T3A: Machine Learning Algorithms

Master of Science in AI and Master of Science in Data Science

@ UPSaclay

2024/2025.

Faïcel Chamroukhi

chamroukhi.com

Outline

Introduction

université PARIS-SACLAY

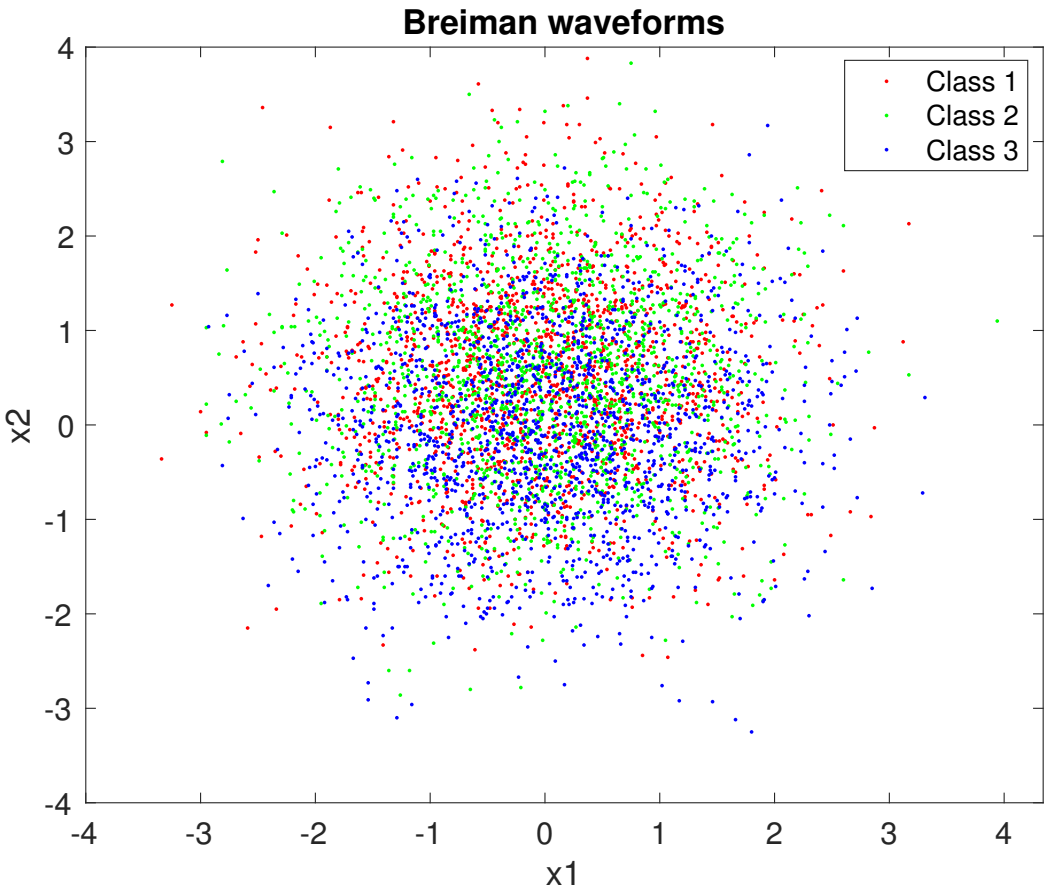
- We assume that we have a set of data collected in some way (e.g., independent or not (i.e sequential), complete or not, etc.),

- to analyze, in some sense (e.g., for prediction, exploration, selection, visualisation, etc.), some scenario or system, in a broad sense.

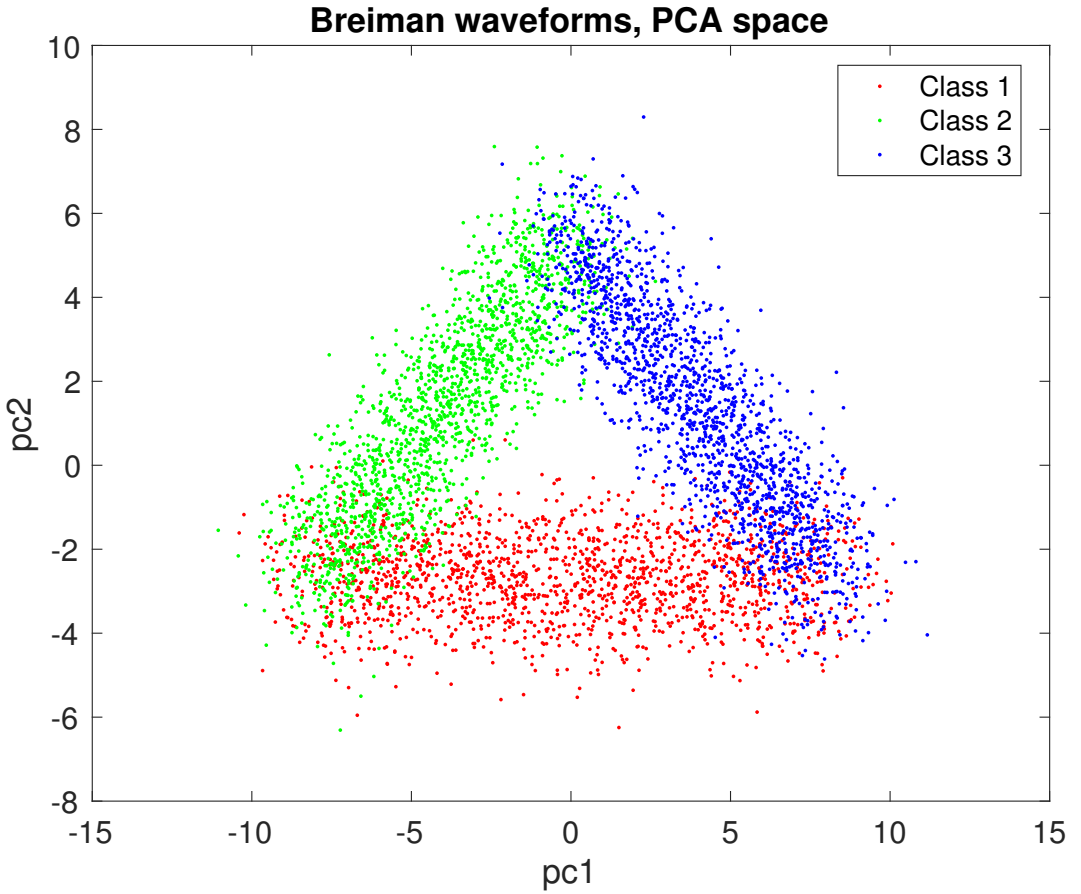
  prediction clustering, dimensionality reduction, visualisation, etc

- We assume that the data are represented by random variables $\hookrightarrow$ statistical learning framework

Representation



Representation



Unsupervised Learning

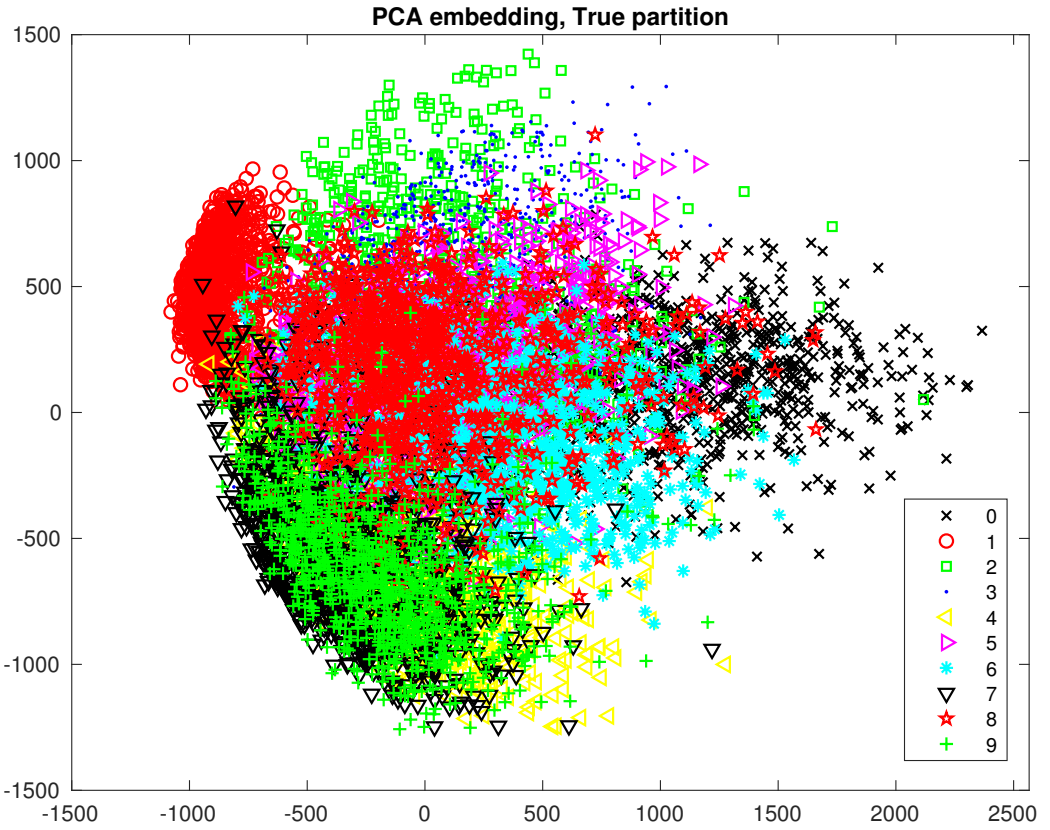
université PARIS-SACLAY

Clustering / Representation / Data viz / Dimensionality reduction

Unsupervised Analysis

université PARIS-SACLAY

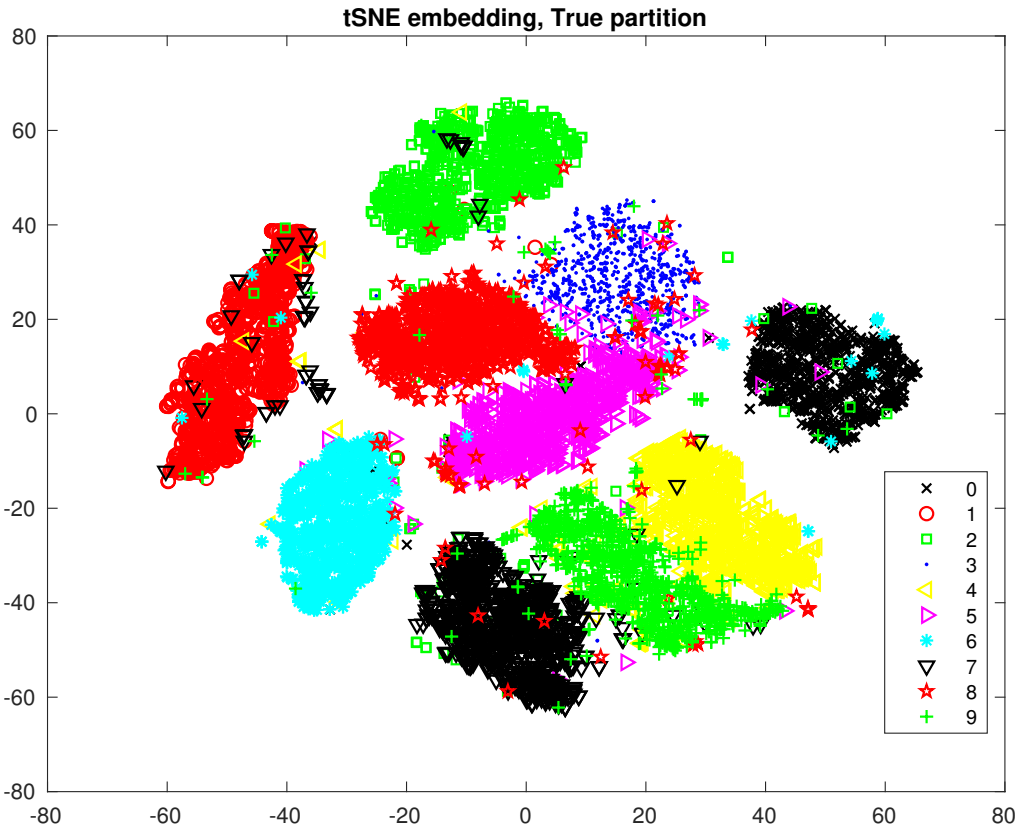
Representation / Data viz / Dimensionality reduction



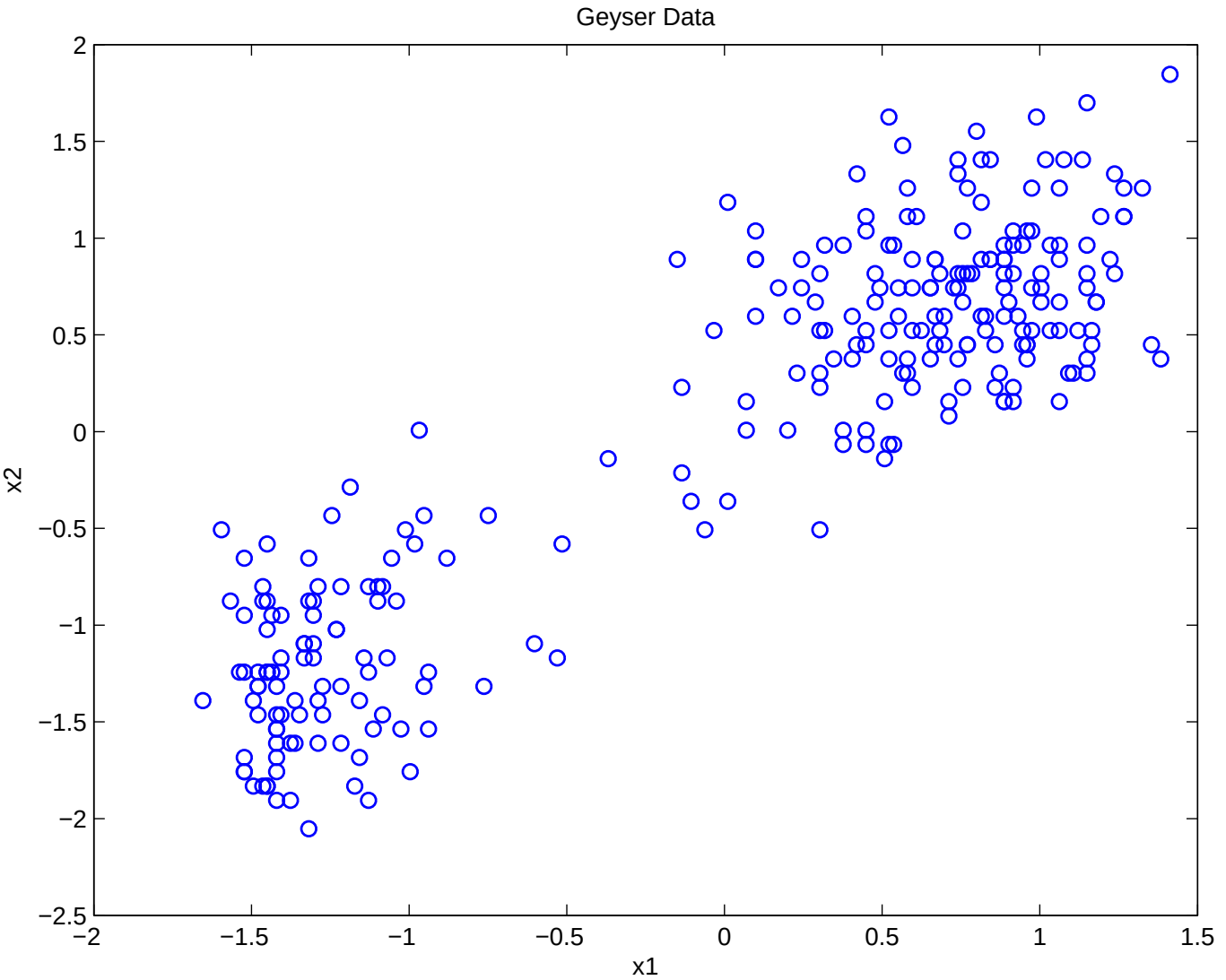
Unsupervised Analysis

université PARIS-SACLAY

Representation / Data viz / Dimensionality reduction



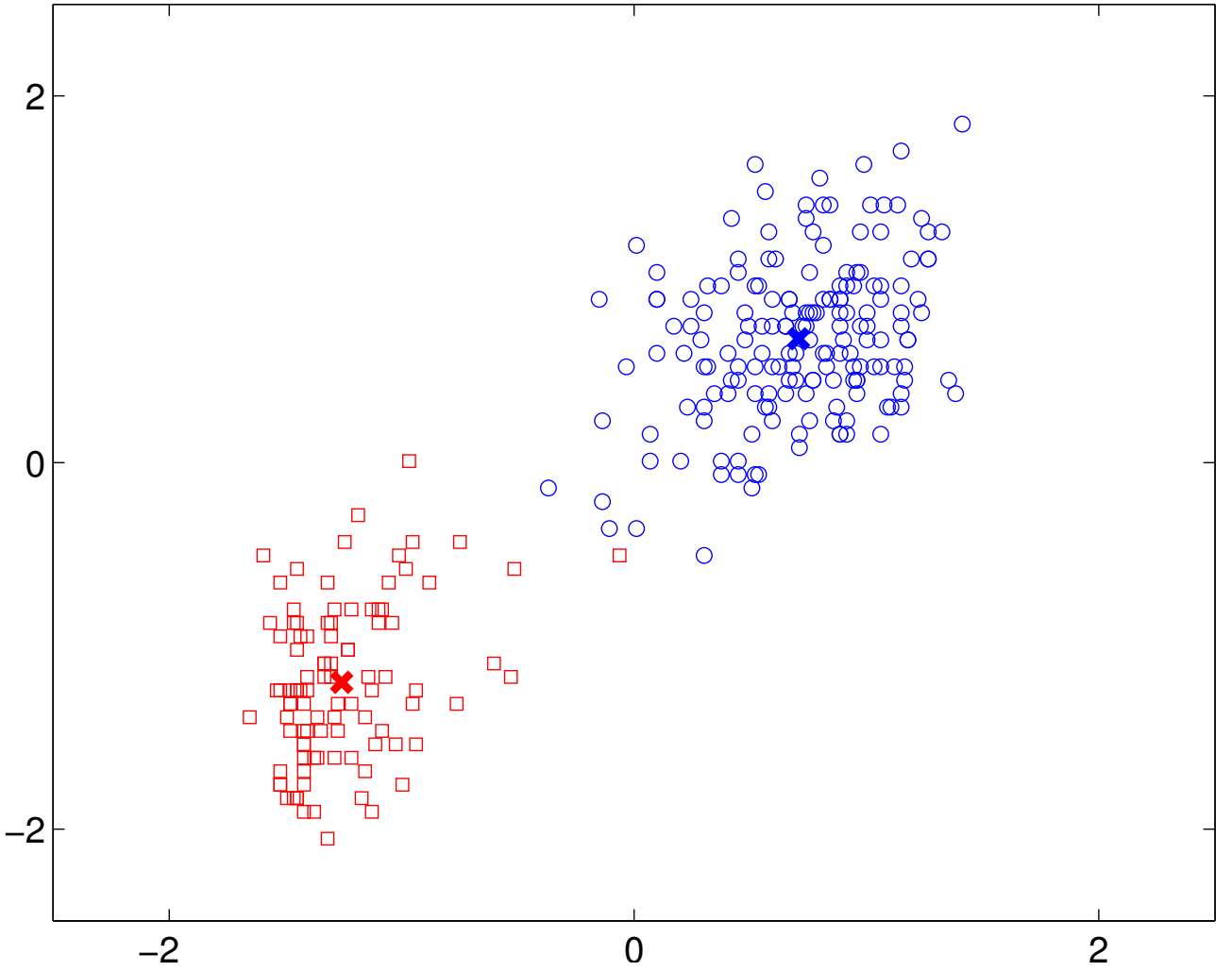
Unsupervised analysis



Unsupervised analysis

université PARIS-SACLAY

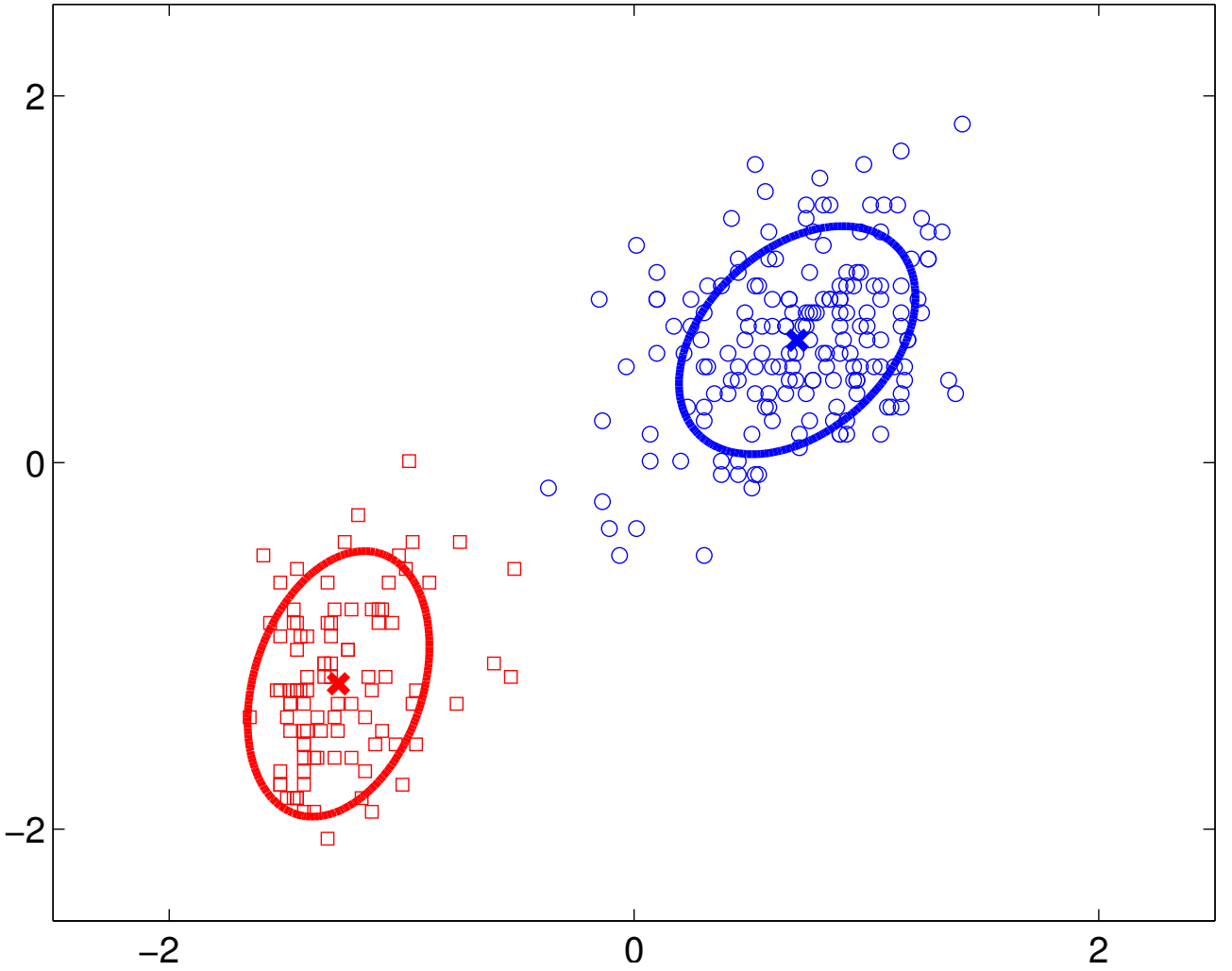
Clustering



Unsupervised analysis

université PARIS-SACLAY

Clustering



Unsupervised analysis

université PARIS-SACLAY

Clustering

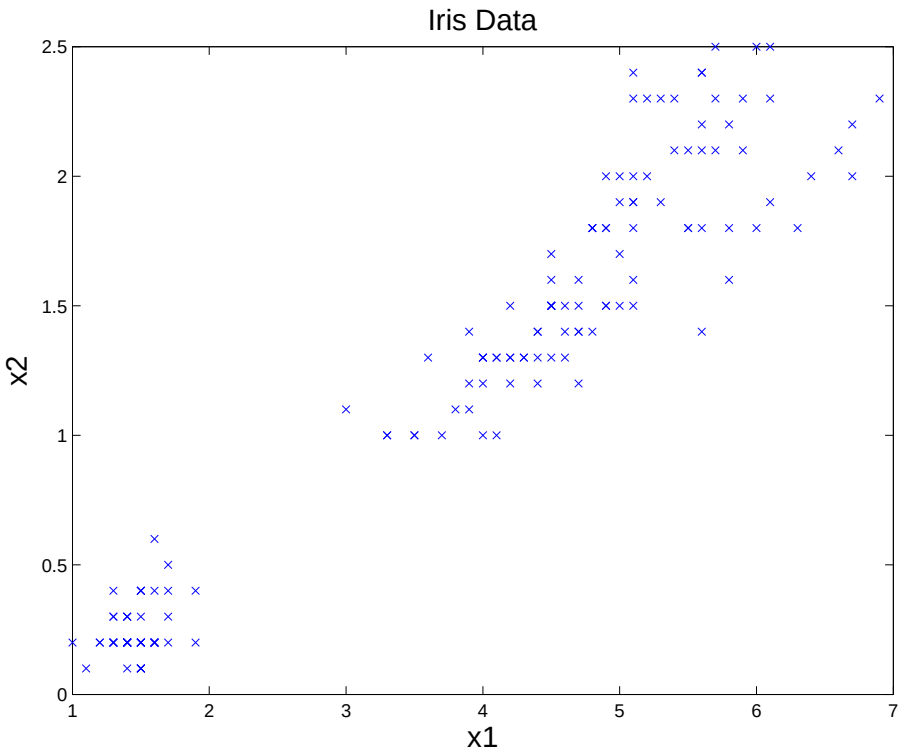


FIGURE – A three-class example of a real data set : Iris data of Fisher.

Unsupervised analysis

université PARIS-SACLAY

Clustering

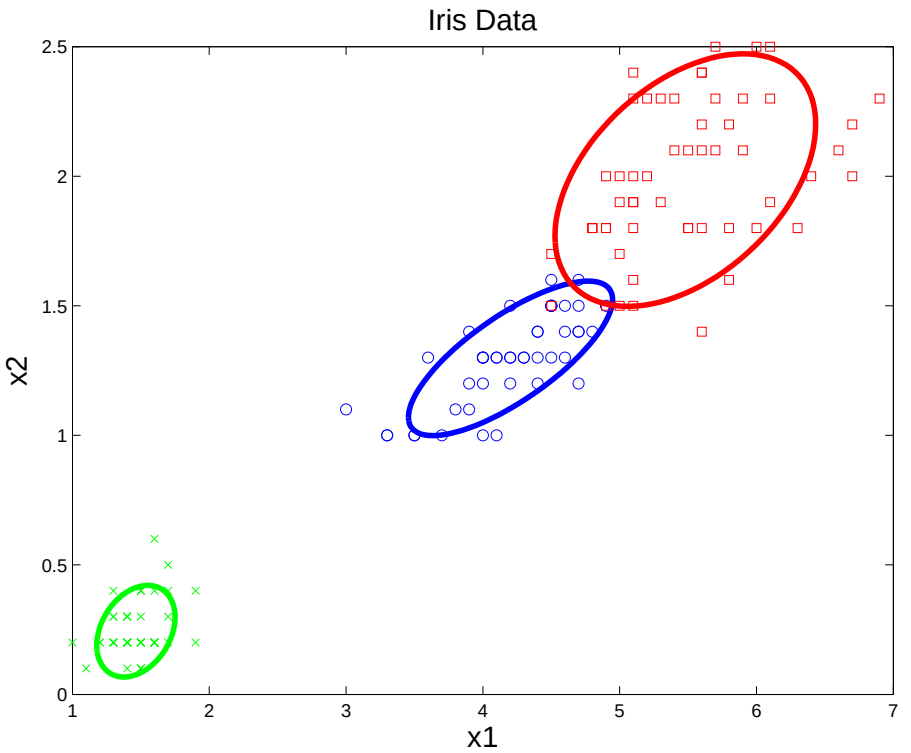


FIGURE – Iris data : Clustering results with EM for a GMM and AIC.

Unsupervised Learning

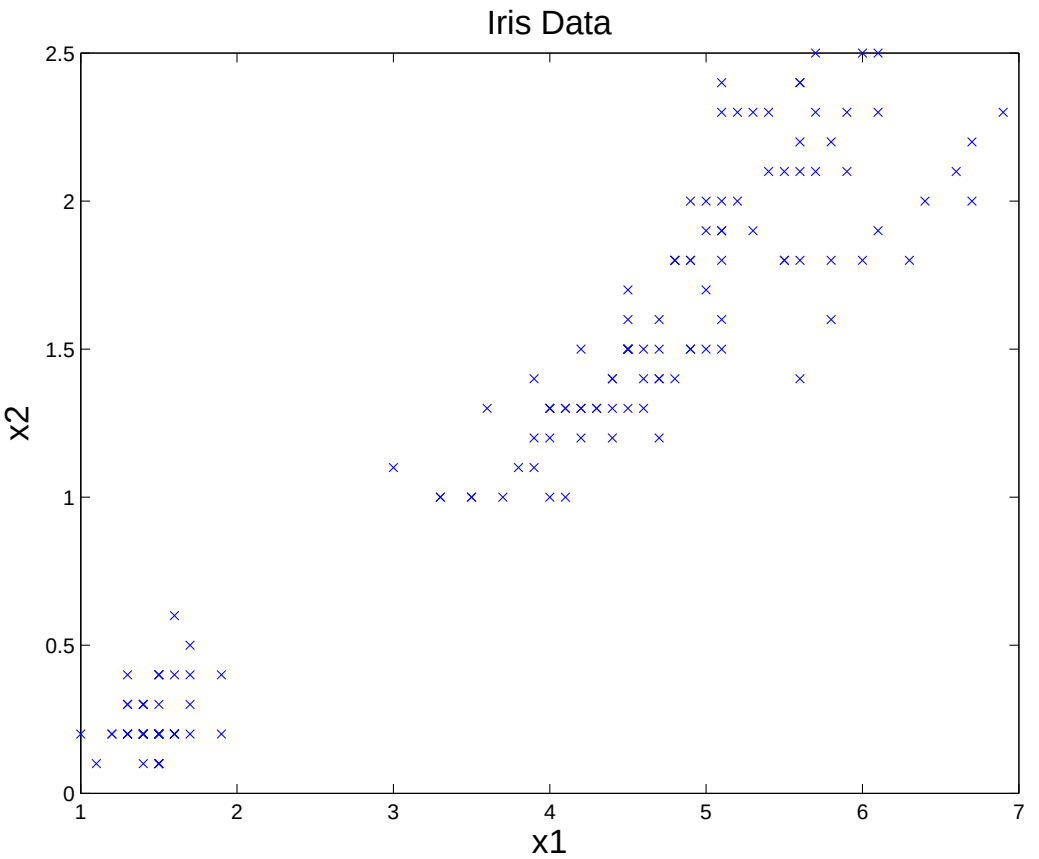
Clustering

université PARIS-SACLAY

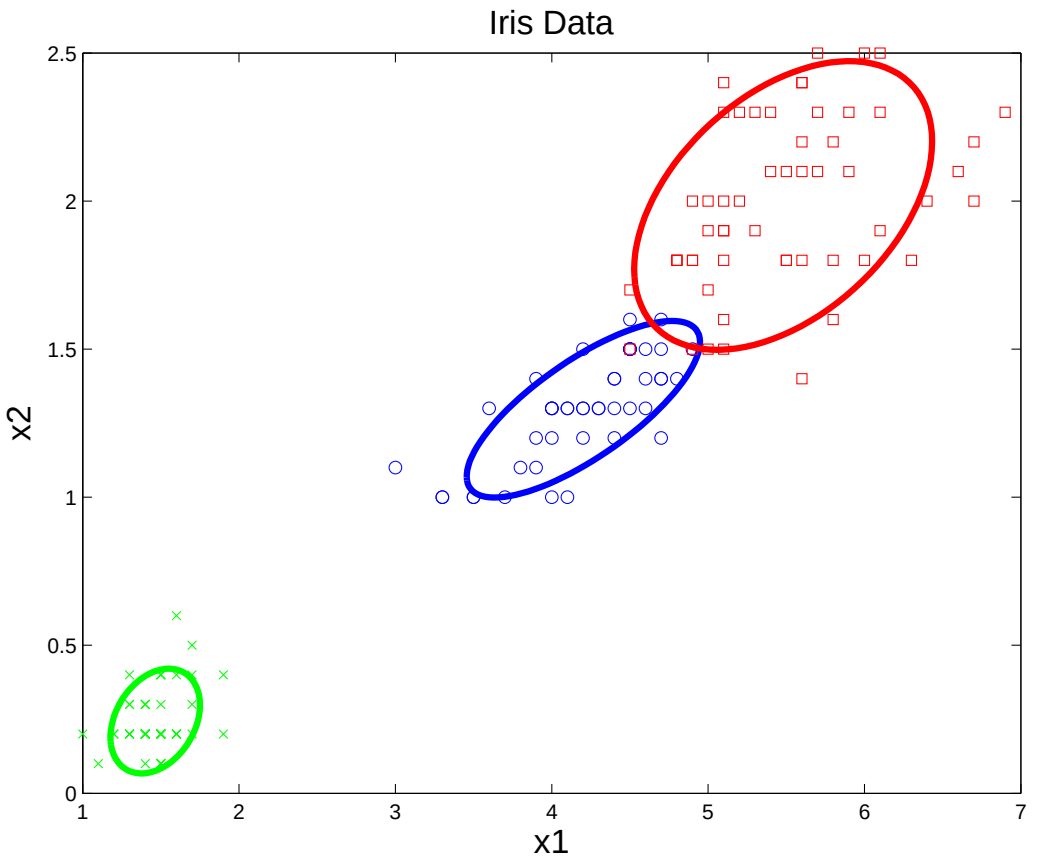
- Clustering is often referred to as unsupervised learning in the sense that the class labels of the data are unknown (missing, hidden). Only the observations $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ are given,
- suitable for many applications where labeled data is difficult to obtain.
- also used to explore and characterize a dataset before running a supervised learning task.
- In clustering, the data are grouped by some notion of dissimilarity.

  $\Rightarrow$ a dissimilarity measure must be defined based on the data.
- the aim of clustering is to find a partition of the data by dividing them into clusters (groups) such that the data within a group tend to be more similar to one another as compared to the data belonging to different groups.
- There is, distance-based, model-based, hierarchical, topographical clustering approaches, etc

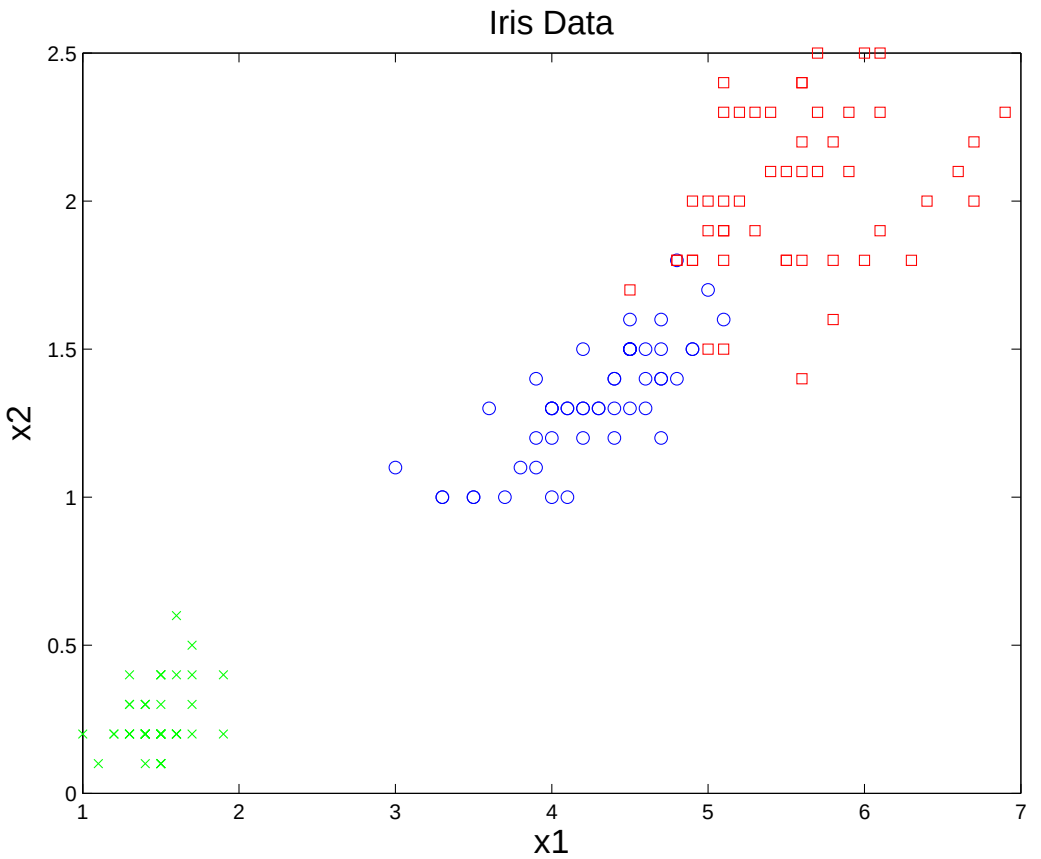
Example 3 : Iris data



Example 3 : Iris data



Example 3 : Iris data



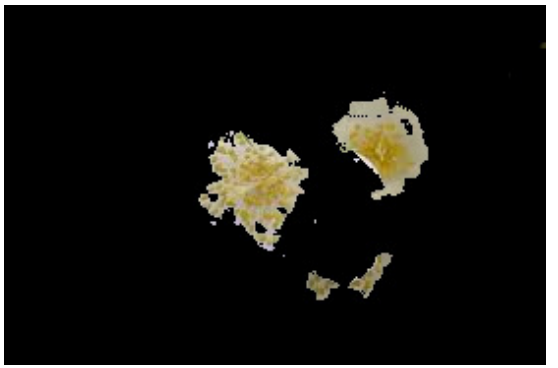
Illustrations

image originale

objects in cluster 1

objects in cluster 2

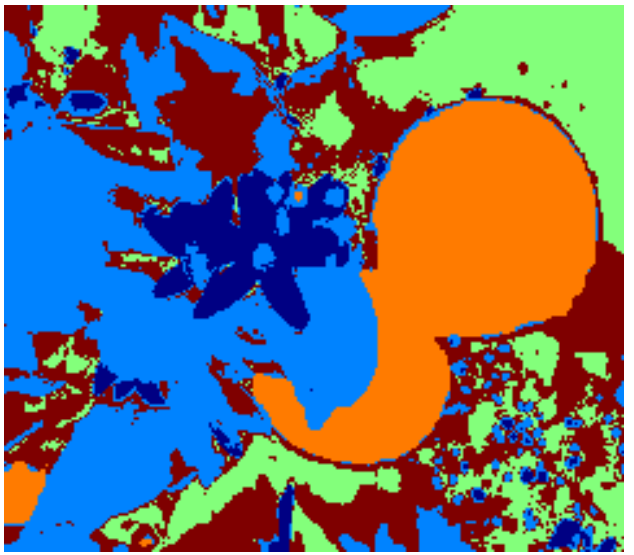
objects in cluster 3



Illustrations

image originale

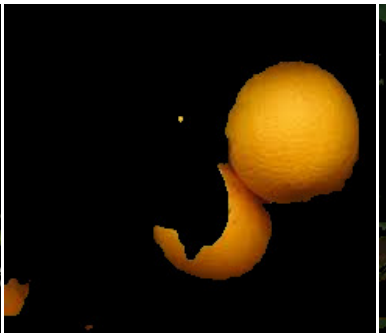
image segmentee par GMM–EM (K=5)



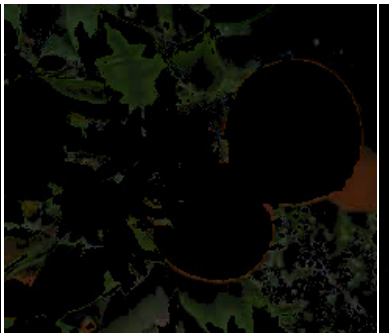
objects in cluster 1

objects in cluster 2

objects in cluster 4



objects in cluster 5

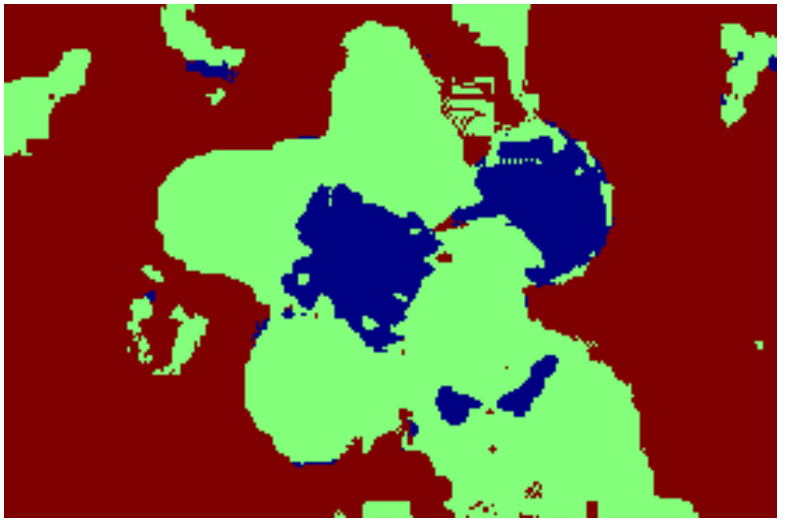


objects in cluster 3

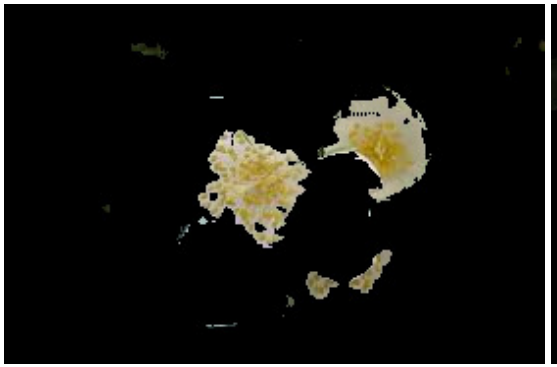
Illustrations

image originale

image segmentee par GMM–EM (K=3)



objects in cluster 1



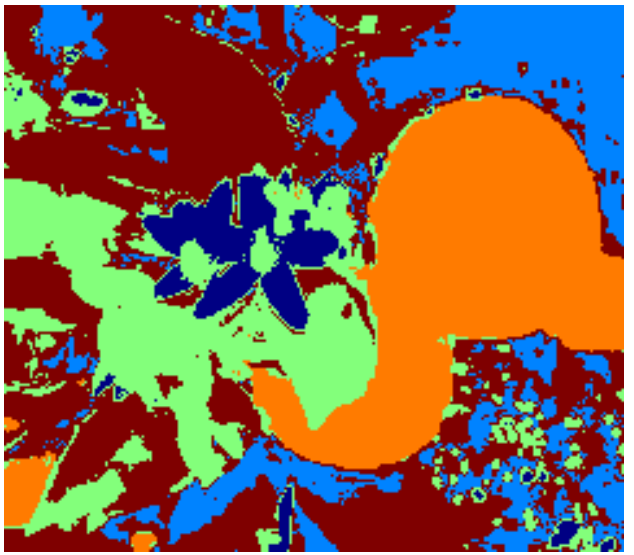
objects in cluster 2

objects in cluster 3

Illustrations

image originale

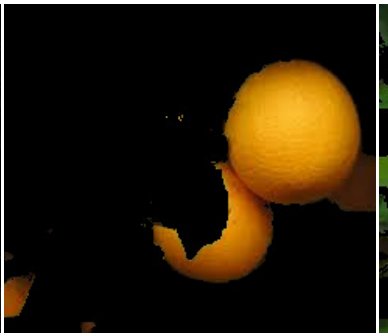
image segmentee par GMM–CEM (K=5)



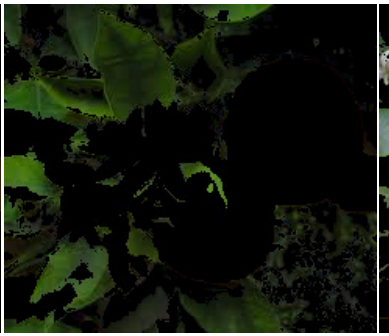
objects in cluster 1

objects in cluster 2

objects in cluster 4

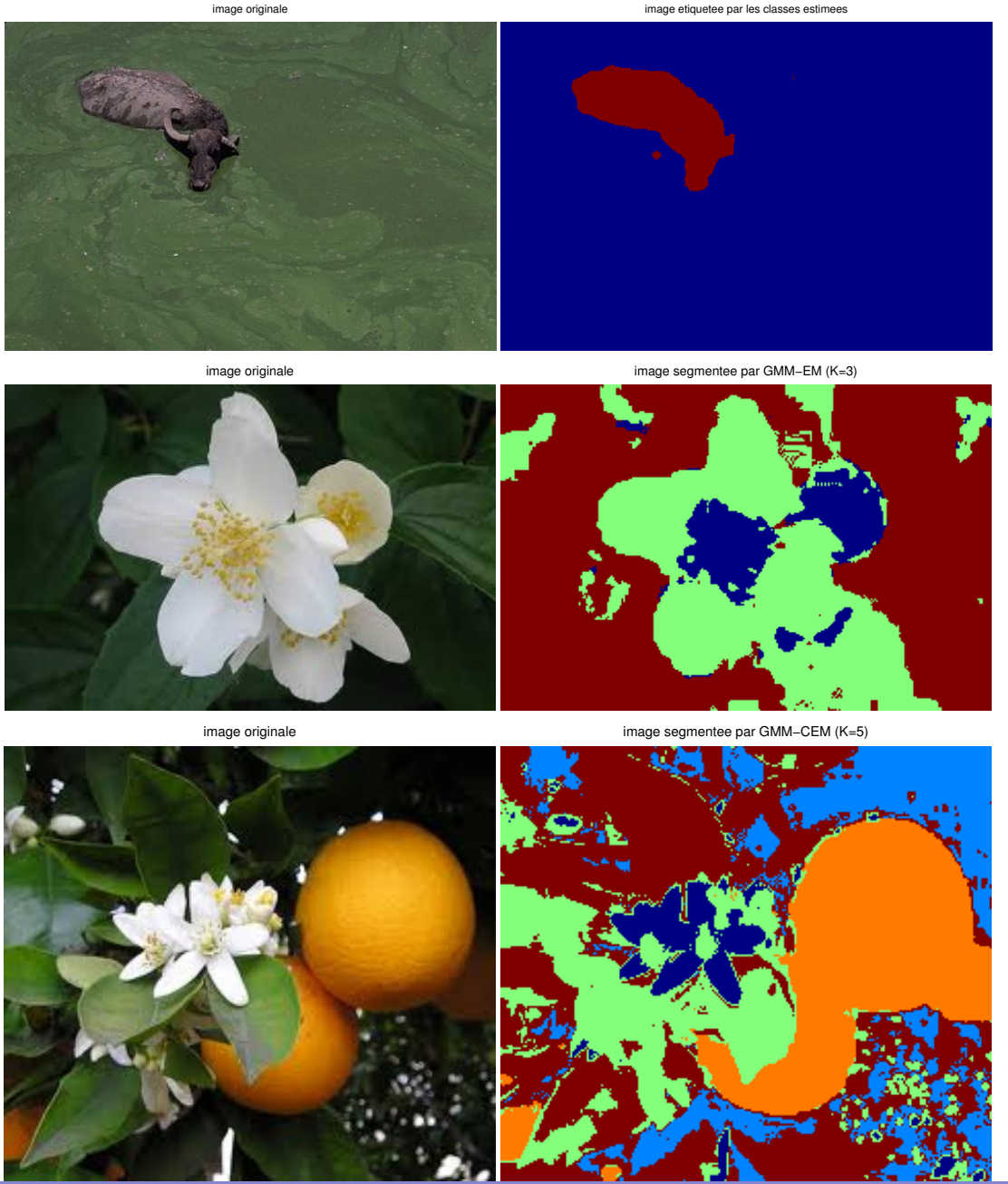


objects in cluster 5



objects in cluster 3

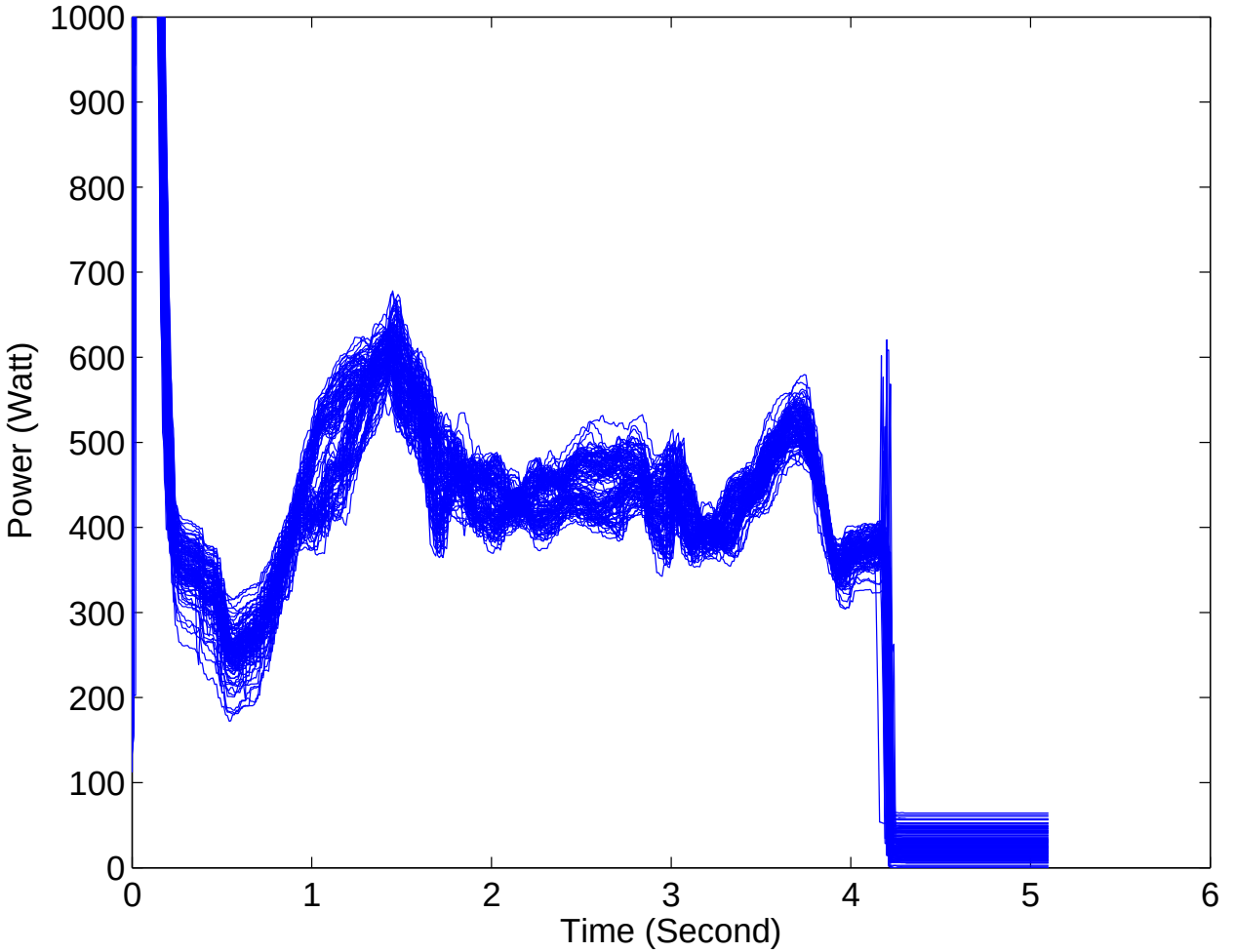
Example : Image segmentation



example 2 : Fault detection

université PARIS-SACLAY

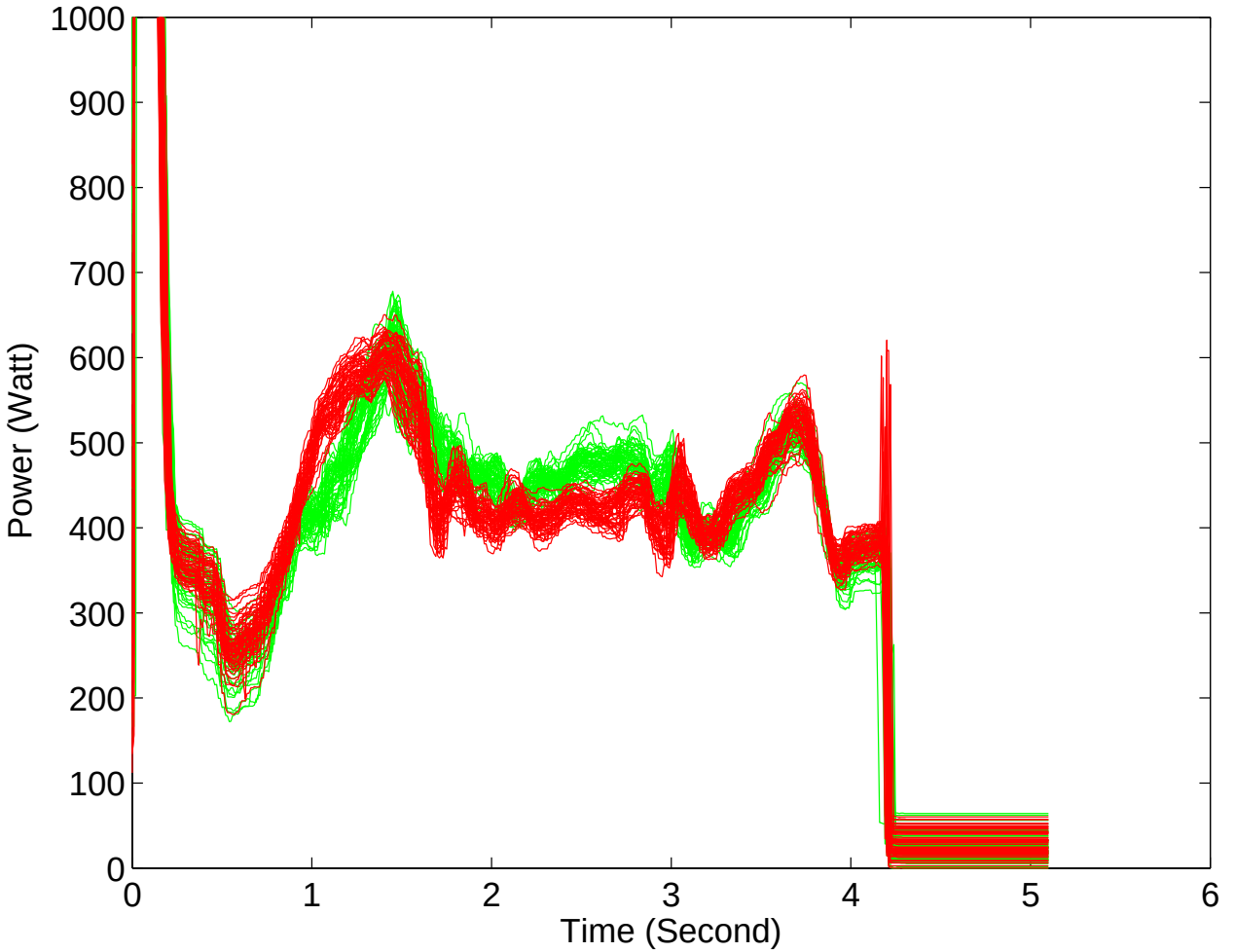
Real data (curves)



example 2 : Fault detection

université PARIS-SACLAY

Clustering results



K-means

université PARIS-SACLAY

- a straightforward and widely used clustering algorithm, is one of the most important algorithms in unsupervised learning.
- Each cluster is represented by its mean (cluster centroid) $\boldsymbol{\mu}_k$ in $\mathbb{R}^d$.

K-means

$$(\widehat{\boldsymbol{\mu}}_1, \ldots, \widehat{\boldsymbol{\mu}}_K, \widehat{\mathbf{z}}) \in \arg \min_{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K, \mathbf{z}} \mathcal{J}(\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K, \mathbf{z})$$

objective function : $\mathcal{J}(\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K, \mathbf{z}) = \sum_{k=1}^{K} \sum_{i=1}^{n} \|\mathbf{x}_i - \boldsymbol{\mu}_{z_i}\|^2$

- Initialization : $(\boldsymbol{\mu}_1^{(0)}, \ldots, \boldsymbol{\mu}_K^{(0)})$ (eg, randomly chosen data points)

1. **Assignment step** : $z_i^{(q)} = \arg \min_{z \in \mathcal{Z}} \|\mathbf{x}_i - \boldsymbol{\mu}_z\|^2$
2. **Relocation step** : $\boldsymbol{\mu}_k^{(q+1)} = \frac{\sum_{i=1}^{n} z_{ik}^{(q)} \mathbf{x}_i}{\sum_{i=1}^{n} z_{ik}^{(q)}}$,

$\Rightarrow$ The K-means algorithm is simple to implement and relatively fast.

K-means

université PARIS-SACLAY

- a straightforward and widely used clustering algorithm, is one of the most important algorithms in unsupervised learning.
- an iterative clustering algorithm that partitions a given dataset into a predefined number of clusters K.
- the value K is chosen by prior knowledge ; how many clusters are desired ; ..
- In K-means, each cluster is represented by its mean (cluster centroid) $\boldsymbol{\mu}_k$ in $\mathbb{R}^d$.
- The default measure of dissimilarity for K-means is the Euclidean distance $\|.\|^2$.
- K-means attempts to minimize the following nonnegative objective function referred to as *distortion measure* :

$$J(\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K, \mathbf{z}) = \sum_{k=1}^{K} \sum_{i=1}^{n} z_{ik} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|^2$$

which corresponds to the total squared Euclidean distance between

K-means

université PARIS-SACLAY

- start with an initial solution $(\boldsymbol{\mu}_1^{(0)}, \ldots, \boldsymbol{\mu}_K^{(0)})$ (eg, by randomly choosing K points in $\mathbb{R}^d$ or some data points)

1. **Assignment step** : Each data point is assigned to its closest centroid using the Euclidian distance : $\forall i = 1, \ldots, n$

$$z_{ik}^{(q)} = \begin{cases} 1 \text{ if } k = \arg\min\limits_{z \in \mathcal{Z}} \|\mathbf{x}_i - \boldsymbol{\mu}_z\|^2 \\ 0 \text{ otherwise.} \end{cases}$$

2. **Relocation step** : Each cluster representative is relocated to the center (i.e., arithmetic mean) of all data points assigned to it :

$$\boldsymbol{\mu}_k^{(q+1)} = \frac{\sum_{i=1}^n z_{ik}^{(q)} \mathbf{x}_i}{\sum_{i=1}^n z_{ik}^{(q)}},$$

q being the current iteration.

$\Rightarrow$ The K-means algorithm is simple to implement and relatively fast.

Algorithm 1 K-means

Input: Données $(\mathbf{x}_1, ..., \mathbf{x}_n)$, nombre de clusters K

Iteration : $t \leftarrow 0$; $\mathbf{\Psi}^{(0)} = (\boldsymbol{\mu}_1^{(0)}, ..., \boldsymbol{\mu}_K^{(0)})$ `/* Initialize the means */`

Distorsion : $\mathcal{J}^{(t)} \leftarrow +\infty$; converge $\leftarrow 0$

while *(converge* $\neq$ *1)* **do**

  for $k \leftarrow 1$ **to** K **do**

    $\mathbf{D}_k = ||\mathbf{X} - \mathbf{1}_n \boldsymbol{\mu}_k^{\top(t)}||^2$ `/* Calculate the Euclidean Distances */`

  end

  $\mathbf{z}^{(t)} = \arg\min_k \mathbf{D}_k$ `/* Classification step */`

  for $k \leftarrow 1$ **to** K **do**

    $\boldsymbol{\mu}_k^{(t+1)} = \frac{\sum_i z_{ik}^{(t)} \boldsymbol{x}_i}{\sum_i z_{ik}^{(t)}}$ `/* Relocation Step */`

  end

  $\mathcal{J}^{(q+1)}$ `/*Calculate the distortion error */`

  `/*Test of convergence*/`

  if $|\mathcal{J}^{(t+1)} - \mathcal{J}^{(t)}| < \epsilon$ **then**

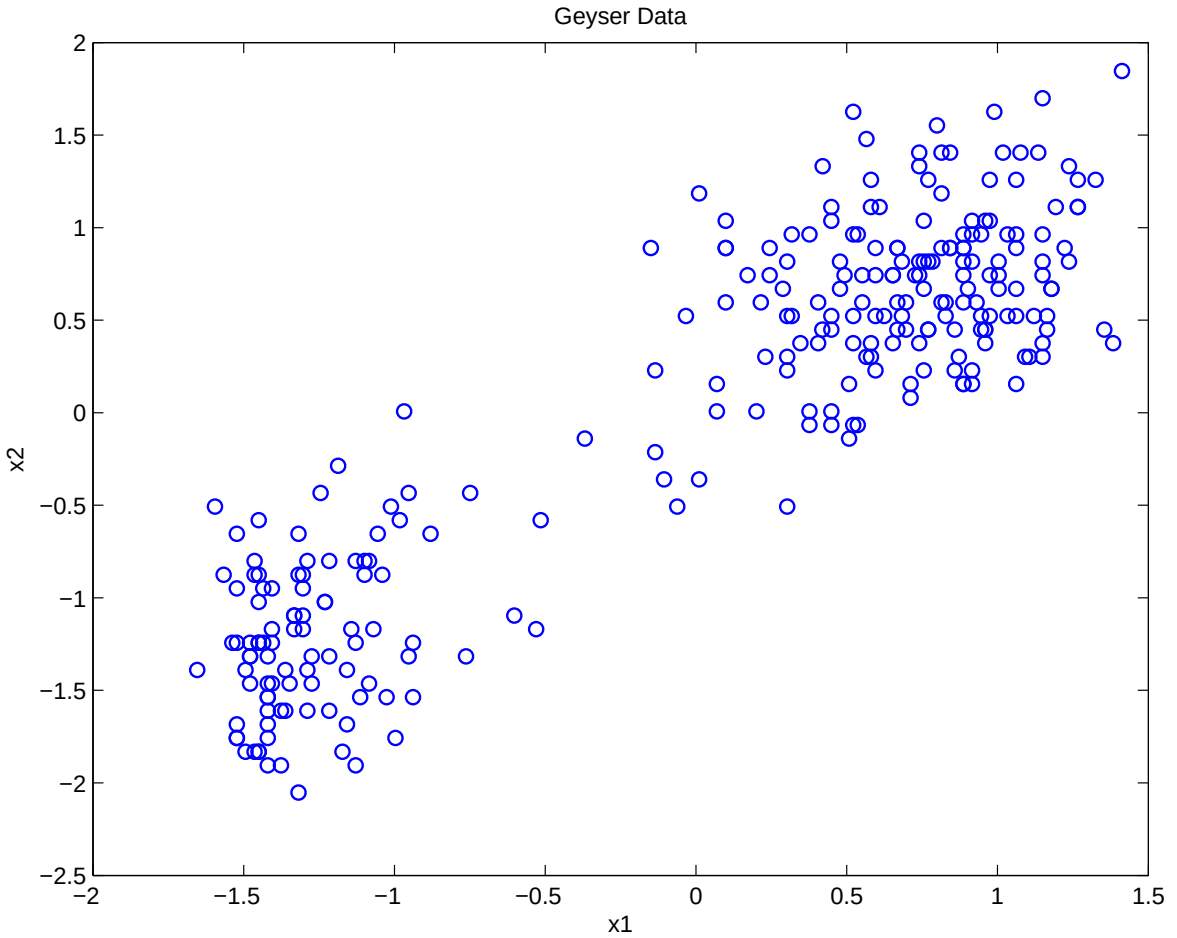
    converge$= 1$

  end

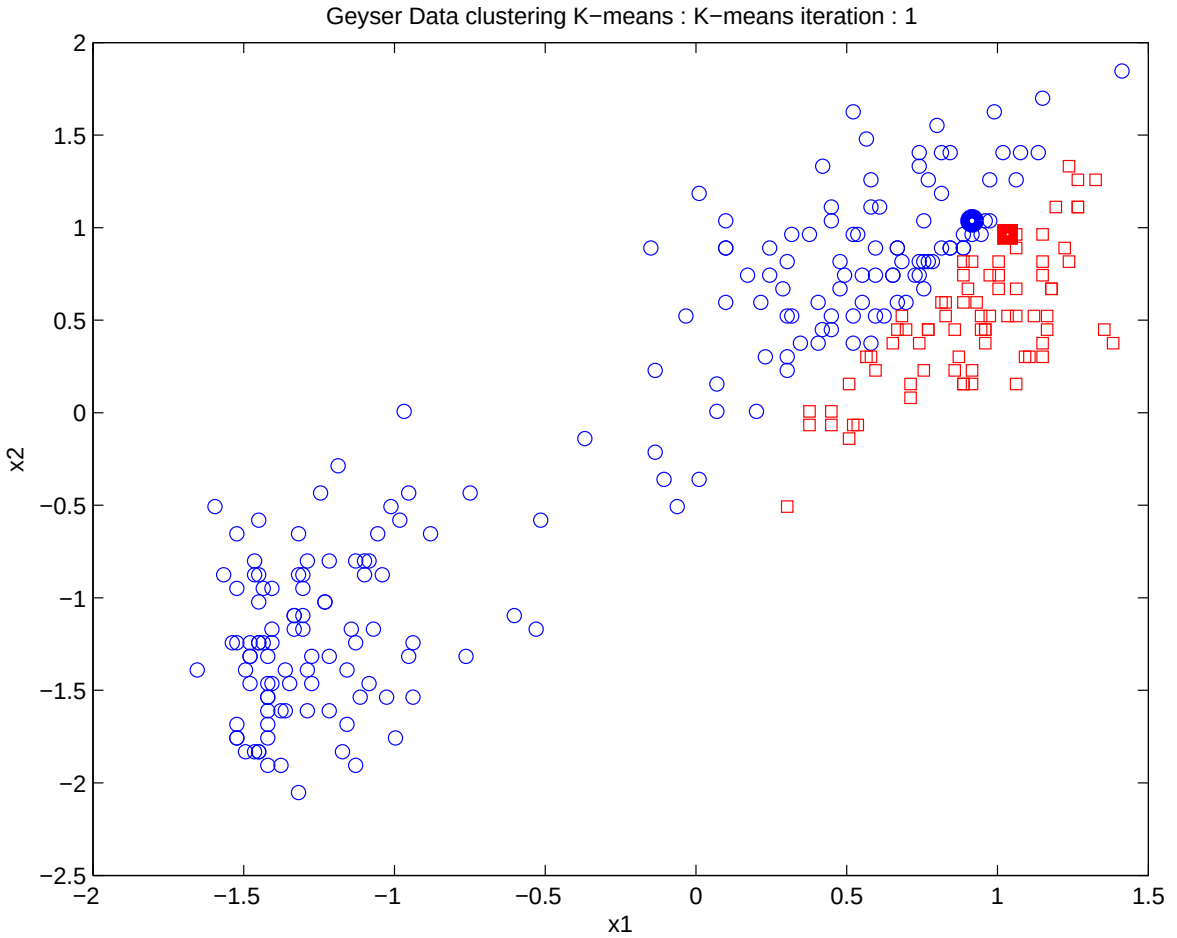
end

Result: les classes $(z_1, \ldots, z_n)$ et les centres des classes $(\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K)$

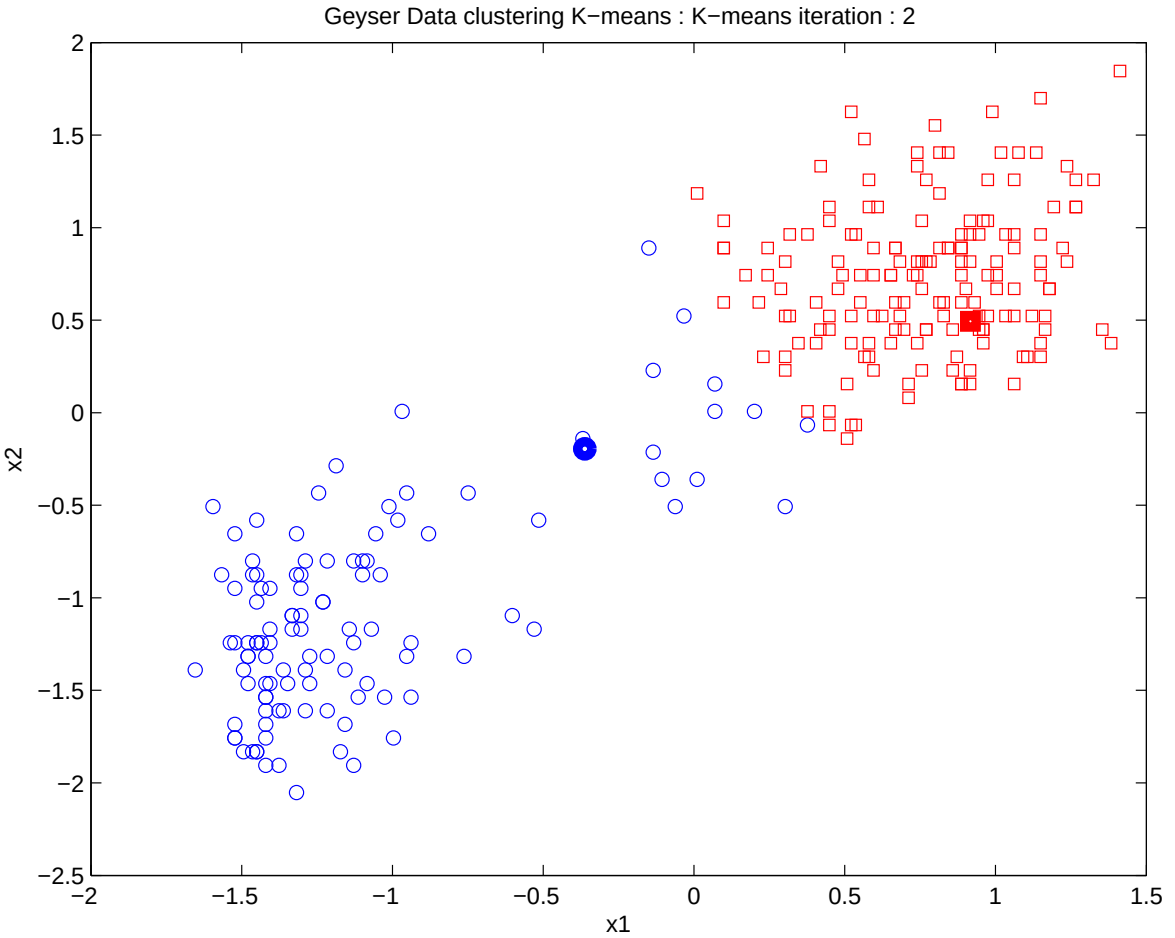
Illustration



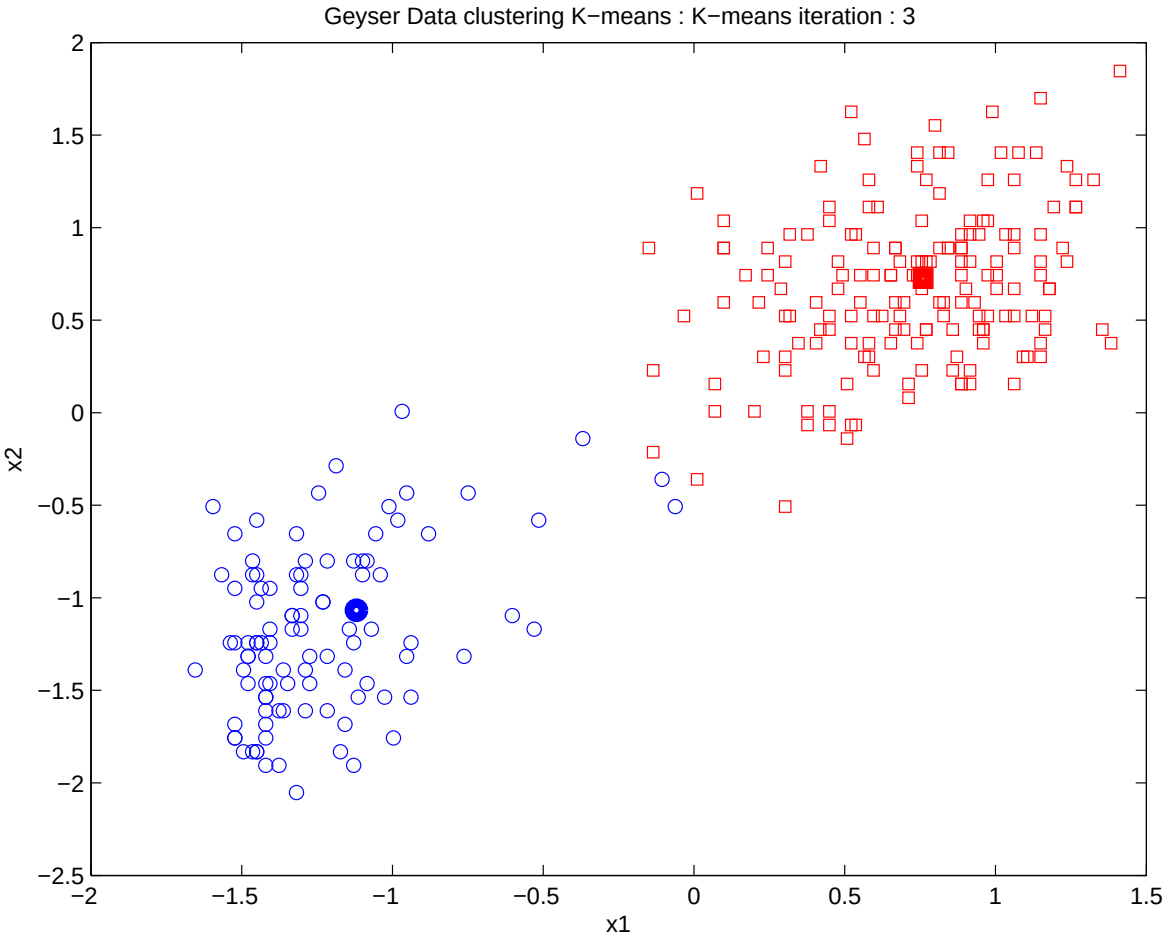
Illustration



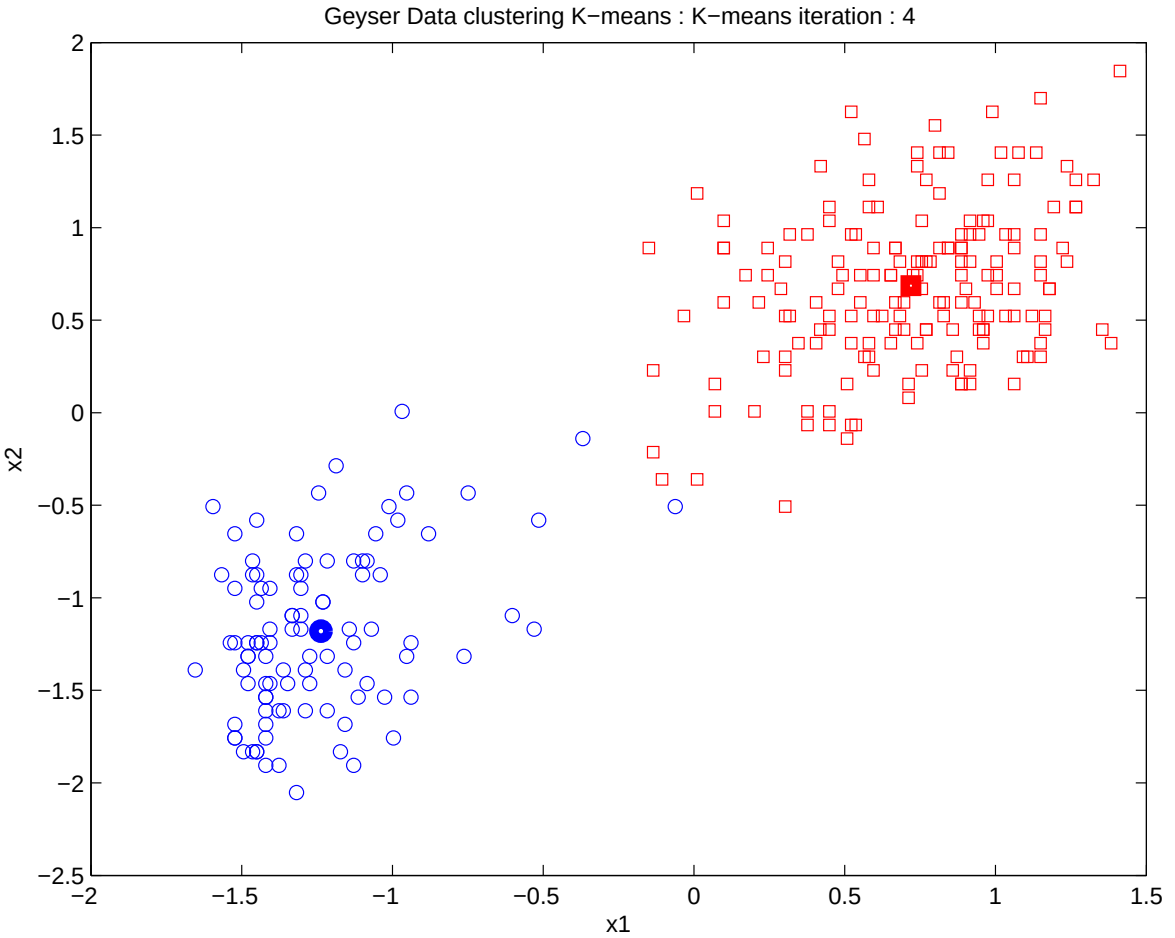
Illustration



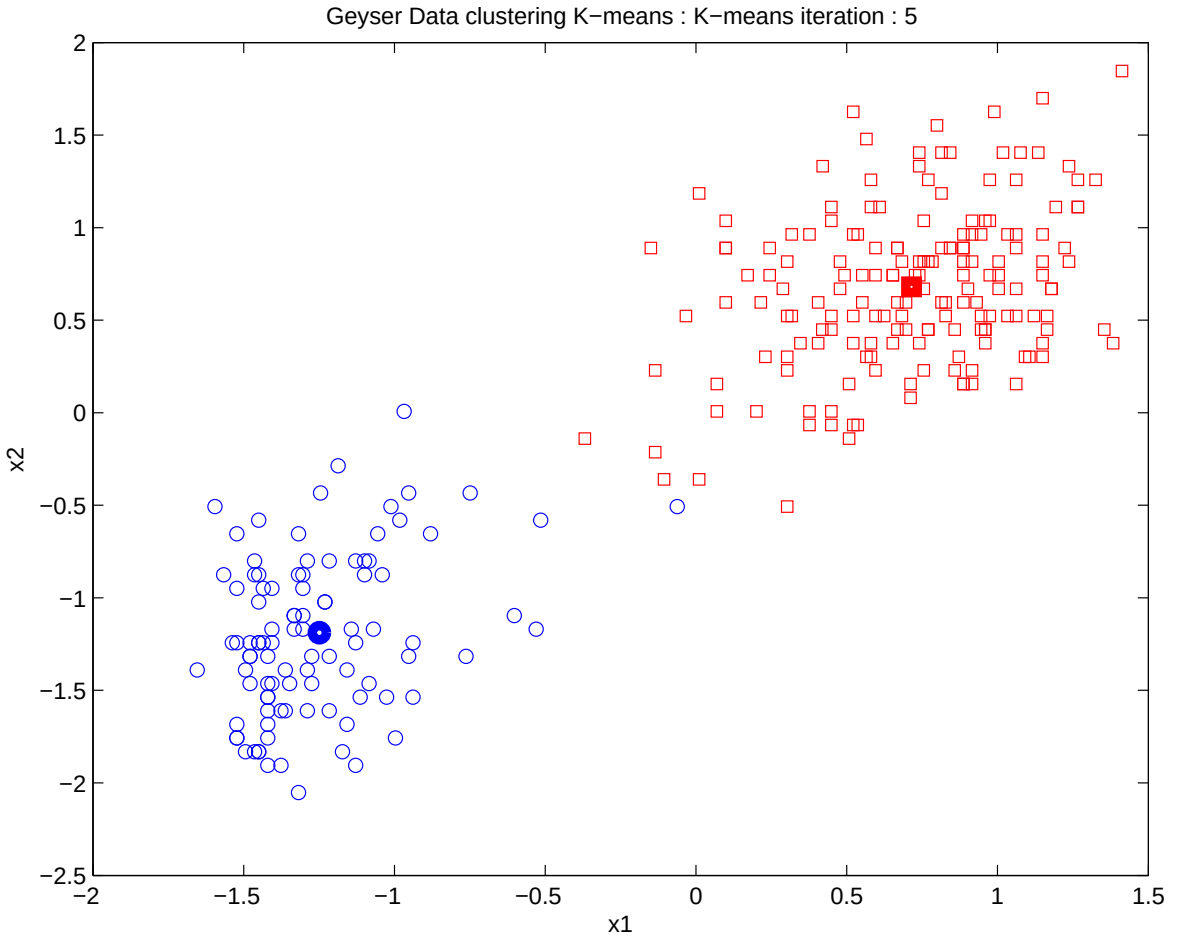
Illustration



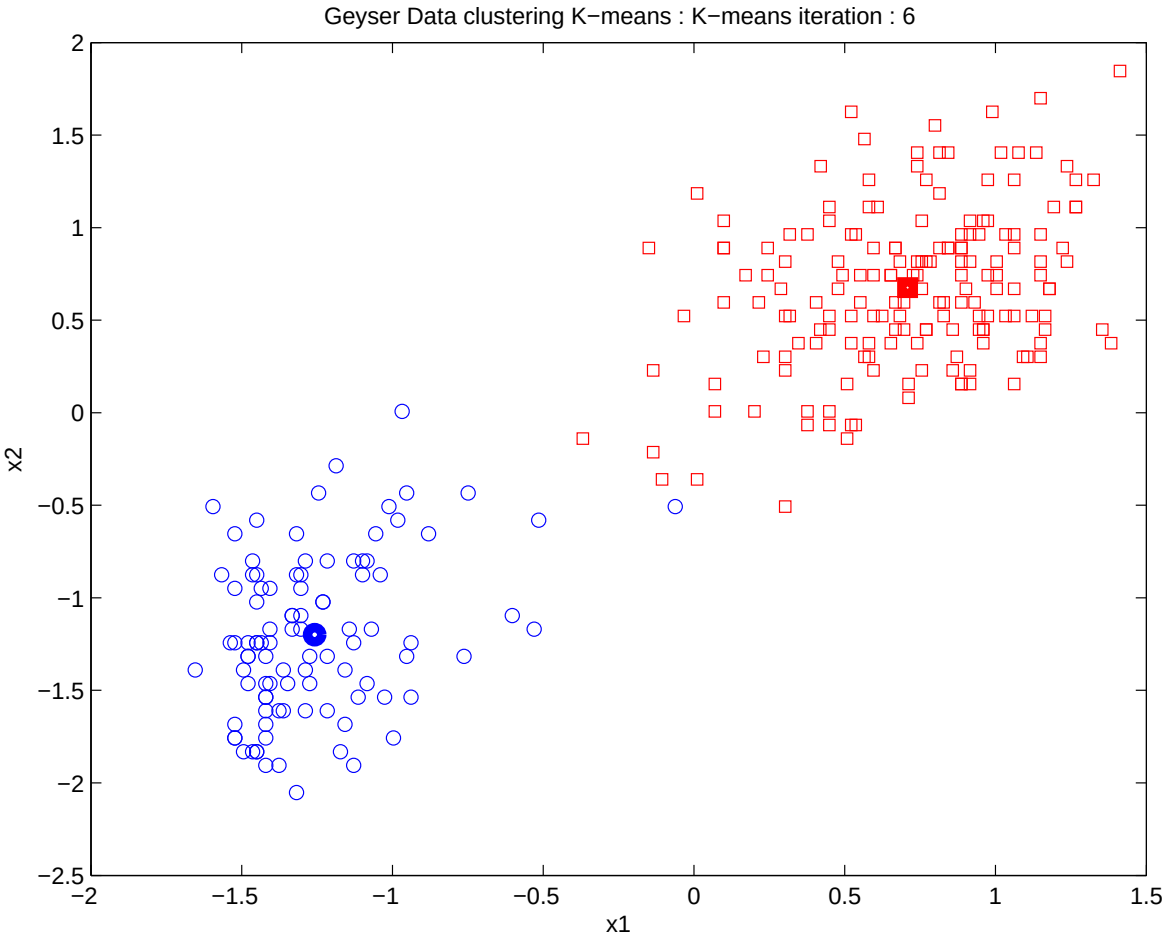
Illustration



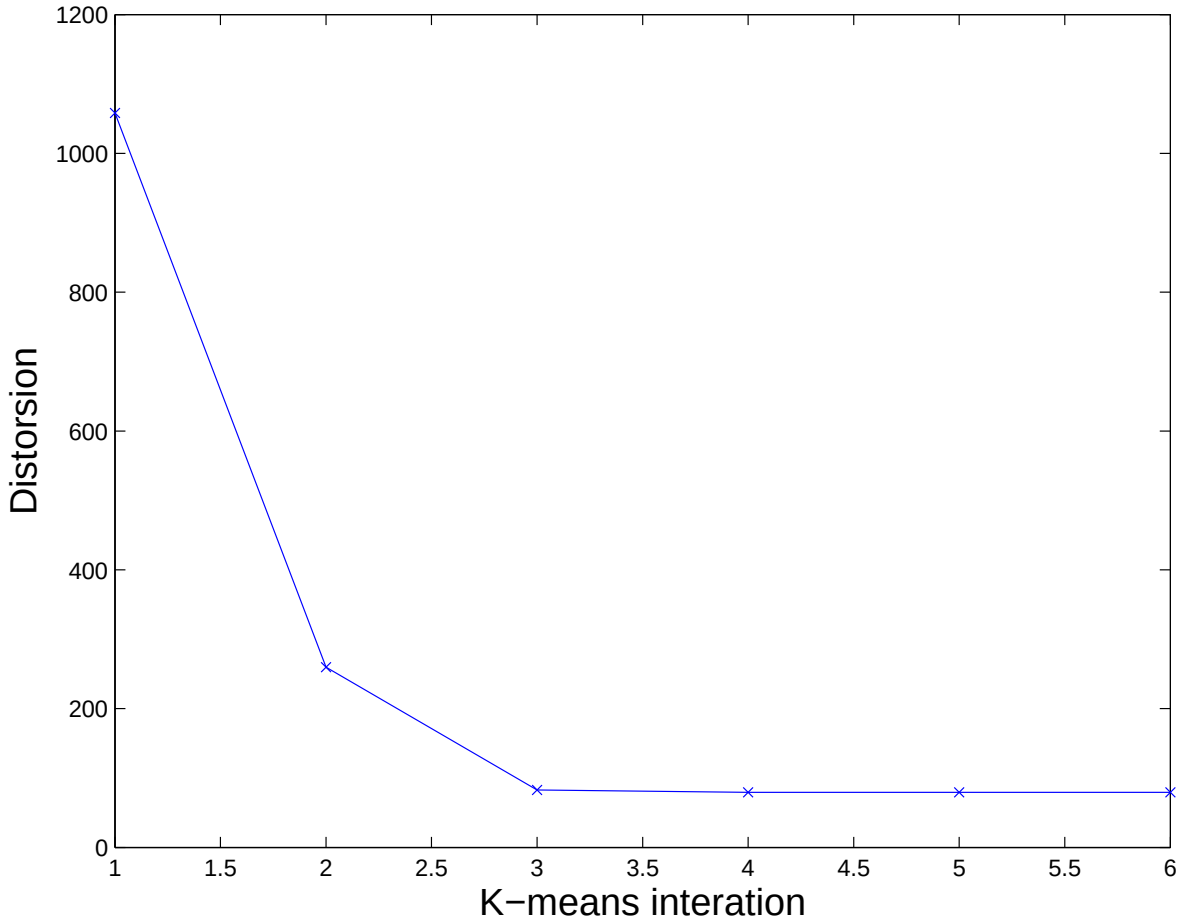
Illustration



Illustration



Illustration



Fuzzy $K-$means

université PARIS-SACLAY

- Easy adaptation of K-means to obtain fuzzy version ofstandard K-means (**?**)
- The minimized criterion is given by :

$$J(\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K) = \sum_{k=1}^{K} \sum_{i=1}^{n} \tau_{ik}^{v} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|^2 \tag{1}$$

$$\text{where } \tau_{ik} \in (0,1) \text{ with } \sum_{k=1}^{K} \tau_{ik} = 1 \text{ and } 1 \leq v < \infty \tag{2}$$

  τ_{ik} denotes the fuzzy cluster membership for the ith data point and v is a constant fixed by the user (typically $v = 2$) that determines the degree of fuzziness (degree of overlap between groups). Standard K-means arises when $v = 1$.
- start with an initial solution $(\boldsymbol{\mu}_1^{(0)}, \ldots, \boldsymbol{\mu}_K^{(0)})$
- **Step 1 :** compute the fuzzy memberships : $\tau_{ik}^{v(q)} = \left(\sum_{\ell=1}^{K} \left(\frac{\|\mathbf{x}_i - \boldsymbol{\mu}_k^{(q)}\|}{\|\mathbf{x}_i - \boldsymbol{\mu}_\ell^{(q)}\|} \right)^{\frac{2}{v-1}} \right)^{-1}$.
- **Step 2 :** Relocation step : Each cluster representative is relocated to the weighted mean with weight the $\tau_{ik}^{v(q)}$'s : $\boldsymbol{\mu}_k^{(q+1)} = \frac{\sum_{i=1}^{n} \tau_{ik}^{v(q)} \mathbf{x}_i}{\sum_{i=1}^{n} \tau_{ik}^{v(q)}}$.

Clustering via finite mixture models

université PARIS-SACLAY

- In the previous section we saw the main common partition-based clustering algorithm, that is K-means.
- Now we describe general clustering methods based on finite mixture models.
- $\Rightarrow$ This approach is known as the *model-based clustering*
- The clustering problem is reformulated as a density estimation problem
- the data probability density function is assumed to be a mixture density, each component density being associated with a cluster.
- $\Rightarrow$ The problem of clustering becomes the one of estimating the parameters of the assumed mixture model (e.g, estimating the means and the covariances for Gaussian mixtures).

Mixture models

université PARIS-SACLAY

- Finite mixture models are an example of latent variable models
- widely used in probabilistic machine learning and pattern recognition.
- very useful to model heterogeneous classes since they assume that each class is composed of sub-classes.
- The finite mixture model decomposes the density of $\mathbf{x}$ into a weighted linear combination of K component densities.
- The mixture model allows for placing K component densities in the input space to approximate the true density.

  $\Rightarrow$ Mixtures provide a natural generalization of the simple parametric density model which is global, to a weighted sum of these models, allowing local adaptation to the density of the data in the input space.

Model definition

université PARIS-SACLAY

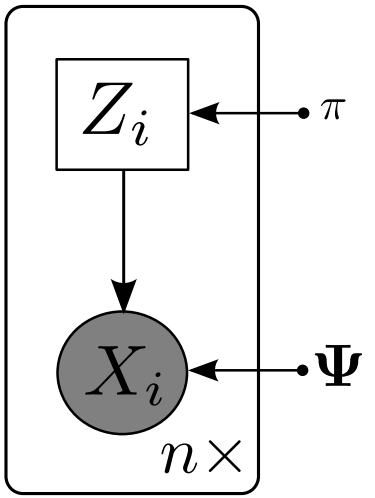
- Let z represent a discrete random variable (binomial or multinomial) which takes its values in the finite set $\mathcal{Z} = \{1, \ldots, K\}$.
- In a general setting, the mixture density of $\mathbf{x}$ is

$$\begin{aligned} f(\mathbf{x}; \mathbf{\Psi}) &= \sum_{k=1}^{K} p(z = k) f(\mathbf{x} | z = k; \mathbf{\Psi}_k) \\ &= \sum_{k=1}^{K} \pi_k f_k(\mathbf{x}; \mathbf{\Psi}_k), \end{aligned}$$

  - $\pi_k = p(z = k)$: the probability that a randomly chosen data point was generated by component k. Referred to as *mixing proportions* $\pi_k \geq 0 \ \forall k$, and $\sum_{k=1}^{K} \pi_k = 1$.
  - $f_1, \ldots, f_K$ are the *component densities*.
  - Each f_k typically consists of a relatively simple parametric model $p(\mathbf{x}|z = k; \mathbf{\Psi}_k)$ (such as a Gaussian distribution with parameters $\mathbf{\Psi}_k = (\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$).

Model definition

FIGURE – Graphical representation of a mixture model.



Parameter estimation for the mixture model

université PARIS-SACLAY

The common parameter estimation methods for mixture models :

- the *maximum likelihood*
- the *Bayesian* methods (Maximum A Posteriori (MAP)) where a prior distribution is assumed for the model parameters

$\Rightarrow$ In this course, we focus on the maximum likelihood framework.

- maximize the observed-data likelihood as a function of the parameters $\boldsymbol{\Psi} = (\pi_1, \ldots, \pi_K, \boldsymbol{\Psi}_k, \ldots, \boldsymbol{\Psi}_K)$, over the parameter space $\boldsymbol{\Omega}$
- The optimization algorithm is the Expectation-Maximization (EM) algorithm

Parameter estimation for the mixture model

université PARIS-SACLAY

- Assume we have an i.i.d sample $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$.
- The observed-data log-likelihood of $\mathbf{\Psi}$ given $\mathbf{X}$ is given by :

$$\begin{aligned} \mathcal{L}(\mathbf{\Psi}; \mathbf{X}) &= \log \prod_{i=1}^{n} p(\mathbf{x}_i; \mathbf{\Psi}) \\ &= \sum_{i=1}^{n} \log \sum_{k=1}^{K} \pi_k f_k(\mathbf{x}_i; \mathbf{\Psi}_k). \end{aligned}$$

- the log-likelihood to be maximized results in a nonlinear function due to the logarithm of the sum
- very difficult to maximize it in a closed form

  $\Rightarrow$ maximize it (locally) using iterative procedures such as gradient ascent, a Newton Raphson procedure or the Expectation-Maximization (EM) algorithm

  $\Rightarrow$ We will focus on the EM algorithm which is widely used and particularly adapted for mixture models.

EM algorithm

université PARIS-SACLAY

- a broadly applicable approach to the iterative computation of maximum likelihood estimates in the framework of latent data models.
- In particular, the EM algorithm simplifies considerably the problem of fitting finite mixture models by maximum likelihood.

- an iterative algorithm where each iteration consists of two steps :
  1. the Expectation step (E-step) : computes the expectation of the complete-data log-likelihood, given the observations $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ and a current value $\boldsymbol{\Psi}^{(q)}$ of the model parameter
  2. the Maximization step (M-step) : Maximize the expected complete-data log-likelihood over the parameter space

EM algorithm

université PARIS-SACLAY

- let $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ be a set of n i.i.d observations with $\mathbf{x}_i \in \mathbb{R}^d$
- $\mathbf{z} = (z_1, \ldots, z_n)$ denote the corresponding unobserved (missing) labels with $z_i \in \mathcal{Z} = \{1, \ldots, K\}$.
- The complete-data : $(\mathbf{X}, \mathbf{z}) = ((\mathbf{x}_1, z_1), \ldots, (\mathbf{x}_n, z_n))$
- The complete-data log-likelihood :

$$\begin{aligned}
\mathcal{L}_c(\mathbf{\Psi}; \mathbf{X}, \mathbf{z}) &= \log p((\mathbf{x}_1, z_1), \ldots, (\mathbf{x}_n, z_n); \mathbf{\Psi}) = \log \prod_{i=1}^{n} p(\mathbf{x}_i, z_i; \mathbf{\Psi}) \\
&= \sum_{i=1}^{n} \log \prod_{k=1}^{K} \Big[p(z_i = k) p(\mathbf{x} | z_i = k; \mathbf{\Psi}_k) \Big]^{z_{ik}} \\
&= \sum_{i=1}^{n} \sum_{k=1}^{K} z_{ik} \log \pi_k f_k \big(\mathbf{x}_i; \mathbf{\Psi}_k \big),
\end{aligned}$$

  where $z_{ik} = 1$ if $z_i = k$ (i.e, when $\mathbf{x}_i$ is generated by the kth component density) and $z_{ik} = 0$ otherwise.
- this log-likelihood depends on the unobservable data $\mathbf{z}$!.

EM algorithm

université PARIS-SACLAY

- The EM algorithm starts with an initial parameter $\mathbf{\Psi}^{(0)}$ and iteratively alternates between the two following steps until convergence :
- **E-step (Expectation) :** computes the expectation of the complete-data log-likelihood given the observations $\mathbf{X}$ and the current value $\mathbf{\Psi}^{(q)}$ of the parameter $\mathbf{\Psi}$ (q being the current iteration).

$$
\begin{aligned}
Q(\mathbf{\Psi}, \mathbf{\Psi}^{(q)}) &= \mathbb{E}\left[\mathcal{L}_c(\mathbf{\Psi}; \mathbf{X}, \mathbf{z}) | \mathbf{X}; \mathbf{\Psi}^{(q)}\right] \\
&= \sum_{i=1}^{n}\sum_{k=1}^{K} \mathbb{E}[z_{ik}|\mathbf{x}_i, \mathbf{\Psi}^{(q)}] \log \pi_k f_k\left(\mathbf{x}_i; \mathbf{\Psi}_k\right) \\
&= \sum_{i=1}^{n}\sum_{k=1}^{K} p(z_{ik} = 1|\mathbf{x}_i; \mathbf{\Psi}^{(q)}) \log \pi_k f_k\left(\mathbf{x}_i; \mathbf{\Psi}_k\right) \\
&= \sum_{i=1}^{n}\sum_{k=1}^{K} \tau_{ik}^{(q)} \log \pi_k f_k\left(\mathbf{x}_i; \mathbf{\Psi}_k\right)
\end{aligned}
$$

EM algorithm

université PARIS-SACLAY

- where

$$\tau_{ik}^{(q)} = p(z_i = k|\mathbf{x}_i; \mathbf{\Psi}^{(q)}) = \frac{\pi_k f_k\left(\mathbf{x}_i; \mathbf{\Psi}_k^{(q)}\right)}{\sum_{\ell=1}^{K} \pi_\ell f_\ell\left(\mathbf{x}_i; \mathbf{\Psi}_\ell^{(q)}\right)}$$

  is the posterior probability that $\mathbf{x}_i$ originates from the kth component density.

- In $\mathbb{E}[z_{ik}|\mathbf{x}_i, \mathbf{\Psi}^{(q)}]$, we used the fact that conditional expectations and conditional probabilities are the same for the indicator binary-valued variables z_{ik} : $\mathbb{E}[z_{ik}|\mathbf{x}_i, \mathbf{\Psi}^{(q)}] = p(z_{ik} = 1|\mathbf{x}_i, \mathbf{\Psi}^{(q)})$.

  $\Rightarrow$ From the expression of $Q(\mathbf{\Psi}, \mathbf{\Psi}^{(q)})$, we can see that this step simply requires the computation of the posterior probabilities $\tau_{ik}^{(q)}$.

EM algorithm

université PARIS-SACLAY

M-step (Maximization) : updates the estimate of $\boldsymbol{\Psi}$ by the value $\boldsymbol{\Psi}^{(q+1)}$ of $\boldsymbol{\Psi}$ that maximizes the Q-function $Q(\boldsymbol{\Psi}, \boldsymbol{\Psi}^{(q)})$ with respect to $\boldsymbol{\Psi}$ over the parameter space $\boldsymbol{\Omega}$:

$$\boldsymbol{\Psi}^{(q+1)} = \arg\max_{\boldsymbol{\Psi}\in\boldsymbol{\Omega}} Q(\boldsymbol{\Psi}, \boldsymbol{\Psi}^{(q)}).$$

We can write

$$Q(\boldsymbol{\Psi}, \boldsymbol{\Psi}^{(q)}) = Q_{\pi}(\pi_1, \ldots, \pi_K, \boldsymbol{\Psi}^{(q)}) + \sum_{k=1}^{K} Q_{\boldsymbol{\Psi}_k}(\boldsymbol{\Psi}_k, \boldsymbol{\Psi}^{(q)})$$

where

$$Q_{\pi}(\pi_1, \ldots, \pi_K, \boldsymbol{\Psi}^{(q)}) = \sum_{i=1}^{n}\sum_{k=1}^{K} \tau_{ik}^{(q)} \log \pi_k$$

$$Q_{\boldsymbol{\Psi}_k}(\boldsymbol{\Psi}_k, \boldsymbol{\Psi}^{(q)}) \quad = \quad \sum_{i=1}^{n} \tau_{ik}^{(q)} \log f_k\big(\mathbf{x}_i; \boldsymbol{\Psi}_k\big)$$

M-Step

$\Rightarrow$ the maximization of the function $Q(\boldsymbol{\Psi}; \boldsymbol{\Psi}^{(q)})$ w.r.t $\boldsymbol{\Psi}$ can be performed by separately maximizing Q_π with respect to the mixing proportions $(\pi_1, \ldots, \pi_K)$ and $Q_{\boldsymbol{\Psi}_k}$ with respect to parameters $\boldsymbol{\Psi}_k$ for each of the K components densities.

- The function Q_π is maximized with respect to $(\pi_1, \ldots, \pi_K) \in [0, 1]^K$ subject to the constraint $\sum_k \pi_k = 1$. This maximization is done in a closed using Lagrange multipliers form and leads to

$$\pi_k^{(q+1)} = \frac{\sum_{i=1}^n \tau_{ik}^{(q)}}{n} = \frac{n_k^{(q)}}{n},$$

- $n_k^{(q)}$ can be viewed as the expected cardinal number of the subpopulation k estimated at iteration q.
- The update of $\boldsymbol{\Psi}_k$ depends on the form of the density f_k (e.g., Gaussian)

EM for Gaussian mixture models (GMMs)

The Gaussian mixture model (GMM) :

$$f(\mathbf{x}_i; \mathbf{\Psi}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_k, \mathbf{\Sigma}_k),$$

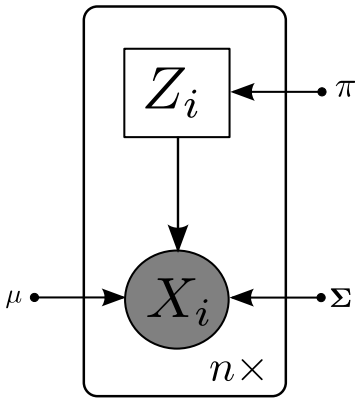


FIGURE – Graphical representation of a Gaussian mixture model.

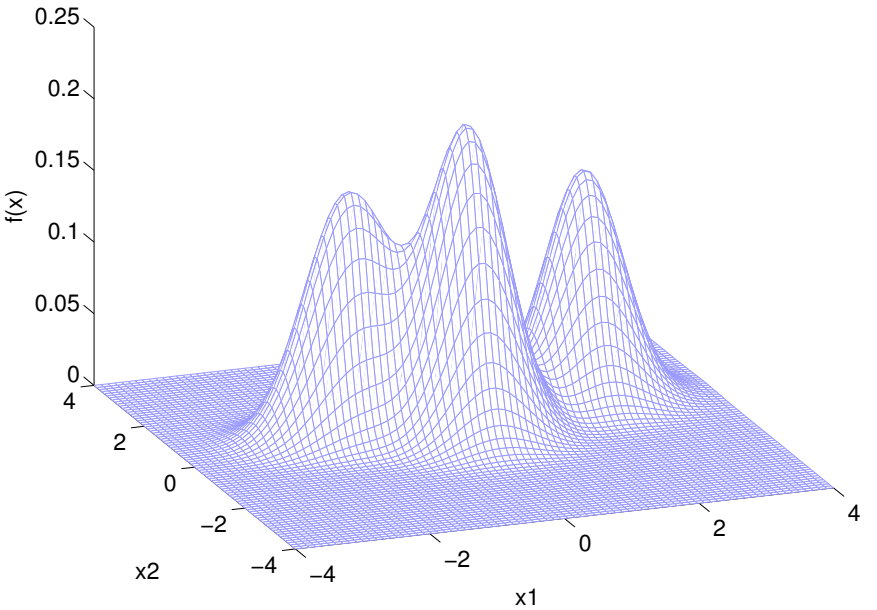


FIGURE – An example of a three-component Gaussian mixture density in $\mathbb{R}^2$.

EM for GMMs

- The observed-data log-likelihood of $\boldsymbol{\Psi}$ for the Gaussian mixture model :

$$\mathcal{L}(\boldsymbol{\Psi};\mathbf{X}) = \sum_{i=1}^{n} \log \sum_{k=1}^{K} \pi_k \mathcal{N}\big(\mathbf{x}_i;\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k\big).$$

- The complete-data log-likelihood of $\boldsymbol{\Psi}$ for the Gaussian mixture model :

$$\mathcal{L}_c(\boldsymbol{\Psi};\mathbf{X},\mathbf{z}) = \sum_{i=1}^{n} \sum_{k=1}^{K} z_{ik} \log \pi_k \mathcal{N}\big(\mathbf{x}_i;\boldsymbol{\mu}_k\boldsymbol{\Sigma}_k\big).$$

EM :

- Starts with an initial parameter $\boldsymbol{\Psi}^{(0)} = (\pi_1^{(0)},\ldots,\pi_K^{(0)},\boldsymbol{\Psi}_1^{(0)},\ldots,\boldsymbol{\Psi}_K^{(0)})$ where $\boldsymbol{\Psi}_k^{(0)} = (\boldsymbol{\mu}_k^{(0)},\boldsymbol{\Sigma}_k^{(0)})$

E-Step for GMMs

- the expected complete-data log-likelihood :

$$
\begin{aligned}
Q(\boldsymbol{\Psi}, \boldsymbol{\Psi}^{(q)}) &= \mathbb{E}\left[\mathcal{L}_c(\boldsymbol{\Psi}; \mathbf{X}, \mathbf{z}) | \mathbf{X}; \boldsymbol{\Psi}^{(q)}\right] \\
&= \sum_{i=1}^{n}\sum_{k=1}^{K} \tau_{ik}^{(q)} \log \pi_k + \sum_{i=1}^{n}\sum_{k=1}^{K} \tau_{ik}^{(q)} \log \mathcal{N}\left(\mathbf{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right).
\end{aligned}
$$

$\Rightarrow$ This step therefore computes the posterior probabilities

$$
\tau_{ik}^{(q)} = p(z_i = k | \mathbf{x}_i, \boldsymbol{\Psi}^{(q)}) = \frac{\pi_k \mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_k^{(q)}, \boldsymbol{\Sigma}_k^{(q)})}{\sum_{\ell=1}^{K} \pi_\ell \mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_\ell^{(q)}, \boldsymbol{\Sigma}_\ell^{(q)})}
$$

that $\mathbf{x}_i$ originates from the kth component density.

M-Step for GMMs

- update the parameter $\boldsymbol{\Psi}$ by the value $\boldsymbol{\Psi}^{(q+1)}$ of $\boldsymbol{\Psi}$ that maximizes the function $Q(\boldsymbol{\Psi}, \boldsymbol{\Psi}^{(q)})$ w.r.t $\boldsymbol{\Psi}$ over the parameter space $\boldsymbol{\Omega}$.

$$\begin{aligned}
\boldsymbol{\mu}_k^{(q+1)} &= \frac{1}{n_k^{(q)}} \sum_{i=1}^{n} \tau_{ik}^{(q)} \mathbf{x}_i, \\
\boldsymbol{\Sigma}_k^{(q+1)} &= \frac{1}{n_k^{(q)}} \sum_{i=1}^{n} \tau_{ik}^{(q)} (\mathbf{x}_i - \boldsymbol{\mu}^{(q+1)})(\mathbf{x}_i - \boldsymbol{\mu}^{(q+1)})^T.
\end{aligned}$$

- The E- and M-steps are alternated iteratively until the change in the log likelihood value are less than some specified threshold.

Algorithm 2 Pseudo code of the EM algorithm for GMMs.

Inputs : a data set $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ and the number of clusters K

fix a threshold $\epsilon > 0$; set $q \leftarrow 0$ (iteration)

Initialize : $\boldsymbol{\Psi}^{(0)} = (\pi_1^{(0)}, \ldots, \pi_K^{(0)}, \boldsymbol{\Psi}_1^{(0)}, \ldots, \boldsymbol{\Psi}_K^{(0)})$ with $\boldsymbol{\Psi}_k^{(0)} = (\boldsymbol{\mu}_K^{(0)}, \boldsymbol{\Sigma}_K^{(0)})$

while increment in log-likelihood $> \epsilon$ **do**

E-step :

for $k = 1, \ldots, K$ **do**

Compute $\tau_{ik}^{(q)} = \frac{\pi_k \mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_k^{(q)}, \boldsymbol{\Sigma}_k^{(q)})}{\sum_{\ell=1}^{K} \pi_\ell \mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_\ell^{(q)}, \boldsymbol{\Sigma}_\ell^{(q)})}$ for $i = 1, \ldots, n$

end for

M-step :

for $k = 1, \ldots, K$ **do**

Compute $\pi_k^{(q+1)} = \frac{\sum_{i=1}^{n} \tau_{ik}^{(q)}}{n}$

Compute $\boldsymbol{\mu}_k^{(q+1)} = \frac{1}{n_k^{(q)}} \sum_{i=1}^{n} \tau_{ik}^{(q)} \mathbf{x}_i$

Compute $\boldsymbol{\Sigma}_k^{(q+1)} = \frac{1}{n_k^{(q)}} \sum_{i=1}^{n} \tau_{ik}^{(q)} (\mathbf{x}_i - \boldsymbol{\mu}^{(q+1)})(\mathbf{x}_i - \boldsymbol{\mu}^{(q+1)})^T$

end for

$q \leftarrow q + 1$

end while

for EM

- The initialization of EM is a crucial point since it maximizes locally the log-likelihood.
- if the initial value is inappropriately selected, the EM algorithm may lead to an unsatisfactory estimation.
- The most used strategy : use several EM tries and select the solution maximizing the log-likelihood among those runs.
- For each run of EM, one can initialize it
  - randomly
  - by Computing a parameter estimate from another clustering algorithm such as K-means, Classification EM, Stochastic EM ...
  - with a few number of steps of EM itself.
- Stop EM when the relative increase of the log-likelihood between two iterations is below a fixed threshold $|\frac{\mathcal{L}^{(q+1)}-\mathcal{L}^{(q)}}{\mathcal{L}^{(q)}}| \leq \epsilon$ or when a predefined number of iterations is reached.

EM properties

université PARIS-SACLAY

- The EM algorithm always monotonically increases the observed-data log-likelihood.
- The sequence of parameter estimates generated by the EM algorithm converges toward at least a local maximum or a stationary value of the incomplete-data likelihood function.
- numerical stability
- simplicity of implementation
- reliable convergence
- In general, both the E- and M-steps will have particularly simple forms when the complete-data probability density function is from the exponential family ;
- Some drawbacks : EM is sometimes very slow to converge especially for high dimensional data ;
  in some problems, the E- or M-step may be analytically intractable (but this can be tackled by using EM extensions)

EM extensions

université PARIS-SACLAY

- The EM variants mainly aim at :
  1. increasing the convergence speed of EM and addressing the optimization problem in the M-step
  2. computing the E-step when it is intractable.
- In the first case, one can speak about deterministic algorithms :
  - e.g., Incremental EM (IEM)
  - Gradient EM
  - Generalized EM (GEM) algorithm
  - Expectation Conditional Maximization (ECM)
  - Expectation Conditional Maximization Either (ECME)
- In the second case, one can speak about stochastic algorithms :
  - e.g., Monte Carlo EM (MCEM)
  - Stochastic EM (SEM)
  - Simulated Annealing EM (SAEM)

Mixture approach/Classification approach

université PARIS-SACLAY

Two main approaches are possible. The former is refereed to as *the mixture approach* or *the estimation approach* and the latter is known as *the classification approach.*

1. **The mixture approach** consists of two steps :
   1. The parameters of the mixture density are estimated by maximizing the *observed-data likelihood* generally via the EM algorithm
   2. After performing the probability density estimation, the posterior probabilities τ_{ik} are then used to determine the cluster memberships through the MAP principle.
2. **The classification approach**
   - consists in optimizing a classification likelihood function which is (can be) the *complete-data likelihood* by using the CEM algorithm **?**.
   - The cluster memberships and the model parameters are estimated simultaneously as the learning proceeds.

Classification EM (CEM) algorithm

université PARIS-SACLAY

- we saw that EM computes the maximum likelihood (ML) estimate of a mixture model.
- The Classification EM (CEM) algorithm **?** estimates both the mixture model parameters and the classes' labels by maximizing the completed-data log-likelihood $\mathcal{L}_c(\mathbf{\Psi}; \mathbf{X}, \mathbf{z}) = \log p(\mathbf{X}, \mathbf{z}; \mathbf{\Psi})$
- start with an initial parameter $\mathbf{\Psi}^{(0)}$

1. **Step 1 :** Compute the missing data $\mathbf{z}^{(q+1)}$ given the observations and the current estimated model parameters $\mathbf{\Psi}^{(q)}$:

$$\mathbf{z}^{(q+1)} = \arg\max_{\mathbf{z} \in \mathcal{Z}^n} \mathcal{L}_c(\mathbf{\Psi}^{(q)}; \mathbf{X}, \mathbf{z})$$

2. **Step 2 :** Compute the model parameters update $\mathbf{\Psi}^{(q+1)}$ by maximizing the complete-data log-likelihood given the current estimation of the missing data $\mathbf{z}^{(q+1)}$:

$$\mathbf{\Psi}^{(q+1)} = \arg\max_{\mathbf{\Psi} \in \mathbf{\Omega}} \mathcal{L}_c(\mathbf{\Psi}; \mathbf{X}, \mathbf{z}^{(q+1)}).$$

CEM for GMMs

université PARIS-SACLAY

- the CEM algorithm, for the case of mixture models, is equivalent to integrating a classification step (C-step) between the E- and the M-steps of the EM algorithm.
- The C-step assigns the observations to the component densities by using the MAP rule :
  1. **E-step :** Compute the conditional posterior probabilities $\tau_{ik}^{(q)}$ that the observation $\mathbf{x}_i$ arises from the kth component density.
  2. **C-step :** Assign each observation $\mathbf{x}_i$ to the component maximizing the conditional posterior probability τ_{ik} :

$$z_i^{(q+1)} = \arg\max_{k \in \mathcal{Z}} \tau_{ik}^{(q)} \quad (i = 1, \ldots, n).$$

     $\Rightarrow$ this step provides a hard partition of the data
  3. **M-step :** Update the mixture model parameters by maximizing the completed-data log-likelihood for the partition provided by the C-step.

Algorithm 3 Pseudo code of the CEM algorithm for GMMs.

Inputs : a data set $\mathbf{X}$ and the number of clusters K

fix a threshold $\epsilon > 0$; set $q \leftarrow 0$ (iteration)

Initialize : $\boldsymbol{\Psi}^{(0)} = (\pi_1^{(0)}, \ldots, \pi_K^{(0)}, \boldsymbol{\Psi}_1^{(0)}, \ldots, \boldsymbol{\Psi}_K^{(0)})$ with $\boldsymbol{\Psi}_k^{(0)} = (\boldsymbol{\mu}_K^{(0)}, \boldsymbol{\Sigma}_K^{(0)})$

while increment in the complete-data log-likelihood $> \epsilon$ **do**

E-step :

for $k = 1, \ldots, K$ **do**

Compute $\tau_{ik}^{(q)} == \frac{\pi_k \mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_k^{(q)}, \boldsymbol{\Sigma}_k^{(q)})}{\sum_{\ell=1}^{K} \pi_\ell \mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_\ell^{(q)}, \boldsymbol{\Sigma}_\ell^{(q)})}$

end for

C-step :

for $k = 1, \ldots, K$ **do**

Compute $z_i^{(q)} = \arg\max_{k \in \mathcal{Z}} \tau_{ik}^{(q)}$ for $i = 1, \ldots, n$

Set $z_{ik}^{(q)} = 1$ if $z_i^{(q)} = k$ and $z_{ik}^{(q)} = 0$ otherwise, for $i = 1, \ldots, n$

end for

M-step :

for $k = 1, \ldots, K$ **do**

Compute $\pi_k^{(q+1)} = \frac{\sum_{i=1}^{n} z_{ik}^{(q)}}{n}$

Compute $\boldsymbol{\mu}_k^{(q+1)} = \frac{1}{n_k^{(q)}} \sum_{i=1}^{n} z_{ik}^{(q)} \mathbf{x}_i$

Compute $\boldsymbol{\Sigma}_k^{(q+1)} = \frac{1}{n_k^{(q)}} \sum_{i=1}^{n} z_{ik}^{(q)} (\mathbf{x}_i - \boldsymbol{\mu}^{(q+1)})(\mathbf{x}_i - \boldsymbol{\mu}^{(q+1)})^T$

end for

$q \leftarrow q + 1$

end while

Output : $\widehat{\boldsymbol{\Psi}} = \boldsymbol{\Psi}^{(q)}$; $\widehat{z}_i = z_i^{(q)}$ $(i = 1, \ldots, n)$

CEM algorithm

université PARIS-SACLAY

- CEM is easy to implement, typically faster to converge than EM and monotonically improves the complete-data log-likelihood as the learning proceeds.
- converges toward a local maximum of the complete-data log-likelihood
- ! CEM provides biased estimates of the mixture model parameters. Indeed, CEM updates the model parameters from a truncated sample contrary to EM for which the model parameters are updated from the whole data through the fuzzy posterior probabilities and therefore the parameter estimations provided by EM are more accurate.
- **link with K-means** :
  - It can be shown that CEM which is formulated in a probabilistic framework, generalizes K-means
  - From a probabilistic point of view, K-means is equivalent to a particular case of the CEM algorithm for a mixture of K Gaussian densities with the same proportions $\pi_k = \frac{1}{K}\ \forall k$ and identical isotropic covariance matrices $\mathbf{\Sigma}_k = \sigma^2 \mathbf{I}\ \forall k$.

Parsimonious Gaussian mixtures

université PARIS-SACLAY

- Parsimonious Gaussian mixture models are statistical models that allow for capturing a specific cluster shapes (e.g., clusters having the same shape or different shapes, spherical or elliptical clusters, etc).
- decompositions of the covariance matrices for the Gaussian mixture model :

$$\mathbf{\Sigma}_k = \lambda_k \mathbf{D}_k \mathbf{A}_k \mathbf{D}_k^T$$

where

  - λ_k represents the volume of the kth cluster (the amount of space of the cluster).
  - $\mathbf{D}_k$ is a matrix with columns corresponding to the eigenvectors of $\mathbf{\Sigma}_k$ that determines the orientation of the cluster.
  - $\mathbf{A}_k$ is a diagonal matrix, whose diagonal entries are the normalized eigenvalues of $\mathbf{\Sigma}_k$ arranged in a decreasing order and its determinant is 1. This matrix is associated with the shape of the cluster.

Parsimonious Gaussian mixtures

université PARIS-SACLAY

- This eigenvalue decomposition provides three main families of models : the spherical family, the diagonal family, and the general family
  and produces 14 different models, according to the choice of the configuration for the parameters λ_k, $\mathbf{A}_k$, and $\mathbf{D}_k$
- In addition to providing flexible statistical models for the clusters, parsimonious Gaussian mixture can be viewed as techniques for reducing the number of parameters in the model.
- imposing constraints on the covariance matrices reduces the dimension of the optimization problem.
- The EM algorithms therefore provide more accurate estimations compared to the full mixture model.

Model selection

université PARIS-SACLAY

- The problem of choosing the number of clusters can be seen as a model selection problem.
- The model selection task consists of choosing a suitable compromise between flexibility so that a reasonable fit to the available data is obtained, and over-fitting.
- A common way is to use a criterion (score function) that ensure the compromise.
- In general, we choose an overall score function that is explicitly composed of two components : a component that measures the goodness of fit of the model to the data, and a penalty component that governs the model complexity :

$$\text{score}(\text{model}) = \text{error}(\text{model}) + \text{penalty}(\text{model})$$

which will be minimized.

Model selection

université PARIS-SACLAY

- The complexity of a model $\mathcal{M}$ is related to the number of its (free) parameters ν, the penalty function then involves the number of model parameters.
- Let $\mathcal{M}$ denote a model, $\mathcal{L}(\widehat{\mathbf{\Psi}})$ its log-likelihood and ν the number of its free parameters. Consider that we fitted M different model structures $(\mathcal{M}_1, \ldots, \mathcal{M}_M)$, from which we wish to choose the "best" one (ideally the one providing the best prediction on future data).
- Assume we have estimated the model parameters $\widehat{\mathbf{\Psi}}_m$ for each model structure $\mathcal{M}_m$ $(m = 1, \ldots, M)$ from a sample of n observations $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ and now we wish to choose among these fitted models.

Model selection

université PARIS-SACLAY

- Akaike Information Criterion (AIC) :

$$\text{AIC}(\mathcal{M}_m) = \mathcal{L}(\widehat{\boldsymbol{\Psi}}_m) - \nu_m$$

- Bayesian Information Criterion (BIC) :

$$\text{BIC}(\mathcal{M}_m) = \mathcal{L}(\widehat{\boldsymbol{\Psi}}_m) - \frac{\nu_m \log(n)}{2}$$

- Integrated Classification Likelihood (ICL) :

$$\text{ICL}(\mathcal{M}_m) = \mathcal{L}_c(\widehat{\boldsymbol{\Psi}}_m) - \frac{\nu_m \log(n)}{2}$$

where $\mathcal{L}_c(\widehat{\boldsymbol{\Psi}}_m)$ is the complete-data log-likelihood for the model $\mathcal{M}_m$ and ν_m denotes the number of free model parameters. For example, in the case of a d-dimensional Gaussian mixture model we have :

$$\nu = \underbrace{(K-1)}_{\pi_k\text{'s}} + \underbrace{K \times d)}_{\{\boldsymbol{\mu}_k\}} + \underbrace{K \times \frac{d \times (d+1)}{2}}_{\{\boldsymbol{\Sigma}_k\}} = \frac{K \times (d+1) \times (d+2)}{2} - 1.$$

Examples

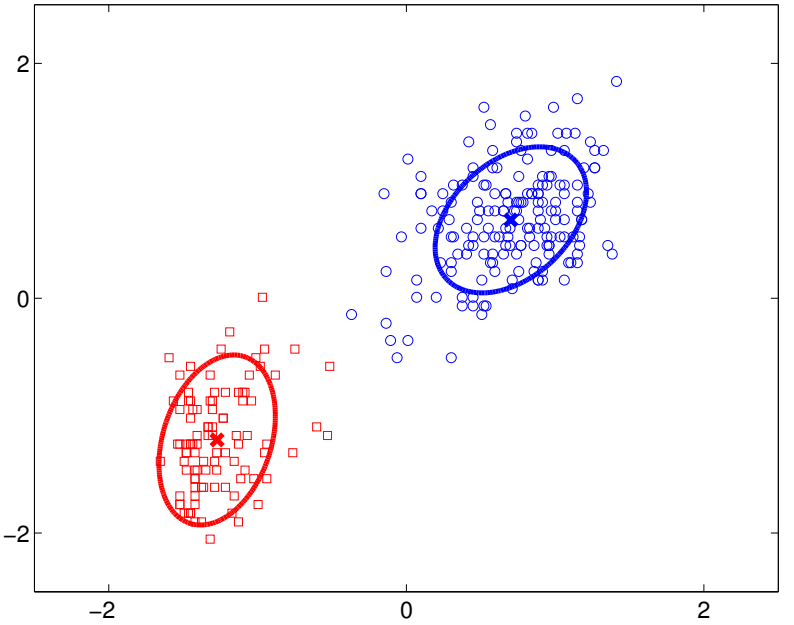


FIGURE – Clustering results obtained with K-means algorithm (left) with $K = 2$ and the EM algorithm (right). The cluster centers are shown by the red and blue crosses and the ellipses are the contours of the Gaussian component densities at level 0.4 estimated by EM. The number of clusters for EM have been chosen by BIC for $K = 1, \ldots, 4$.

Examples

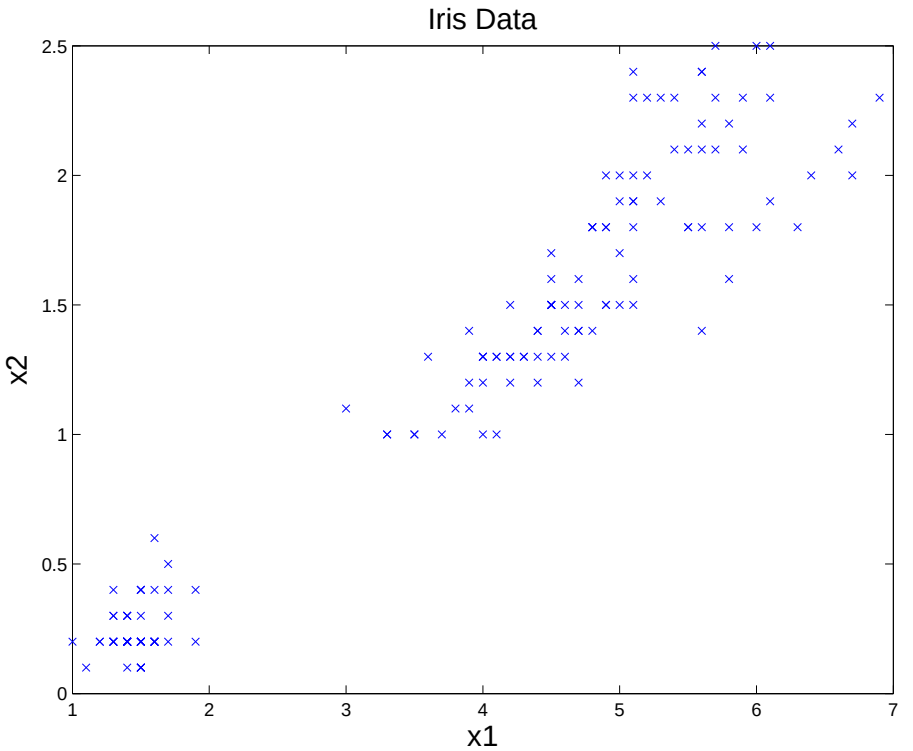


FIGURE – A three-class example of a real data set : Iris data of Fisher.

Examples

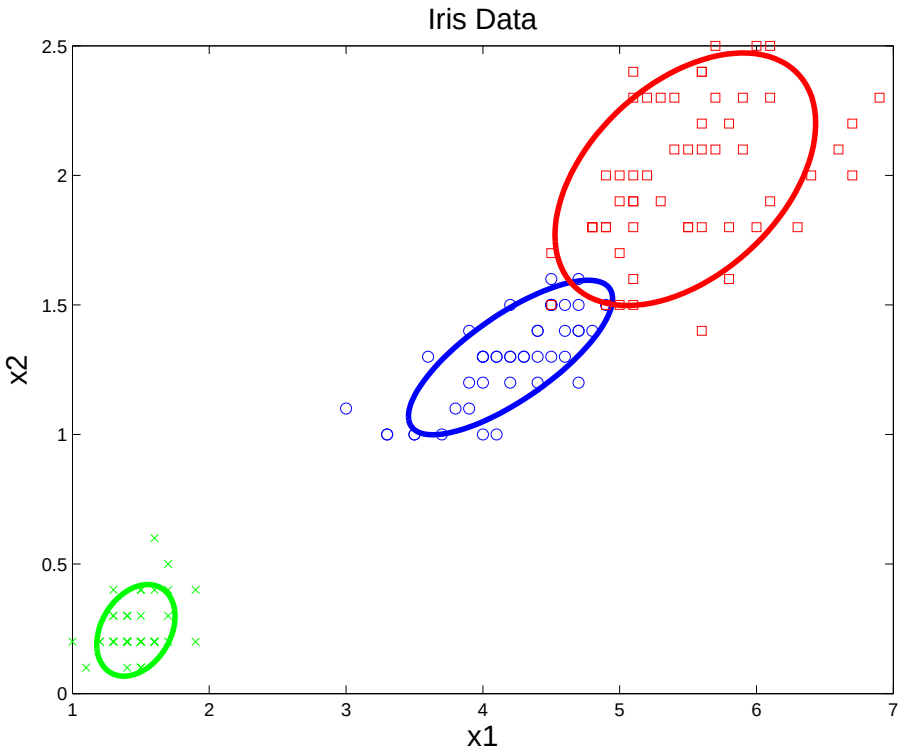


FIGURE – Iris data : Clustering results with EM for a GMM and AIC.

Examples

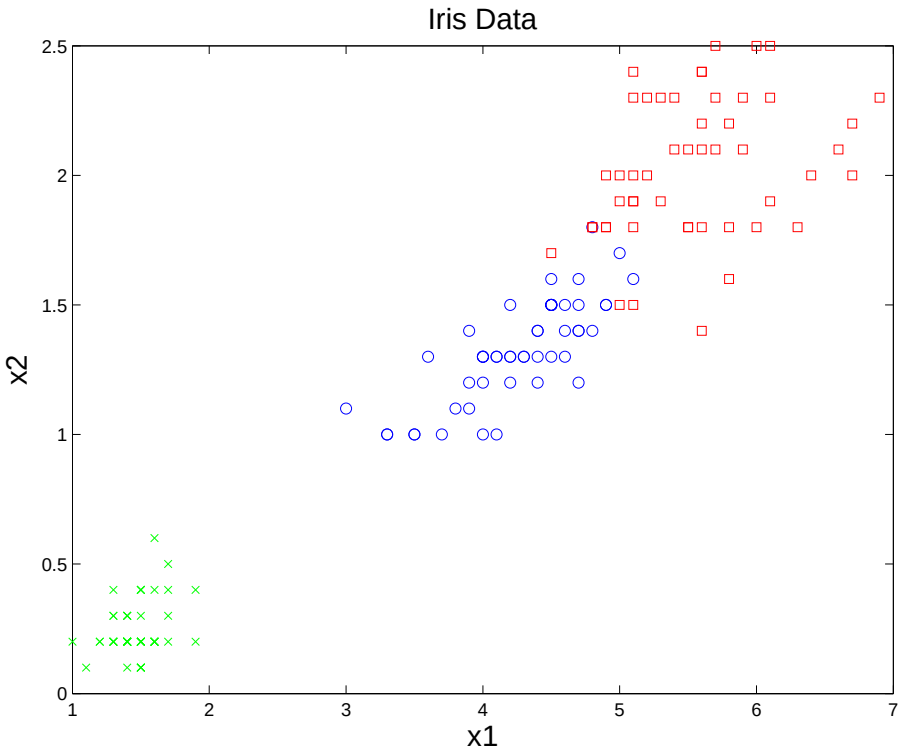
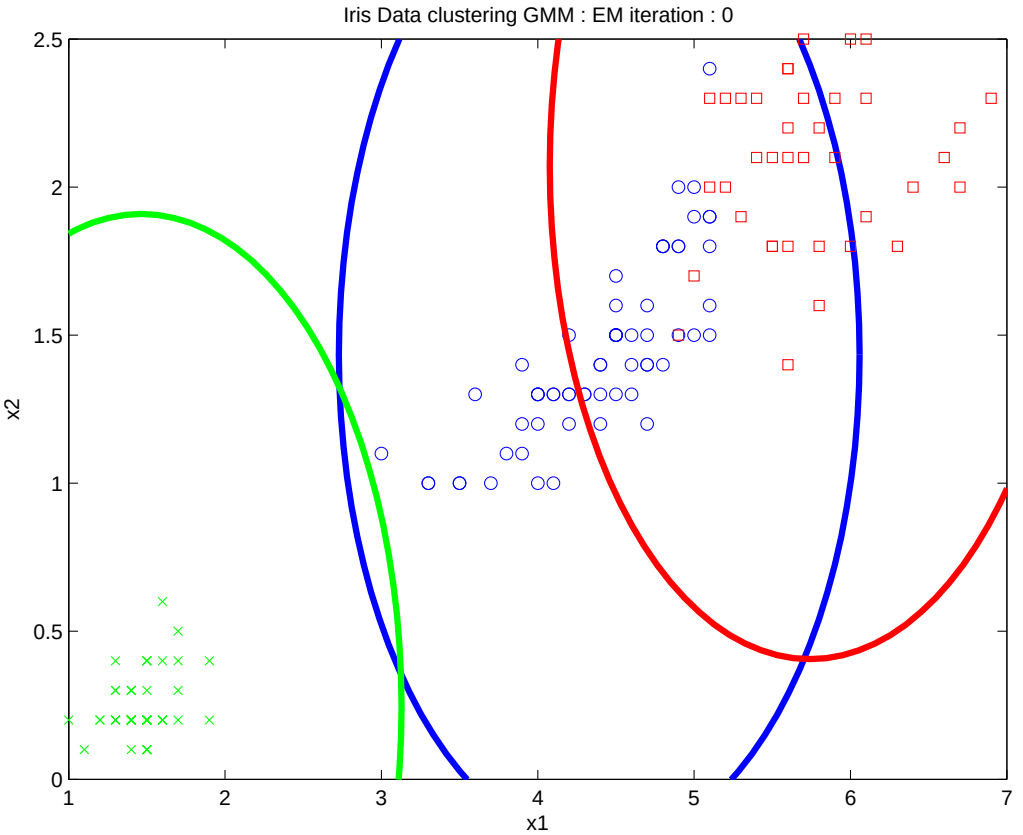
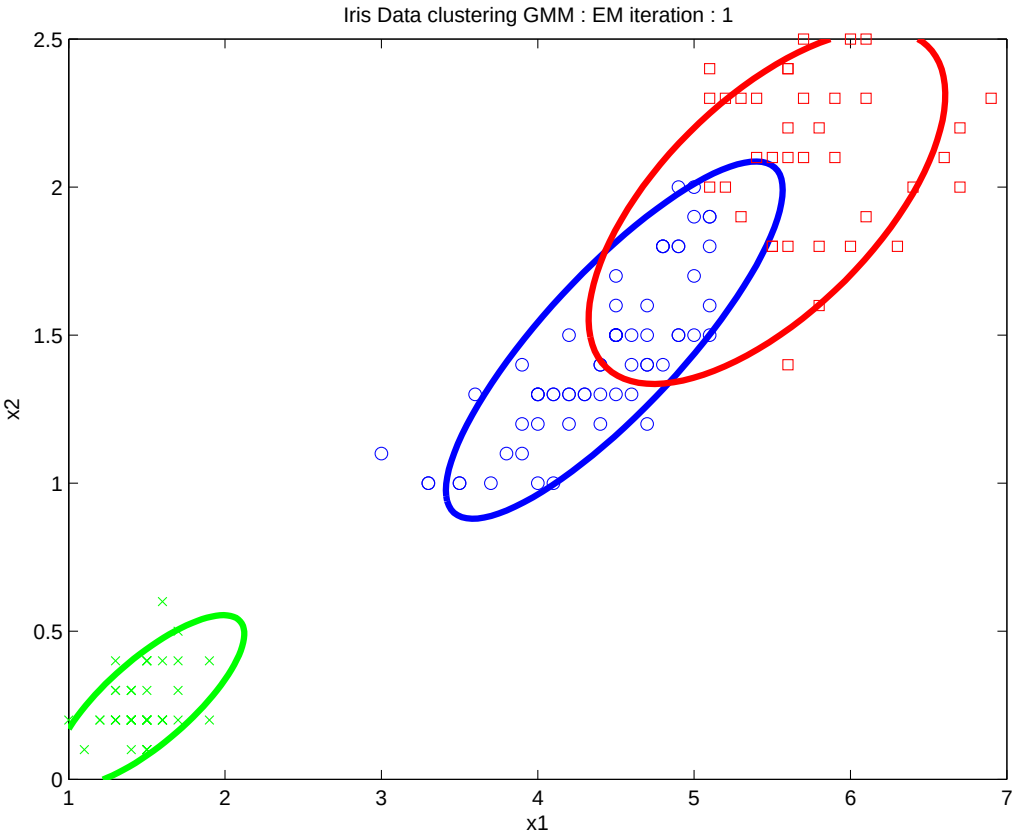


FIGURE – Iris data of Fisher : The data are colored according to the true partition.

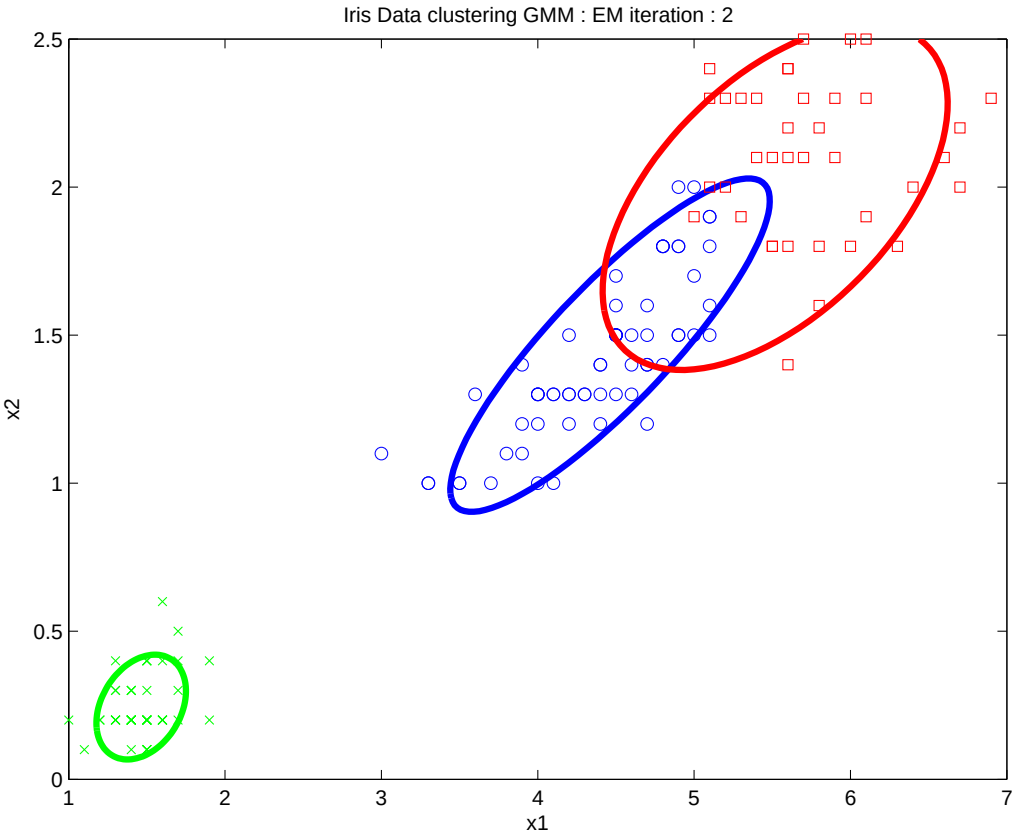
Examples


Iris Data clustering GMM : EM iteration : 0
x2
x1
2.5
2
1.5
1
0.5
0
1
2
3
4
5
6
7

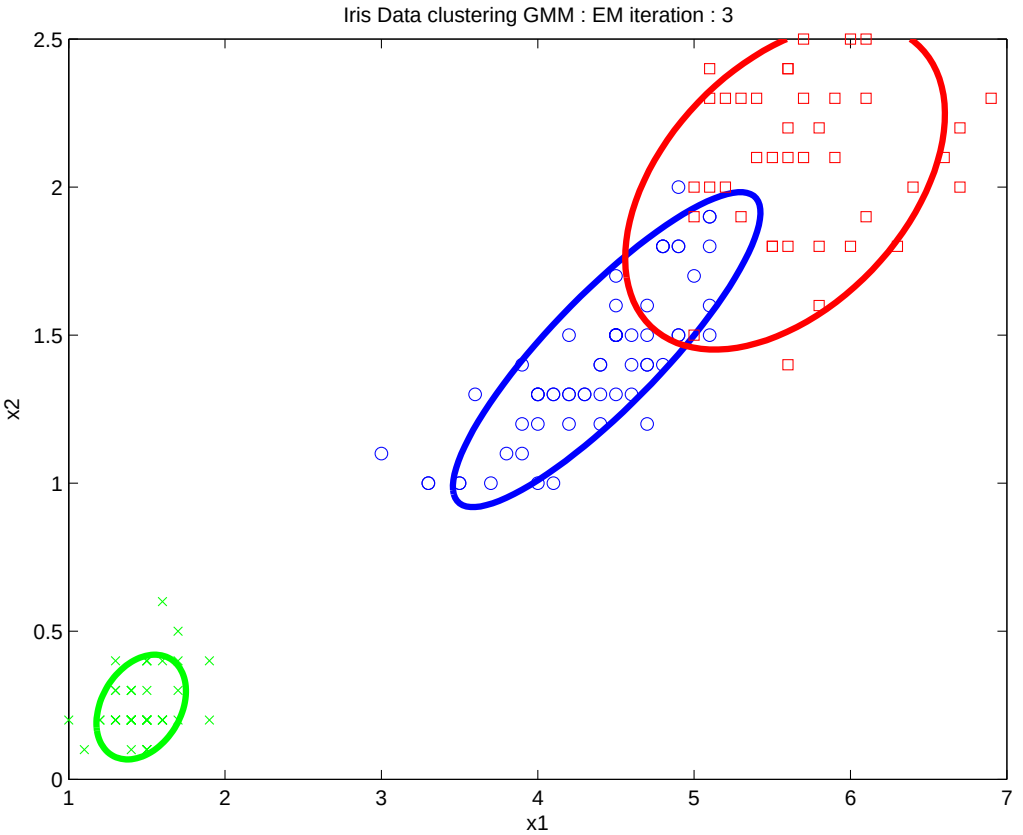
Examples


Iris Data clustering GMM : EM iteration : 1
x2
x1
2.5
2
1.5
1
0.5
0
1
2
3
4
5
6
7

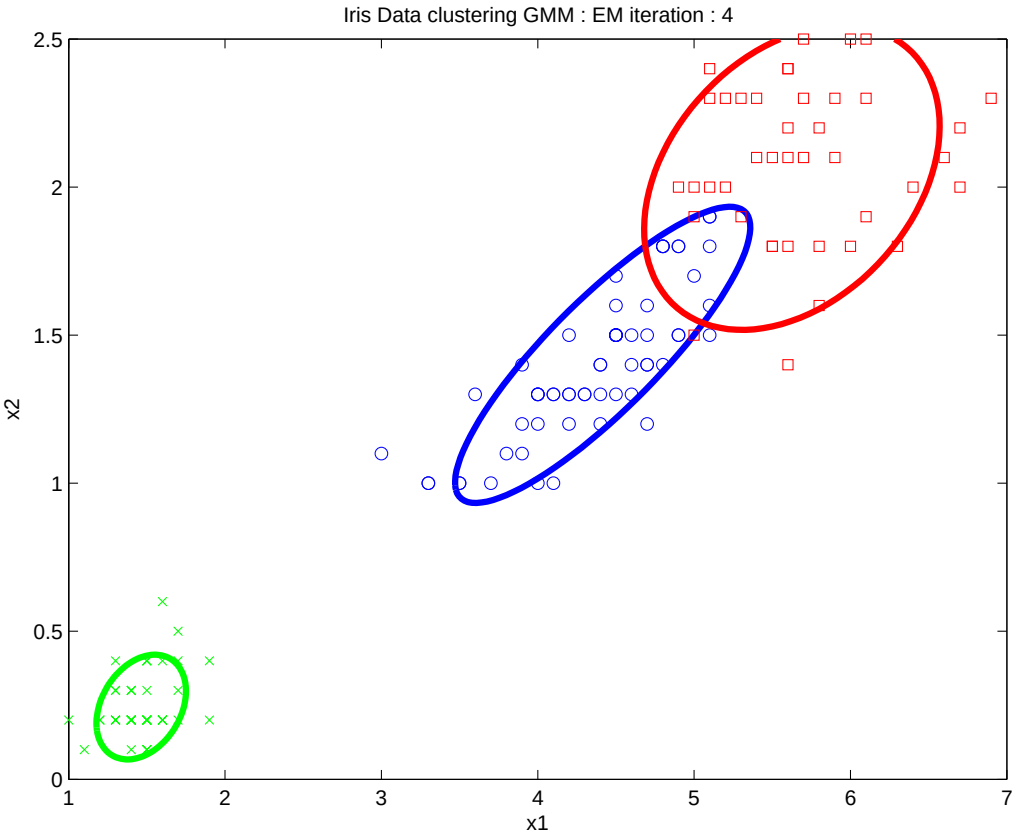
Examples


Iris Data clustering GMM : EM iteration : 2
x2
x1
2.5
2
1.5
1
0.5
0
1
2
3
4
5
6
7

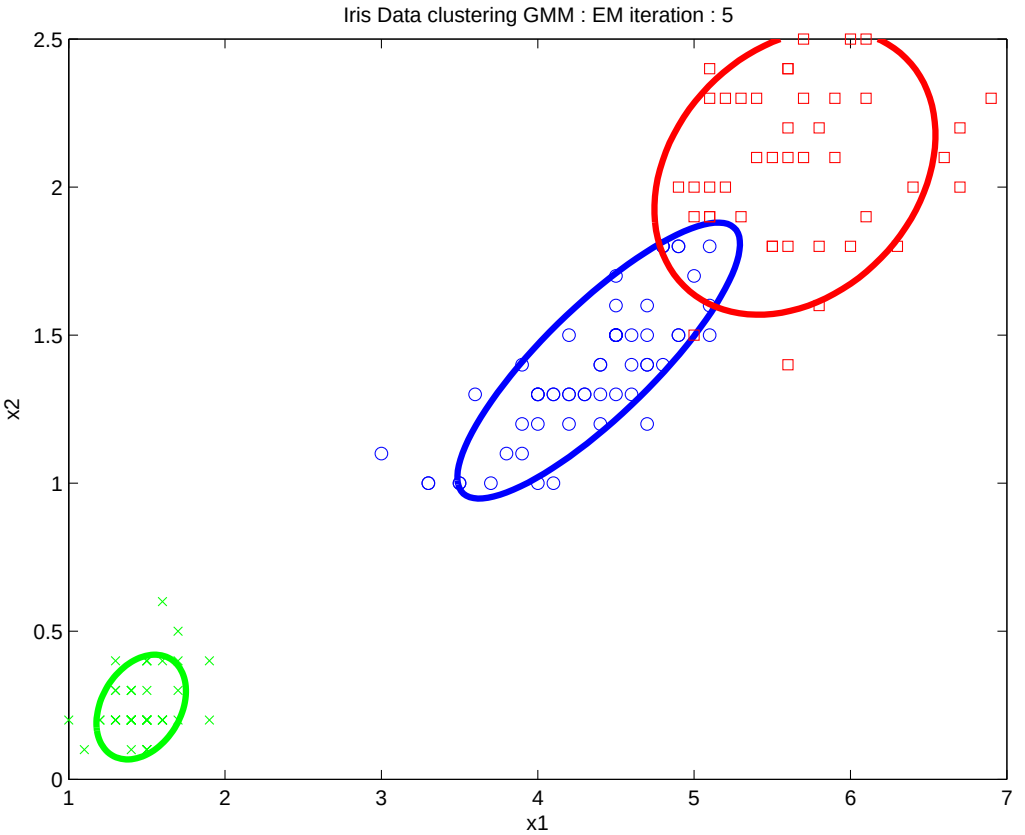
Examples


Iris Data clustering GMM : EM iteration : 3
x2
x1
2.5
2
1.5
1
0.5
0
1
2
3
4
5
6
7

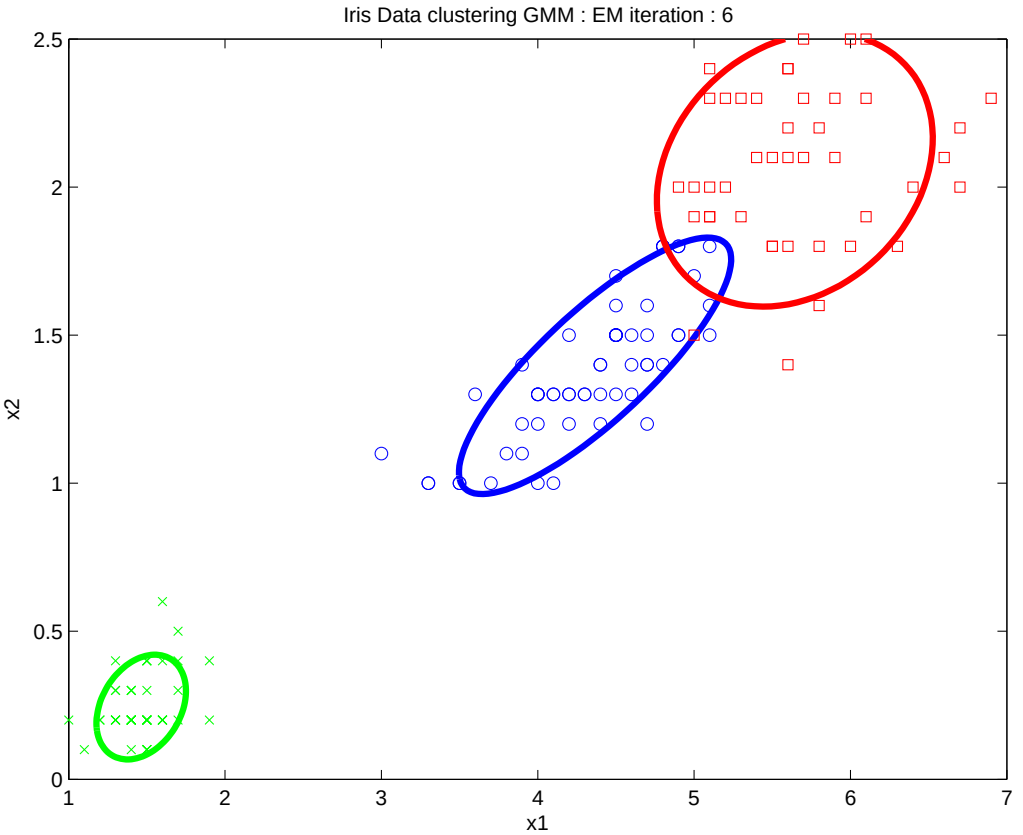
Examples


Iris Data clustering GMM : EM iteration : 4
x2
x1
2.5
2
1.5
1
0.5
0
1
2
3
4
5
6
7

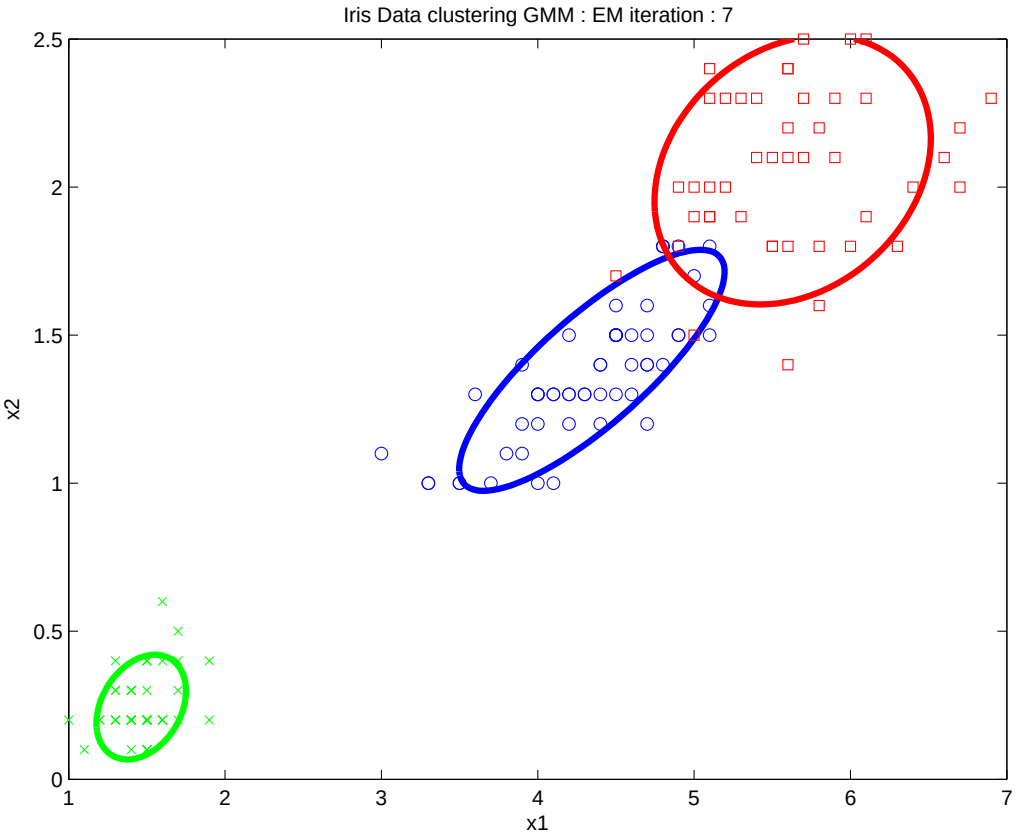
Examples


Iris Data clustering GMM : EM iteration : 5
x2
x1
2.5
2
1.5
1
0.5
0
1
2
3
4
5
6
7

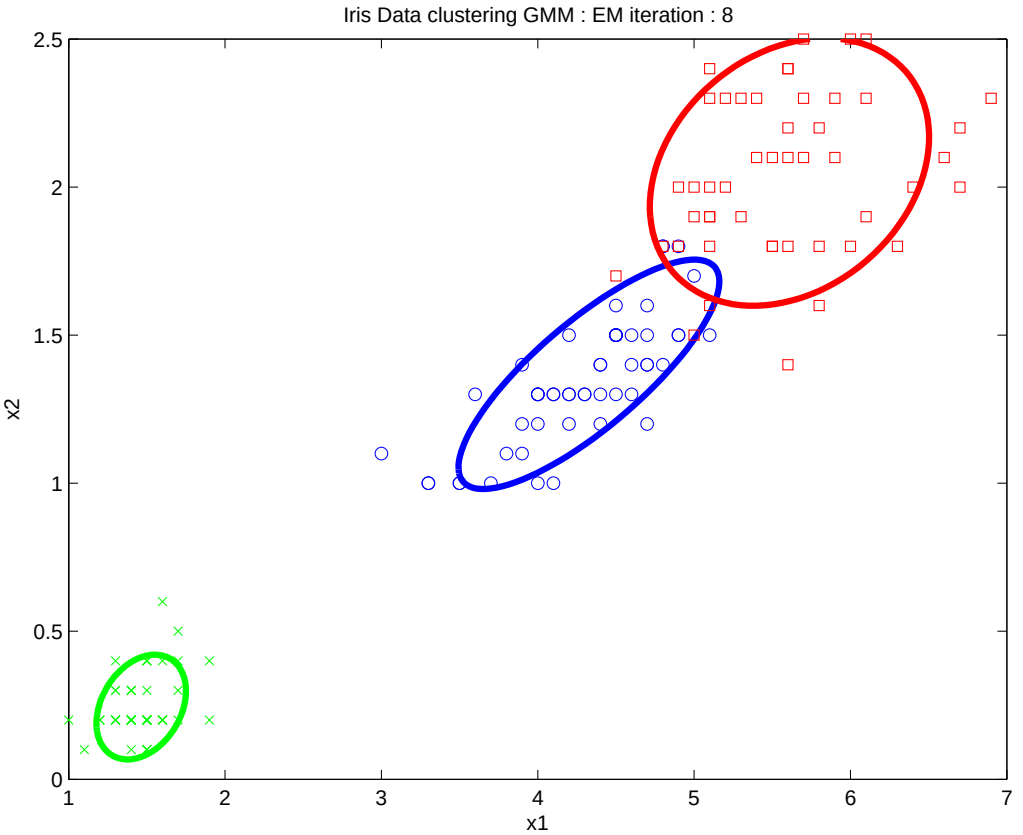
Examples


Iris Data clustering GMM : EM iteration : 6
x2
x1
2.5
2
1.5
1
0.5
0
1
2
3
4
5
6
7

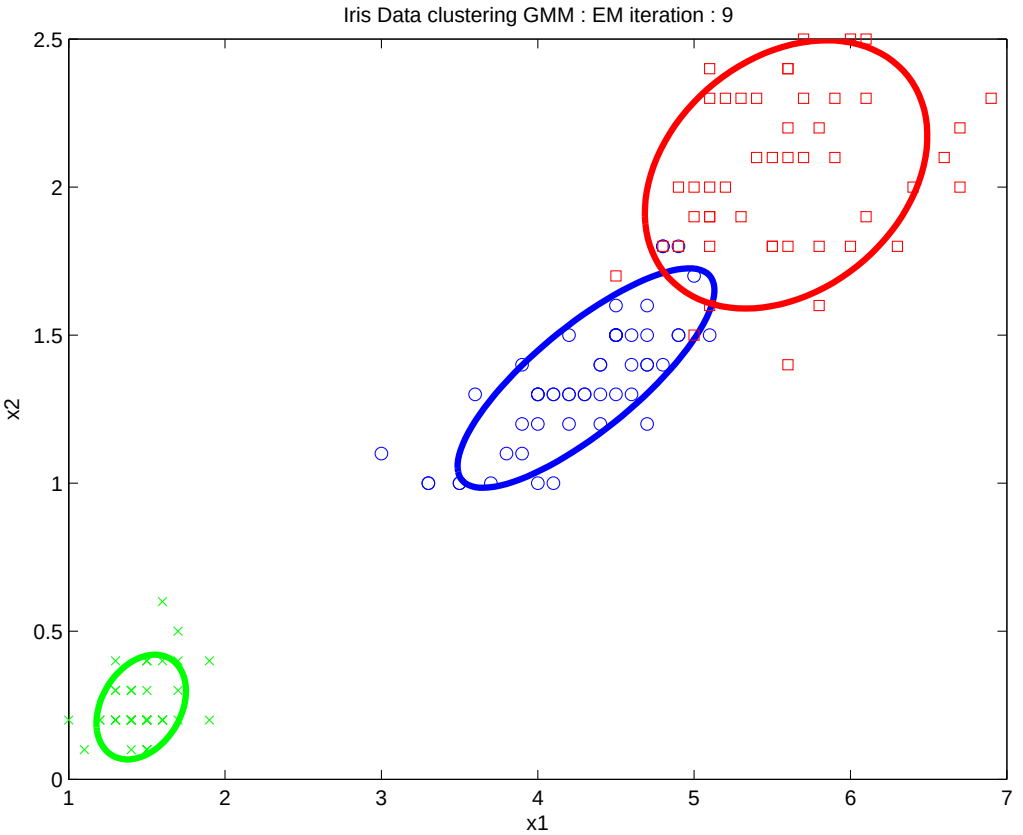
Examples


Iris Data clustering GMM : EM iteration : 7
x2
x1
2.5
2
1.5
1
0.5
0
1
2
3
4
5
6
7

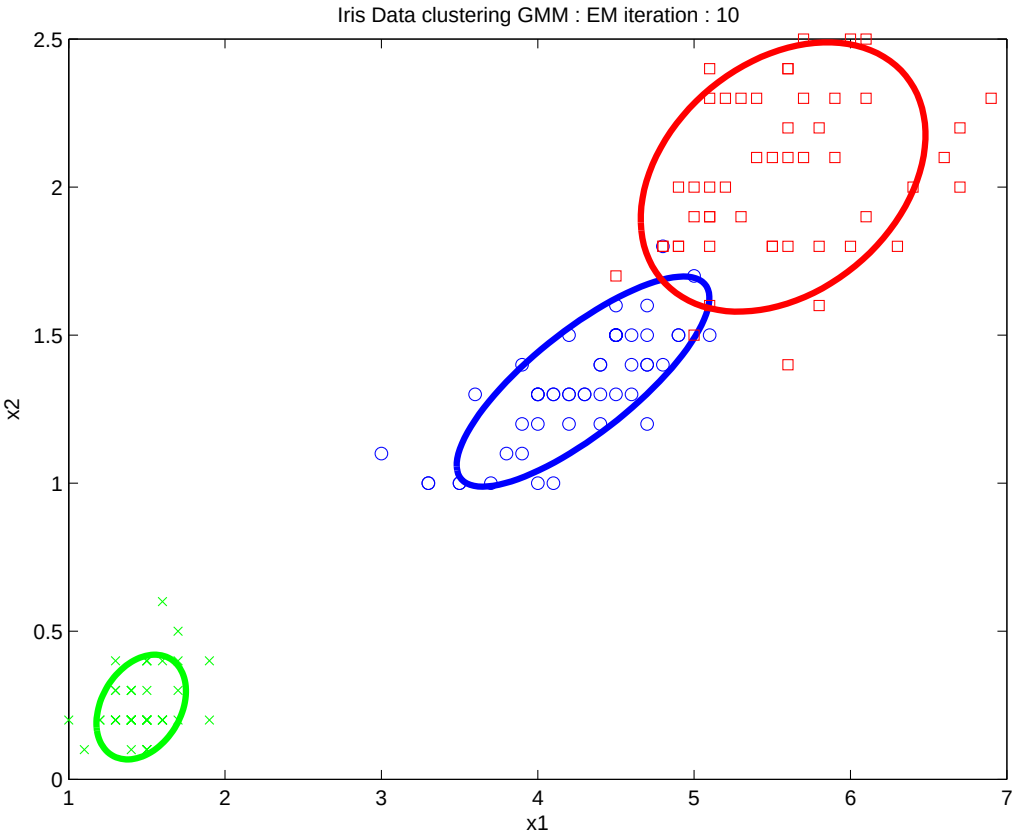
Examples


Iris Data clustering GMM : EM iteration : 8
x2
x1
2.5
2
1.5
1
0.5
0
1
2
3
4
5
6
7

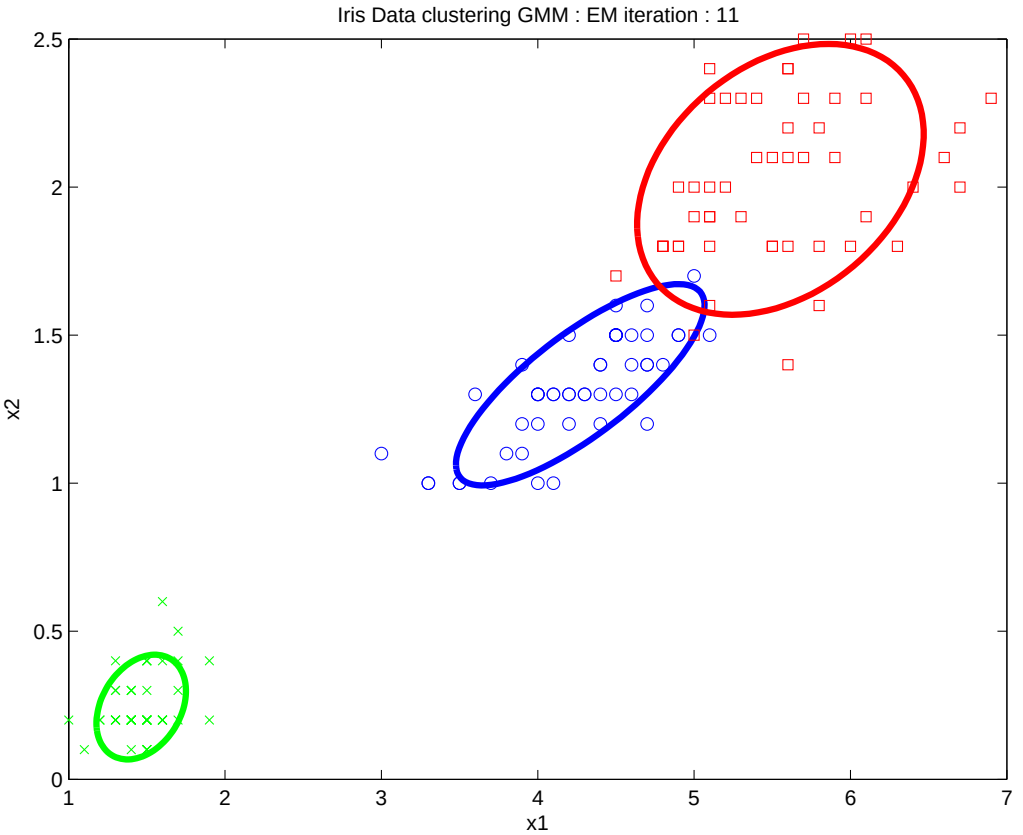
Examples


Iris Data clustering GMM : EM iteration : 9
x2
x1
2.5
2
1.5
1
0.5
0
1
2
3
4
5
6
7

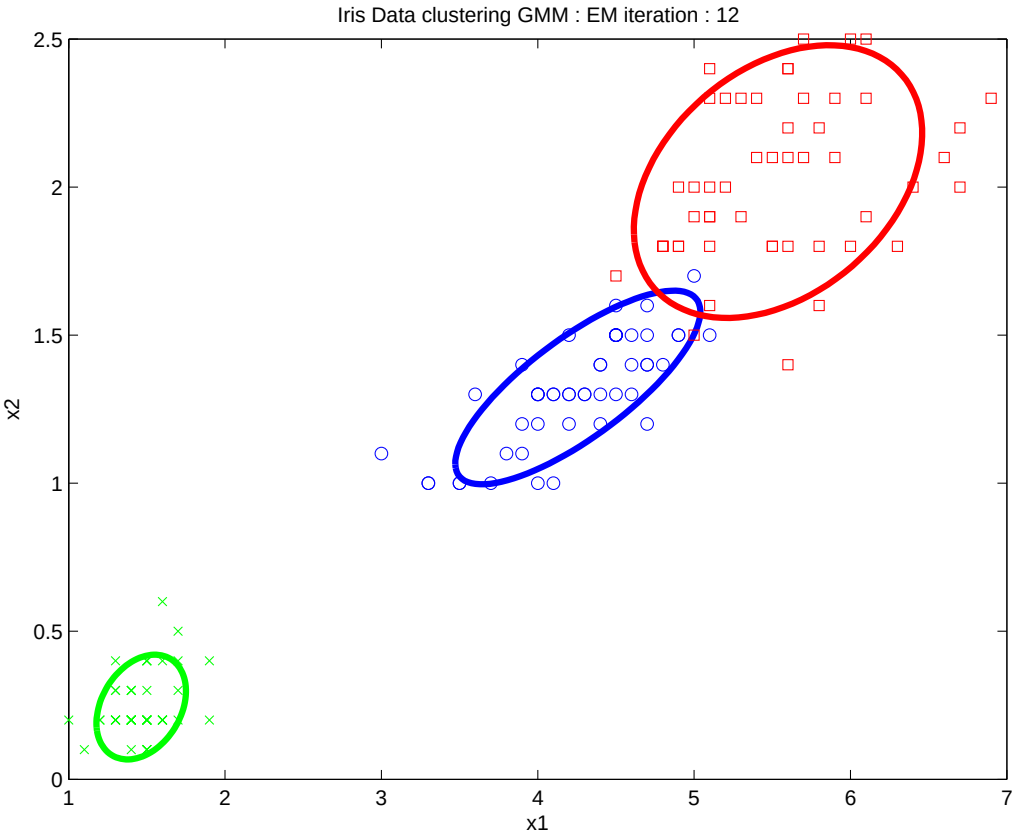
Examples


Iris Data clustering GMM : EM iteration : 10
x2
x1
2.5
2
1.5
1
0.5
0
1
2
3
4
5
6
7

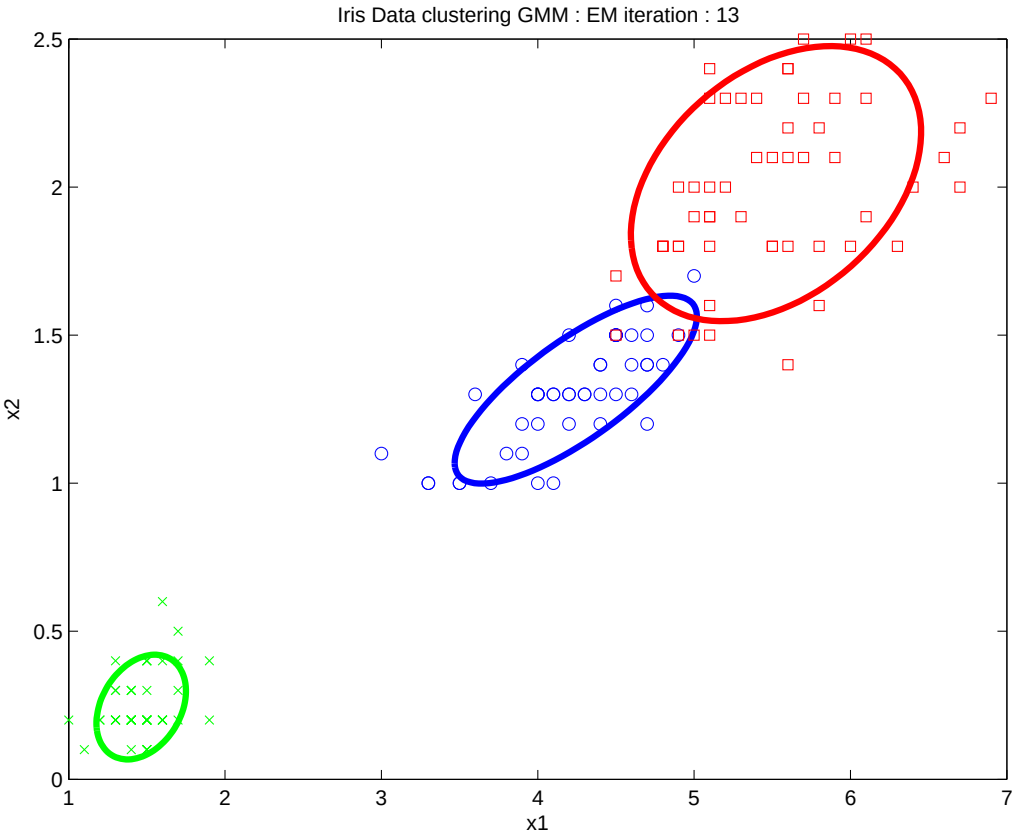
Examples


Iris Data clustering GMM : EM iteration : 11
x2
x1
2.5
2
1.5
1
0.5
0
1
2
3
4
5
6
7

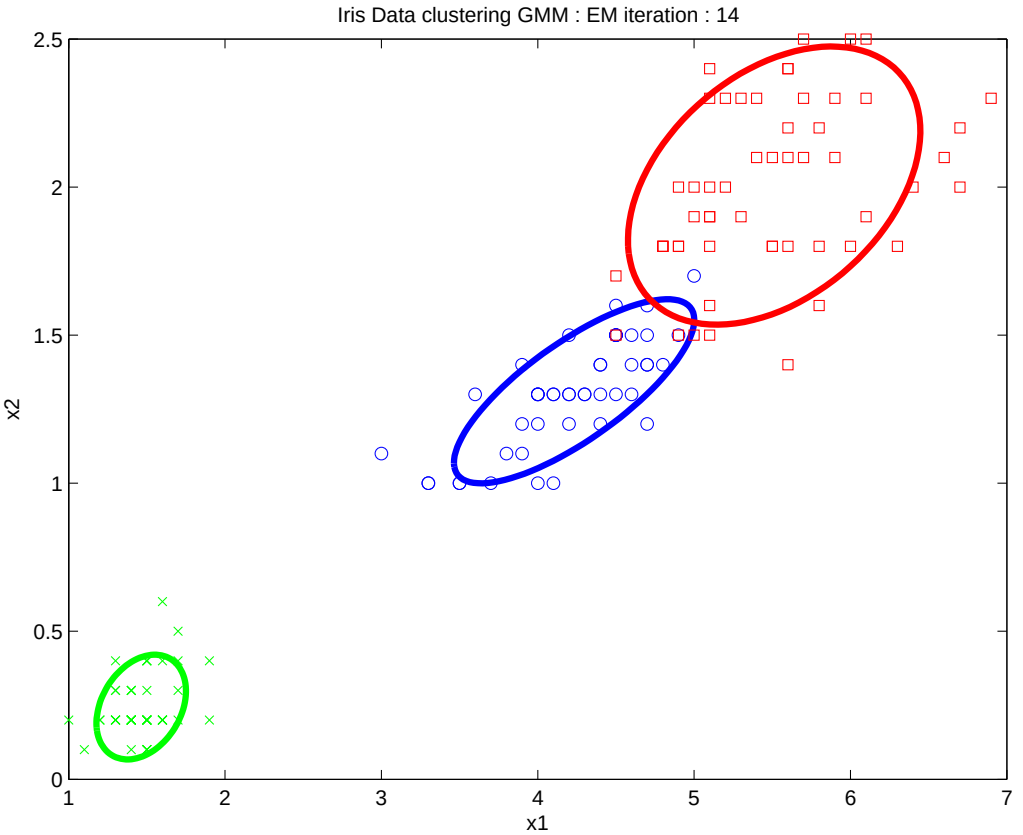
Examples


Iris Data clustering GMM : EM iteration : 12
x2
x1
2.5
2
1.5
1
0.5
0
1
2
3
4
5
6
7

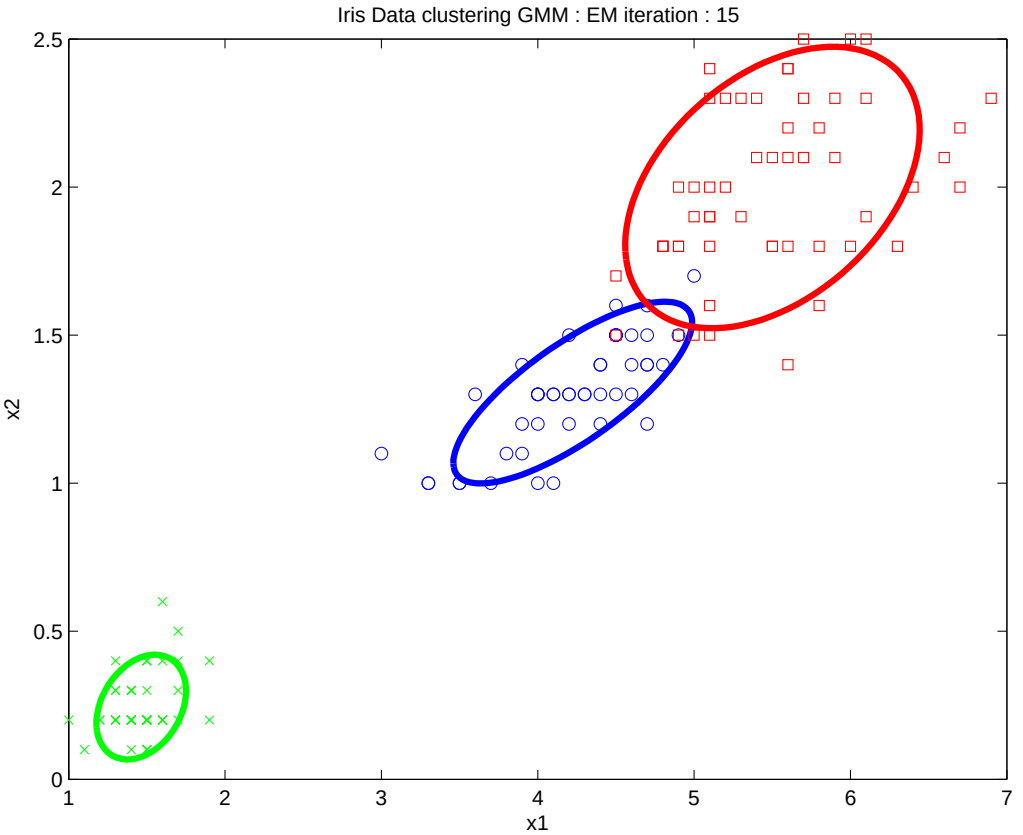
Examples


Iris Data clustering GMM : EM iteration : 13
x2
x1
2.5
2
1.5
1
0.5
0
1
2
3
4
5
6
7

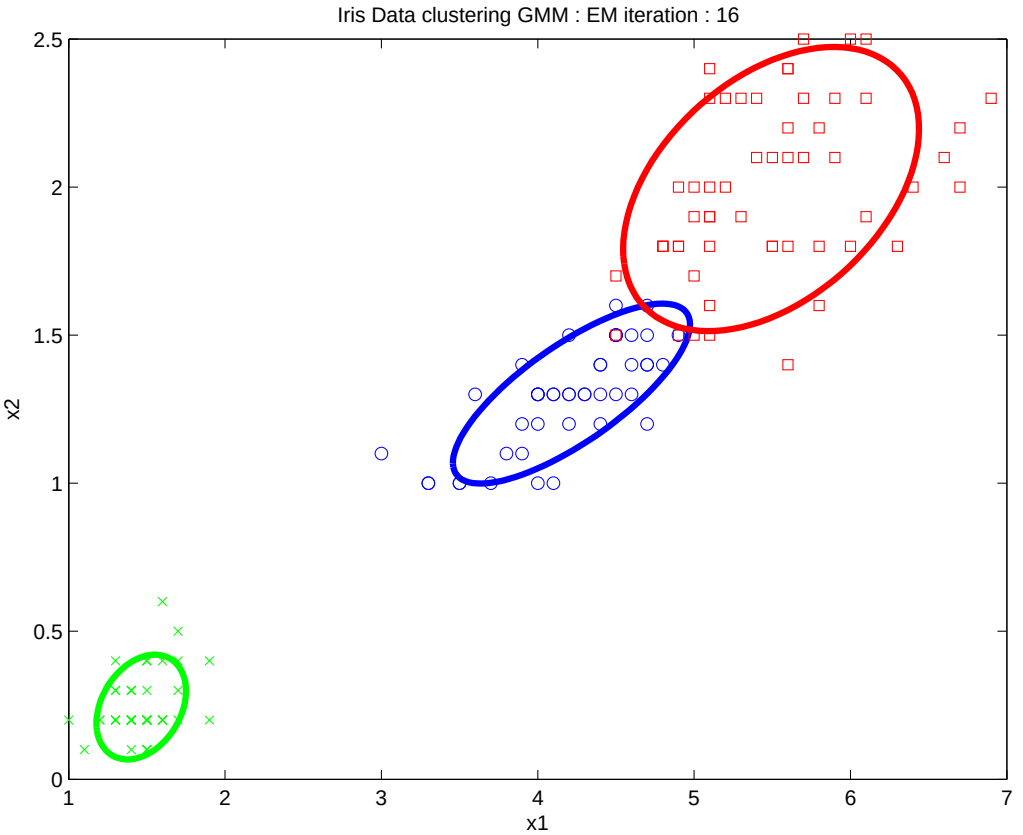
Examples


Iris Data clustering GMM : EM iteration : 14
x2
x1
2.5
2
1.5
1
0.5
0
1
2
3
4
5
6
7

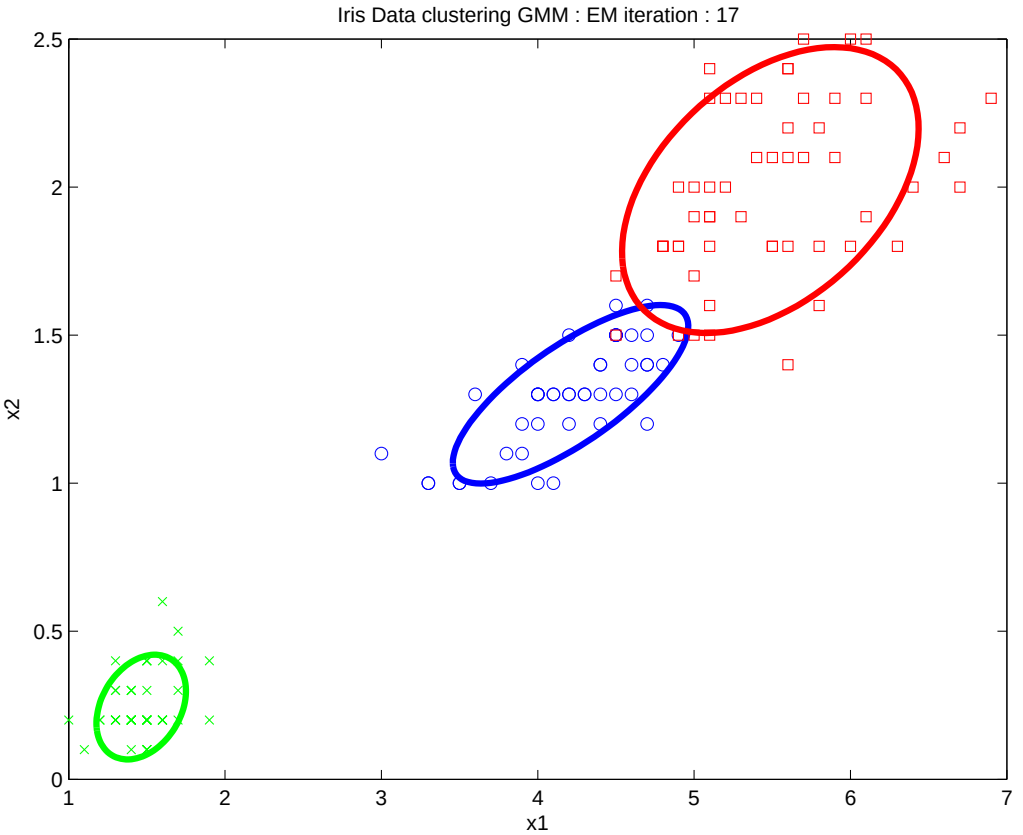
Examples


Iris Data clustering GMM : EM iteration : 15
x2
x1
2.5
2
1.5
1
0.5
0
1
2
3
4
5
6
7

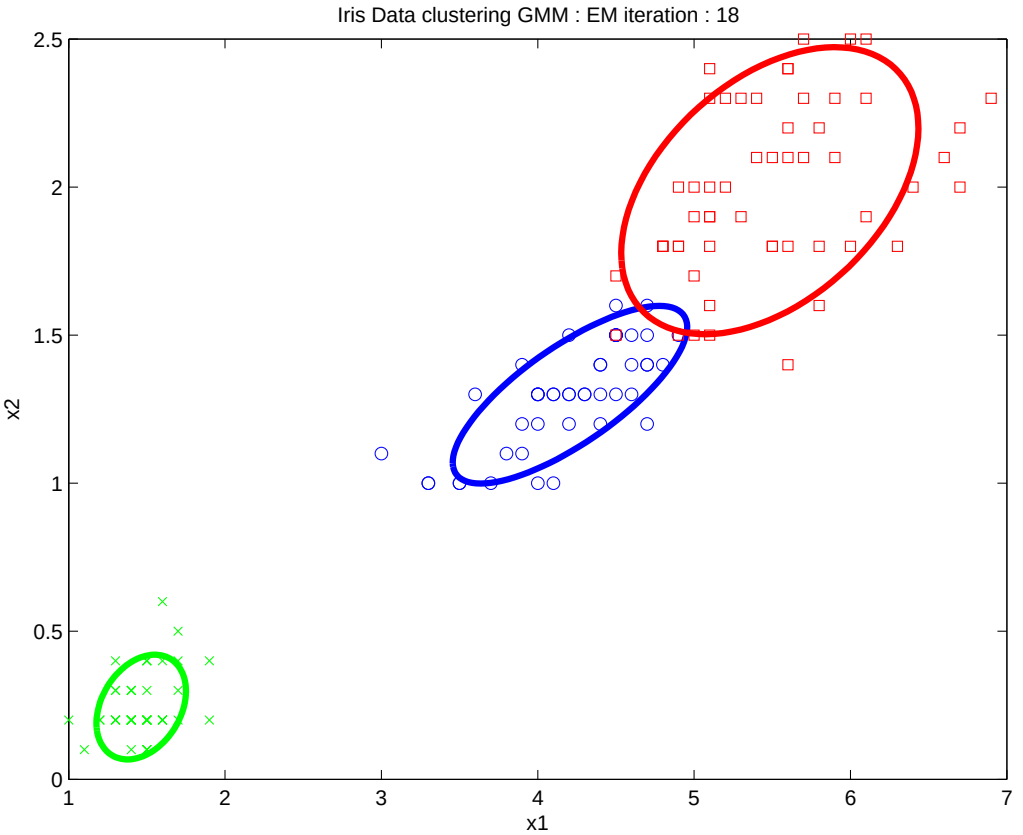
Examples


Iris Data clustering GMM : EM iteration : 16
x2
x1
2.5
2
1.5
1
0.5
0
1
2
3
4
5
6
7

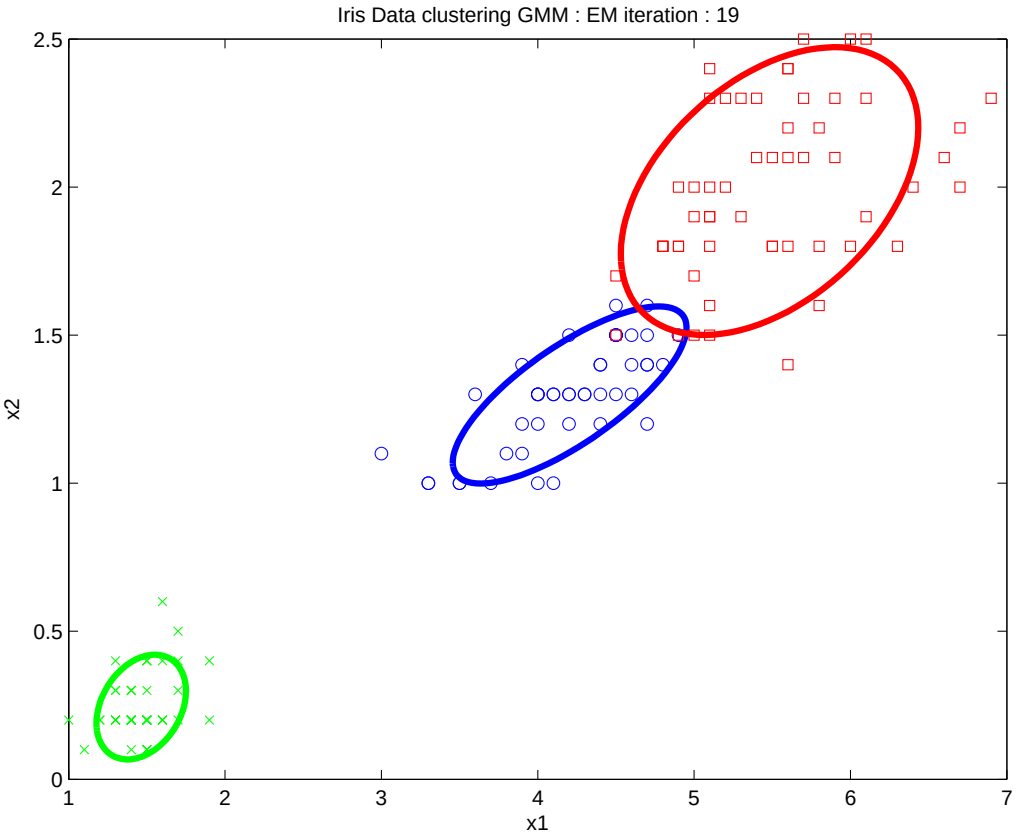
Examples


Iris Data clustering GMM : EM iteration : 17
x2
x1
2.5
2
1.5
1
0.5
0
1
2
3
4
5
6
7

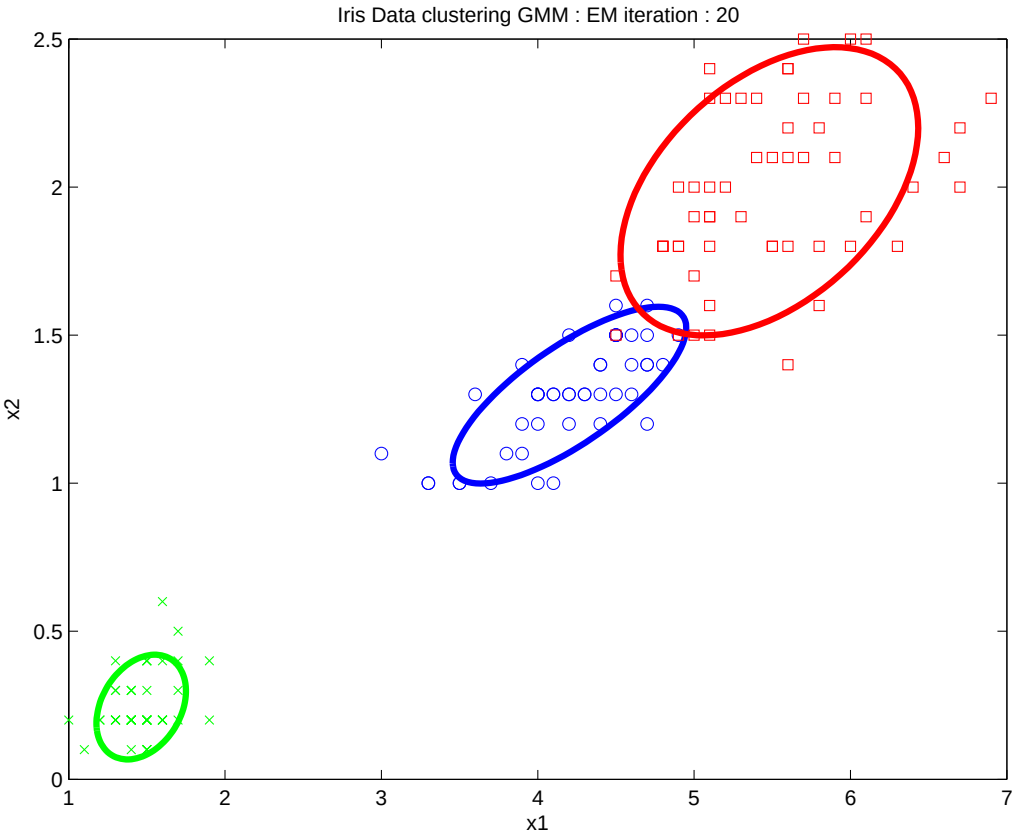
Examples


Iris Data clustering GMM : EM iteration : 18
x2
x1
2.5
2
1.5
1
0.5
0
1
2
3
4
5
6
7

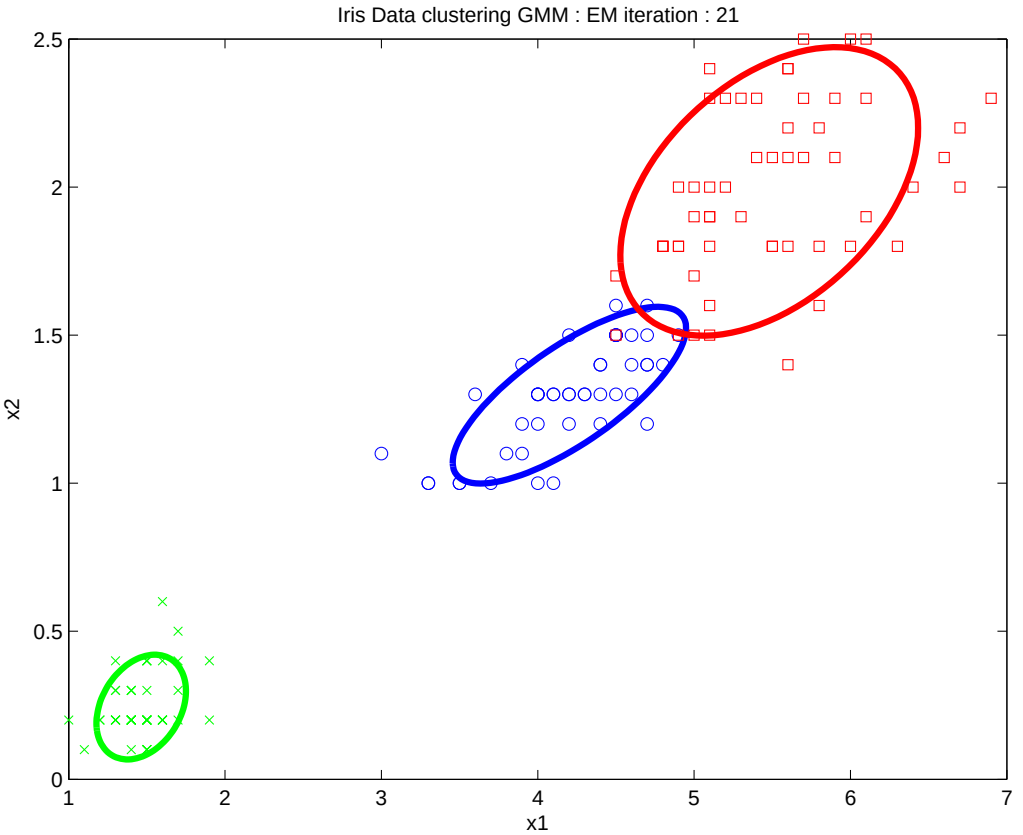
Examples


Iris Data clustering GMM : EM iteration : 19
x2
x1
2.5
2
1.5
1
0.5
0
1
2
3
4
5
6
7

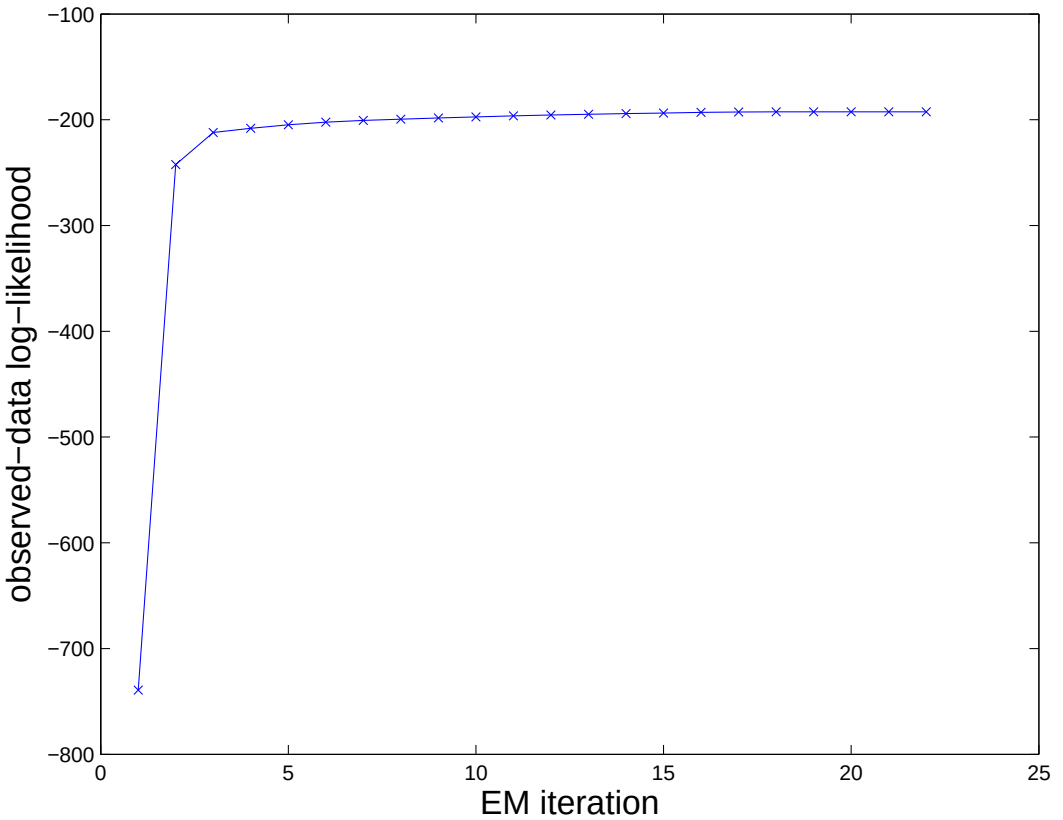
Examples


Iris Data clustering GMM : EM iteration : 20
x2
x1
2.5
2
1.5
1
0.5
0
1
2
3
4
5
6
7

Examples


Iris Data clustering GMM : EM iteration : 21
x2
x1
2.5
2
1.5
1
0.5
0
1
2
3
4
5
6
7

Examples


−100
−200
−300
−400
−500
−600
−700
−800
0
5
10
15
20
25
observed−data log−likelihood
EM iteration