K-means, EM, Mélanges de Gaussiennes, Théorie des graphes **2010/2011**

Cours 3 — 13 octobre

Enseignant: Francis Bach *Scribe: Alexandre Boulc'h, Bertrand Decoster*

Pour parler d'estimation de paramêtres "cachés", les Français et les Anglais utilisent des appelations qui peuvent porter à confusion. Dans un cadre supervisé, les Anglais parleront de *classification*, alors que les Français utiliseront *discrimination*. Dans un contexte non-supervisé, les Anglais parleront cette fois de *clustering*, alors que les Français utiliseront *classification*.

3.1 K-means

K-means est un algorithme de quantification vectorielle (clustering en anglais). K-means est un algorithme de minimisation alternée qui, étant donné un entier K, va chercher à séparer un ensemble de points en K clusters (Figure 3.1).

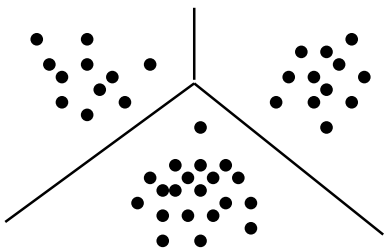

Fig. 3.1. Clustering sur un ensemble de points 2D, 3 clusters.

3.1.1 Notations, Mesure de distorsion

On utilise les notations suivantes :
- Les $x_i \in \mathbb{R}^p$, $i \in \{1, ..., n\}$ sont les points à séparer.
- Les z_i^k sont des variables indicatrices associées aux x_i telles que $z_i^k = 1$ si x_i appartient au cluster k, $z_i^k = 0$ sinon. z est la matrice des z_i^k.
- μ est le vecteur des $\mu_k \in \mathbb{R}^p$, où μ_k est le centre du cluster k.

On définit de plus *la mesure de distorsion* $J(\mu, z)$ par :

$$J(\mu, z) = \sum_{i=1}^{n} \sum_{k=1}^{n} z_i^k \|x_i - \mu_k\|^2$$

3.1.2 Algorithme

Le but de l'algorithme est de minimiser $J(\mu, z)$, il se présente sous la forme d'un algorithme de minimisation alternée :

- Etape 0 : "choisir le vecteur μ"
- Etape 1 : on minimise J par rapport à z : $z_i^k = 1$ pour $k \in \arg\min \|x_i - \mu_k\|$, *ie* on associe à x_i le centre μ_k le plus proche.
- Etape 2 : on minimise J par rapport à μ : $\mu_k = \frac{\sum_i z_i^k x_i}{\sum_i z_i^k}$.
- Etape 3 : retour à l'étape 1 jusqu'à convergence.

Remarque 3.1.1 *L'étape de minimisation par rapport à z revient à répartir les x_i selon les cellules de Voronoï dont les centres sont les μ_k.*

Remarque 3.1.2 *Dans l'étape de minimisation selon μ, μ_k est obtenu en annulant la k-ième coordonnée du gradient de J selon μ.*

3.1.3 Convergence et initialisation

On peut montrer que cet algorithme converge en un nombre fini d'opérations. Cependant la convergence et locale, ce qui pose le problème de l'initialisation.

Une méthode classique consiste à lancer plusieurs fois l'algorithme en prenant les moyennes μ_k aléatoirement à chaque fois, puis on compare leur mesure de distorsion. On choisit la répartition qui possède la distorsion minimale.

Dans le pire des cas, cet algorithme peut se révéler arbitrairement mauvais, mais dans la pratique, il réalise de très bons résultats.

3.1.4 Choix de K

Le choix de K n'est pas universel, on remarque que si on augmente K, la distorsion diminue, et s'annule lorsque chaque point est centre de son cluster. Pour pallier à ce phénomène il possible de rajouter un terme en fonction de K dans l'expression de J, mais là encore son choix est arbitraire.

3.2 EM : Expectation Maximisation

EM est un algorithme qui permet d'étudier un modèle qui possède des variables latentes ou cachées.

Les algorithmes précédents étaient des algorithmes visant à estimer le paramètre θ, maximisant la vraisemblance des $p_\theta(x)$, où x est le vecteur des données observées.

En pratique les données sont parfois perdues, latentes (elles sont là mais ne sont pas observées) ou encore non observées.

3.2.1 Exemple

La densité de représenté sur la figure 3.2.1 s"apparente à une moyenne de deux gaussiennes. Il est donc naturel d'utiliser un modèle de mélange, et d'introduire une variable cachée z, de Bernouilli définissant l'appartenance à l'une ou l'autre des gaussiennes.

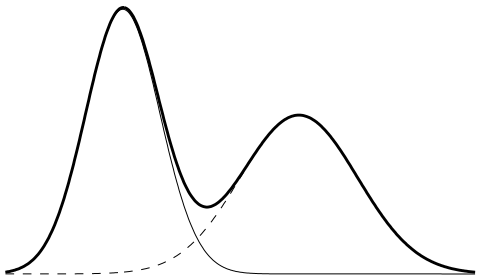

Fig. 3.2. Densité moyenne de deux densités gaussiennes, pour lequel est naturel d'introduire un modèle de mélange.

On a donc : $z \in \{1, 2\}$ et $x|z = i \sim \mathcal{N}(\mu_i, \Sigma_i)$. La densité $p(x)$ est une densité combinaison convexe de densités normales :

$$p(x) = p(x, z = 1) + p(x, z = 2) = p(x|z = 1)p(z = 1) + p(x|z = 2)p(z = 2)$$

C'est un modèle de mélange, un moyen de modéliser simplement des phénomènes compliqués.

3.2.2 Objectif : maximum de vraisemblance

Soit z la variable cachée, x sont les données observées. On suppose que les x_i, $i \in \{1, ..., n\}$ sont des données i.i.d..

Le but est de maximiser la vraisemblance :

$$p_\theta(x) = \prod_i p_\theta(x_i) = \prod_i \sum_z p_\theta(x_i, z)$$

$$\log p_\theta(x) = \sum_i \log \sum_z p_\theta(x_i, z)$$

.

Il est possible d'envisager deux manières différentes de résoudre ce problème :

1. de manière directe, si le problème le permet, par exemple par montée de gradient.
2. en utilisant l'algorithme EM.

3.2.3 Inégalité de Jensen

Nous allons utiliser dans la suite les deux propriétés suivantes :

1. si $f : \mathbb{R} \to \mathbb{R}$ est convexe et si X est une v.a. intégrable :
$$E(f(X)) \geq f(E(X))$$
2. si $f : \mathbb{R} \to \mathbb{R}$ est strictement convexe, on a égalité dans la propriété précédente si et seulement si $X =$ contante p.s.

3.2.4 L'algorithme EM

On introduit dans l'expression de la vraisemblance la fonction $q(z)$ telle que $q(z) \geq 0$ et $\sum_z q(z) = 1$, et on a :

$$\begin{aligned}
\log p_\theta(x) &= \log \sum_z p_\theta(x, z) \\
&= \log \sum_z \left(\frac{p_\theta(x, z)}{q(z)} \right) q(z) \\
&\geq \sum_z q(z) \log \frac{p_\theta(x, z)}{q(z)}, \text{ par l'inégalité de Jensen, et la concavité de } log \\
&= \sum_z q(z) \log p_\theta(x, z) - \sum_z q(z) \log q(z) \\
&= \mathcal{L}(q, \theta)
\end{aligned}$$

avec égalité ssi $q(z) = \frac{p_\theta(x,z)}{\sum_z p_\theta(x,z)} = p_\theta(z|x)$

Proposition 3.1 *$\forall \theta$, $\forall q$ $\log p_\theta(x) \geq \mathcal{L}(q, \theta)$ avec égalité si et seulement si $q(z) = p_\theta(z|x)$.*

Exemple 3.2.1 *On a créé une fonction qui est toujours en dessous la fonction $log(p_\theta(x))$*

L'algorithme EM est un algorithme de maximisation alternée par rapport q et θ.

On initialise à θ_0 puis on itère pour $t > 0$, en alternant les étapes suivantes jusqu'à convergence :

- E-Step : $q_{t+1} \in argmax_q(\mathcal{L}(q, \theta_t))$
- M-Step : $\theta_{t+1} \in argmax_\theta(\mathcal{L}(q_{t+1}, \theta))$

Les propriétés de l'algorithme

- C'est un algorithme de montée : $\forall t$ $log(p_{\theta_t}) \geq log(p_{\theta_{t-1}})$.
- L'algorithme converge.
- Cependant il ne converge pas globalement mais vers un maximum local car on est dans un cas non convexe. Souvent, l'optimum global est infini.
- Comme pour K-means, pour avoir un résultat plus sûr, on réitère plusieurs fois l'algorithme et on choisit l'essai avec la meileure vraisemblance.

Initialisation Comme EM donne un maximum local, il est intéressant de choisir un θ_0 relativement proche de la solution finale. Il est usuel d'initialiser EM par un K-means, la solution de K-means fournissant le θ_0.

La Recette EM On rappelle le but qui est de maximiser la vraisemblance *incomplète* $log(p_\theta(x))$. Pour cela on peut utiliser la recette suivante :

1. Écrire la vraisemblance *complète* $l_c = log(p_\theta(x, z))$.
2. E-Step : écrire l'espérance de la vraisemblance $E[q(z)p_\theta(x, z)]$, voulant $q(z) = p_\theta(z|x)$.
3. M-Step : maximiser par rapport à θ.

3.2.5 Mélanges de Gaussiennes

Soient les couples (x_i, z_i), pour $i \in \{1, ..., n\}$ avec $x_i \in \mathbb{R}^p$ i.i.d, $z_i \sim$ Multinomiale(Π) i.i.d. et $(x_i|z_i = q) \sim \mathcal{N}(\mu_q, \Sigma_q)$.

Calcul de $p(z|x)$ On utilise la formule de Bayes pour exprimer $p(x_i)$:

$$\begin{aligned} p(x_i) &= \sum_{z_i} p(x_i, z_i) = \sum_{z_i} p(x_i|z_i)p(z_i) \\ &= \sum_{q=1}^{k} p(x_i|z_i = q)p(z_i = q) \end{aligned}$$

Puis on exprime $p(z|x)$:

$$\begin{aligned} p(z_i = q|x_i) &= \frac{p(x_i|z_i = q)p(z_i = q)}{p(x_i)} \\ &\propto p(x_i|z_i = q)p(z_i = q) \\ &= \frac{\Pi_q \mathcal{N}(x_i|\mu_q, \Sigma_q)}{\sum_{l=1}^{k} \Pi_l \mathcal{N}(x_i|\mu_l, \Sigma_l)} \\ &= \tau_i^q(\theta) \end{aligned}$$

On rappelle que $\mathcal{N}(x_i|\mu, \Sigma) = -\frac{1}{(2\pi)^{\frac{d}{2}}|\Sigma|^{\frac{1}{2}}} exp(-\frac{1}{2}(x - \mu)^T\Sigma^{-1}(x - \mu))$.

Vraisemblance complète

$$\begin{aligned} l_c = \log p_\theta(x,z) &= \sum_{i=1}^{n} \log p_\theta(x_i, z_i) \\ &= \sum_{i=1}^{n} \log(p(z_i)p(x_i|z_i)) \\ &= \sum_{i=1}^{n}\sum_{q=1}^{k} log(p(z_i=q)) + log(p(x_i|z_i=q)) \\ &= \sum_{i=1}^{n}\sum_{q=1}^{k} log(\Pi_q) + log(\frac{1}{(2\pi)^{\frac{d}{2}}} + log(\frac{1}{|\Sigma_q|^{\frac{1}{2}}}) - \frac{1}{2}(x_i-\mu_q)^T\Sigma_q^{-1}(x_i-\mu_q)) \end{aligned}$$

On écrit maintenant $E[p(z|x)]$, quantité que l'on va maximiser par la recette EM.

$$E[p(z|x)] = \sum_{i=1}^{n}\sum_{q=1}^{k} \tau_i^q(\theta_{précédent})(log(\Pi_q) + log(\frac{1}{(2\pi)^{\frac{k}{2}}} + log(\frac{1}{|\Sigma_q|^{\frac{1}{2}}}) - \frac{1}{2}(x_i-\mu_q)^T\Sigma_q^{-1}(x_i-\mu_q)))$$

E-Step La première étape de l'algorithme EM est de maximiser $E[p(z|x)]$ par rapport à Π. En maximisant selon les méthodes vues au premier chapitre, on obtient :

$$\Pi_q = \frac{\sum_i \tau_i^q}{\sum_i \sum_l \tau_i^l} = \frac{1}{n}\sum_{i=1}^{n} \tau_i^q.$$

M-Step On cherche ici à maximiser par rapport à μ et Σ. En prenant les gradients selon les μ_q puis selon les Σ_q, on obtient que :

$$\mu_q(t+1) = \frac{\sum_i \tau_i^q(t)x_i}{\sum_i \tau_i^q(t)}$$

$$\Sigma_q(t+1) = \frac{\sum_i \tau_i^q(t)(x_i-\mu_q(t+1))(x_i-\mu_q(t+1))^T}{\sum_i \tau_i^q(t)}$$

Le M-Step dans EM correspond à l'estimation des moyennes dans K-means.

3.3 Théorie des Graphes

3.3.1 Graphe

Définition 3.2 *Soit V un ensemble de sommets (vertex, vertices, noeuds, variables aléatoires, ...). Soit $E \subset V \times V$ l'ensemble des arètes (arcs, edges). Un graphe G est couple (V, E).*

Dans ce cours on ne considérera que des graphes sans self-loop, c'est à dire qu'on autorisera pas d'arète ayant ses extrémités identiques.

3.3.2 Les graphes non orientés

Définition 3.3 *$G = (V, E)$ est dit graphe non-orienté lorsque $\forall (u, v) \in V \times V$, $u \neq v$ alors $(u, v) \in E \iff (v, u) \in E$ (Figure 3.3.2).*

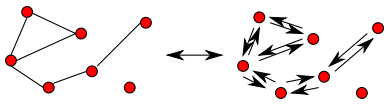

Fig. 3.3. Deux manières de représenter un graphe non-orienté.

Définition 3.4 *L'ensemble des voisins de $u \in V$ dans G est noté $\mathcal{N}(u)$, tel que $\mathcal{N}(u) = \{v \in V, (v, u) \in E\}$ (Figure 3.3.2).*

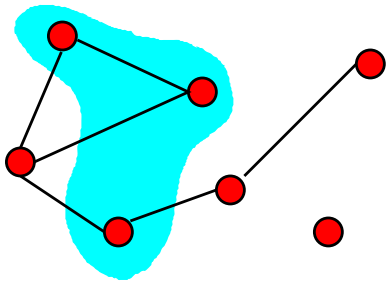

Fig. 3.4. Un sommet et ses voisins.

Définition 3.5 *Une clique est un sous ensemble de V totalement connecté (de cardinalité > 1) ou un singleton (Figure 3.3.2).*

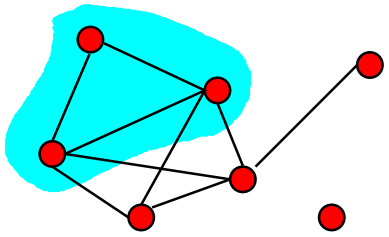

Fig. 3.5. Une clique.

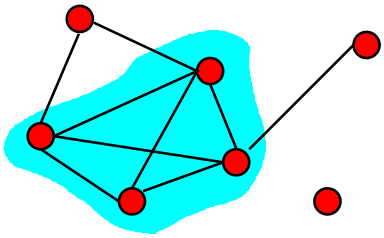

Fig. 3.6. Une clique maximale.

Définition 3.6 *Une clique maximale est une clique maximale pour l'ordre de l'inclusion i.e si C est une clique maximale, $\nexists v \in V$, tq $v \notin V$ et $v \cup C$ soit une clique (Figure 3.3.2).*

Définition 3.7 *Un chemin est une suite d'éléments de V, consécutivement voisins et globalements distincts (Figure 3.3.2).*

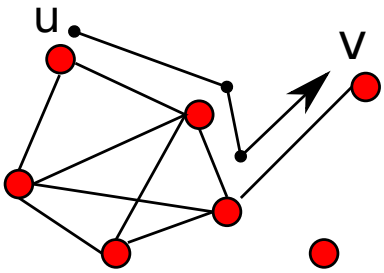

Fig. 3.7. Un chemin de u vers v.

Définition 3.8 *Un cycle est une suite d'éléments de V, $v_1, ..., v_n$ telle que :*
- $\forall i (v_i, v_{i+1}) \in E$
- $v_1 = v_n$
- $\forall i, j \ v_i \neq v_j$ *sauf si* $\{i, j\} = \{1, n\}$

Définition 3.9 *Soit A, B, C des ensembles disjoints de V. C sépare A et B ssi tous les chemins de A vers B passent par C (Figure 3.3.2).*

Définition 3.10 *Une composante connexe est une classe d'équivalence pour la relation d'équivalence $u \mathcal{R} v \iff \exists$ un chemin de u vers v (Figure 3.3.2).*

Dans ce cours : on considère qu'il y a une seule composante connexe. Dans le cas contraire, on les traite de manière indépendante.

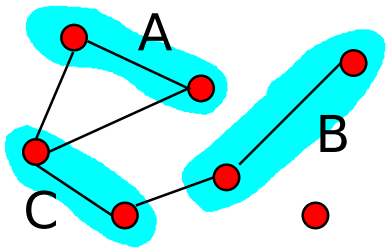

Fig. 3.8. C sépare A et B.

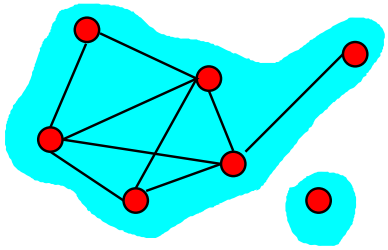

Fig. 3.9. Un graphe avec 2 composantes connexes.

3.3.3 Les graphes orientés

Définition 3.11 *u est un parent de $v \iff (u,v) \in E$.*

Définition 3.12 *u est un enfant de $v \iff (v,u) \in E$.*

Définition 3.13 *u est un ancètre de $v \iff \exists$ un chemin de u vers v.*

Définition 3.14 *u est un descendant de $v \iff \exists$ un chemin de v vers u.*

Définition 3.15 *Un cycle est une séquence $u_1, ..., u_n$ (Figure 3.3.3) tq :*
- $u_1 = u_n$
- $\forall i \in \{1, ..., n-1\}$, u_i *est parent de* u_{i+1}
- $u_1, ..., u_{n-1}$ *sont distincts*

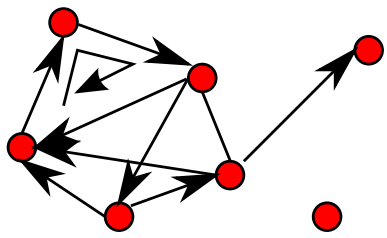

Fig. 3.10. Un graphe orienté avec un cycle.

Définition 3.16 *Un DAG (Directed Acyclic Graph) est un graphe qui ne contient pas de cycle.*

Définition 3.17 *Un ordre topologique est une ordre tel que les parents arrivent avant les enfants :*
$I : V \to \mathbb{N}$ *tq* $\forall (u, v) \in E \Longrightarrow I(u) < I(v)$.

Proposition 3.18 *G est un DAG* $\Longleftrightarrow$ $\exists$ *un ordre topologique sur G.*