-2), reconstructions by MPCA (rows 3-4) and PCA (rows 5-6).** *Figure taken from : Hung Hung (2012). On multilinear principal component analysis of order-two tensors. Biometrika, 99(3) : 569–583.*

An example : different performances of MPCA and PCA

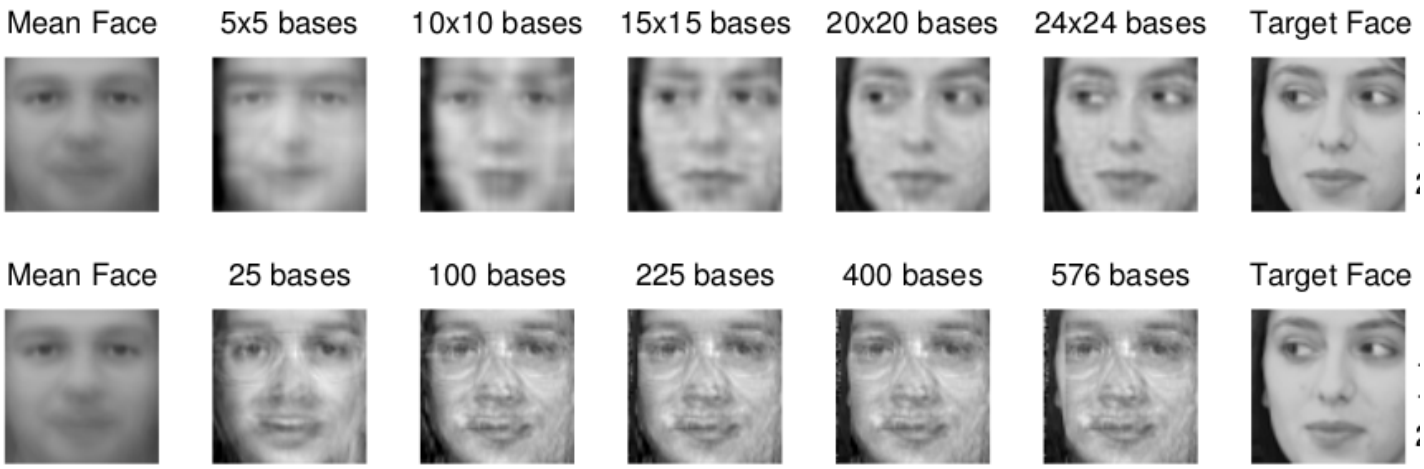


Figure : **Image reconstruction, MPCA (top) and PCA (bottom).** *Figure taken from : Hung Hung (2012). On multilinear principal component analysis of order-two tensors. Biometrika, 99(3) : 569–583.*

➤ **Better performance of MPCA** for this example.

MPCA

An example : leading 100 basis

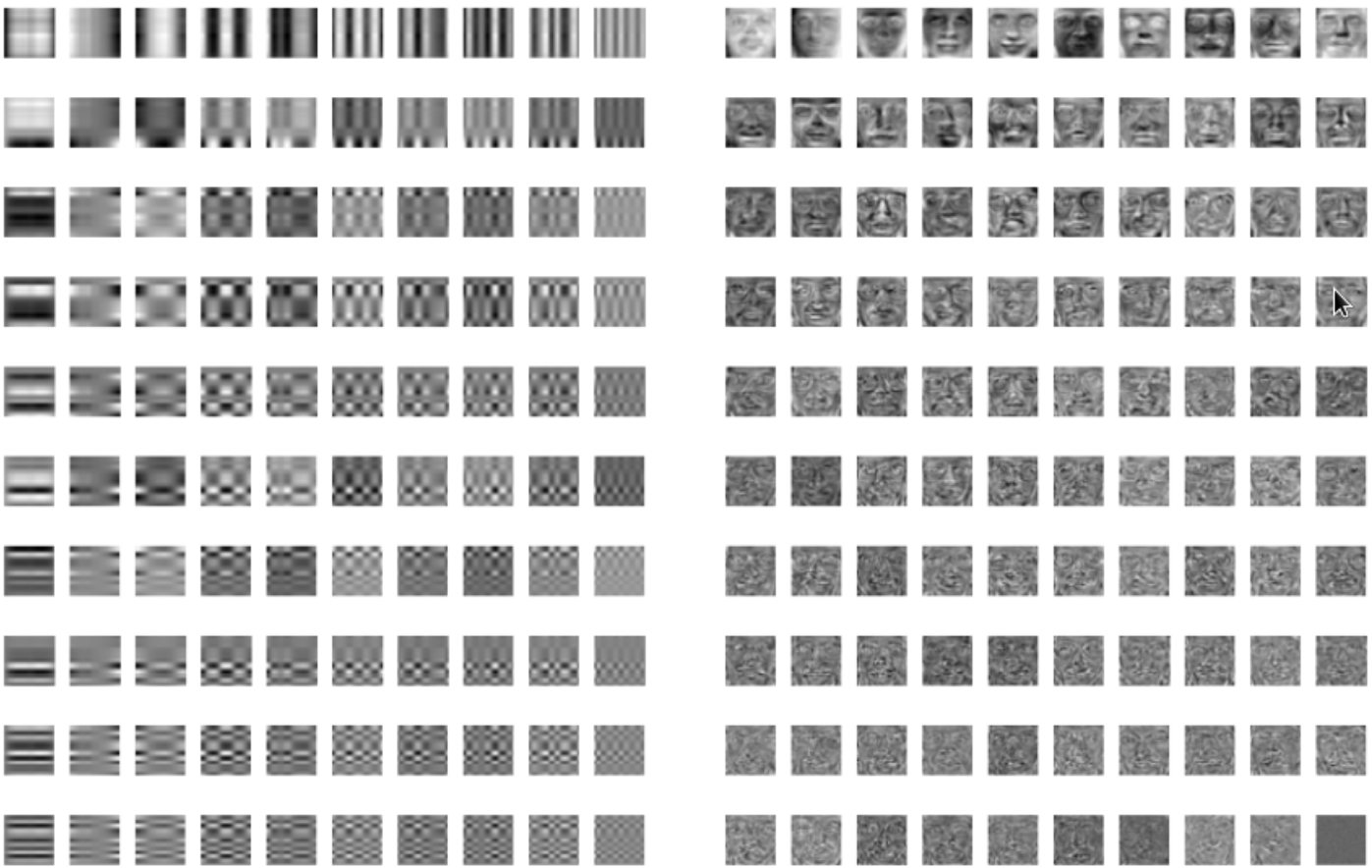


Figure : **MPCA : more module oriented, PCA : too much information in a basis.** *Figure taken from : Hung Hung (2012). On multilinear principal component analysis of order-two tensors. Biometrika, 99(3) : 569–583.*

Conclusion

- **Survey of some important dimension reduction methods, but there are many other ones.**
  - Latent probabilistic models (FA, PPCA, PICA,...)
  - Dimension reduction methods for functional data.
  - PLS
  - Projection Pursuit
  - Random projection
  - ...
- **The reason of the dimension reduction is an important issue**
  - Compression
  - Visualization
  - Regression
  - Classification
  - Simulation
  - ....

to choose the best method !

Conclusion

PCA, ICA, NMF, MPCA or non-linear methods ?

- **Depends on what you want to do**
  - PCA works well for dimension reduction.
  - ICA provides a clean output, obtains maximal independence.
  - NMF provides interpretable output, but only for non-negative data and has no particular statistical properties.
  - MPCA is well suited for observations that have a tensorial stucture.
  - Non-linear methods can be used to extract features or to do classification on manifolds, but not for compression or simulation.
- **When in doubt, try them all !**

Thank you for your attention

A first reference for each method

- **ICA** : Hyvärinen, Aapo and Oja, Erkki (2000). **Independent component analysis : algorithms and applications.** *Neural networks*, 13(4), 411–430. Elsevier.
- **NMF** : Paatero, Pentti and Tapper, Unto (1994). **Positive matrix factorization : A non-negative factor model with optimal utilization of error estimates of data values.** *Environmetrics*, 5(2), 111–126. Wiley Online Library.
- **Kernel PCA** : Schölkopf, Bernhard and Smola, Alexander and Müller, Klaus-Robert (1997). **Kernel principal component analysis.** *Artificial Neural Networks—ICANN'97*, 583-588. Springer.
- **ISOMAP** : Tenenbaum, Joshua B and De Silva, Vin and Langford, John C (2000). **A global geometric framework for nonlinear dimensionality reduction.** *Science*, 290(5500), 2319–2323. American Association for the Advancement of Science.

A first reference for each method

- **LLE** : Roweis, Sam T and Saul, Lawrence K (2000). **Nonlinear dimensionality reduction by locally linear embedding.** *Science*, 290(5500), 2323–2326, American Association for the Advancement of Science.
- **MPCA/ HO-SVD** : De Lathauwer, Lieven and De Moor, Bart and Vandewalle, Joos (2000). **A multilinear singular value decomposition**. *SIAM journal on Matrix Analysis and Applications*, 21(4), 1253–1278, SIAM.

R Packages

- **PCA :** https ://cran.r-project.org/web/packages/FactoMineR/index.html
- **ICA :** https ://cran.r-project.org/web/packages/fastICA/index.html
- **NMF :** https ://cran.r-project.org/web/packages/NMF/index.html
- **Kernel PCA :**
  https ://cran.r-project.org/web/packages/kernlab/index.html
- **ISOMAP et LLE :**
  https ://www.bioconductor.org/packages/3.3/bioc/vignettes/RDRToolbox/inst/d
- **MPCA :** https ://cran.r-project.org/web/packages/rTensor/index.html.