Graphical models - MVA 2017/2018
Homework 2

Rémi Lespinet

1 Conditional independence and factorizations

1. Let X, Y and Z be real random variables.

$$p(x\,|\,y,z)p(y,z) = p(x,y,z) = p(x,y\,|\,z)p(z) \tag{$\star$}$$

Sufficient condition

If $X \perp\!\!\!\perp Y \ | \ Z$, then

$$\forall (x,y,z) \in E \times F \times G,\ p(x,y\,|\,z) = p(x\,|\,z)p(y\,|\,z)$$

and we can rewrite ($\star$) as

$$p(x\,|\,y,z)p(y,z) = p(x\,|\,z)p(y\,|\,z)p(z)$$

Which leads to

$$p(x\,|\,y,z)p(y,z) = p(x\,|\,z)p(y,z)$$

We conclude that

$$\forall (x,y,z) \in E \times F \times G,\ p(y,z) > 0 \Rightarrow p(x\,|\,y,z) = p(x\,|\,z)$$

Necessary condition

Let $(x,y,z) \in E \times F \times G$, we suppose $p(y,z) > 0 \Rightarrow p(x\,|\,y,z) = p(x\,|\,z)$ (For this property to make sense, we also suppose $p(z) > 0$)

<u>If $p(y,z) = 0$</u>

Then we have $\forall x \in E,\ p(x,y\,|\,z) = 0$ and $p(y\,|\,z) = 0$ so

$$\forall x \in E,\ p(x,y\,|\,z) = p(x\,|\,z)p(y\,|\,z) = 0$$

<u>If $p(y,z) > 0$</u>

By ($\star$), we have

$$\forall x \in E,\ p(x\,|\,z)p(y,z) = p(x,y\,|\,z)p(z)$$

Hence

$$\forall x \in E,\ p(x|z)p(y|z)p(z) = p(x,y|z)p(z)$$

which by division by $p(z) > 0$ gives

$$\forall x \in E,\ p(x,y|z) = p(x|z)p(y|z)$$

2. if p factorizes in G, we have the following factorization

$$p(x,y,z,t) = p(x)p(y)p(z|x,y)p(t|z)$$

In this graph, considering the chain $X \to Z \to Y$, this chain is not blocked at Z since X, Z, Y is a v-structure and T is a descendant of Z. There exists a chain from X to Y that is not blocked by T. Hence X and Y are not d-separated by T, which means that there exist $p \in \mathcal{L}(G)$ such that $X \not\perp\!\!\!\perp Y \mid T$

Looking at the problem with Bayes ball algorithm, the ball can move take the path $X \to Z \to T \to Z \to Y$, we see that $X \to Z \to T$ is a markov chain with Z unobserved so the ball is not blocked at Z, $Z \to T \to Z$ is a v-structure with T observed so the ball is not blocked at T, and $T \to Z \to Y$ is also a markov chain, so the ball is not blocked at its second passage by Z.

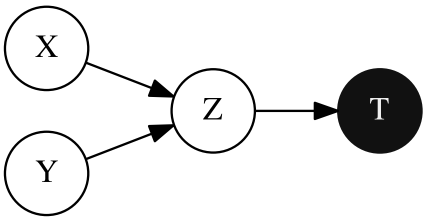


Figure 1: Representation of Bayes ball algorithm on Graph G to determine if X and Y are conditionally independant given T

3.

(a) Let (X,Y,Z) be random variables on a finite space with Z binary. Suppose $X \perp\!\!\!\perp Y \mid Z$ and $X \perp\!\!\!\perp Y$.

If $p(Z=1) = 0$ or $p(Z=1) = 1$ we obviously have $X \perp\!\!\!\perp Z$ and $Y \perp\!\!\!\perp Z$, so we can suppose without loss of generality that $p(Z=1) \neq 0$ and $p(Z=0) \neq 1$.

To simplify the notation, let us define π such that

$$\pi = P(Z=1)$$

We have

$$p(x,y) = p(x,y|Z=1)\pi + p(x,y|Z=0)(1-\pi)$$

which by hypothesis can be rewriten as

$$p(x, y) = p(x|Z = 1)p(y|Z = 1)\pi + p(x|Z = 0)p(y|Z = 0)(1 - \pi) \tag{$\triangle$}$$

We also have using the second hypothesis :

$$p(x, y) = p(x)p(y) = [p(x|Z = 1)\pi + p(x|Z = 0)(1 - \pi)] \cdot [p(y|Z = 1)\pi + p(y|Z = 0)(1 - \pi)]$$

Developping this expression, we obtain

$$\begin{aligned} p(x, y) = {} & p(x|Z = 1)p(y|Z = 1)\pi^2 \\ & + p(x|Z = 0)p(y|Z = 1)\pi(1 - \pi) \\ & + p(x|Z = 1)p(y|Z = 0)\pi(1 - \pi) \\ & + p(x|Z = 0)p(y|Z = 0)(1 - \pi)^2 \end{aligned}$$

Since

$$\pi^2 = \pi(\pi - 1) + \pi$$

and

$$(1 - \pi)^2 = \pi(\pi - 1) + 1 - \pi$$

By replacing in the previous equation,

$$\begin{aligned} p(x, y) = {} & p(x|Z = 1)p(y|Z = 1)(\pi(\pi - 1) + \pi) \\ & + p(x|Z = 0)p(y|Z = 1)\pi(1 - \pi) \\ & + p(x|Z = 1)p(y|Z = 0)\pi(1 - \pi) \\ & + p(x|Z = 0)p(y|Z = 0)(\pi(\pi - 1) + 1 - \pi) \end{aligned}$$

Hence,

$$\begin{aligned} p(x, y) = {} & p(x|Z = 1)p(y|Z = 1)\pi + p(x|Z = 0)p(y|Z = 0)(1 - \pi) \\ & - p(x|Z = 1)p(y|Z = 1)\pi(1 - \pi) \\ & + p(x|Z = 0)p(y|Z = 1)\pi(1 - \pi) \\ & + p(x|Z = 1)p(y|Z = 0)\pi(1 - \pi) \\ & - p(x|Z = 0)p(y|Z = 0)\pi(1 - \pi) \end{aligned}$$

By ($\triangle$), the first line of the left member is $p(x, y)$, hence

$$\begin{aligned} 0 = \pi(1 - \pi)\Big[& - p(x|Z = 1)p(y|Z = 1) \\ & + p(x|Z = 0)p(y|Z = 1) \\ & + p(x|Z = 1)p(y|Z = 0) \\ & - p(x|Z = 0)p(y|Z = 0)\Big] \end{aligned}$$

Which gives

$$\pi(1 - \pi)\,[p(x|Z = 0) - p(x|Z = 1)]\,[p(y|Z = 0) - p(y|Z = 1)] = 0$$

Since we have supposed that $P(Z = 0) \neq 0$ and $P(Z = 1) \neq 0$, we have

$$p(x|Z = 0) = p(x|Z = 1) = 0 \text{ or } p(y|Z = 0) = p(y|Z = 1)$$

- If $p(x|Z=0) = p(x|Z=1)$, then

  $$p(x) = p(x|Z=0)p(Z=0) + p(x|Z=1)p(Z=1)$$

  gives

  $$p(x) = p(x|Z=0) = p(x|Z=1)$$

  Thus,

  $$X \perp\!\!\!\perp Z$$

- else, we have $p(y|Z=0) = p(y|Z=1)$, and we can apply the exact same by symmetry of the variables x and y,

  $$Y \perp\!\!\!\perp Z$$

2 Distributions factorizing in a graph

1. The chain rules gives us

$$p(x_i, x_j, x_{\pi_i}) = p(x_{\pi_i})p(x_i|x_{\pi_i})p(x_j|x_{\pi_i}, x_i)$$

and

$$p(x_i, x_j, x_{\pi_i}) = p(x_{\pi_i})p(x_j|x_{\pi_i})p(x_i|x_{\pi_i}, x_j)$$

From that, we deduce that

$$p(x_i|x_{\pi_i})p(x_j|x_{\pi_i}, x_i) = p(x_j|x_{\pi_i})p(x_i|x_{\pi_i}, x_j) \qquad (\square)$$

Let π_i be the set of parents of node i in G and ρ_i be the set of parent of i in G'. First we notice that

$$\forall k \in \{1 \dots N\} \setminus \{i,j\}, \rho_k = \pi_k$$

and we have

$$\rho_i = \pi_i \cup \{j\}$$

$$\rho_j = \pi_i$$

Suppose that p factorizes in G, we have

$$p(x) = \prod_{1\leq k\leq N} p(x_k|x_{\pi_k}) = p(x_i|x_{\pi_i})p(x_j|x_{\pi_i}, x_i) \prod_{\substack{1\leq k\leq N \\ k\notin\{i,j\}}} p(x_k|x_{\pi_k})$$

By $(\square)$,

$$p(x) = p(x_j|x_{\pi_i})p(x_i|x_{\pi_i}, x_j) \prod_{\substack{1\leq k\leq N \\ k\notin\{i,j\}}} p(x_k|x_{\pi_k})$$

Which yields

$$p(x) = p(x_j|x_{\rho_j})p(x_i|x_{\rho_i}) \prod_{\substack{1\leq k\leq N \\ k\notin\{i,j\}}} p(x_k|x_{\rho_k}) = \prod_{1\leq k\leq N} p(x_k|x_{\rho_k})$$

This proves that $\mathcal{L}(G) \subset \mathcal{L}(G')$. By applying the same operation on the graph G' with the covered edge (j, i), we obtain that $\mathcal{L}(G') \subset \mathcal{L}(G)$

Hence $\mathcal{L}(G) = \mathcal{L}(G')$

2. We start by a lemma,

Lemma 2.1. *The maximum cardinal of a clique in an undirected tree is* 2

Proof. If we had a clique of cardinal $n > 2$, $x_0, x_1, \dots x_n$, by definition of the clique, (x_0, x_1, x_2) and (x_0, x_2) would be two distinct path from x_0 to x_2. By definition, of an undirected tree, two vertices are connected by exactly one path, hence the contradiction. □

Let $p \in \mathcal{L}(G)$, then we have

$$p(x) = \prod_{i=1}^{N} p(x_i | x_{\pi_i})$$

Based on the lemma, if we index G with a topological order (starting from 0 for the root, to n) and G' with the same indices, the cliques of cardinal 2 in G' form the set $\{(x_i, x_{\pi_i}) \ ; \ 1 \leq i \leq n\}$ where x_{π_i} is the set of parent of the node x_i (it contains at most one element). the cliques of cardinal 1 are obviously defined by the set of nodes $\{x_i \ ; \ 0 \leq i \leq n\}$.

Therefore, we can write

$$p(x) = \prod_{i=0}^{N} \phi_i(x_i, x_{\pi_i})$$

or equivalently

$$p(x) = \prod_{C \in \mathcal{C}} \phi_C(x_C)$$

with

$$\phi_C = \begin{cases} p(x_i | x_{\pi_i}) & \text{if } C = (x_i, x_{\pi_i}) \\ p(x_i) & \text{if } C = (x_i) \text{ and } x_{\pi_i} = \varnothing \\ 1 & \text{if } C = (x_i) \text{ and } x_{\pi_i} \neq \varnothing \end{cases}$$

Hence $p \in \mathcal{L}(G)$.

Suppose now that $p \in \mathcal{L}(G')$. By the previous reasonement on the cliques of an undirected tree, and using the same indexing as before we can write

$$p(x) = \prod_{i=0}^{N} \phi_i(x_i, x_{\pi_i})$$

3 Entropy and Mutual Information

1. Let X be a discrete random variable on $\mathcal{X}$. Let $p_X(x)$ denote its probability mass function.

(a) For all $x \in \mathcal{X}$, $0 \leq p_X(x) \leq 1$, hence

$$\forall x \in \mathcal{X}, -\log p_X(x) \geq 0$$

Hence,

$$\forall x \in \mathcal{X}, -p_X(x) \log p_X(x) \geq 0$$

Summing over the set $\mathcal{X}$ we obtain that $H(X) \geq 0$.

Let us prove that $H(X) = 0$ if and only if X is a constant with probability 1.

Necessary condition

if X is constant with probability 1,

$$\exists c \in \mathcal{X}, \begin{cases} p_X(c) = 1 \\ \forall x \neq c, p_X(x) = 0 \end{cases}$$

Hence each term of the sum in the entropy is null

$$\forall x \in \mathcal{X},\ p_X(x) \log p_X(x) = 0$$

And

$$H(X) = \sum_{x \in \mathcal{X}} p_X(x) \log p_X(x) = 0$$

Sufficient condition

Suppose $H(X) = 0$.

H is a sum of positive terms. If there were a $\tilde{x} \in \mathcal{X}$, such that $0 < p_X(\tilde{x}) < 1$, then we would have $-p_X(\tilde{x}) \log p_X(\tilde{x}) > 0$, hence

$$-\sum_{x \in \mathcal{X}} p_X(x) \log p_X(x) \geq -p_X(\tilde{x}) \log p_X(\tilde{x}) > 0$$

(One of the term of H(X) would be strictly positive and H would be strictly positive)

We deduce that

$$\forall x \in \mathcal{X},\ p_X(x) = 0 \text{ or } p_X(x) = 1 \qquad (\Diamond_1)$$

Moreover by definition,

$$\sum_{x \in \mathcal{X}} p_X(x) = 1 \qquad (\Diamond_2)$$

In order to satisfy both $(\Diamond_1)$ and $(\Diamond_2)$, we must have exactly one $c \in \mathcal{X}$ such that $p_X(c) = 1$ and $\forall x \in \mathcal{X} \setminus \{c\},\ p_X(x) = 0$, which means that X is constant with probability 1.

(b) We have $\forall x \in \mathcal{X} q(x) = \frac{1}{k}$, and the Kullback-Leibler divergence between p and q is finite.

We can write

$$D(p\|q) = \sum_{x\in\mathcal{X}} p(x)\log(p(x)) - \sum_{x\in\mathcal{X}} p(x)\log\left(\frac{1}{k}\right)$$

Hence,

$$D(p\|q) = -H(X) + \log(k)$$

(c) Since we know that $D(p\|q) \geq 0$, we obtain immediately

$$H(X) \leq \log(k)$$

We note that the upper bound on the entropy is the entropy of a random variable uniformely distributed over $\mathcal{X}$.

$$H(Y) = -\sum_{x\in\mathcal{X}} \frac{1}{k}\log\left(\frac{1}{k}\right) = \log(k)$$

2. Let (X_1, X_2) a pair of discrete random variables defined over $\mathcal{X}_1 \times \mathcal{X}_2$.

(a) We have

$$I(X_1,X_2) = \sum_{x_1\in\mathcal{X}_1}\sum_{x_2\in\mathcal{X}_2} p(x_1,x_2)\log\frac{p(x_1,x_2)}{p(x_1)p(x_2)}$$

This can be rewriten as

$$I(X_1,X_2) = \sum_{x_1\in\mathcal{X}_1}\sum_{x_2\in\mathcal{X}_2} p(x_1)p(x_2)\frac{p(x_1,x_2)}{p(x_1)p(x_2)}\log\frac{p(x_1,x_2)}{p(x_1)p(x_2)}$$

Let f be the function

$$\begin{array}{lcl} \mathbb{R}^+ & \rightarrow & \mathbb{R} \\ x & \mapsto & x\log x \end{array}$$

This function is convex (twice differentiable, with $f''(x) = 1/x$), and since

$$\sum_{x_1\in\mathcal{X}_1}\sum_{x_2\in\mathcal{X}_2} p(x_1)p(x_2) = \left(\sum_{x_1\in\mathcal{X}_1} p(x_1)\right)\left(\sum_{x_2\in\mathcal{X}_2} p(x_2)\right) = 1$$

Jensen inequality gives us

$$f\left(\sum_{x_1\in\mathcal{X}_1}\sum_{x_2\in\mathcal{X}_2} p(x_1)p(x_2)\frac{p(x_1,x_2)}{p(x_1)p(x_2)}\right) \leq \sum_{x_1\in\mathcal{X}_1}\sum_{x_2\in\mathcal{X}_2} p(x_1)p(x_2) f\left(\frac{p(x_1,x_2)}{p(x_1)p(x_2)}\right)$$

Since

$$f\left(\sum_{x_1\in\mathcal{X}_1}\sum_{x_2\in\mathcal{X}_2} p(x_1)p(x_2)\frac{p(x_1,x_2)}{p(x_1)p(x_2)}\right) = f\left(\sum_{x_1\in\mathcal{X}_1}\sum_{x_2\in\mathcal{X}_2} p(x_1,x_2)\right) = f(1) = 0$$

And

$$\sum_{x_1 \in \mathcal{X}_1} \sum_{x_2 \in \mathcal{X}_2} p(x_1)p(x_2) f\left(\frac{p(x_1, x_2)}{p(x_1)p(x_2)}\right) = I(X_1, X_2)$$

We obtain

$$I(X_1, X_2) \geq 0$$

(b)

We can rewrite the mutual information as

$$\begin{aligned} I(X_1, X_2) &= \sum_{x_1 \in \mathcal{X}_1} \sum_{x_2 \in \mathcal{X}_2} p(x_1, x_2) \log\Big(p(x_1, x_2)\Big) \\ &\quad - \sum_{x_1 \in \mathcal{X}_1} \sum_{x_2 \in \mathcal{X}_2} p(x_1, x_2) \log\Big(p(x_1)\Big) \\ &\quad - \sum_{x_1 \in \mathcal{X}_1} \sum_{x_2 \in \mathcal{X}_2} p(x_1, x_2) \log\Big(p(x_2)\Big) \end{aligned}$$

We can simplify

$$\begin{aligned} I(X_1, X_2) &= \sum_{x_1 \in \mathcal{X}_1} \sum_{x_2 \in \mathcal{X}_2} p(x_1, x_2) \log\Big(p(x_1, x_2)\Big) \\ &\quad - \sum_{x_1 \in \mathcal{X}_1} \left(\sum_{x_2 \in \mathcal{X}_2} p(x_1, x_2)\right) \log\Big(p(x_1)\Big) \\ &\quad - \sum_{x_2 \in \mathcal{X}_2} \left(\sum_{x_1 \in \mathcal{X}_1} p(x_1, x_2)\right) \log\Big(p(x_2)\Big) \end{aligned}$$

Hence,

$$\begin{aligned} I(X_1, X_2) &= \sum_{x_1 \in \mathcal{X}_1} \sum_{x_2 \in \mathcal{X}_2} p(x_1, x_2) \log\Big(p(x_1, x_2)\Big) \\ &\quad - \sum_{x_1 \in \mathcal{X}_1} p(x_1) \log\Big(p(x_1)\Big) \\ &\quad - \sum_{x_2 \in \mathcal{X}_2} p(x_2) \log\Big(p(x_2)\Big) \end{aligned}$$

If we consider the variable $X = (X_1, X_2)$, we can write its entropy as

$$\begin{aligned} H(X) &= - \sum_{x \in \mathcal{X}_1 \times \mathcal{X}_2} p_X(x) \log p_X(x) \\ &= - \sum_{x \in \mathcal{X}_1 \times \mathcal{X}_2} p(x_1, x_2) \log\Big(p(x_1, x_2)\Big) \end{aligned}$$

Hence,

$$I(X_1, X_2) = -H(X_1, X_2) + H(X_1) + H(X_2)$$

(c) From the two previous question, we have

$$I(X_1, X_2) = -H(X_1, X_2) + H(X_1) + H(X_2) \geq 0$$

e.g.

$$H(X_1, X_2) \leq H(X_1) + H(X_2)$$

And if we note p the joint probablity distribution that make X_1 and X_2 independent, and X the random variable associated with this probability distribution, e.g.

$$\forall (x_1, x_2) \in \mathcal{X}_1 \times \mathcal{X}_2,\ p(x_1, x_2) = p(x_1)p(x_2)$$

$$\begin{aligned}
H(X) &= - \sum_{x \in \mathcal{X}_1 \times \mathcal{X}_2} p(x_1, x_2) \log \Big(p(x_1, x_2) \Big) \\
&= - \sum_{x \in \mathcal{X}_1 \times \mathcal{X}_2} p(x_1)p(x_2) \log \Big(p(x_1) \Big) - \sum_{x \in \mathcal{X}_1 \times \mathcal{X}_2} p(x_1)p(x_2) \log \Big(p(x_2) \Big) \\
&= - \sum_{x_1 \in \mathcal{X}_1} p(x_1) \log \Big(p(x_1) \Big) - \sum_{x_2 \in \mathcal{X}_2} p(x_2) \log \Big(p(x_2) \Big) \\
&= H(X_1) + H(X_2)
\end{aligned}$$

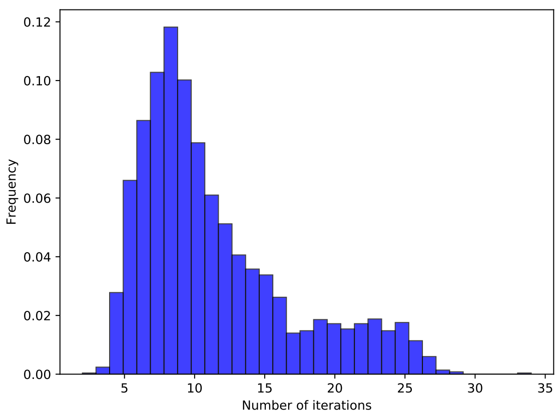
Which means that the maximum entropy is obtained when X_1 and X_2 are independant.

4 Implementation - Gaussian mixtures

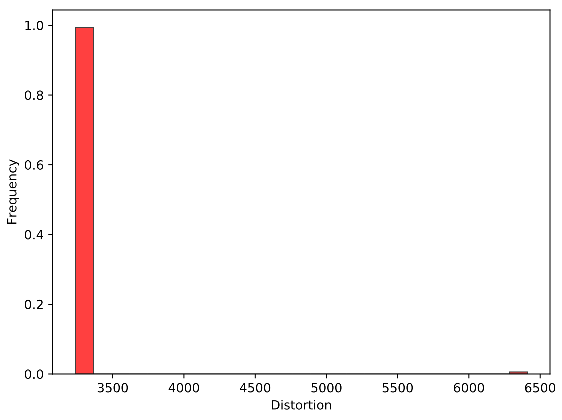
(a) The file *clustering.py* contains my implementation in python of the *K-means* algorithm. It uses random points as initialization. I used a function that I found online to show the voronoi cells in the case of K-means.

For $K = 4$, the algorithm seems to almost always converge to the same local optimum (3237.8 distortion), I ran the algorithm 5000 times and plotted the histogram representing the number of iterations to convergence (Figure 2a) and the histogram representing the distortion (Figure 2b) In the distortion histogram, the second bar represents (29 run of the algorithm over 5000, which means that the algorithm has converged to the global optimum 99.42% of the 5000 runs)

The result obtained 99.42% of the runs by the algorithm is presented in figure 3, and the evolution of distortion is given in figure 4. Since the histogram of the distortion presents cases where the algorithm converges to a worst local optimum, I managed to find one such example, it's presented in the figure 5 (this K-means execution resulted in a distortion of 6378.84 and converged in 5 steps)



(a) Histogram of the number of iterations



(b) Histogram of the distortion

Figure 2: Histogram representing the number of iterations and the distortion for 5000 iterations of the algorithm K-means with $K = 4$ on the training data

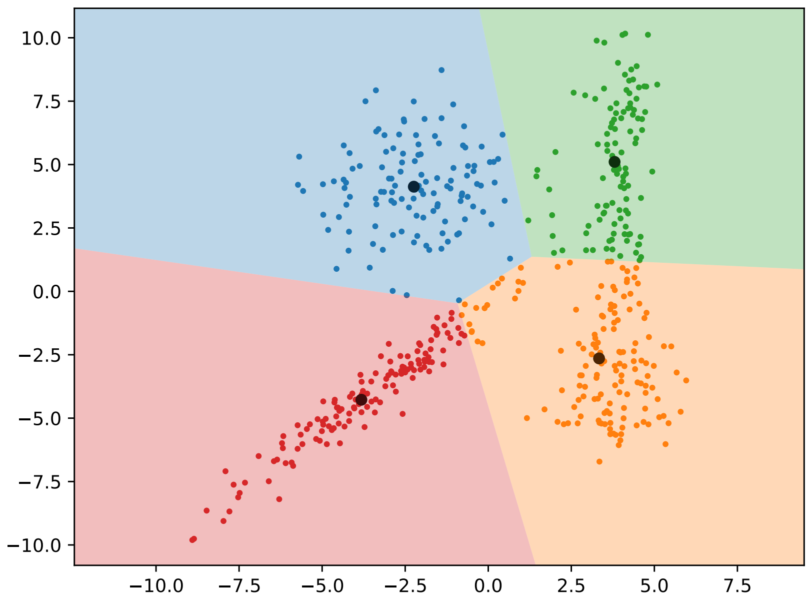


Figure 3: Representation of the training points, the means returned by an execution of the K-means algorithm ($K = 4$) (and the Voronoi partition of the space associated) (distortion = 3237.78, convergence in 7 iterations)

(b) Let's rewrite the EM algorithm under the hypothesis

$$\forall k \in \{1 \dots K\}, \Sigma_k = \sigma_k^2 \cdot \mathrm{Id}_d$$

we have

$$\forall k \in \{1 \dots K\}, \det \Sigma_k = \sigma_k^{2d}$$

E step

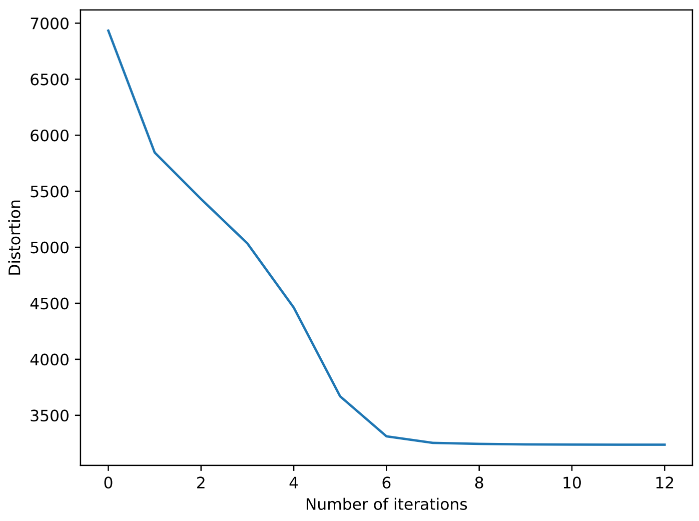


Figure 4: Evolution of the distortion with respect to the number of iterations for an execution of the Kmeans algorithm

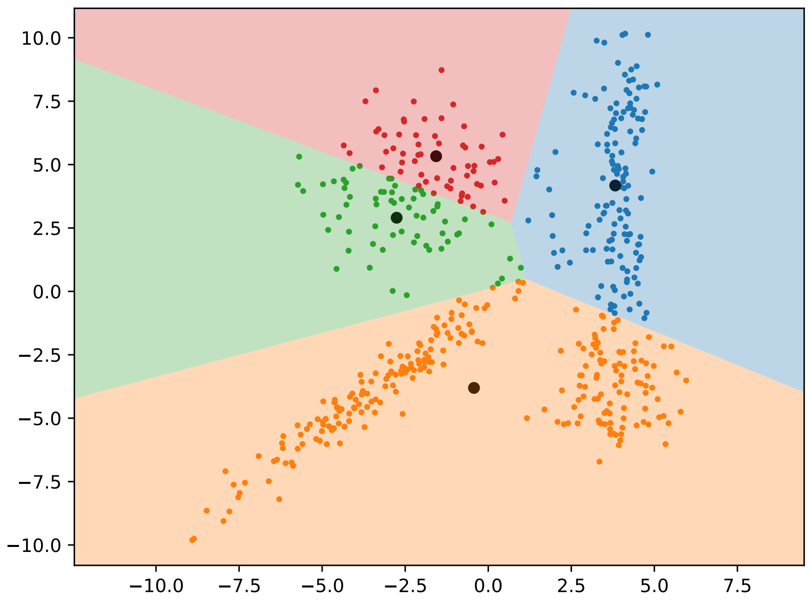


Figure 5: Representation of the training points, the means returned by an execution of the K-means algorithm ($K = 4$) that has converged to a bad local minimum (distortion $= 6378.84$, convergence in 5 iterations)

The probability $p_\theta(z_i = j\,|\,x_i)$ is

$$p_\theta(z_i = j\,|\,x_i) = \frac{\frac{\pi_j}{\sigma_j^d}\exp\left(-\frac{\|x_i - \mu_j\|^2}{2\sigma_j^2}\right)}{\sum_{k=1}^{K}\frac{\pi_k}{\sigma_k^d}\exp\left(-\frac{\|x_i - \mu_k\|^2}{2\sigma_k^2}\right)} = \tau_i^j$$

The complete log likelihood is given by

$$\begin{aligned}\log p_\theta(x,z) = &\sum_{i=1}^{N}\sum_{j=1}^{K} z_i^j \log \pi_j \\ &- \frac{dN}{2}\log 2\pi - \frac{d}{2}\sum_{i=1}^{N}\sum_{j=1}^{K} z_i^j \log \sigma_j^2 \\ &- \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{K} z_i^j \frac{\|x_i-\mu_j\|^2}{\sigma_j^2}\end{aligned}$$

As explained in the lesson, taking the expectation with respect to the conditionnal distribution of Z given X is equivalent to replacing z_i^j by

$$E_{Z|X}[z_i^j] = 0\cdot p(z_i^j=0|x) + 1\cdot p(z_i^j=1|x) = p(z_i=j\,|\,x_i)$$

Hence,

$$\begin{aligned}E_{Z|X}[\log p_\theta(x,z)] = &\sum_{i=1}^{N}\sum_{j=1}^{K} p_\theta(z_i=j\,|\,x_i)\log \pi_j \\ &- \frac{dN}{2}\log 2\pi - \frac{d}{2}\sum_{i=1}^{N}\sum_{j=1}^{K} p_\theta(z_i=j\,|\,x_i)\log \sigma_j^2 \\ &- \sum_{i=1}^{N}\sum_{j=1}^{K} p_\theta(z_i=j\,|\,x_i)\frac{\|x_i-\mu_j\|^2}{2\sigma_j^2}\end{aligned}$$

Which by replacing the expression of $p_\theta(z_i=j\,|\,x_i)$, gives

$$\begin{aligned}E_{Z|X}[\log p_\theta(x,z)] = &\sum_{i=1}^{N}\sum_{j=1}^{K} \tau_i^j(\theta)\log \pi_j \\ &- \frac{dN}{2}\log 2\pi - \frac{d}{2}\sum_{i=1}^{N}\sum_{j=1}^{K} \tau_i^j \log \sigma_j^2 \\ &- \sum_{i=1}^{N}\sum_{j=1}^{K} \tau_i^j(\theta)\frac{\|x_i-\mu_j\|^2}{2\sigma_j^2}\end{aligned} \tag{ECL}$$

M step

In this sum, we can maximize separately over π and (μ,σ^2), because the expression of the expected complete log-likelihood is a sum of a function which only depends on π and a function which only depends on (μ,σ^2)

Maximizing over π

We have the following problem

$$\begin{aligned}&\underset{\pi}{\text{maximize}} && \sum_{i=1}^{N}\sum_{j=1}^{K} \tau_i^j \log \pi_j \\ &\text{subject to} && \pi_1+\cdots+\pi_K = 1\end{aligned}$$

This is a convex problem, with affine constraints, it's feasible, hence by slater's constraint, strong duality holds. If we derive w.r.t π_k the following

$$\sum_{i=1}^{N}\sum_{j=1}^{K}\tau_i^j \log \pi_j + \lambda\left(\sum_{j=1}^{K}\pi_j - 1\right)$$

we obtain

$$\sum_{i=1}^{N}\frac{\tau_i^k}{\pi_k} + \lambda = 0$$

If we let $\hat{\pi}$ such that

$$\forall k \in \{1\dots K\}\,,\ \hat{\pi}_k = \frac{\sum_{i=1}^{N}\tau_i^k}{\sum_{i=1}^{N}\sum_{j=1}^{K}\tau_i^j}$$

and

$$\hat{\lambda} = \sum_{i=1}^{N}\sum_{j=1}^{K}\tau_i^j$$

Then, $(\hat{\pi}, \hat{\lambda})$ statisfy the K.K.T conditions, hence $\hat{\pi}$ is the primal solution of the problem.

Maximizing over μ

The expected complete likelihood function is concave w.r.t (μ, σ^2) (not strictly) (the hessian has 2 non zero eigenvalues which are negative).

The gradient of the expected complete likelihood w.r.t to μ_k is zero when

$$\sum_{i=1}^{N}\tau_i^k\frac{(x_i-\hat{\mu}_k)}{2\sigma_k^2} = 0 \iff \hat{\mu}_k = \frac{\sum_{i=1}^{N}\tau_i^k x_i}{\sum_{i=1}^{N}\tau_i^k}$$

Maximizing over σ^2

The gradient of the expected complete likelihood w.r.t to σ_k^2 is zero when

$$\begin{aligned}-\frac{d}{2}\sum_{i=1}^{N}\frac{\tau_i^k}{\hat{\sigma}_k^2} + \frac{1}{2}\sum_{i=1}^{N}\tau_i^k\frac{\|x_i-\mu_k\|^2}{\hat{\sigma}_k^4} = 0 &\iff -d\sum_{i=1}^{N}\tau_i^k + \sum_{i=1}^{N}\tau_i^k\frac{\|x_i-\mu_k\|^2}{\hat{\sigma}_k^2} = 0\\ &\iff \hat{\sigma}_k^2 = \frac{\sum_{i=1}^{N}\tau_i^k\|x_i-\mu_k\|^2}{d\sum_{i=1}^{N}\tau_i^k}\end{aligned}$$

Results

The figure 6 represents an execution of the algorithm. The points of the training data are represented in the color of the most probable value of their corresponding latent variable. For each gaussian distribution, the mean is represented in black, the ellipse containing 90% of the distribution is also represented. The evolution of the likelihood with respect to the number of iteration is given in figure 7.

The algorithm stops when the absolute difference of the incomplete likelihood is less than a given threshold ϵ from one iteration to the next. (I divide by the number of point so that the

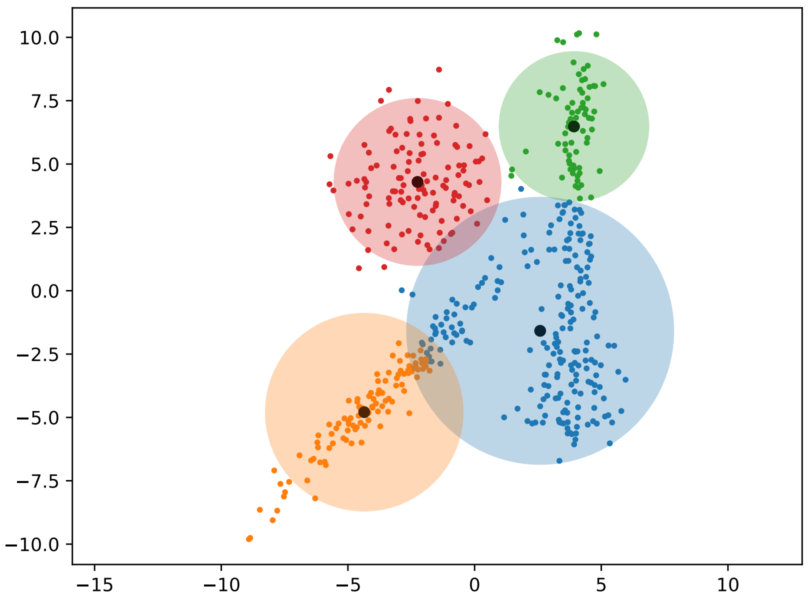


Figure 6: Representation of the training points, the means returned by anw execution of the spherical EM algorithm ($K = 4$). the disks represented contain 90% of the mass of the associated gaussian distribution (likelihood is -2645.53, convergence in 71 iterations, for $\epsilon = 10^{-6}$)

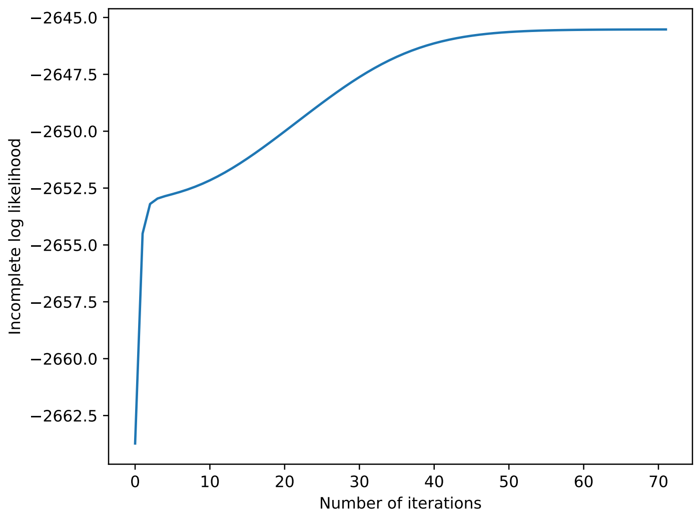
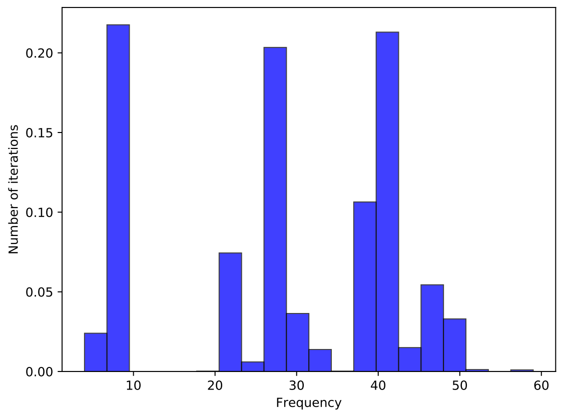


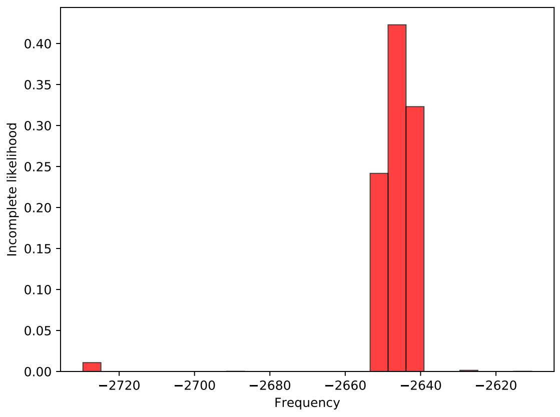
Figure 7: Evolution of the incomplete log likelihood with respect to the number of iterations for an execution of the EM algorithm for gaussian mixture with isotropic covariance

threshold does not depends on the number of point, and also because the threshold in *scikit-learn* corresponds exactly to this quantity, which makes it possible to compare speed for a given threshold (see in conclusion))

I've run 5000 iteration of the algorithm on training sample, and plotted the histogram representing the number of iteration and the estimated complete likelihood it has converged to (Figure 8)). By looking at the histogram, we can see that there are cases where the algorithm sometimes converges to a local minimum that is "far" from the optimum, figure 9 represents one such case



(a) Histogram of the number of iterations



(b) Histogram of the incomplete likelihood

Figure 8: Histogram representing the number of iterations and the expected complete likelihood for 5000 iterations of the spherical EM algorithm with $K = 4$ and a tolerance of 10^{-4} on the training data

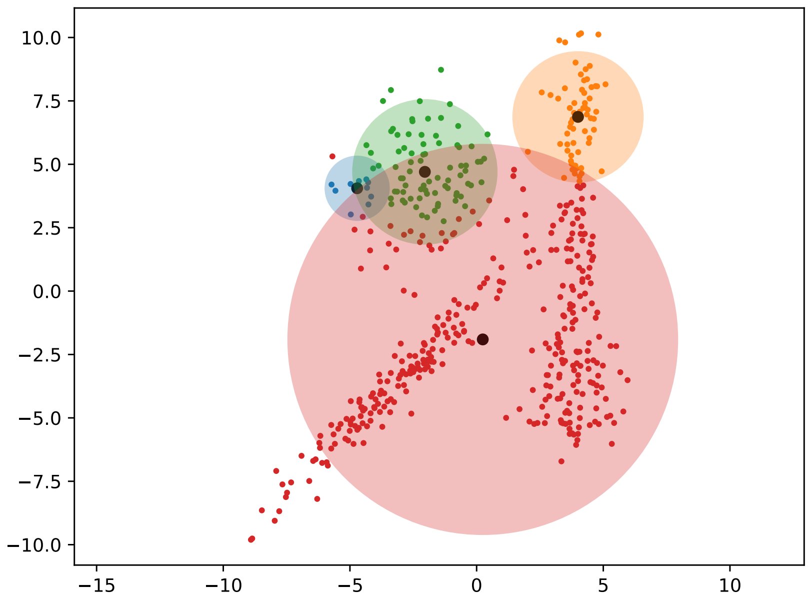


Figure 9: Representation of the training points, the means returned by an execution of the spherical EM algorithm ($K = 4$) that is stuck in a local minimum. the disks represented contain 90% of the mass of the associated gaussian distribution (it fails to converge to the global minimum) (likelihood = -2727.57, convergence in 128 iterations, for $\epsilon = 10^{-6}$)

(c) The result for an execution of the EM algorithm for gaussian mixture with general covariance is represented on figure 10. As for the spherical case, each point of the training example is represented in the color of the most probable value of the corresponding latent variable. For each of the K gaussian distribution, the mean is represented in black, the ellipse containing 90% of the distribution is also represented. The evolution of the incomplete log likelihood is represented in the figure 11.

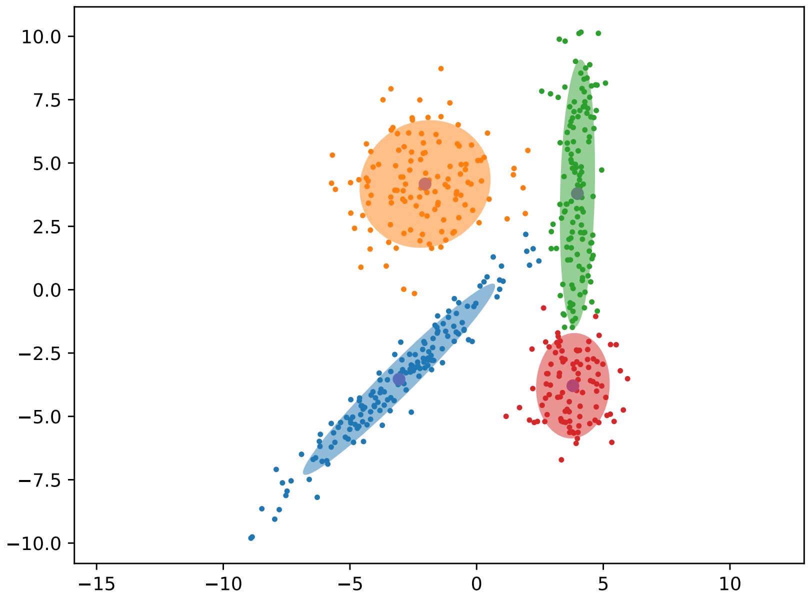


Figure 10: Representation of the training points, the means returned by an execution of the full EM algorithm ($K = 4$). the disks represented contain 90% of the mass of the associated gaussian distribution (likelihood $= -2327.72$, convergence in 26 iterations with $\epsilon = 10^{-6}$)

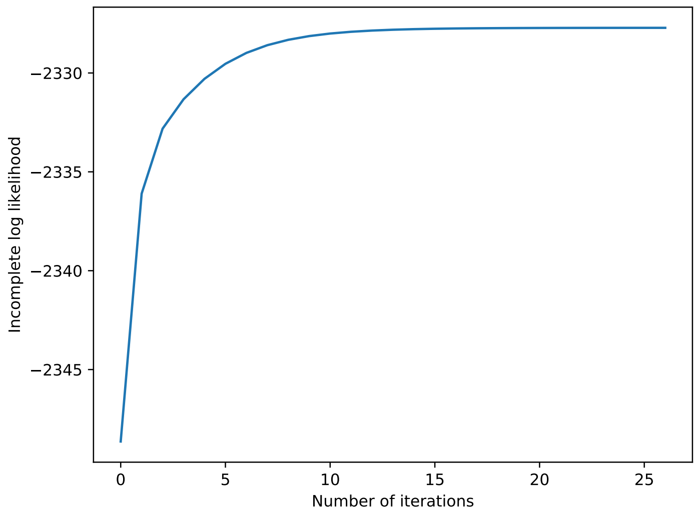
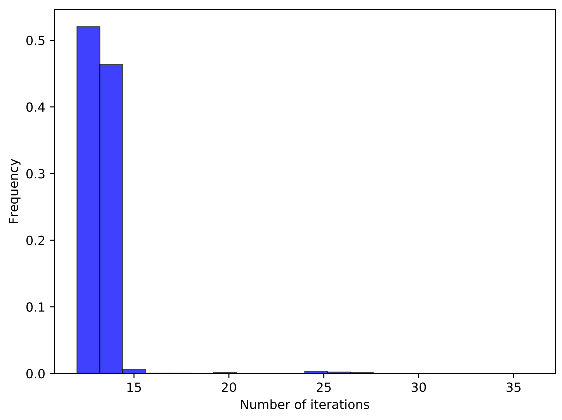


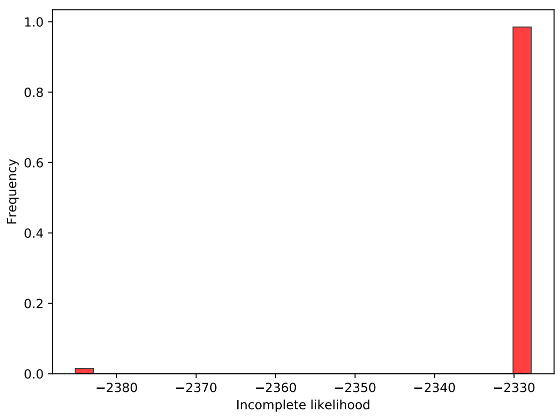
Figure 11: Evolution of the incomplete log likelihood with respect to the number of iterations for an execution of the EM algorithm for gaussian mixture with general covariance (converges a likelihood $= -2327.72$, in 26 iterations with $\epsilon = 10^{-6}$)

I've also run 5000 iteration of the algorithm on training sample, and plotted the histogram representing the number of iteration and the incomplete log likelihood it has converged to (Figure 12)). In the histogram, the estimated frequencies corresponding to each of the two bars are 1.42% and 98.58%. The figure 13 presents a case where the algorithm has converged to a different local minimum (corresponding to the first bar in the histogram).

The algorithm has the same stopping criterion as in the spherical case.



(a) Histogram of the number of iterations



(b) Histogram of the incomplete likelihood

Figure 12: Histogram representing the number of iterations and the expected complete likelihood for 5000 iterations of the full EM algorithm with $K = 4$ and a tolerance of 10^{-4} on the training data

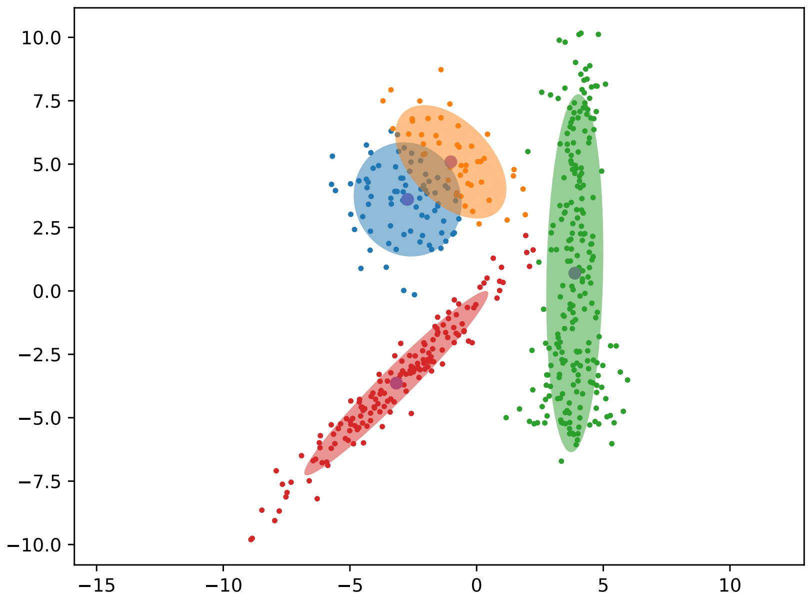


Figure 13: Representation of the training points, the means returned by an execution of the full EM algorithm ($K = 4$). the disks represented contain 90% of the mass of the associated gaussian distribution (likelihood = -2383.64, convergence in 99 iterations, with $\epsilon = 10^{-6}$)

(d) I've trained each algorithm (Kmeans, EM for gaussian mixture model with isotropic and general covariance) 500 times, for each train, I've measured the distortion / incomplete likelihood after training for both the training and the test data. The results are given in the table 1.

	Train		**Test**	
Method	Mean	Std	Mean	Std
Kmeans (Distortion)	3382.60	891.92	3224.83	742.64
Kmeans++ (Distortion)	3293.13	355.21	3152.40	325.07
EM Isotropic GMM (Likelihood)	-2652.89	15.17	-2654.39	23.01
EM General GMM (Likelihood)	-2332.29	17.97	-2417.30	19.77

Table 1: Comparison of the mean and standard deviation of the distortion / likelihood between the train and test data (obtained for 500 trains of the algorithm)

Kmeans

The first thing to notice is that the Kmeans is not really adapted to cluster these king of data, the data has clearly been generated by a weighted sum of gaussian kernels of general covariances.

We see that the standard deviation of the distortion provided by Kmeans is very large even though the mean is close from what we have observed as the best local minimum (3237.78), this shows that the Kmeans algorithm sometimes fails and get stuck in a "bad" local minimum, this agrees with what we have seen in the dedicated question (in particular with the histogram presented in figure 2), we have seen in this histogram that the algorithm behaved well 99.42% of the time on 5000 trains. To have a good probability of getting a good behavior, we could restart the algorithm N times and keep the best distortion. Then non formally, the probability of getting a "good" result will be $1-(1-p)^N$ where p is the probability of having a "good" result.

Despite the fact that it is not really adapted to the data, the output of Kmeans is quite consistent.

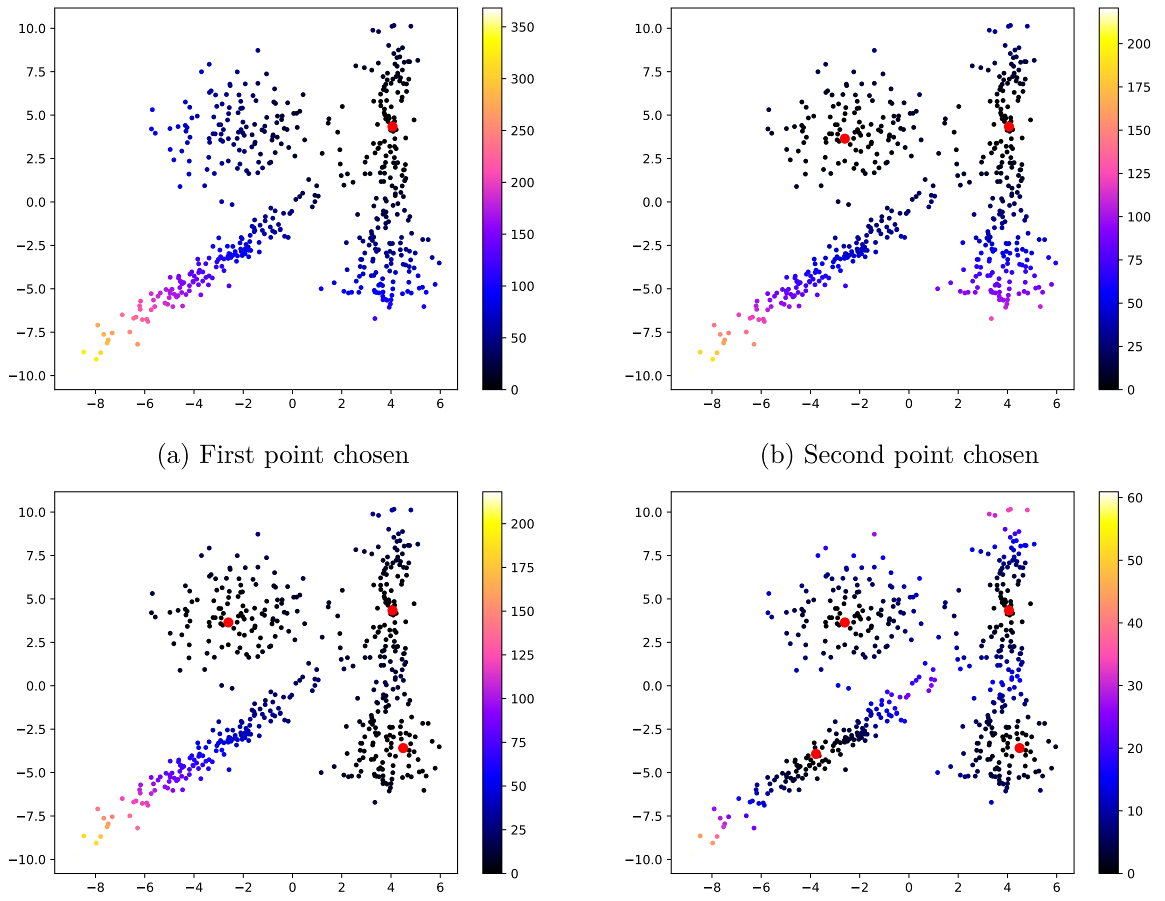
Isotropic vs general covariance

The EM algorithm for gaussian mixture with general covariance performs better than the one for gaussian mixture with isotropic covariance, this is not a surprise, because this model is more general than the other, and the data are particularly adapted to this model (they really seem to be generated by a weighted sum of gaussian kernels that do not have isotropic covariance matrix). Despite that, the EM for isotropic gaussian mixture has a low standard deviation, and also gives a consistent result (see the histogram in figure 8).

We can see that in the case of the general covariance, the EM algorithm converges faster compared to the isotropic case. We can see in the histograms on figure 8 and 12, and by comparing the evolution of the incomplete log likelihood for both cases (7 and 11), we see that in the general covariance case this fonction converges really fast, whereas in the isotropic case, the function goes through unstable points, which traduces that the maximization problem is less regular.

Kmeans++

I've also implemented Kmeans++, because I wanted to know how much it would change the results. Unfortunately, I did not have time to exploit the results. I've put the result of *Kmeans++* in the table 1, and the figure 14 represents the different steps for the initialization of the points with the algorithm, the distances, and hence the probabilities of beeing chosen at the next step, are represented in color scale for each point.



(a) First point chosen

(b) Second point chosen

(c) Third point chosen

(d) Last point chosen

Figure 14: Representation of the points chosen by the K-means++ algorithm and probability of selecting points at each steps

5 Conclusion

I've spent a lot of time improving the code, it uses extensively the numpy broadcasting mechanism and do computations on multidimensional arrays to make the computation very efficient (there's basically no loop). I've measured the performances to be more or less equivalent to scikit-learn for the given data (It's about 10% slower *scikit-learn* for the isotropic case and 10% faster for the general case, when tested on 500 runs of the training data for $\epsilon = 10^{-6}$ on my matchine). I would have loved to optimize more but it would have seriously impacted the readability of my code.

6 Content of the archive

- clustering.py : contains all the clustering method (Kmeans, Kmeans++, EM isotropic GMM, EM general GMM), this is the main class everything depends on.

- comp_likelihood.py : print the values of the distortion and likelihood for each methods (produces the table 1)
- comp_scikit.py : compares the performances of my algorithm and *scikit-learn*'s
- kmeans.py run Kmeans algorithm and plots the result
- spherical_em.py : runs EM for gaussian mixture with isotropic covariance and plots the result
- full_em.py : runs EM for gaussian mixture with general covariance and plots the result
- histograms.py : plots the histograms presented in the figures 2, 8 and 12
- likelihood.py : plots the distortion or likelihood w.r.t the number of iteration
- voronoi.py : function found on the internet (https://gist.github.com/pv/8036995) to plot a voronoi graph given points.