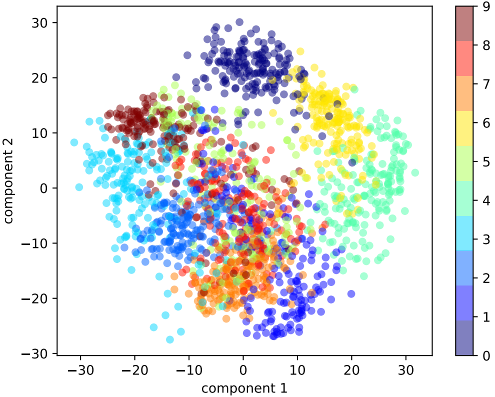
Machine learning II, unsupervised learning and agents: dimensionality reduction



Overview

Motivation

Local averaging

Linear dimensionality reduction (Principal component analysis)

Nonlinear dimensionality reduction (manifold learning)

- Multidimensional scaling (MDS)
- t-SNE
- Others famous methods
- Comments

Why is the dimension so important ?

Recall 2 important constants in machine leaning :

- n : number of samples
- d : dimension of the samples (number of features)

The space $\mathcal{X}$ that contains the data, for either supervised or unsupervised learning is often included in $\mathbb{R}^d$.

Dimensionality reduction

- $\mathcal{X} \subset \mathbb{R}^d$.
- If d is large (e.g. $\geq 10^4$), the algorithms that run on the data might become too slow to be used, as their algorithmic complexity depends on d (potentially in a quadratic or exponential way, curse of dimensionality)
- we will illustrate this problem with local averaging.

Dimensionality reduction

We consider the space $\mathcal{X}$ that contains the data, for either supervised or unsupervised learning. In machine learning, we often have $\mathcal{X} \in \mathbb{R}^d$.

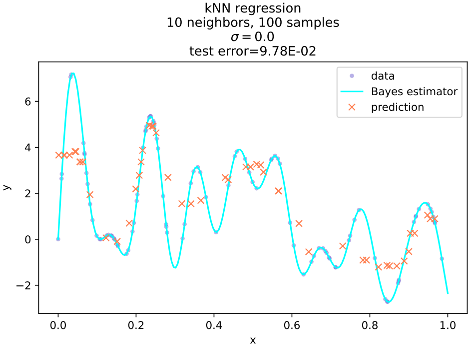
- If d is large (e.g. $\geq 10^4$), the algorithms that run on the data might become too slow to be used, as their algorithmic complexity depends on d (potentially in a quadratic or exponential way, curse of dimensionality)
- **However**, often the data might actually occupy a **subspace** of lower dimension q, or it may be possible to project the data on such a subspace without loosing too much information.
  - Working in a subspace of lower dimension might speed up the algorithms.
  - It may also allow visualization of the data.

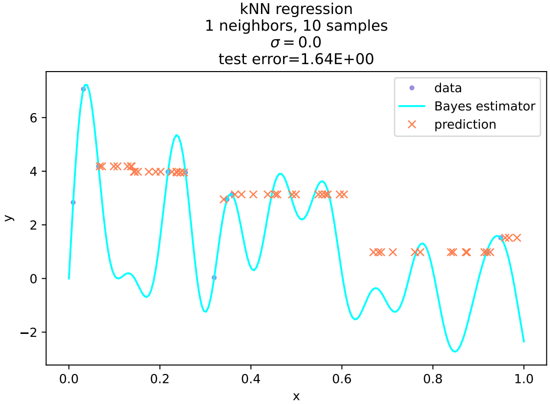
Methods of dimensionality reduction

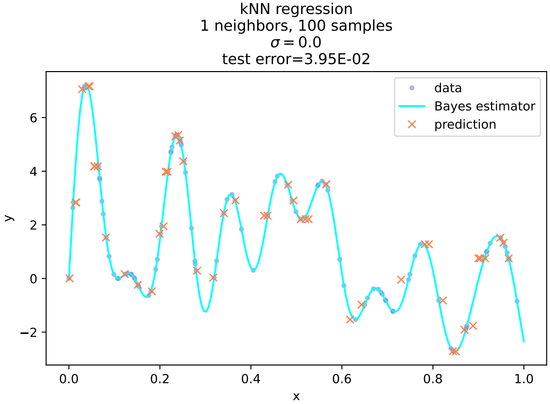
- **feature selection** : selecting a subset of the original dimensions.
- **feature extraction** : computing new features from the original features.

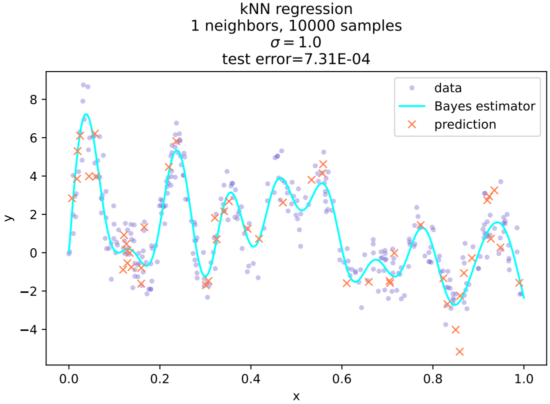
Local averaging

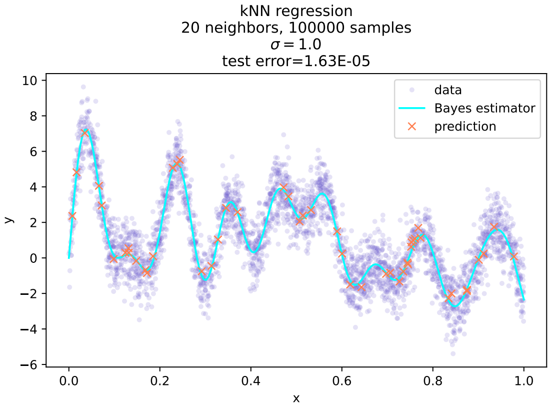
Local averaging is a prediction method that is not based on any opitmization, but only based on the dataset itself.




kNN regression
1 neighbors, 10 samples
$\sigma = 0.0$
test error=1.64E+00
data
Bayes estimator
prediction
y
x
0.0
0.2
0.4
0.6
0.8
1.0
−2
0
2
4
6


kNN regression
1 neighbors, 100 samples
$\sigma = 0.0$
test error=3.95E-02
data
Bayes estimator
prediction
y
x
0.0
0.2
0.4
0.6
0.8
1.0
−2
0
2
4
6


kNN regression
1 neighbors, 10000 samples
$\sigma = 1.0$
test error=7.31E-04
data
Bayes estimator
prediction
y
x
8
6
4
2
0
−2
−4
0.0
0.2
0.4
0.6
0.8
1.0


kNN regression
20 neighbors, 100000 samples
$\sigma = 1.0$
test error=1.63E-05
data
Bayes estimator
prediction
x
y
10
8
6
4
2
0
−2
−4
−6
0.0
0.2
0.4
0.6
0.8
1.0

The problem with local averaging is that the number of samples that we need in order to garantee a prediction error of order of magnitude ϵ is

$$n_\epsilon \geq \frac{\epsilon^{-d} d^{d/2}}{\alpha^{d/2}} \tag{1}$$

where $\alpha >$ is a constant. n_ϵ hence grows exponentially with $1/\epsilon$, and this is an example of curse of dimensionality.

https://scikit-learn.org/stable/modules/unsupervised_reduction.html
https://en.wikipedia.org/wiki/Principal_component_analysis

Introduction

- Principal Component Analysis is a dimensionality reduction method.

Introduction

- Principal Component Analysis is a dimensionality reduction method.
- It is based on statistical and geometrical considerations.
- It was invented by Pearson at the beginning of XXth century but is still used today.

Introduction

- Principal Component Analysis is a dimensionality reduction method.
- It is based on statistical and geometrical considerations.
- Applications include :
  - dimensionality reduction
  - noise filtering
  - prediction
  - general data visualization and analysis

Example

In this paper, astrophysicists use PCA in order to test a new star temperature prediction method [Bermejo et al., 2013]

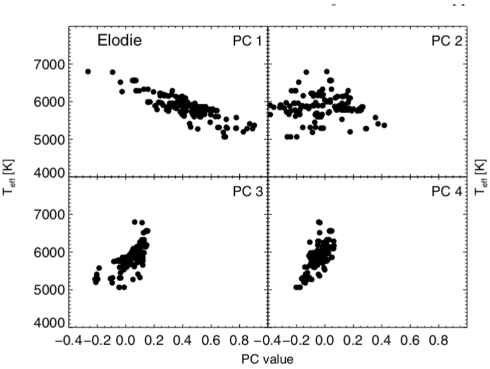
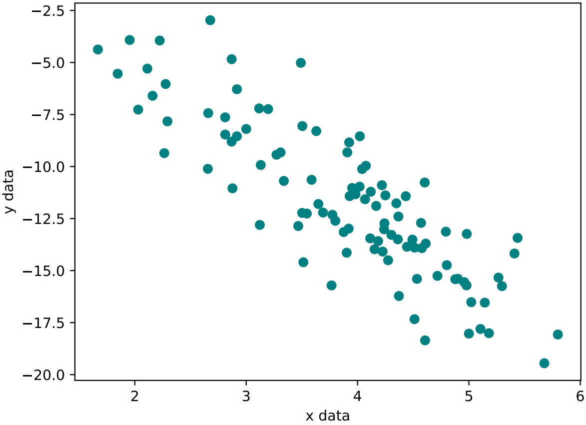


Figure – PCA used in order to predict temperature.

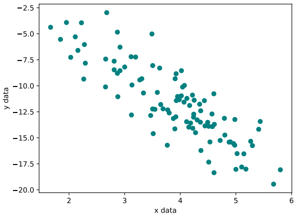
Problem statement

- We have multidimensional data.



Problem statement

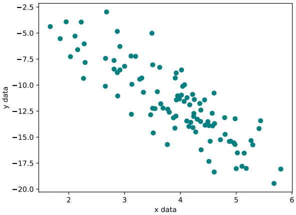
- We have multidimensional data.



We look for the axes that "explain" or carry the most variations in the data.

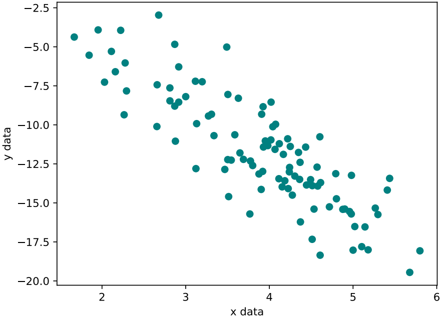
Problem statement

- We have multidimensional data.



We look for the axes that "explain" or carry the most variations in the data.
Thoses axes will be called the **principal components.**

Visual intuition



When looking at this image, what axis would you suggest in order to explain the biggest variation among the data ?

Formalization

We need a mathematical criterion in order to formalize this intuition.

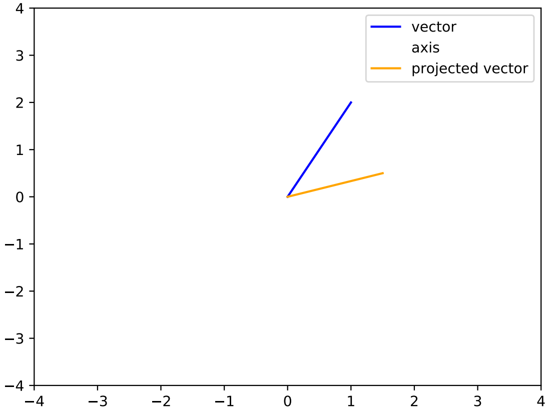
Formalization

We need a **mathematical criterion** in order to formalize this intuition.
In the case of a larger problem with more dimensions, it will not be possible to use visual feedback in order to choose the principal components (the axis).
Furthermore, even in 2D, using only visualization, we can only do a **approximation** of the most relevant axis.

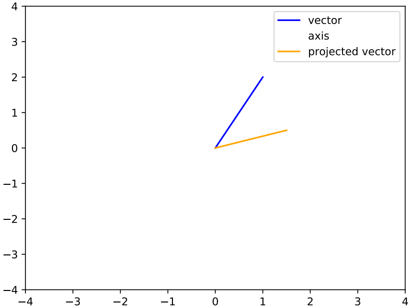
Orthogonal projections

We will restate the problem in a mathematical way, using tools coming from **linear algebra.**
In particular, the concept of **orthogonal projection** is essential.

Orthogonal projection



Orthogonal projection



Exercice 1 : **Mathematical problem** Can you link orthogonal projections to the problem of finding the most relevant axis ?

Inertia

- The **inertia** related to an axis will be a measure of the relevance of an axis.
- If x_i is a sample taken from n samples, and Δ an axis, we call $p_{x_i \to \Delta}$ the orthogonal projection of x_i on the axis Δ.

Inertia

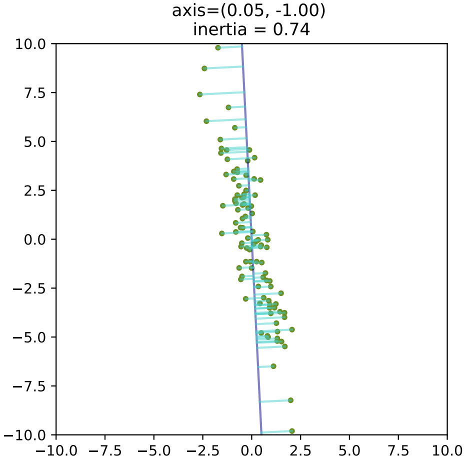
- The **inertia** related to an axis will be a measure of the quality of an axis.
- If x_i is a sample from n samples, and Δ an axis, we call $p_{x_i \to \Delta}$ the orthogonal projection of x_i on the axis Δ.
- The inertia related to Δ is :

$$I_\Delta = \frac{1}{n} \sum_{i=1}^{n} d^2(x_i, p_{x_i \to \Delta}) \tag{2}$$

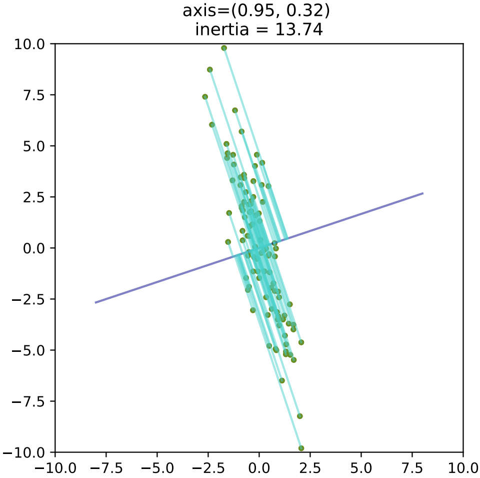
Inertia

Exercice 2 : **No inertia**
In what situations could we have $I_\Delta = 0$?

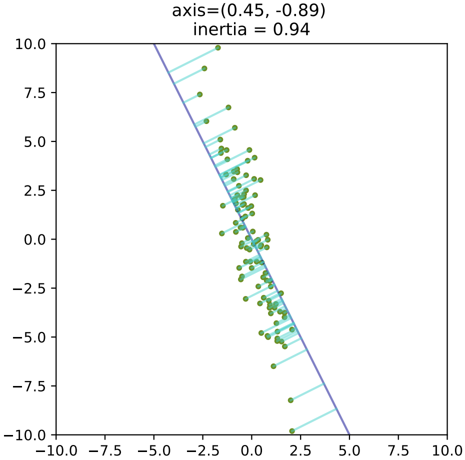
Inertia


axis=(0.05, -1.00)
inertia = 0.74
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

Inertia


axis=(0.95, 0.32)
inertia = 13.74
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

Inertia



axis=(0.45, -0.89)
inertia = 0.94
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

Inertia

Exercice 3 : **Computing an inertia**

Please **cd pca/custom data/** and use the file **inertia.py** in order to represent the projections of the data onto a chosen axis and to compute the inertia of this axis.

You need to add the computation of the inertia to the script (starting at line 75).

What is your optimal axis ?

Remark

Minimizing I_Δ is the same as maximizing I_{Δ^*} where Δ^* is the supplementary orthogonal space.

Several principal components

- In $2D$ (like in the former example), we can only have 1 or 2 principal component.

Several principal components

- In $2D$ (like in the former example), we can only have 1 or 2 principal component.
- However, when the data have more dimensions, it is possible to have **several principal components** (as many as the dimensionality of the data).
- the data are then projected on these components.

Link with expected values

We will connect PCA and the inertia to **expected values and variances.**

Expected value (espérance)

- For a discrete random variable X that takes the values x_i with probability p_i :

$$E(X) = \sum_{i=1}^{n} p_i x_i \tag{3}$$

- For a continuous random variable X with density p(x) :

$$E(X) = \int x p(x) dx \tag{4}$$

Variance

$$var(X) = E\left(\left(X - E(X)\right)^2\right) \tag{5}$$

Variance and Covariance

$$var(X) = E\Big(\big(X - E(X)\big)^2\Big) \tag{6}$$

$$cov(X, Y) = E\Big(\big(X - E(X)\big)\big(Y - E(Y)\big)\Big) \tag{7}$$

Probability vs Statistics

(Advanced notions)

- from a dataset of samples $(x_1, \dots, x_n)$, we can only compute **statistics**
- for instance
  - sample mean :

$$\hat{x} = \frac{1}{n} \sum_{i=1}^{n} x_i \tag{8}$$

  - unbiased sample variance :

$$\frac{1}{n-1} \sum_{i=1}^{n} (x_i - \hat{x})^2 \tag{9}$$

The $\frac{1}{n-1}$ factor instead of $\frac{1}{n}$ requires some theory to be properly understood.

Variance and Inertia

Exercice 4 : **Linking variance and inertia**
What is the relationship between variance and inertia ?

Optimization

- In real applications, algorithms or analytic solutions are used in order to find the optimal axes.

Optimization with analytic solution

- Methods include the **Lagrange multiplier method**
- The method shows that we need to compute the **eigenvectors** of the **covariance matrix.**
- See also : Gram matrix

Optimization with algorithms

- One can also use an approximation method to find the first principal component, called the **power iteration** algorithm.
- It is useful when the dimensionality is high, when the Lagrange multiplier method becomes too slow, since it involves computation of the covariance matrix.

Optimization with algorithms

Exercice 5 : **Complexity of computing the covariance matrix.** If n is the number of datapoints, and p the number of variables, what is the complexity of this calculation ?

Optimization with algorithms

Exercice 5 : **Complexity of computing the covariance matrix.**
If n is the number of datapoints, and p the number of variables, what is the complexity of this calculation ?
The power iteration algorithm only involves cnp operations, where c is a constant way smaller than p, so it is way faster.

PCA with scikit-learn

- We will use scikit-learn in order to perform PCA on our dataset.
- https://scikit-learn.org/stable/modules/decomposition.html
- https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html

PCA with scikit-learn

Exercice 6 : **scikit-learn**

Please use the file **pca_sklearn.py** in order to find the principal components on the dataset of the previous exercise.

Iris dataset

- We can perform PCA on the iris dataset as in the file **iris/pca-iris.py**.

Iris dataset

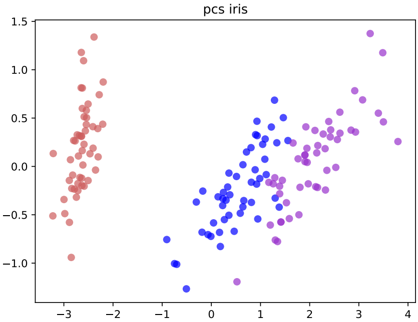
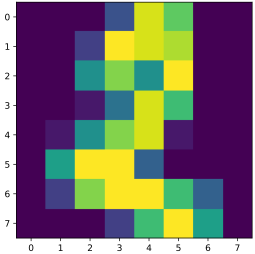


Figure – PCA performed on the iris dataset. We see that the principal components are able to separate the data.

PCA on digits

- We will also perform the PCA on a dataset consisting in 8×8 pixels images of digits.
- The idea is to see if the PCA can help us visualize structure in the data.

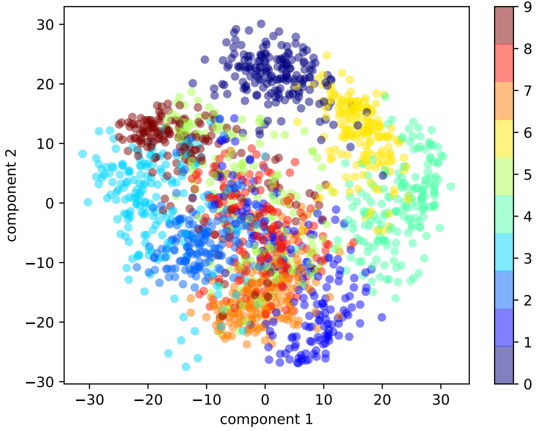


PCA on digits

Exercice 7 : **Performing PCA**

Please use the file **pca_digits.py** in order to apply PCA to this dataset.

PCA on digits



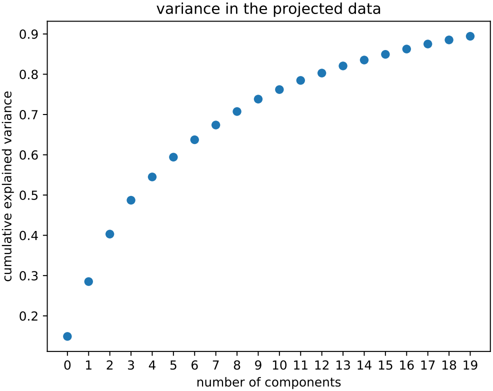
Number of components

What is a relevant number of components for PCA ?

Number of components

Exercice 8 : **Choosing the number of components**
Use the file **pca_digits_variance.py** in order to determine how many components are necessary in order to keep 75% of the variance in the digits dataset.

Number of components



Reconstruction

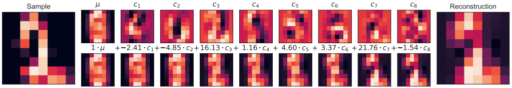


Figure – Reconstruction of a sample with 8 components

Reconstruction

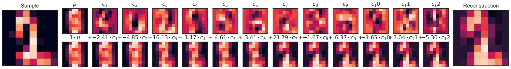
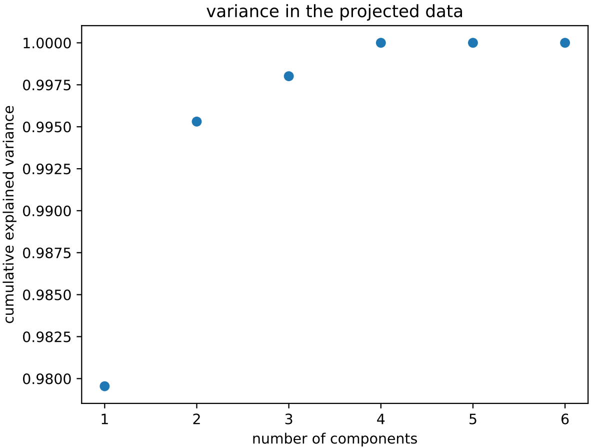


Figure – Reconstruction of a sample with 12 components

Number of components

Exercice 9 : **Redundancy**
What happens with the data contained in **redundancy/redundant_data.npy** ? You can analyze them with the file **redundancy/pca_variance.py**.

Number of components



Number of components

Conclusion : PCA can help detemine whether some components carry no information in the data.

Shortcomings of PCA

PCA is sensitive to :

- outliers
- initial data scaling

Shortcomings of PCA

PCA is sensitive to :

- outliers
- initial data scaling
- Many variants and heuristics exist on this topic.

Asset of PCA

Reducing the dimensionality of the data might be very helpful in a situation where you need to train a classification algorithm on the projected data. The algorithm might be way faster on data that are in a smaller space. However, a sufficient amount of information should be kept during the reduction, hence the necessity of heuristics.

See also

- kernel-PCA
- Randomized PCA

https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_digits.html

https://scikit-learn.org/stable/modules/manifold.html
https://en.wikipedia.org/wiki/Nonlinear_dimensionality_reduction

Manifolds

Fundamental hypothesis (as before) :
In some cases, altough the dimensionality of the data is high (as for images), the data lie on a specific subpart of the linear space containing them.
This "subpart" is assumed to be a **manifold** (variété).

Manifolds

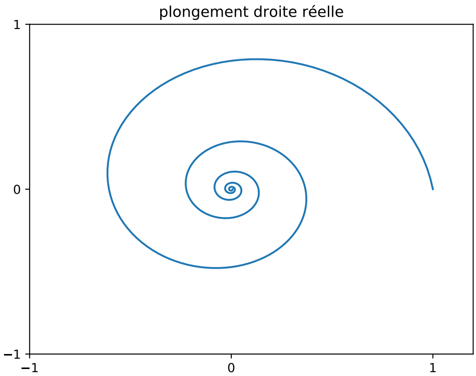


Figure – 1 dimensional manifold, in $\mathbb{R}^2$

Linearity

- In the case of PCA, the manifolds are **linear subspaces** of the ambient space : PCA is a **linear method** and does **linear projections.**
- A linear mapping in a linear space E is a mapping u such that

$$\forall x, y \in E, u(x+y) = u(x) + u(y) \tag{10}$$

Nonlinearity

- However, in many situations, the data might lie on **nonlinear** manifolds.
- **Manifold learning** is the unsupervised learning of these manifolds in order to study the structure of the data.

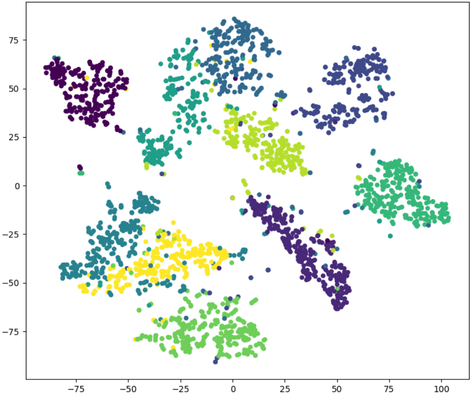
Multidimensional scaling (MDS) / Positionnement multidimensionnel

- Projects the data in a smaller subspace trying to preserve pairwise similarities between points.

t-SNE

- t-Stochastic Neighbor Embedding [Van Der Maaten and Hinton, 2008]
- https://lvdmaaten.github.io/tsne/

t-SNE on MNIST



Seel also

- Isomap
- LLE

Comments

Some disadvantages of non linear manifold methods :

- Hard to determine a good output dimension (whereas in PCA we can use explained variance) and it is hard to interpret the embedded dimensions (whereas in PCA we know what they mean).
- Depends on the number of neighbors chosen (if relevant)
- Often compuationally slower.

References

Bermejo, J. M., Ramos, A. A., and Prieto, C. A. (2013).
Astrophysics A PCA approach to stellar effective temperatures.
95 :1–9.

Van Der Maaten, L. J. P. and Hinton, G. E. (2008).
Visualizing high-dimensional data using t-sne.
Journal of Machine Learning Research, 9 :2579–2605.