MAP 534

Introduction to machine learning

Dimension reduction

Principal component analysis (PCA) & Kernel PCA

Outline

Machine Learning in a few words

Classification

$\rightharpoonup$ Learn whether an individual from $\mathbb{R}^d$ belongs to some class.

$\rightharpoonup$ Focus usually set on learning with a known number M of classes: an individual is associated with a label in $\{1, \ldots, M\}$.

$\rightharpoonup$ The statistical model is then given by $(X, Y) \in \mathbb{R}^p \times \{1, \ldots, M\}$ and the objective is to define a function $f : \mathbb{R}^p \to \{1, \ldots, M\}$, called classifier, such that $f(X)$ is the best prediction of Y in a given context.

Regression

$\rightharpoonup$ The observation associated with X is assumed to be given by

$$Y = f(X) + \varepsilon\,,$$

where ε is a centered noise independent of X.

$\rightharpoonup$ The statistical model is then given by $(X, Y) \in \mathbb{R}^p \times \mathbb{R}^m$ and the objective is to define the best estimator of f in a given context.

State sequence X characterizing the system:

$\rightarrow$ discrete/continuous-time, discrete/continuous state space ;

$\rightarrow$ law given by a parametric or nonparametric probabilistic model ;

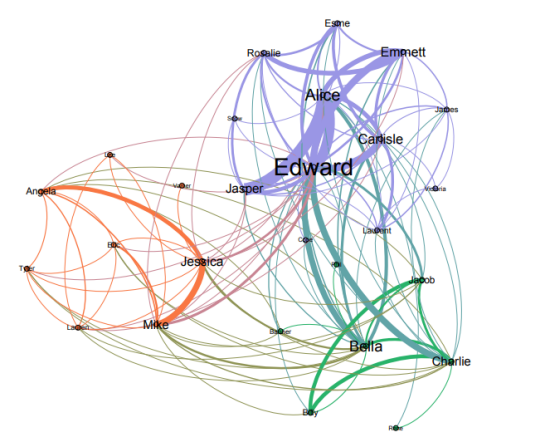
$\rightarrow$ independent state or distribution governed by either a Markovian or a slowly mixing behavior.

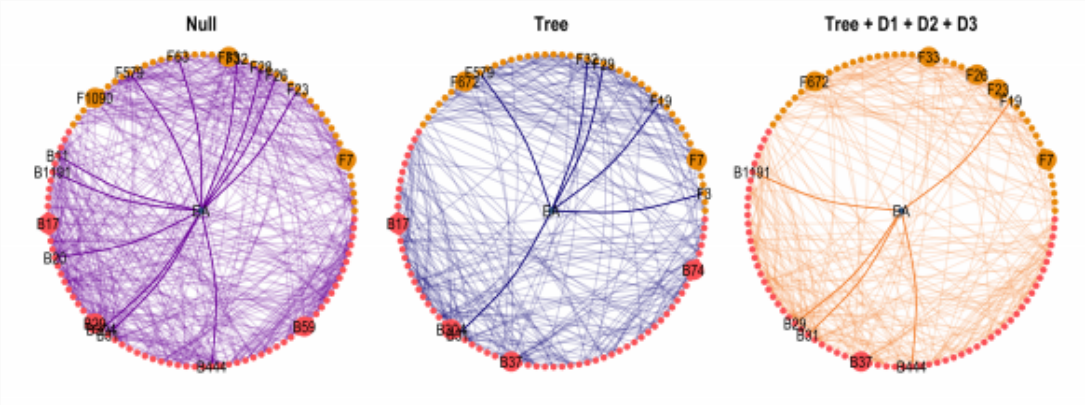
Sequence of observations Y characterizing at each time step this system:

$\rightarrow$ discrete/continuous-time, discrete/continuous observation space ;

$\rightarrow$ parametric or nonparametric probabilistic model.

Machine Learning in a few words - clustering





→ Task: group characters/news in texts/journals, infer an ecological network[1].

→ Performance: quality of the clusters.

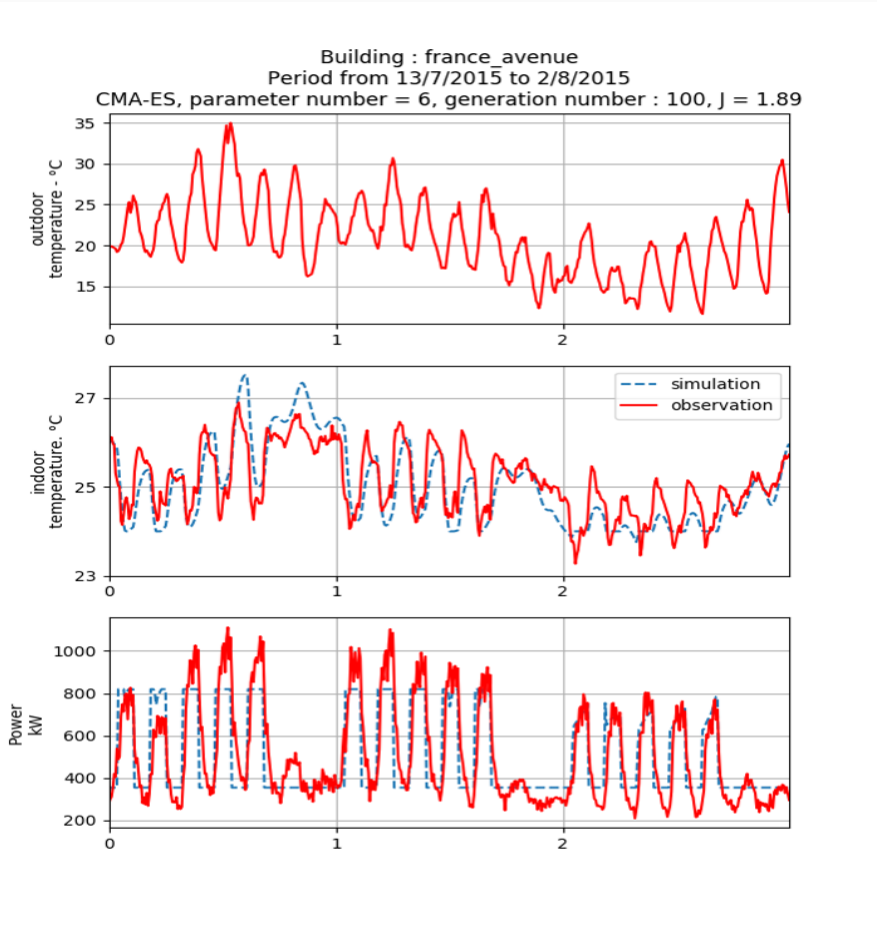
→ Experience: number of occurrences of two key names both within a specified distance in the text / abundance of all species in all sites.

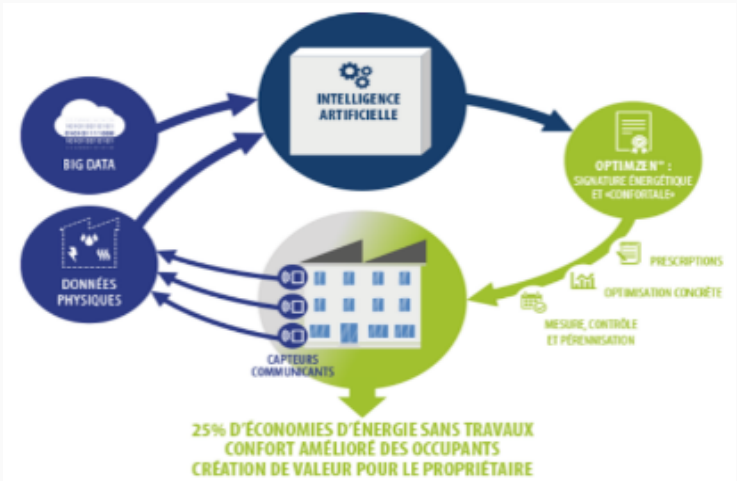
[1] **Tree-based Reconstruction of Ecological Network from Abundance Data**, Momal, R. et al., ***ArXiv:1905.02452***

Machine Learning in a few words - object recognition

- Task: say if an object is present or not in the image.
- Performance: number of errors.
- Experience: set of previously seen labeled images.

Machine Learning in a few words - prediction of time series



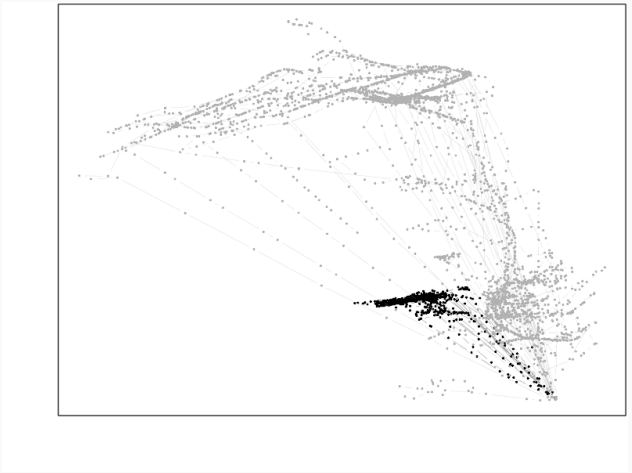
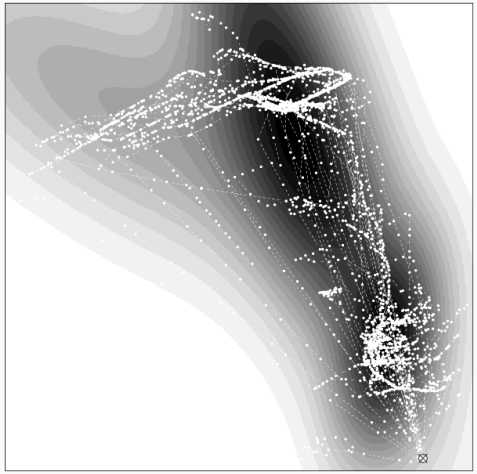
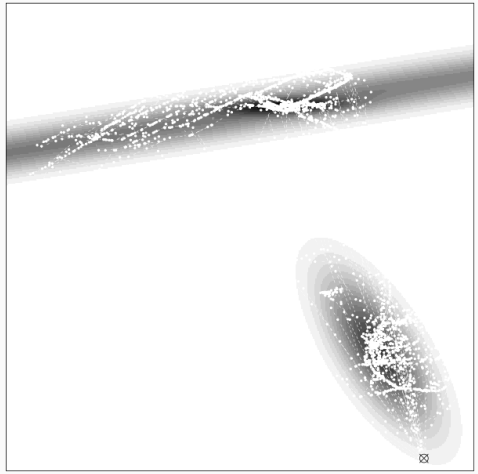


→ Task: predict future consumptions, gas emissions and temperatures.

→ Performance: comfort/ecological scores.

→ Experience: millions of observations obtained from hundreds of sensors.

Machine Learning in a few words - Continuous time target tracking

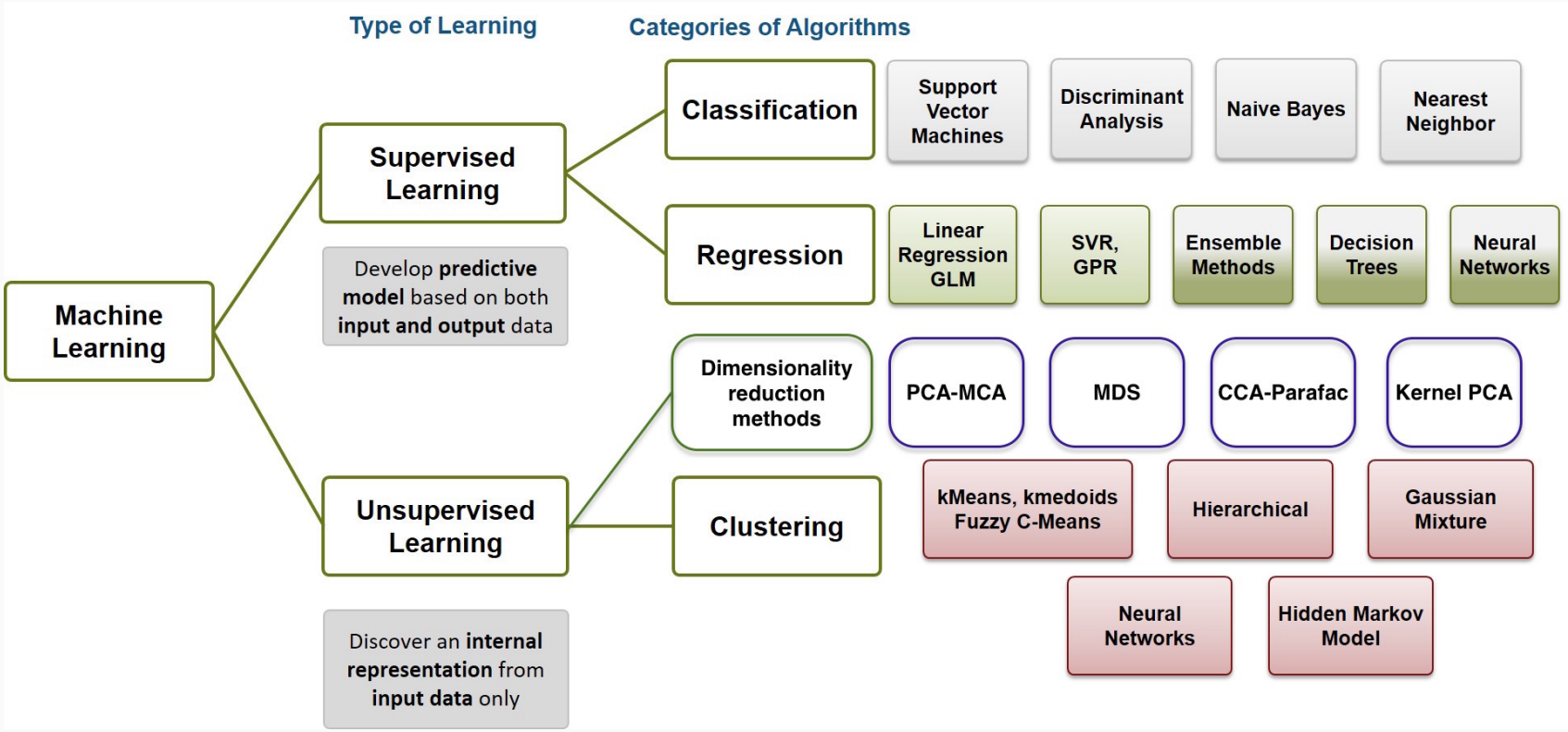
- $(X_t)_{t \geqslant 0}$ is the **continuous time position** (coordinates, orientation, etc.):

$$\mathrm{d}X_t = \alpha_\theta(X_t)\mathrm{d}t + \sigma_\theta(X_t)\mathrm{d}W_t \,.$$

- An observation (GPS...) at each **discrete time** t_k.

$$Y_k = f_\star(X_k) + \varepsilon_k \,.$$

Machine Learning in a few words



MAP 534 - Objectives

$\rightarrow$ Master the statistical learning framework and its challenges.

$\rightarrow$ Know the inner mechanism of some classical machine learning methods. *Principal component analysis, Logistic regression, Support Vector Machines, Neural networks, gradient descent*.

$\rightarrow$ Know how to implement (R and Python) the most classical learning algorithms.

$\rightarrow$ Understand some optimization tools used in ML as well as some theoretical aspects of ML.

Evaluation

$\rightarrow$ A practical lab on optimization (5 pt).

$\rightarrow$ A final exam (15 pt).

MAP 534 - Team

Lectures: Sylvain Le Corff

Practical sessions: Thomas Kerdreux, Sylvain Le Corff

Dimensionality reduction

Find a **low-dimensional representation** that captures the statistical properties of high-dimensional data.

- Compression, denoising, data completion, anomaly detection.
- Preprocessing before supervised learning (improve performances / regularization to reduce overfitting).
- Simplify the description of massive datasets.
- Provide tools to analyze both observations and variables.

descriptive methods to better understand the data (difficult to plot and interpret).

- Principal component Analysis (PCA): continuous data.
- Correspondence analysis (CA): contingency table.
- Multiple correspondence analysis (MCA): categorical data.
- Multiple factor analysis (MFA): multi-table, array data.

$\rightarrow$ Implicit assumption that the data **lies in a low-dimensional manifold of the orginial space**.

Outline

Singular value decomposition

For all $\mathbb{R}^{n\times d}$ matrix A with rank r, there exist $\sigma_1 \geqslant \ldots \geqslant \sigma_r > 0$ such that

$$A = \sum_{k=1}^{r} \sigma_k u_k v_k^T \ ,$$

where $\{u_1, \ldots, u_r\} \in (\mathbb{R}^n)^r$ and $\{v_1, \ldots, v_r\} \in (\mathbb{R}^d)^r$ are two orthonormal families.

- The real numbers $\{\sigma_1, \ldots, \sigma_r\}$ are called singular values of A.
- The vectors $\{u_1, \ldots, u_r\}$ are the left-singular vectors of A.
- The vectors $\{v_1, \ldots, v_r\}$ are the right-singular vectors of A.

$\rightharpoonup$ The rank of A is the **dimension of the space generated by the columns of A ($r \leqslant n$)** and is **equal to the dimension of the space generated by the rows of A ($r \leqslant d$)**.

$\rightharpoonup$ The rank of A is the number of nonzero singular values of A.

Singular value decomposition

If U denotes the $\mathbb{R}^{n\times r}$ matrix with columns given by $\{u_1, \ldots, u_r\}$ and V denotes the $\mathbb{R}^{p\times r}$ matrix with columns given by $\{v_1, \ldots, v_r\}$, then the singular value decomposition of A may also be written as

$$A = UD_rV^T \ ,$$

where $D_r = \mathrm{diag}(\sigma_1, \ldots, \sigma_r)$.

The singular value decomposition is closely related to the spectral theorem for symmetric semipositive definite matrices.

The matrices A^TA and AA^T are positive semidefinite such that

$$A^TA = VD_r^2V^T \quad \text{and} \quad AA^T = UD_r^2U^T \ .$$

Singular value decomposition

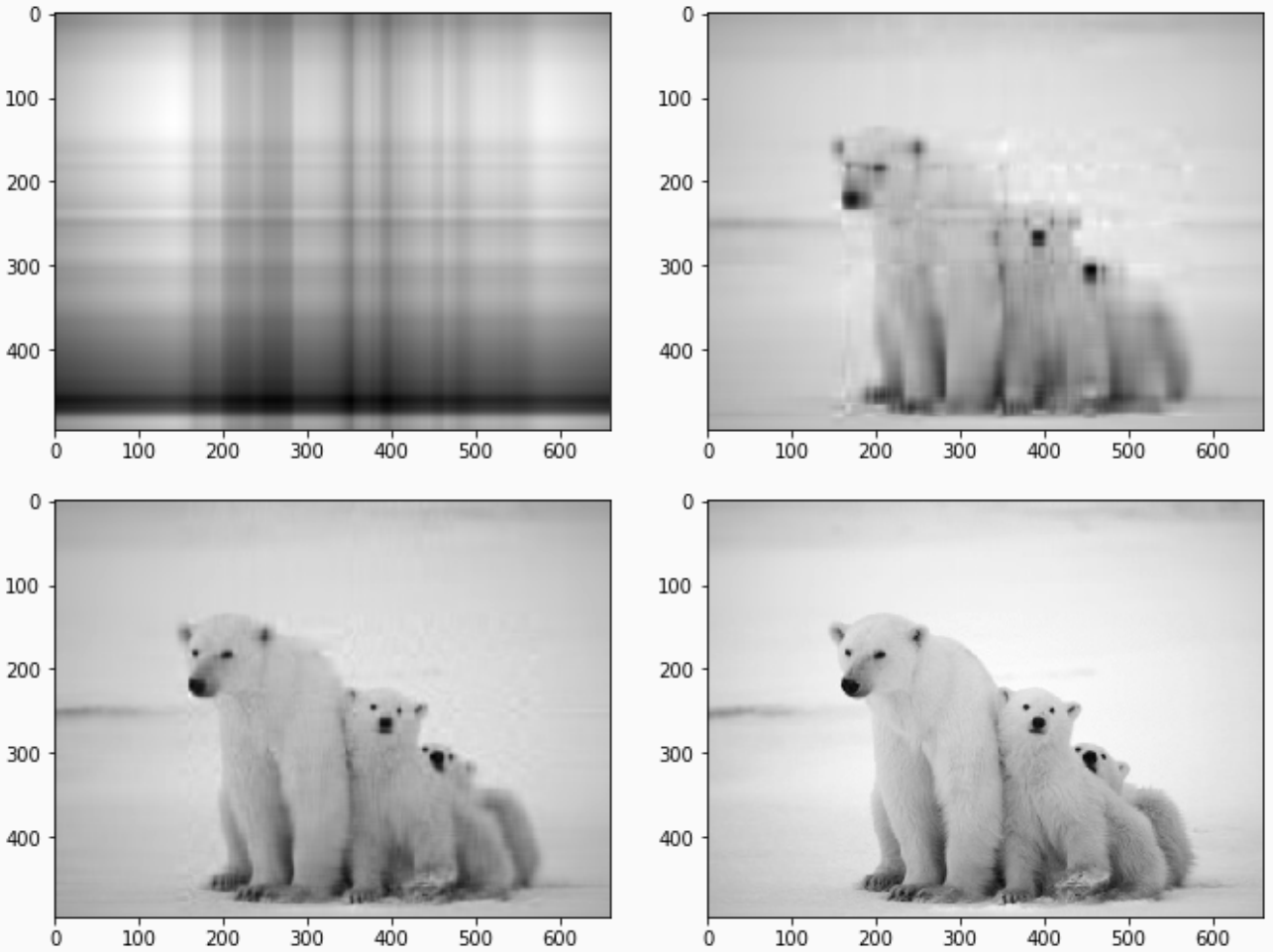
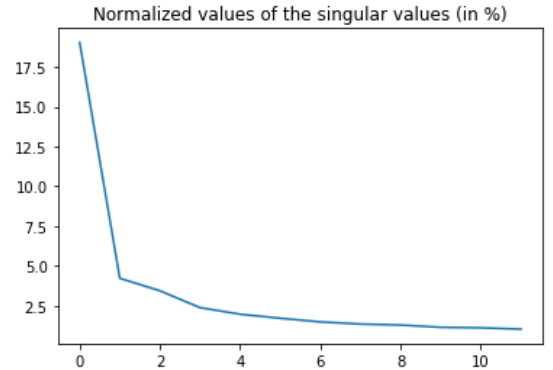


Figure 1: The original image is a $\mathbb{R}^{495\times660}$ matrix of pixels (grey scale). The image is then reconstructed using only the largest (top left), the first ten (top right), the first 25 (bottom left) and the first 100 (bottom right) singular values.

Singular value decomposition

```python
# write a function with input the path of an image "path_image" and an integer "k"
# and return in gray scale the reconstructed picture with the first k singular values
def svd_decomposition(path_image,k):
    img           = Image.open(path_image)
    img_mat       = np.array(list(img.getdata(band=0)), float)
    img_mat.shape = (img.size[1], img.size[0])
    img_mat       = np.matrix(img_mat)

    # Perform Singular Value Decomposition
    U, sigma, V = np.linalg.svd(img_mat)
    print('Size left singular eigenvectors  ' + str(np.shape(U)))
    print('Size right singular eigenvectors  ' + str(np.shape(V)))
    print('Size eigenvalues matrix ' + str(np.shape(sigma)))

    # Image reconstruction
    reconstimg = np.matrix(U[:, :k]) * np.diag(sigma[:k]) * np.matrix(V[:k, :])

    fig = plt.figure(1)
    plt.plot(sigma[0:12]*100/np.sum(sigma))
    plt.title("Normalized values of the singular values (in %)")
    fig = plt.figure(2)
    plt.title("Image reconstruction with %g singular values"%k)
    plt.imshow(reconstimg, cmap='gray')
    plt.axis('off')
```

```
Size left singular eigenvectors  (1415, 1415)
Size right singular eigenvectors  (2122, 2122)
Size eigenvalues matrix (1415,)
```



Outline

Principal component analysis

Principal component analysis is a multivariate technique which allows to analye the statistical structure of high dimensional dependent observations by representing data using orthogonal variables called *principal components*.

Let $(X_i)_{1 \leqslant i \leqslant n}$ be i.i.d. random variables in $\mathbb{R}^d$ and consider the matrix $X \in \mathbb{R}^{n \times d}$ such that the i-th row of X is the observation X_i^T.

It is assumed that data are preprocessed so that the columns of X are centered. This means that for all $1 \leqslant p \leqslant d$, $\sum_{i=1}^n X_{i,k} = 0$. Let Σ_n be the empirical covariance matrix:

$$\Sigma_n = n^{-1} \sum_{i=1}^n X_i X_i^T \ .$$

Note that $\sum_{i=1}^n X_i X_i^T = X^T X$ so that $\Sigma_n = (X/\sqrt{n})^T (X/\sqrt{n})$ and the eigenvalue decomposition of Σ_n allows to recover the singular values of $X/\sqrt{n}$ and its right-singular vectors.

PCA for which data?

→ **Environmental data**

Waters - physico-chemical analyses, towns - temperature.

→ **Economy**

Countries - economic indicators.

→ **Biology**

Cheeses - microbiological analyses, tumors - genes expression.

Observations study

Similarity between observations with respect to all the variables, partition between observations.

Variables study

Linear relationships between variables, visualization of the correlation matrix, find synthetic variables.

Link between the two studies

Characterization of the groups of observations with variables, specific observations to understand links between variables.

Fit the observations cloud

Find the subspace which best sums up the data.

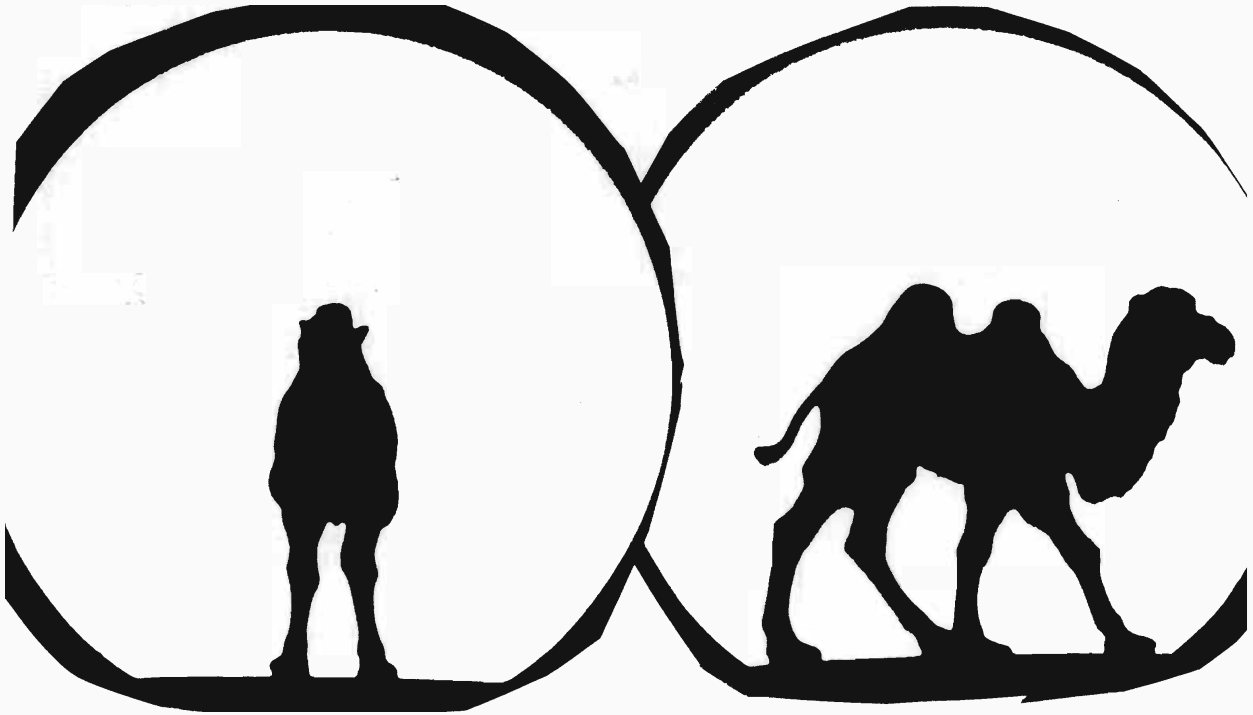


Figure 2: Camel vs dromedary? (J.P. Fenelon)

→ Closest representation by projection.

→ Best representation of the diversity, variability of the observations.

Fit the observations cloud - a compressed sensing approach

$\rightharpoonup$ **Reducing the dimensionality of the observations** $(X_i)_{1 \leqslant i \leqslant n}$ using a *compression* matrix $W \in \mathbb{R}^{p \times d}$ with $p \leqslant d$ so that for each $1 \leqslant i \leqslant n$, WX_i ia a low dimensional representation of X_i.

$\rightharpoonup$ The original observation may then be partially recovered using another matrix $U \in \mathbb{R}^{d \times p}$.

$\rightharpoonup$ Principal Component Analysis computes U and W using the **least squares approach**:

$$(U_\star, W_\star) \in \underset{(U,W) \in \mathbb{R}^{d \times p} \times \mathbb{R}^{p \times d}}{\operatorname{argmin}} \sum_{i=1}^{n} \|X_i - UWX_i\|^2 ,$$

$\rightharpoonup$ Let $(U_\star, W_\star) \in \mathbb{R}^{d \times p} \times \mathbb{R}^{p \times d}$ be a solution. **Then, the columns of $U_\star$ are orthonormal and $W_\star = U_\star^T$.**

Fit the observations cloud - a compressed sensing approach

Principal Component Analysis computes U and W using the **least squares approach**:

$$U_\star \in \underset{U \in \mathbb{R}^{d\times p}\ ; U^T U = I_p}{\operatorname{argmin}} \ \sum_{i=1}^n \|X_i - UU^T X_i\|^2 \ ,$$

For all $U \in \mathbb{R}^{d\times p}$ such that $U^T U = I_p$,

$$\begin{aligned}
\sum_{i=1}^n \|X_i - UU^T X_i\|^2 &= \sum_{i=1}^n \|X_i\|^2 + \sum_{i=1}^n X_i^T UU^T X_i - 2\sum_{i=1}^n X_i^T UU^T X_i \ , \\
&= \sum_{i=1}^n \|X_i\|^2 - \sum_{i=1}^n X_i^T UU^T X_i \ , \\
&= \sum_{i=1}^n \|X_i\|^2 - \operatorname{trace}(U^T XX^T U) \ .
\end{aligned}$$

Therefore, solving the PCA problem boils down to computing

$$U_\star \in \underset{U \in \mathbb{R}^{d\times p} \ , \ U^T U = I_p}{\operatorname{argmax}} \ \{\operatorname{trace}(U^T \Sigma_n U)\} \ .$$

Fit the observations cloud - a compressed sensing approach

Let $\{\vartheta_1, \ldots, \vartheta_d\}$ be **orthonormal eigenvectors associated with the eigenvalues** $\lambda_1 \geqslant \ldots \geqslant \lambda_d$ **of** Σ_n. Then a solution to the PCA problem is given by the matrix $U_\star$ with columns $\{\vartheta_1, \ldots, \vartheta_p\}$ and $W_\star = U_\star^T$.

$\rightharpoonup$ Compute the matrix Σ_n, obtain its eigenvectors sorted by eigenvalues.

$\rightharpoonup$ Build the matrices $U_\star$ and $W_\star = U_\star^T$.

$\rightharpoonup$ Compressed data in $\mathbb{R}^p$ given by $W_\star X_i$ for all $1 \leqslant i \leqslant n$.

Fit the observations cloud - a projection approach

For any dimension $1 \leqslant p \leqslant d$, let $\mathcal{F}_d^p$ be the set of all vector suspaces of $\mathbb{R}^d$ with dimension p.

Define the linear span V_p as

$$V_p \in \underset{V \in \mathcal{F}_d^p}{\operatorname{argmin}} \sum_{i=1}^n \|X_i - \pi_V(X_i)\|^2 \,,$$

where π_V is the orthogonal projection onto the linear span V.

For all $1 \leqslant p \leqslant d$, a solution is given by $V_p = \operatorname{span}\{v_1, \ldots, v_p\}$ where

$$v_1 \in \underset{v \in \mathbb{R}^d \,;\, \|v\|=1}{\operatorname{argmax}} \sum_{i=1}^n \langle X_i, v\rangle^2 \quad \text{and for } k \geqslant p \,, \quad v_k \in \underset{\substack{v \in \mathbb{R}^d \,;\, \|v\|=1 \,; \\ v \perp v_1, \ldots, v \perp v_{k-1}}}{\operatorname{argmax}} \sum_{i=1}^n \langle X_i, v\rangle^2 \,.$$

$\rightharpoonup$ By induction, $\{\vartheta_1, \ldots, \vartheta_p\}$, the **orthonormal eigenvectors associated with the eigenvalues $\lambda_1 \geqslant \ldots \geqslant \lambda_p$ of Σ_n**, are solutions to this problem.

Fit the observations cloud - first dimension

The first dimension is defined by

$$v_1 \in \underset{v \in \mathbb{R}^d \,;\, \|v\|=1}{\operatorname{argmax}} \sum_{i=1}^{n} \langle X_i, v \rangle^2$$

The projection of each observation on this dimension is

$$\pi_{V_1}(X_i) = \langle X_i, \vartheta_1 \rangle \vartheta_1 \ .$$

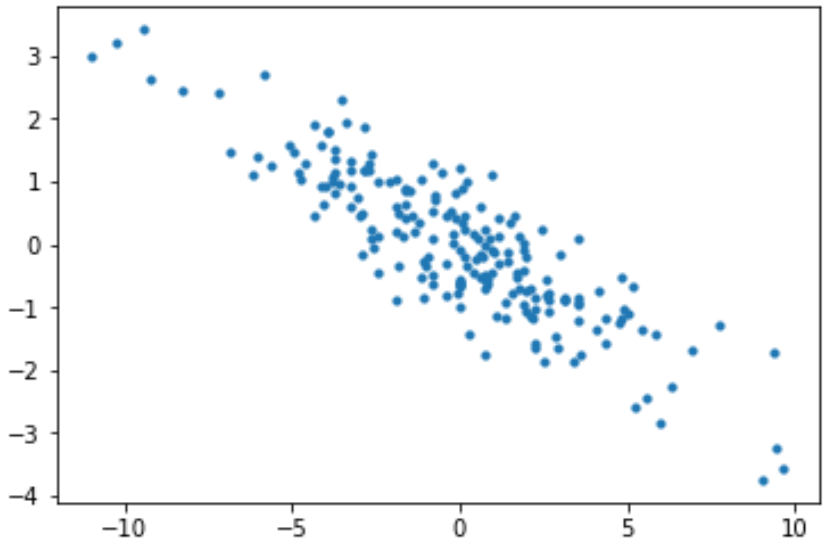
The first principal component is the vector with i-th entry equal to the coordinate of X_i in V_1:

$$c_1 = X\vartheta_1 \ , \quad \pi_{V_1}(X_i) = c_1(i)\vartheta_1 \ .$$

Fit the observations cloud - first dimension

```
# Build a data set with n=100 bi-dimensional i.i.d. data with centered Gaussian distribution
X = np.dot(np.random.normal(0,1,[2,2]), np.random.normal(0,1,[2,200])).T
plt.scatter(X[:, 0], X[:, 1], s = 10)
```

```
<matplotlib.collections.PathCollection at 0x2b7a92b99e8>
```



Fit the observations cloud - first dimension

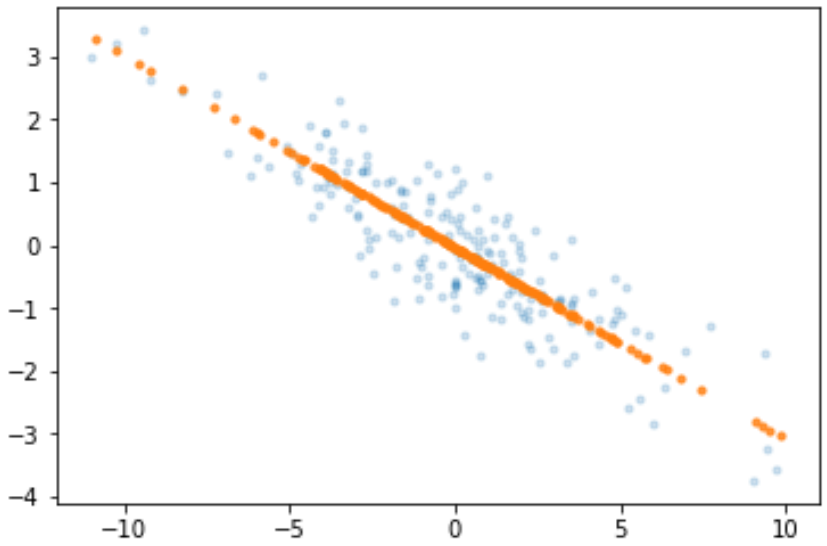
```python
# perform a PCA with one component using PCA(n_components=1) and the function fit.
from sklearn.decomposition import PCA
pca = PCA(n_components=1)
pca.fit(X)
print('The principal component is:')
print(pca.components_)

print('The explained variance is %g'%pca.explained_variance_)
print('The associated singular value is %g'%pca.singular_values_)

# Apply the dimensionality reduction on X
# X_pca contains the coordinates of each data in the space generated by the principal components
X_pca = pca.transform(X)

# in this case pca.components_[k] contains the coordinates of the k-th principal component in
# the original space (here the usual Euclidian plane). In a general case pca.components_[k] is a
# d-dimensional vector.
# X_pca[i] contains the coordinates of the i-th data in the vector space generated by the principal
# components.
# Therefore, X_pca[i] is a vector with n_components entries.
```

```
The principal component is:
[[-0.95642601  0.29197481]]
The explained variance is 13.5057
The associated singular value is 51.8423
```

Fit the observations cloud - first dimension

```
# transform the reduced data set in the original space
# X_inverse[i,:] contains the coordinates of the projection of Xi in the original space
X_inverse = pca.inverse_transform(X_pca)
plt.scatter(X[:, 0], X[:, 1], alpha=0.2, s=10)
plt.scatter(X_inverse[:, 0], X_inverse[:, 1], alpha=0.8, s=10)
```

```
<matplotlib.collections.PathCollection at 0x2b7ab9442e8>
```



Fit the observations cloud - additional dimensions

As $V_p = \text{span}\{\vartheta_1, \ldots, \vartheta_p\}$, for all $1 \leqslant i \leqslant n$,

$$\pi_{V_p}(X_i) = \sum_{k=1}^{p} \langle X_i, \vartheta_k \rangle \vartheta_k = \sum_{k=1}^{p} (X_i^T \vartheta_k) \vartheta_k = \sum_{k=1}^{p} c_k(i) \vartheta_k \ ,$$

where for all $1 \leqslant k \leqslant p$, the k-th principal component is defined as $c_k = X\vartheta_k$.

As already highlighted, $U_*^T X_i$ contains the coordinates of $\pi_{V_p}(X_i)$ in V_p.

Representation quality (dimensionality reduction loses information)

$\rightharpoonup$ Total variance of the observations (total inertia):

$$\frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{d} (X_{ik})^2 = \text{trace}(\Sigma_n) = \sum_{k=1}^{d} \lambda_k \ .$$

$\rightharpoonup$ Variance of the projected observations (p-dimensional representation):

$$\text{Var}(c_1) + \ldots + \text{Var}(c_p) \ .$$

$\rightharpoonup$ Percentage of inertia explained: $\sum_{k=1}^{p} \lambda_k / \sum_{k=1}^{d} \lambda_k$.

Fit the observations cloud - graphical representation

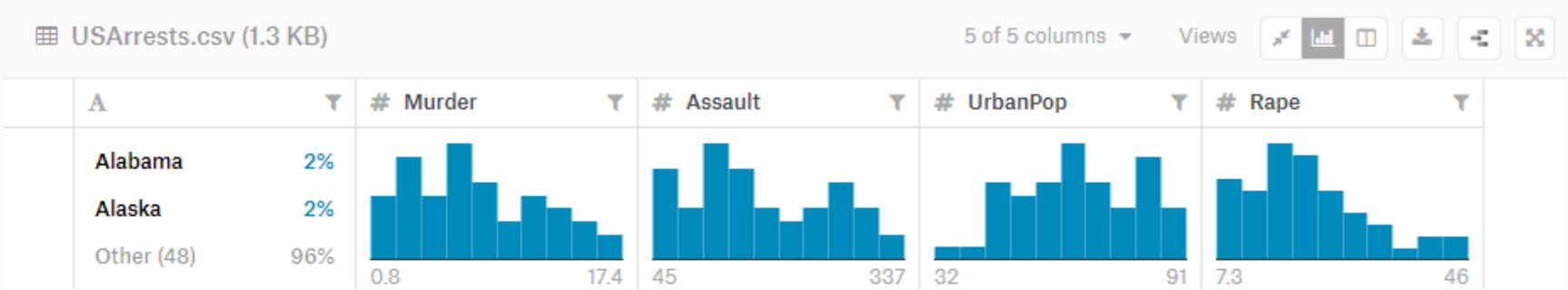
The first principal component is $c_1 = X\vartheta_1$ so that for aeach observation $1 \leqslant i \leqslant n$, $c_1(i) = X_i^T \vartheta_1$.

$\rightharpoonup$ ϑ_1 is also known as the first loading vector.

$\rightharpoonup$ $(c_1(i) = X_i^T \vartheta_1)_{1 \leqslant i \leqslant n}$ are the scores of the first principal component.

Application to the USArrests dataset[2]

$\rightharpoonup$ Arrests per 100.000 residents for assault, murder and rape for the 50 US states in 1973. Percent of the population living in urban areas are also given.



[2] https://www.kaggle.com/deepakg/usarrests

Fit the observations cloud - graphical representation

$\rightarrow$ Principal component loading vectors ϑ_1 and ϑ_2.

	PC1	PC2
Murder	0.5358995	−0.4181809
Assault	0.5831836	−0.1879856
UrbanPop	0.2781909	0.8728062
Rape	0.5434321	0.1673186

In this setting $n = 50$ (number states) and $d = 4$ (number of variables - Murder, Assault, Rape and UrbanPop).

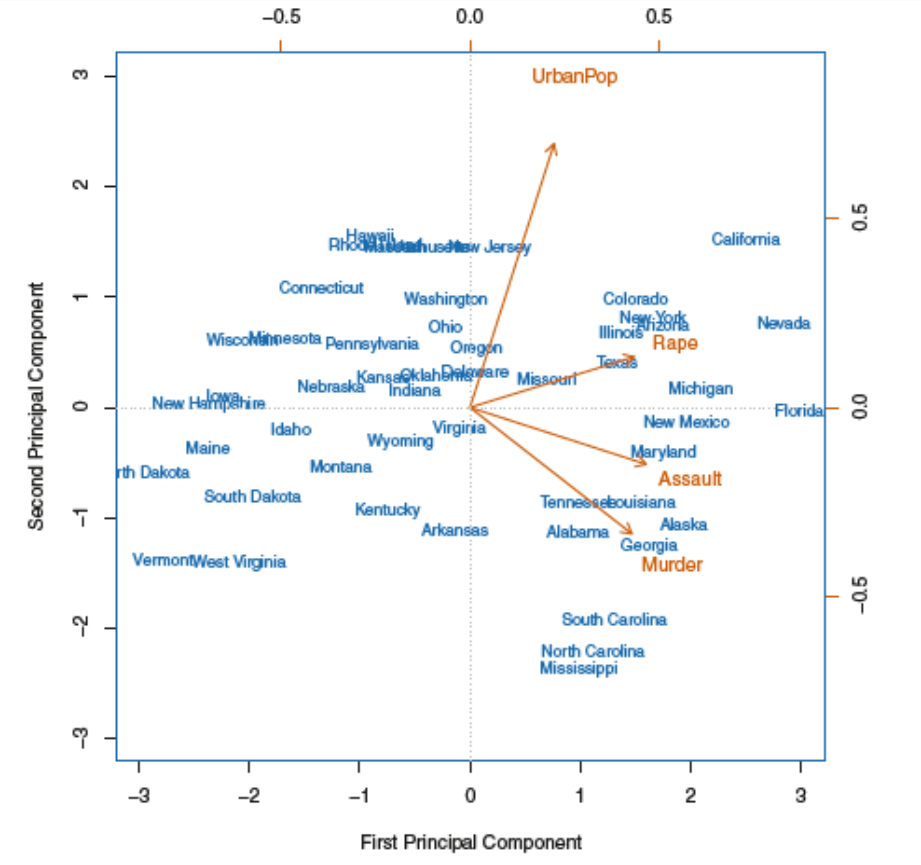
$$\vartheta_1 = (0.53, 0.58, 0.28, 0.54) ,$$
$$\vartheta_2 = (-0.42, -0.19, 0.87, 0.17) .$$

Fit the observations cloud - graphical representation

$\rightarrow$ **Biplot**[3].

States names in blue are the the scores for the two first principal components (coordinates of the projected observations).

$$\pi_{V_2}(X_i) = (X_i^T \vartheta_1)\vartheta_1 + (X_i^T \vartheta_2)\vartheta_2 \ .$$



[3] **An introduction to statistical learning**, James, G. and Witten, D. and Hastie, T. and Tibshirani, R.

Fit the observations cloud - graphical representation

Let W_p be the vector subspace of $\mathbb{R}^n$ generated by $\{c_1, \ldots, c_p\}$. Since $(c_j)_{1 \leqslant j \leqslant d}$ form a orthogonal basis of W_p, for all $1 \leqslant j \leqslant d$,

$$\pi_{W_p}(X_{.,j}) = \sum_{\ell=1}^{p} \frac{\langle c_\ell, X_{.,j} \rangle}{\|c_\ell\|^2} c_\ell \ .$$

As for all $1 \leqslant \ell \leqslant d$, $\|c_\ell\|^2 = n\lambda_\ell$ and

$$\langle c_\ell, X_{.,j} \rangle = \langle X\vartheta_\ell, X_{.,j} \rangle = X_{.,j}^T X \vartheta_\ell = (X^T X \vartheta_\ell)_j = (n\Sigma_n \vartheta_\ell)_j = n\lambda_\ell \vartheta_\ell(j) \ .$$

This yields, for all $1 \leqslant j \leqslant d$, $\pi_{W_p}(X_{.,j}) = \sum_{\ell=1}^{p} \vartheta_\ell(j) c_\ell$.

$\rightharpoonup$ The coordinates of each variable j in the space generated by the principal components are the loadings values $(\vartheta_\ell(j))_{1 \leqslant \ell \leqslant p}$.

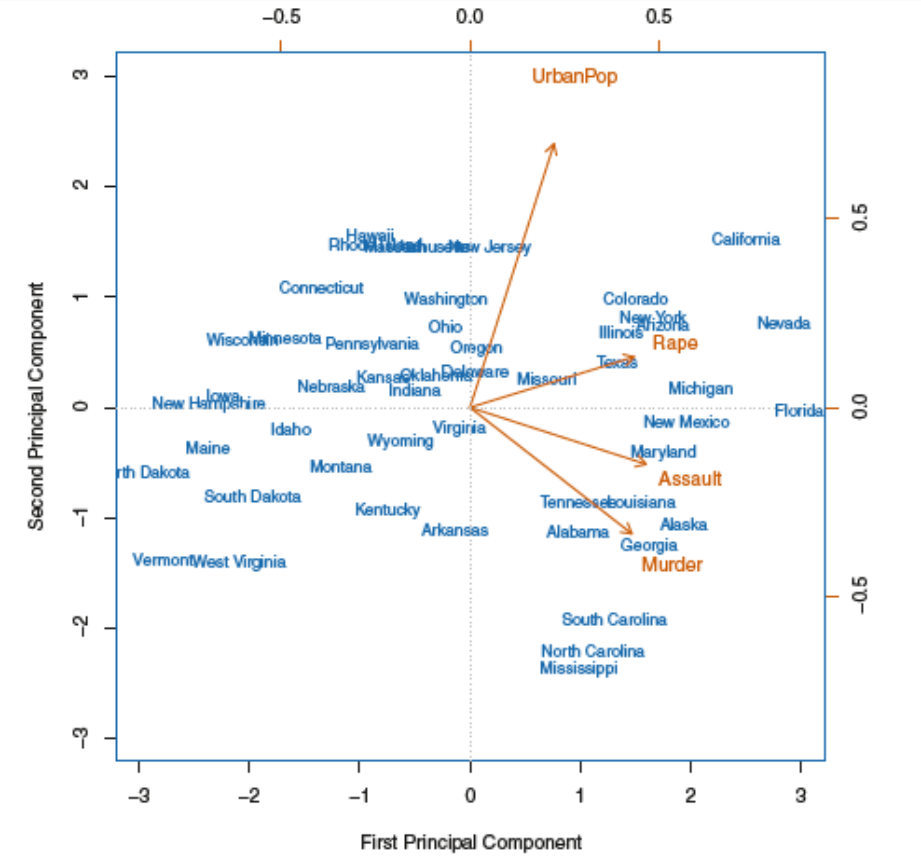
$\rightharpoonup$ The variables can be plotted as points in the component space using their loadings as coordinates.

Fit the observations cloud - graphical representation

$\rightharpoonup$ **Biplot**[4].

Orange arrows are the loadings for the two first principal components.

$$\pi_{W_2}(X_{\cdot,j}) = \sum_{\ell=1}^{2} \vartheta_\ell(j) c_2 \ .$$



[4] **An introduction to statistical learning**, James, G. and Witten, D. and Hastie, T. and Tibshirani, R.

A classical example - wine data

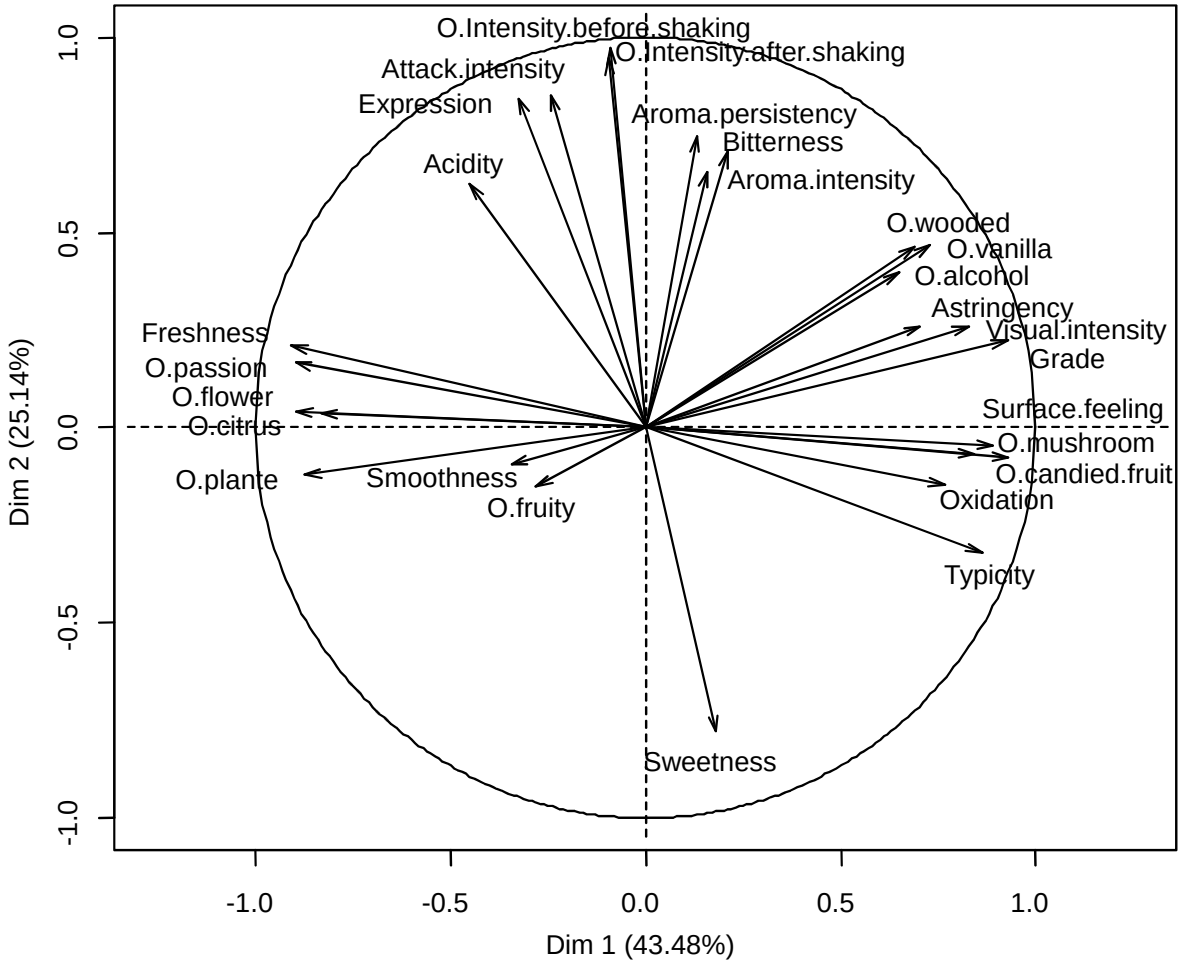
→ **10 observations (rows)**: white wines from Val de Loire.

→ **30 variables (columns)**:

- 27 continuous variables: sensory descriptors.
- 2 continuous variables: odour and overall preferences.
- 1 categorical variable: label of the wines (Vouvray - Sauvignon).

	O.fruity	O.passion	O.citrus	...	Sweetness	Acidity	Bitterness	Astringency	Aroma.intensity	Aroma.persistency	Visual.intensity	Odor.preference	Overall.preference	Label
S Michaud	4,3	2,4	5,7	...	3,5	5,9	4,1	1,4	7,1	6,7	5,0	6,0	5,0	Sauvignon
S Renaudie	4,4	3,1	5,3	...	3,3	6,8	3,8	2,3	7,2	6,6	3,4	5,4	5,5	Sauvignon
S Trotignon	5,1	4,0	5,3	...	3,0	6,1	4,1	2,4	6,1	6,1	3,0	5,0	5,5	Sauvignon
S Buisse Domaine	4,3	2,4	3,6	...	3,9	5,6	2,5	3,0	4,9	5,1	4,1	5,3	4,6	Sauvignon
S Buisse Cristal	5,6	3,1	3,5	...	3,4	6,6	5,0	3,1	6,1	5,1	3,6	6,1	5,0	Sauvignon
V Aub Silex	3,9	0,7	3,3	...	7,9	4,4	3,0	2,4	5,9	5,6	4,0	5,0	5,5	Vouvray
V Aub Marigny	2,1	0,7	1,0	...	3,5	6,4	5,0	4,0	6,3	6,7	6,0	5,1	4,1	Vouvray
V Font Domaine	5,1	0,5	2,5	...	3,0	5,7	4,0	2,5	6,7	6,3	6,4	4,4	5,1	Vouvray
V Font Brûlés	5,1	0,8	3,8	...	3,9	5,4	4,0	3,1	7,0	6,1	7,4	4,4	6,4	Vouvray
V Font Coteaux	4,1	0,9	2,7	...	3,8	5,1	4,3	4,3	7,3	6,6	6,3	6,0	5,7	Vouvray

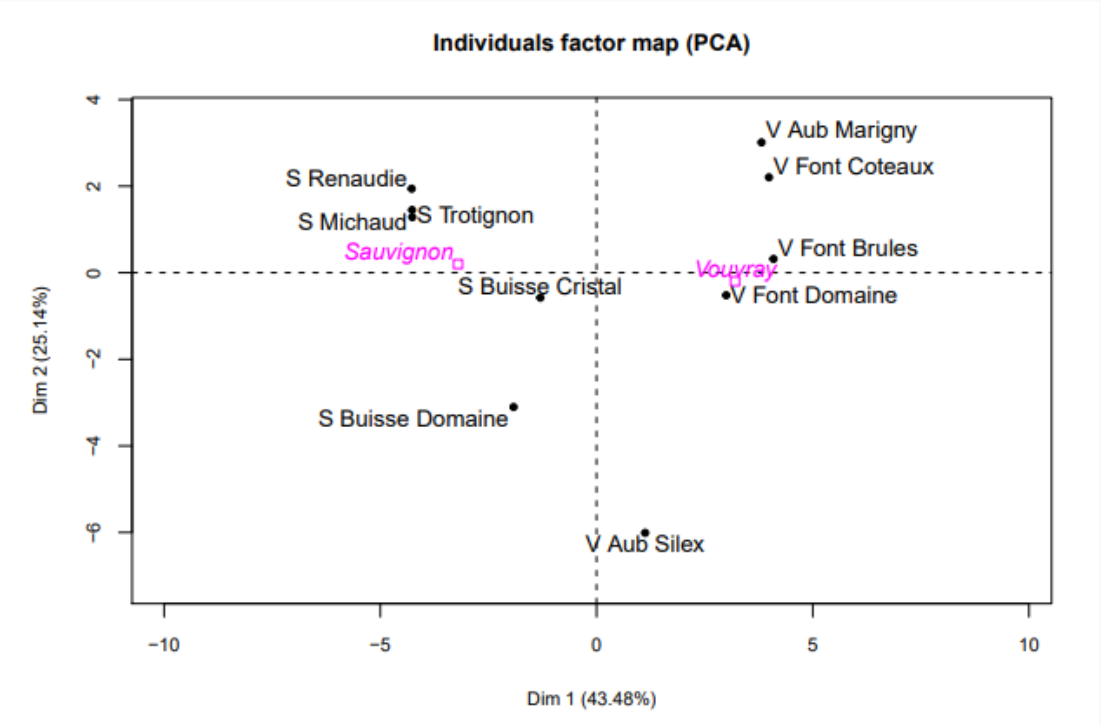
Interpretation of the observations graph with the variables



Supplementary information

- Supplementary variables do not impact PCA results but may provide insights on their interpretability.
- In R, supplementary continuous variables added by quanti.sup and discrete variables added with quali.sup.
- In the case of discrete variables, the barycentre of the observations which take each category is displayed in the observations plot.

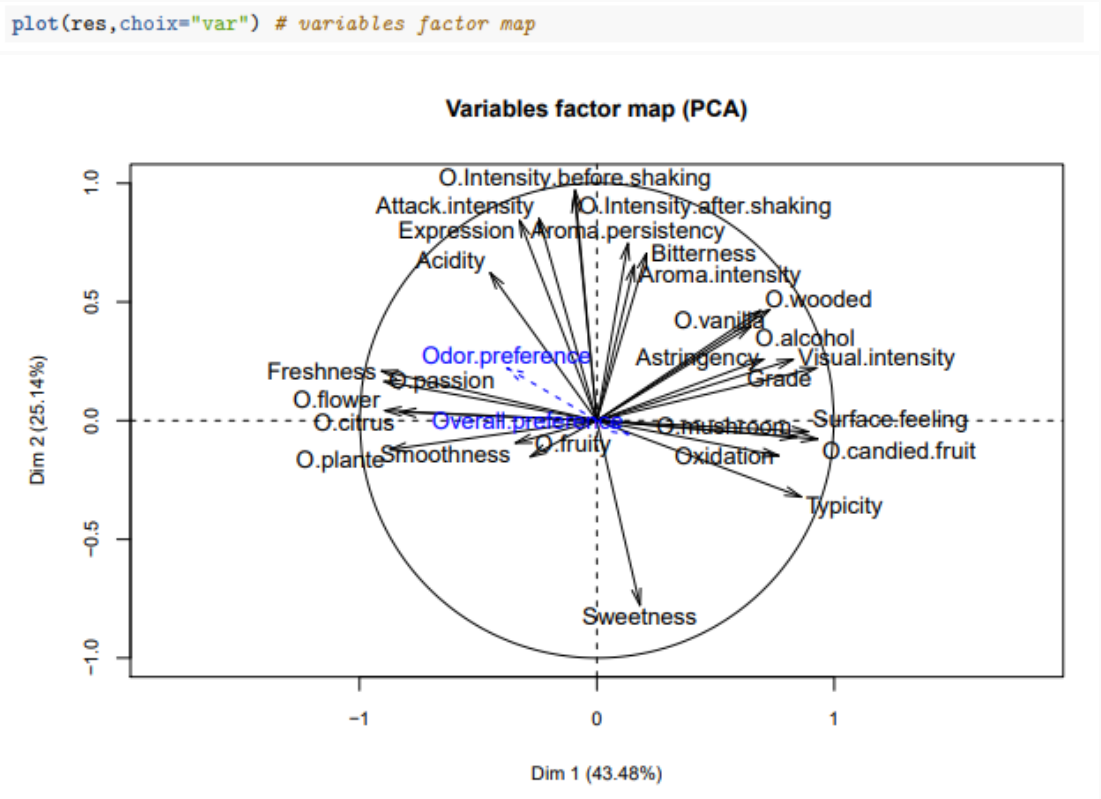
```
res <- PCA(wine,quali.sup=1,quanti.sup=29:30,graph=FALSE)
par(cex=0.8,cex.main=1,cex.lab=0.8,cex.axis=0.8) # change graphical parameters (font size)
plot(res,choix="ind") # individuals factor map
```



Supplementary information

- In the case of continuous variables, additional variables are projected in the factor map.

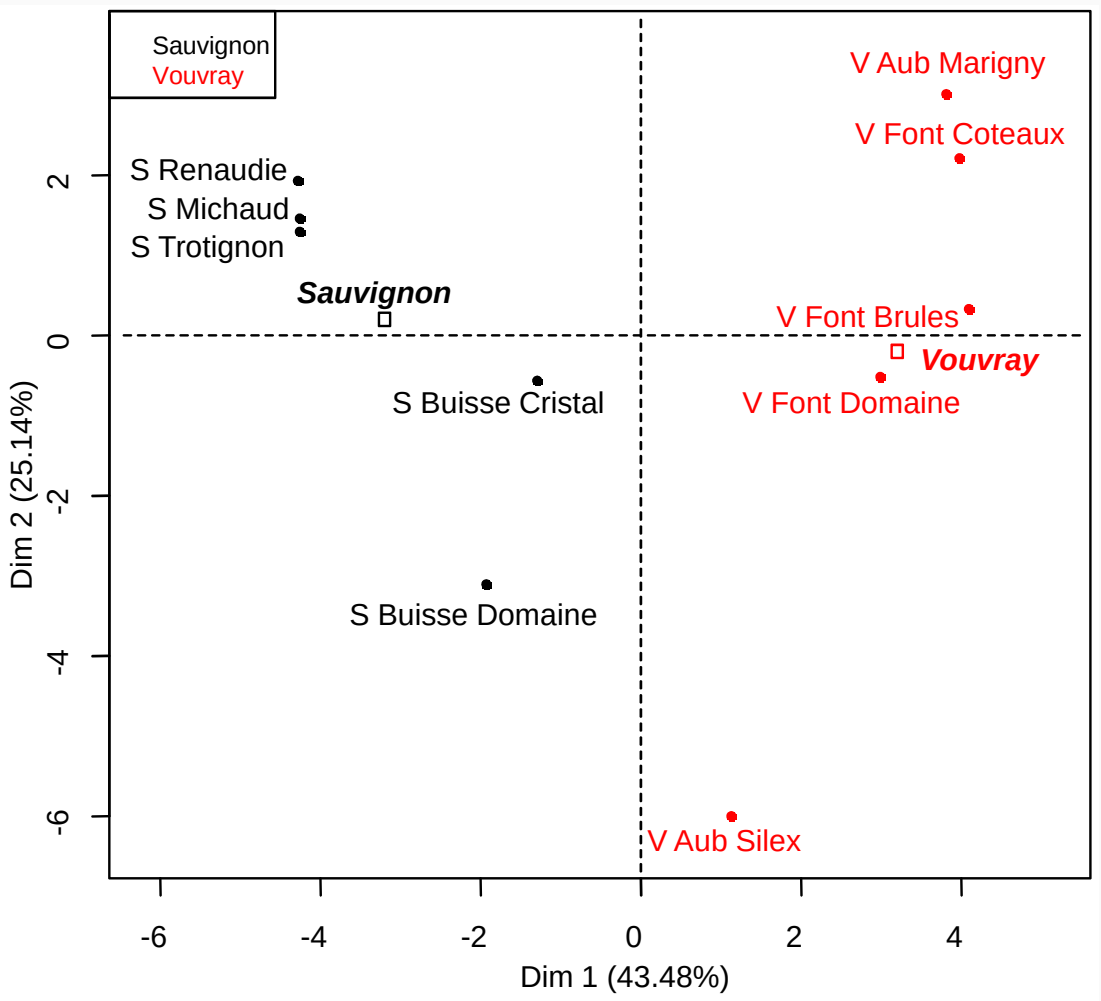
```
plot(res,choix="var") # variables factor map
```



Supplementary information

- Drawing with a color according to the label.

```
plot(res,habillage="Label")
```

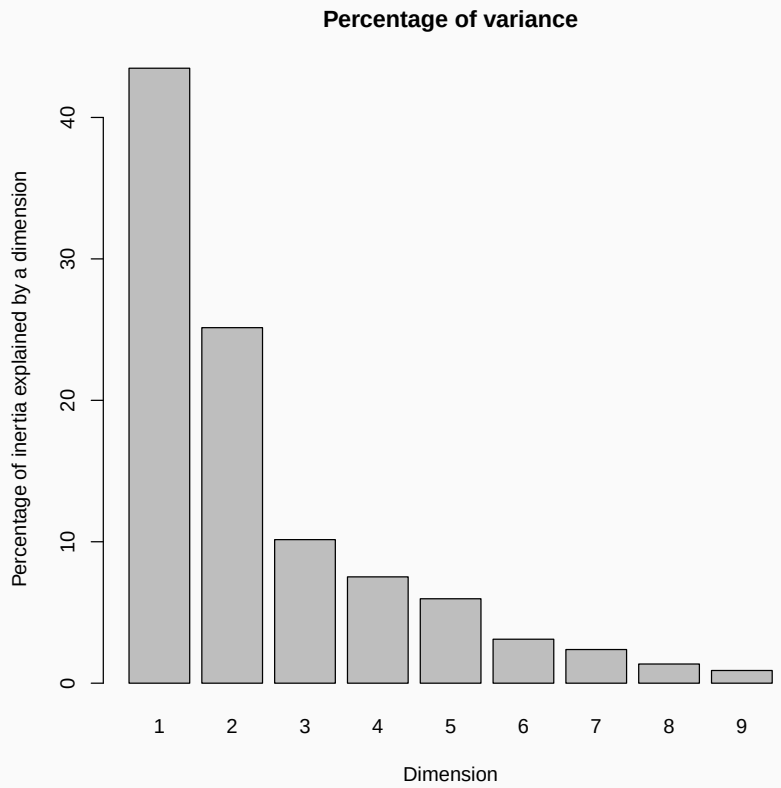


Choosing the number of components

→ The problem is to figure out which amount of information should be extracted from the original dataset. **How many components need to be considered?**

→ Open problem for which a few practical guidelines exist.

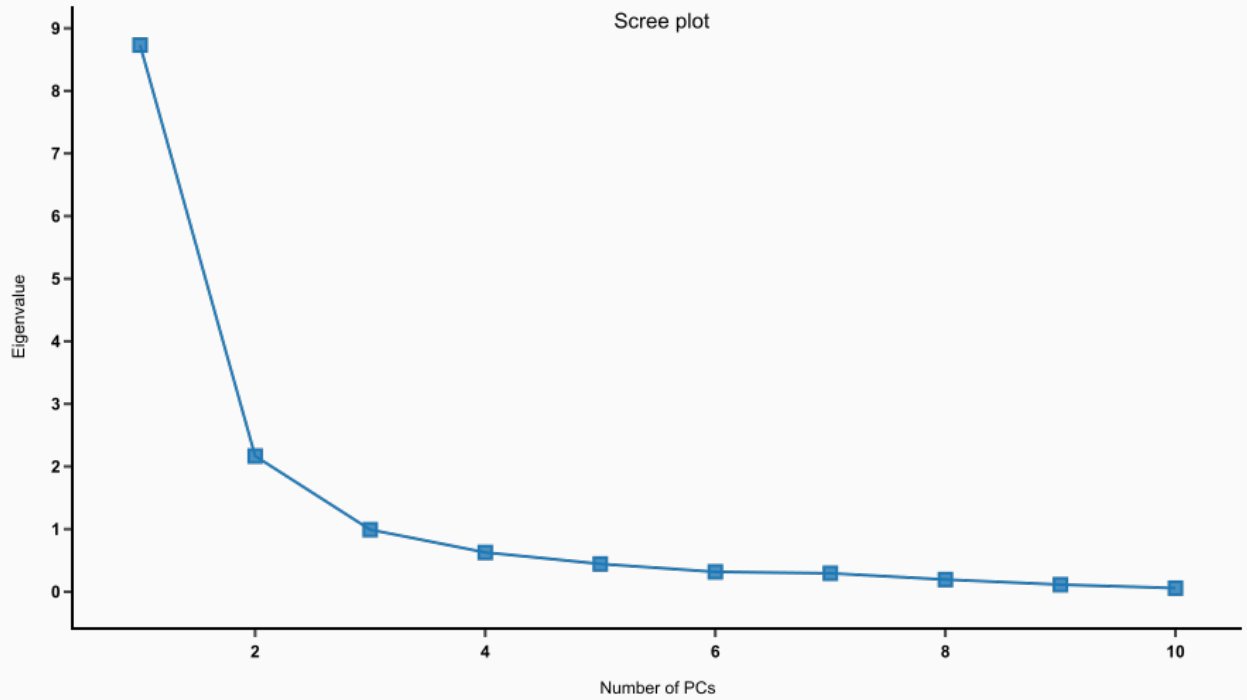
→ **Bar plot for illustrative purposes**.



Choosing the number of components

$\rightarrow$ A scree plot may be used to plot the eigenvalues according to their size.

$\rightarrow$ Detect the elbow, i.e. the eigenvalue where the slope in this graph goes from steep to flat: keep only the components which are before the elbow.



Choosing the number of components - going further

$\rightarrow$ estim_ncpPCA may be used to estimate the number of components using cross validation.

$\rightarrow$ In the default case, leave-one-out cross validation, each value X_{ik} is iteratively removed from the dataset. This value is then estimated by $\widehat{X}_{ik}^{r}$ using a PCA with r components obtained from the dataset that excludes this cell (with imputePCA).

$\rightarrow$ The estimated number of components is then

$$r \mapsto \frac{1}{nd} \sum_{i=1}^{n} \sum_{k=1}^{d} \left(X_{ik} - \widehat{X}_{ik}^{r} \right)^2 .$$

$\rightarrow$ K-fold cross validation is also possible with a given proportion of data to be removed.

Contribution

$\rightarrow$ Contribution of an observation to the dimension k (percentage of variability).

$$\text{Ctr}_k(X_i) = \frac{c_k(i)^2/n}{\sum_{i=1}^{n} c_k(i)^2/n} = \frac{c_k(i)^2}{n\lambda_k} \ .$$

```
round(res.pca$ind$contrib,2)
                 Dim.1 Dim.2
S Michaud        15.49  3.10
S Renaudie       15.56  5.56
S Trotignon      15.46  2.43
```

$\rightarrow$ Observations with large coordinate contribute the most.

Outline

Kernel PCA

$\rightharpoonup$ PCA only allows dimensionality reduction based on principal components which are linear combinations of the variables.

$\rightharpoonup$ When the data has more complex structures which cannot be well represented in a linear subspace, standard PCA fails.

$\rightharpoonup$ kernel PCA[5] allows us to generalize standard PCA to nonlinear dimensionality reduction (Scholkopf et al., 1999)[6].

A function $k : \mathbb{R}^d \times \mathbb{R}^d :\rightarrow \mathbb{R}$ is said to be a **positive definite kernel** if and only if it is symmetric and if for all $n \geqslant 1$, $(x_1, \ldots, x_n) \in (\mathbb{R}^d)^n$ and all $(a_1, \ldots, a_n) \in \mathbb{R}^n$,

$$\sum_{1 \leqslant i,j \leqslant n} a_i a_j k(x_i, x_j) \geqslant 0 \ .$$

[5] **Kernel principal component analysis**, Scholkopf, B. et al., ***International Conference on Artificial Neural Networks***, p. 583-588, 1997

[6] **Advances in kernel methods: support vector learning**, Scholkopf et al., ***MIT Press***

Kernel PCA

$\rightharpoonup$ Each observation X_i is mapped into a m-dimensional space using a function $\varphi : \mathbb{R}^d \to \mathbb{R}^m$.

$\rightharpoonup$ The new features are assumed to be centered:

$$\sum_{i=1}^{n} \varphi(X_i) = 0 \ .$$

$\rightharpoonup$ The covariance matrix of the new features is

$$\Sigma_n = \frac{1}{n} \sum_{i=1}^{n} \varphi(X_i)(\varphi(X_i))^T \ .$$

Let ϑ_k be the eigenvectors of Σ_n associated with the eigenvalues $\lambda_1 \geqslant \lambda_2 \geqslant \ldots \geqslant \lambda_m$:

$$\Sigma_n \vartheta_k = \lambda_k \vartheta_k \ .$$

Kernel PCA

$\rightharpoonup$ For each eigenvector ϑ_k, there exist $(\alpha_{ki})_{1 \leqslant i \leqslant n}$ such that

$$\vartheta_k = \sum_{i=1}^n \alpha_{ki} \varphi(X_i) \ .$$

This yields

$$\frac{1}{n} \sum_{i=1}^n \varphi(X_i)(\varphi(X_i))^T \left(\sum_{j=1}^n \alpha_{kj} \varphi(X_j) \right) = \lambda_k \sum_{i=1}^n \alpha_{ki} \varphi(X_i) \ .$$

$\rightharpoonup$ Defining the kernel $\kappa(X_i, X_j) = (\varphi(X_i))^T \varphi(X_j))$, for all $1 \leqslant p \leqslant n$,

$$\frac{1}{n} \sum_{i=1}^n \sum_{j=1}^n \alpha_{kj} \kappa(X_p, X_i) \kappa(X_i, X_j) = \lambda_k \sum_{i=1}^n \alpha_{ki} \kappa(X_p, X_i) \ .$$

$\rightharpoonup$ Let K be the matrix defined as $\mathrm{K}_{ij} = \kappa(X_i, X_j) = (\varphi(X_i))^T \varphi(X_j))$ and $\bar{\alpha}_k$ be the column vector with entries $(\alpha_{ki})_{1 \leqslant i \leqslant n}$, so that,

$$\mathrm{K}^2 \bar{\alpha}_k = n \lambda_k K \bar{\alpha}_k \ .$$

Kernel PCA

$\rightharpoonup$ Only requires to compute $\mathrm{K}_{ij} = \kappa(X_i, X_j) = (\varphi(X_i))^T \varphi(X_j))$, to find the vector $\bar{\alpha}_k$ associated with the nonzero eigenvalue λ_k such that

$$\mathrm{K}\bar{\alpha}_k = n\lambda_k \bar{\alpha}_k \ .$$

$\rightharpoonup$ Principal components given by

$$c_k(i) = \varphi(X_i)^T \vartheta_k = \sum_{j=1}^{n} \alpha_{jk} K(X_i, X_j) \ .$$

No need to evaluate $\varphi(X_i)$, only the kernel matrix is necessary.

$\rightharpoonup$ If the new features are not centered, the matrix K may be replaced by

$$\widetilde{\mathrm{K}} = 1_n \mathrm{K} - \mathrm{K} 1_n + 1_n \mathrm{K} 1_n \ ,$$

where 1_n is the $n \times n$ matrix with all elements equal to $1/n$.

Kernel PCA in practice

$\rightharpoonup$ **Compute the kernel matrix** K, for instance based on the Gaussian radial basis function:

$$\mathrm{K}(X_i, X_j) = \exp\left(-\gamma \|X_i - X_j\|^2\right) \ .$$

$\rightharpoonup$ If the new features are not centered, the matrix K may be replaced by

$$\widetilde{\mathrm{K}} = 1_n \mathrm{K} - \mathrm{K} 1_n + 1_n \mathrm{K} 1_n \ .$$

$\rightharpoonup$ **Compute the vector** $\bar{\alpha}_k$ by solving

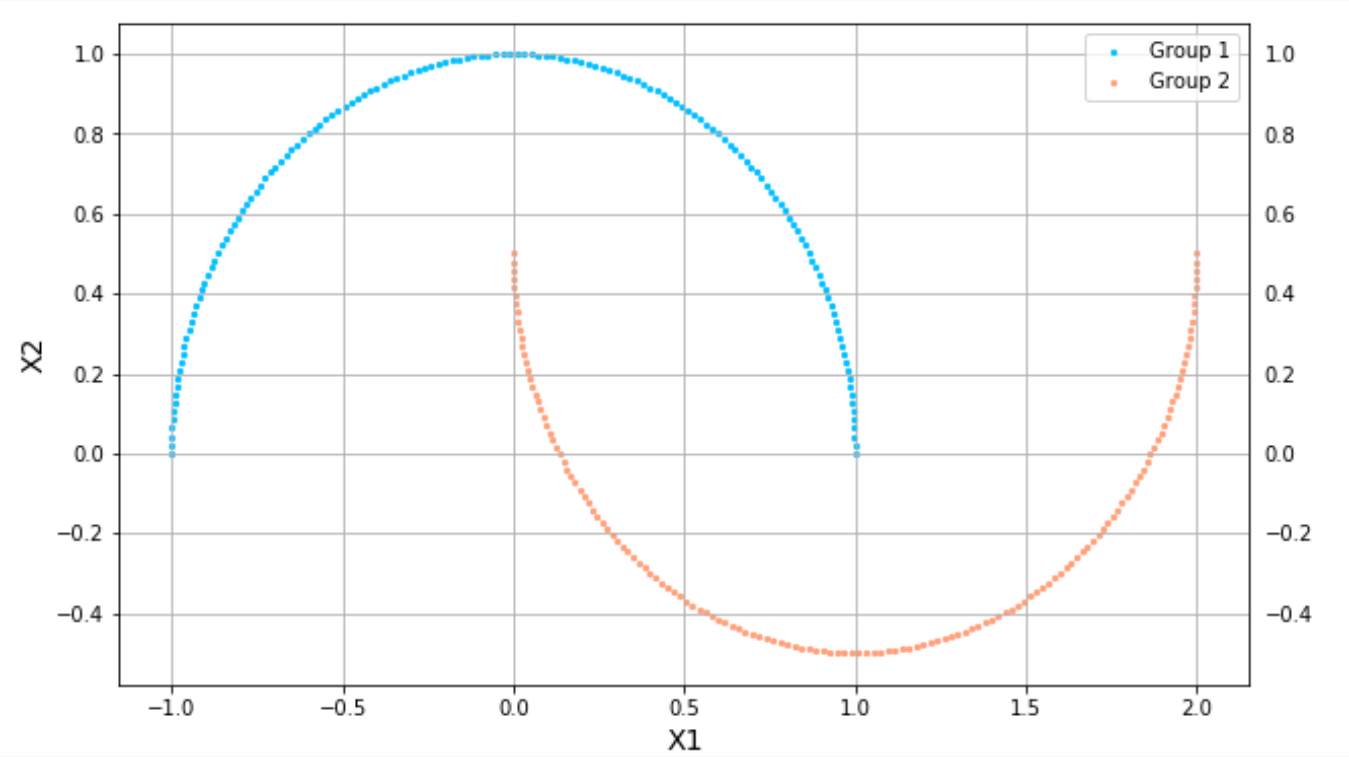
$$\mathrm{K}\bar{\alpha}_k = n\lambda_k \bar{\alpha}_k \ .$$

$\rightharpoonup$ **Compute the principal components**:

$$c_k(i) = \varphi(X_i)^T \vartheta_k = \sum_{j=1}^{n} \alpha_{jk} K(X_i, X_j) \ .$$

Kernel PCA in practice

```python
import matplotlib.pyplot as plt
from sklearn.datasets import make_moons

X, y = make_moons(n_samples = 300)

plt.figure(figsize=(10,6))
plt.scatter(X[y==0, 0], X[y==0, 1], color ='deepskyblue', s = 5, label = 'Group 1')
plt.scatter(X[y==1, 0], X[y==1, 1], color ='lightsalmon', s = 5, label = 'Group 2')

plt.ylabel('X2', fontsize=14)
plt.xlabel('X1', fontsize=14)
plt.tick_params(labelright=True)
plt.grid('on')
plt.legend()

plt.show()
```

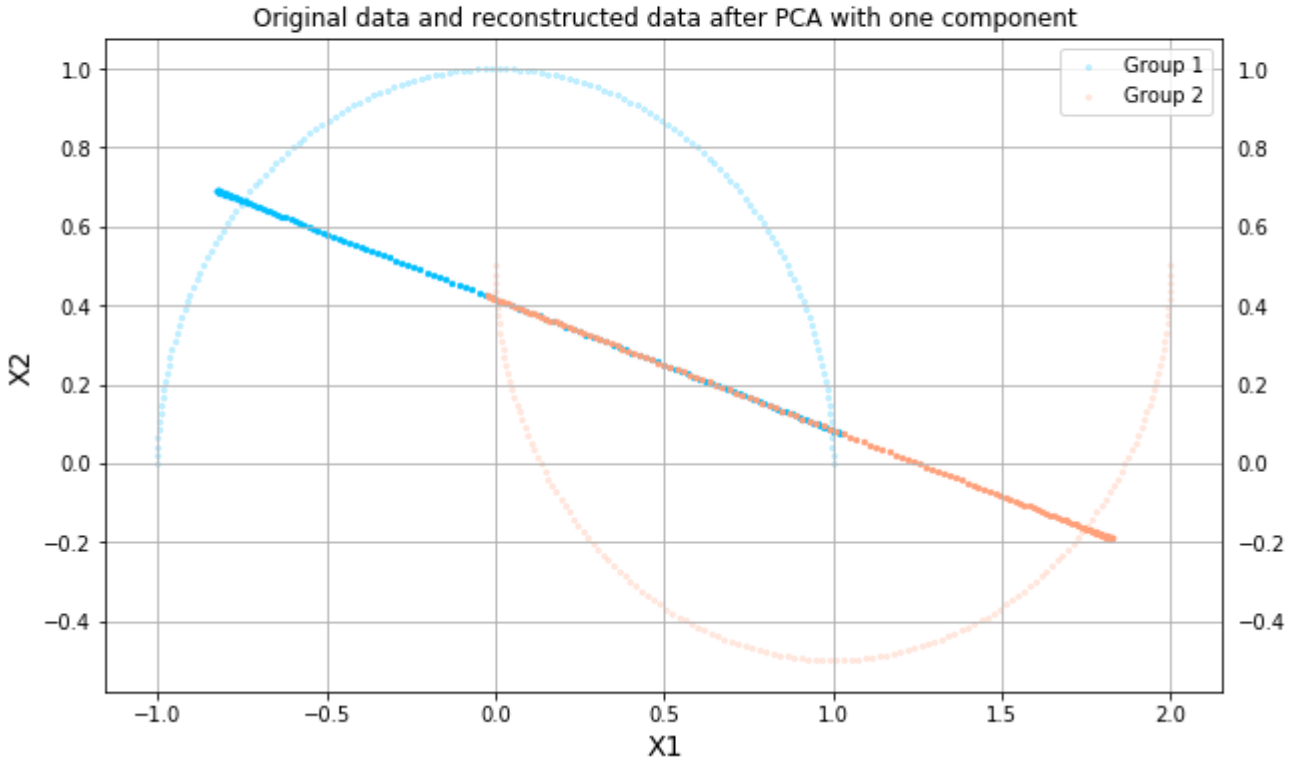


Kernel PCA in practice

```
pca = PCA(n_components=1)
pca.fit(X)
print('The principal component is:')
print(pca.components_)
print('The explained variance is %g'%pca.explained_variance_)
print('The associated singular value is %g'%pca.singular_values_)

X_pca = pca.transform(X)

# transform the reduced data set in the original space
# X_inverse[i,:] contains the coordinates of the projection of Xi in the original space
X_inverse = pca.inverse_transform(X_pca)
plt.figure(figsize=(10,6))
plt.scatter(X[y==0, 0], X[y==0, 1], color ='deepskyblue', s = 5, label = 'Group 1', alpha = 0.2)
plt.scatter(X[y==1, 0], X[y==1, 1], color ='lightsalmon', s = 5, label = 'Group 2', alpha = 0.2)

plt.scatter(X_inverse[y==0, 0], X_inverse[y==0, 1], color ='deepskyblue', s = 5)
plt.scatter(X_inverse[y==1, 0], X_inverse[y==1, 1], color ='lightsalmon', s = 5)

plt.tick_params(labelright=True)
plt.grid('on')
plt.legend()
plt.ylabel('X2', fontsize=14)
plt.xlabel('X1', fontsize=14)
plt.title('Original data and reconstructed data after PCA with one component')
plt.show()
```

Kernel PCA in practice

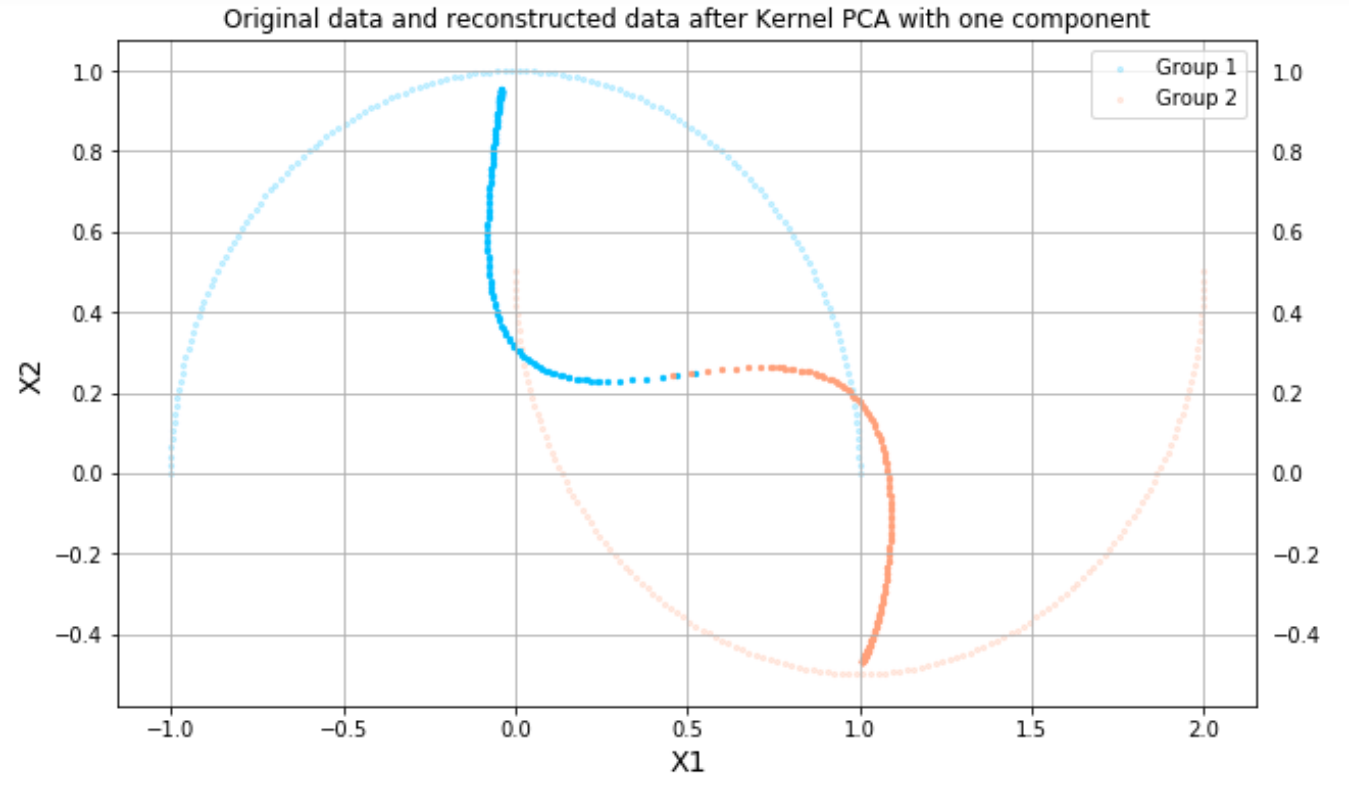
```
The principal component is:
[[-0.94876358  0.31598683]]
The explained variance is 0.819737
The associated singular value is 15.6557
```



Kernel PCA in practice

```
from sklearn.decomposition import KernelPCA
kpca = KernelPCA(n_components = 1, kernel='rbf', fit_inverse_transform = True, gamma = 10)

kpca.fit(X)
X_pca = kpca.transform(X)

# transform the reduced data set in the original space
# X_inverse[i,:] contains the coordinates of the projection of Xi in the original space
X_inverse = kpca.inverse_transform(X_pca)
plt.figure(figsize=(10,6))
plt.scatter(X[y==0, 0], X[y==0, 1], color ='deepskyblue', s = 5, label = 'Group 1', alpha = 0.2)
plt.scatter(X[y==1, 0], X[y==1, 1], color ='lightsalmon', s = 5, label = 'Group 2', alpha = 0.2)

plt.scatter(X_inverse[y==0, 0], X_inverse[y==0, 1], color ='deepskyblue', s = 5)
plt.scatter(X_inverse[y==1, 0], X_inverse[y==1, 1], color ='lightsalmon', s = 5)

plt.tick_params(labelright=True)
plt.grid('on')
plt.legend()
plt.ylabel('X2', fontsize=14)
plt.xlabel('X1', fontsize=14)
plt.title('Original data and reconstructed data after Kernel PCA with one component')
plt.show()
```

Kernel PCA in practice



Outline

Linear independent component analysis

$\rightharpoonup$ **Linear independent component analysis** of a random vector X amounts to estimating the following model

$$X = AS + \varepsilon \,,$$

where $A \in \mathbb{R}^{d \times n}$ is an unknown mixing matrix, $S \in \mathbb{R}^n$ is a random vector with independent entries and ε is an independent noise.

$\rightharpoonup$ Noise free formulation of this problem, i.e. $X = AS$, was proposed in the signal processing litterature[7]

[7] **Blind separation of sources, part I: an adaptive algorithm based on neuromimetic architecture.** Jutten, C. et al., ***Signal Processing, 2(4), 1-10***, 1991

Linear independent component analysis - identifiability

The **noise free linear independent component analysis** is identifiable under the following (sufficient conditions).

$\rightharpoonup$ The components S_i are not Gaussian random variables (with the possible exception of one component).

$\rightharpoonup$ $d \geqslant n$, i.e. the number of oservations is greater than the number of independent components.

$\rightharpoonup$ The matrix A has full rank.

Linear independent component analysis - identifiability

Assume that $d \geqslant n$, and that the components S_i are not Gaussian random variables (with the possible exception of one component) in the model

$$X = AS \ .$$

Assume also that the probability density p_S of S belogns to a subset E of probability densities on $\mathbb{R}^n$ which admit (at least) a moment up to order 2.

Then, if $X = \tilde{A}\tilde{S}$, where the probability density $p_{\tilde{S}}$ of $\tilde{S}$ lies in E and if the contrast function Ψ is discriminant over E, $\Psi(p_{\tilde{S}}) = \Psi(p_S)$ **if and only if** $\tilde{A} = A\Lambda P$ **where** Λ **is an invertible diagonal matrix and** P **is a permutation matrix**.

Linear independent component analysis - Fast ICA

Preprocessing proceeds in two steps.

$\rightharpoonup$ **Centering**

For all $1 \leqslant i \leqslant d$,

$$\tilde{X}_{ij} = X_{ij} - \frac{1}{n} \sum_{\ell=1}^{n} X_{i\ell} \, .$$

$\rightharpoonup$ **Whitening**

A linear transformation is used to transform data into a new dataset with unit covariance matrix. A standard approach to do so is to perform an eigenvalue decomposition of the empirical covariance matrix Σ of the centered data $\tilde{X} = (\tilde{x}_{ij})_{1 \leqslant i \leqslant d, 1 \leqslant j \leqslant n}$:

$$\Sigma = P \Delta P^{-1} \, ,$$

where Δ is the diagonal matrix containing all the eigenvalues of Σ. The unit covariance covariance data is the given by:

$$\widehat{X} = \Delta^{-1/2} P^T \tilde{X} \, .$$

Linear independent component analysis - Fast ICA

The notion of independence maximized in Fast ICA is a measure of non-Gaussianity.

In the case of Fast ICA, non-Gaussianity is computed using a nononlinear function g applied to the projected observations based on the current weight. The common choice for g is $g : u \mapsto \tanh(u)$.

After the first $c-1$ independent components/weights have been estimated w_1, ..., w_{c-1} the algorithm is run to produce w_c:

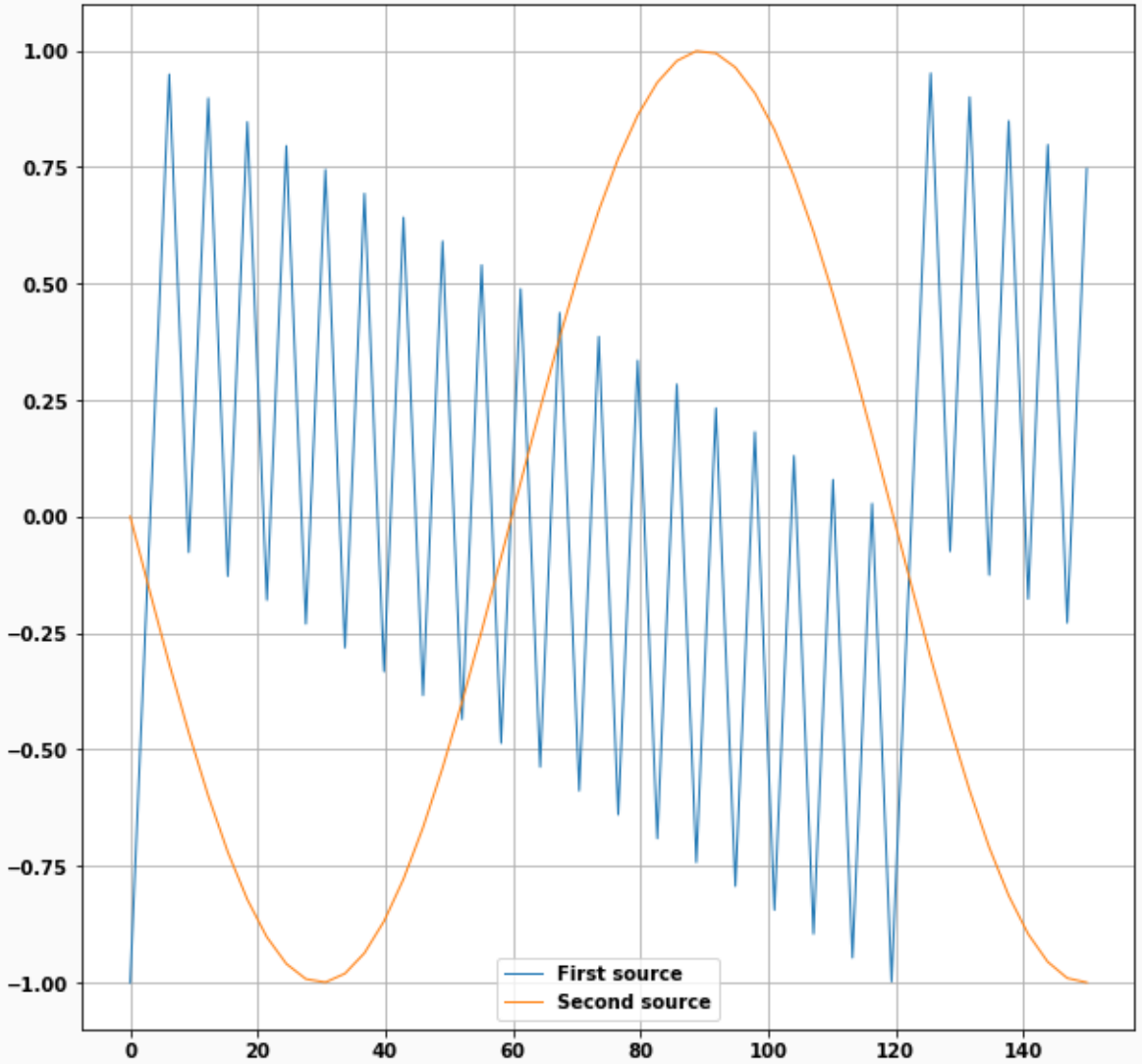
$$w_c = \frac{1}{n}\sum_{i=1}^{n} X_i g(w_c^T X_i) - \frac{1}{n}\sum_{i=1}^{n} g'(w_c^T X_i) w \,.$$

Then, an additive is introduced to subtract the projections onto w_1, ..., w_{c-1}:

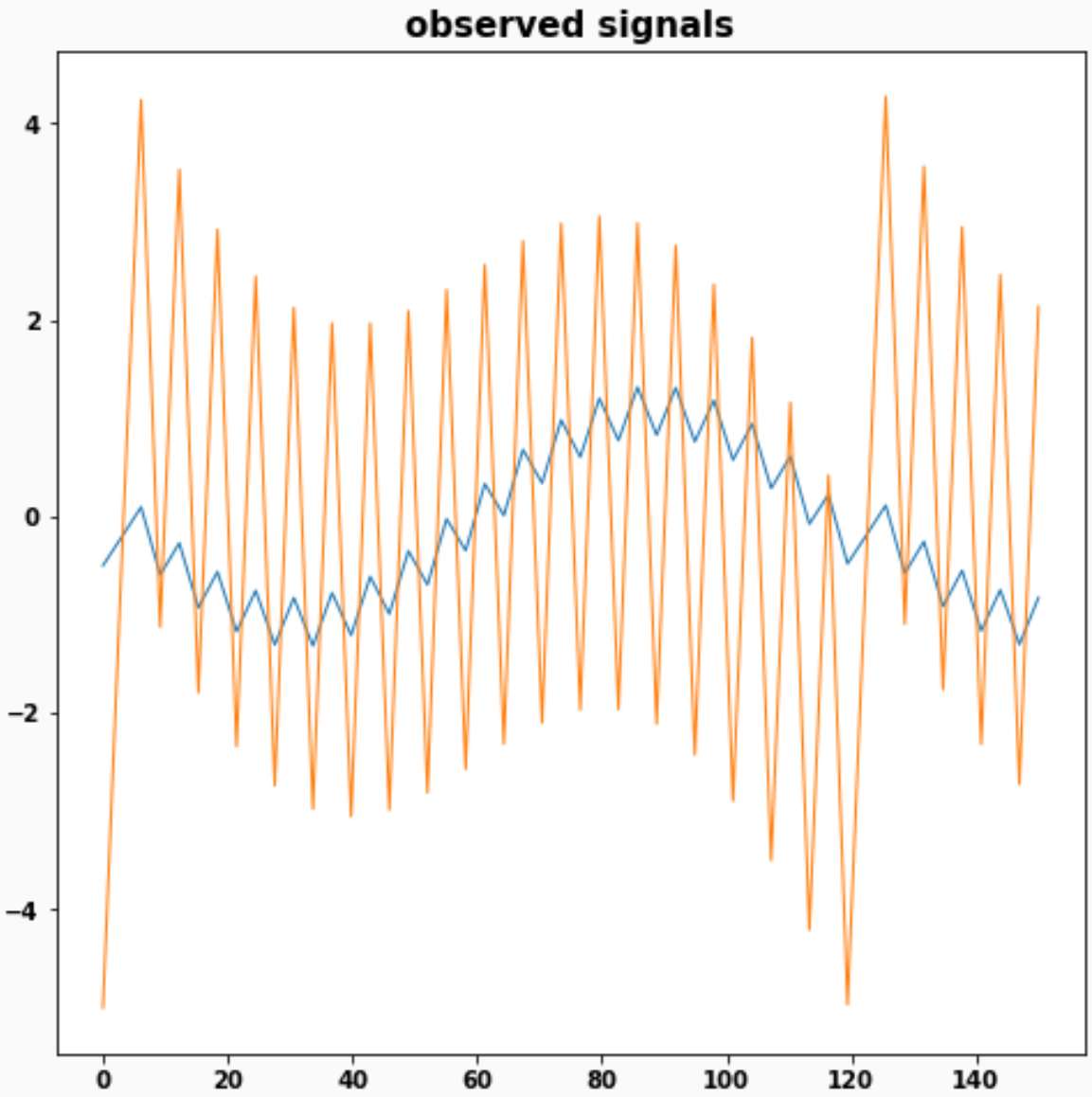
$$w_c = w_c - \sum_{j=1}^{c-1} w_c^T w_j w_j \,.$$

These steps are repeated until convergence.

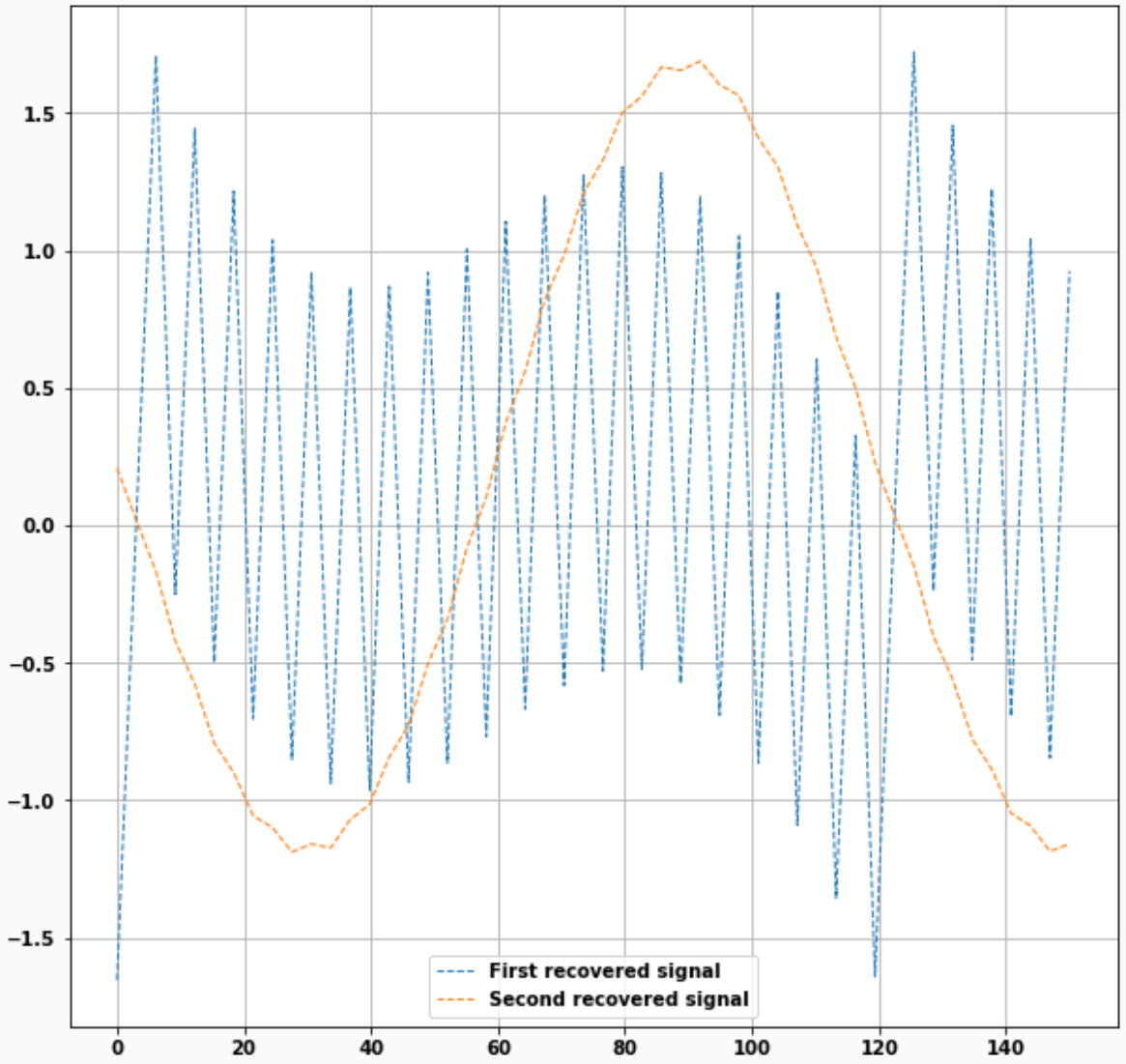
Linear independent component analysis - Fast ICA



Linear independent component analysis - Fast ICA



Linear independent component analysis - Fast ICA



Outline

Correspondence analysis

$\rightharpoonup$ Similar to principal component analysis, but applied to categorical data.

$\rightharpoonup$ Let X be a $n \times d$ contingency matrix, i.e. a matrix of nonnegative integers.

$\rightharpoonup$ The correspondence matrix is then

$$P = (1_n^T X 1_d)^{-1} X \ ,$$

where $1_n^T X 1_d$ is the total number of counts.

$\rightharpoonup$ Principal components obtained by the singular value decomposition of the standardized residuals:

$$\Sigma = \mathrm{diag}(P 1_d)^{-1/2} \left(P - P 1_d 1_n^T P\right) \mathrm{diag}(P^T 1_n)^{-1/2} \ .$$

Other dimensionality reduction techniques

- Nonlinear independent component analysis (ICA)[8].
- Locally linear embedding (LLE)[9].
- Uniform manifold approximation (UMA)[10].
- Random forests[11].

---

[8] **Independent component analysis: algorithms and applications**, Hyvarinen, A. et al., ***Neural networks, 13(4-5), 411-430***, 2000

[9] **Nonlinear dimensionality reduction by locally linear embedding**, Row, S.T. et al., ***Science, 290(5500), 2323-2326***, 2000

[10] **Umap: Uniform manifold approximation and projection for dimension reduction**, McInnes, L. et al., ***arXiv:1802.03426***, 2018

[11] **Random forests**, Breiman, L. et al., ***Machine learning, 45(1), 5-32***, 2001