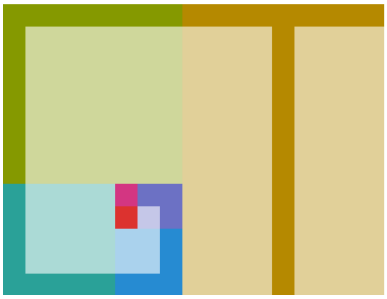


Maths - Bio

# Analyse en groupement

Guilhem Doulcier

Séances des 20 et 27 novembre novembre 2013

# Table des matières



# Avant propos

Ce document est issu de notes qui avaient été mises en ligne à l'adresse

`www.eleves.ens.fr/home/doulcier/projects/math/clustering.html`

mais a subi quelques modifications depuis. La page Web ci-dessus contient des figures interactives qu'il n'a pas été possible d'intégrer dans le pdf – elles ont donc été remplacées par des liens. Mais comme la durée de vie de la page Web en question n'est pas très bien définie, ces figures sont également hébergées sur le site du GT. Elles sont regroupées sur la page :

`www.gt-mathsbio.biologie.ens.fr/interactif/clustering`

De plus, les séances d'introduction à l'apprentissage d'Alexis Jacq traitaient de questions similaires à celles présentées dans ce document, mais avec un point de vue très différent. Les deux points de vue (celui de ce document, qui balaie très large, et celui des notes d'Alexis Jacq, plus précis) se complètent et il est donc possible de lire les deux en parallèle.

# 1 Introduction : Pourquoi vouloir classer ?

La capacité d'abstraction, c'est à dire la capacité à former une pensée abstraite à partir d'observations individuelles en "oubliant" leurs particularités pour en tirer des principes généraux, est un des fondements de la méthode empirique et est nécessaire à toute entreprise scientifique. Il s'agit également d'un processus cognitif complexe et difficilement formalisable.

L'analyse en groupement (*cluster analysis* en anglais) est un domaine à l'interface entre mathématiques (essentiellement par les statistiques) et informatique (essentiellement par l'intelligence artificielle) qui s'intéresse à l'apprentissage automatique non supervisé. On entend par là le développement de méthodes *automatisables* permettant le partitionnement d'un jeu de données en un certain nombre de sous ensembles partageant des propriétés communes.

L'analyse en groupement est un problème dit "non supervisé", c'est à dire que l'on suppose que l'on ne possède pas de connaissances préalables sur les données (contrairement au cas de l'apprentissage supervisé, où l'on dispose d'un certain nombre d'exemples de réalisation de la tâche à apprendre). L'enjeu est seulement de mettre en évidence la structure sous-jacente du jeu de données. Déceler ainsi des régularités dans un ensemble d'observations est la première étape pour la formation d'une pensée abstraite à son propos. Pour le dire simplement : *c'est la première étape pour dire quelque chose d'intéressant à propos d'un jeu de données complexe.*

Ces méthodes sont extensivement utilisées en biologie pour répondre à de nombreux problèmes : la comparaison de séquences en phylogénie et en taxonomie ou de profils d'expressions en transcriptomique, la comparaison de communautés en écologie, mais aussi l'analyse d'images.

Au cours de ces deux séances d'introduction à l'analyse en groupement je vous propose tout d'abord de réfléchir un instant sur la notion de similarité : comment formaliser la notion intuitive de "deux objets se ressemblent plus ou moins" ? Ensuite, nous passerons en revue les principes de quelques grandes familles d'algorithmes de classification non supervisée et enfin nous poserons la question de l'évaluation d'un partitionnement de données.

# 2 Distance et Similarités

La première chose qu'il nous faut lorsque l'on souhaite classer des objets dans différents groupes est une mesure de leur similarité.

## 2.1 Distance métriques

On appelle distance sur un ensemble $E$ toute application $d$ définie sur l'ensemble $E \times E$, à valeurs dans l'ensemble $\mathbb{R}^+$ des réels positifs, $E \times E \to \mathbb{R}^+$ et vérifiant les propriétés suivantes :

**Symétrie**

$$\forall a, b \in E,\ d(a,b) = d(b,a)$$

**Séparation**

$$\forall a, b \in E,\ d(a,b) = 0 \Leftrightarrow a = b$$

**Inégalité triangulaire (ou sub-additivité)**

$$\forall a, b, c \in E,\ d(a,c) \leq d(a,b) + d(b,c)$$

Ces trois propriétés assurent que la grandeur que l'on appelle "distance" en mathématiques correspond bien à l'idée "intuitive" que l'on se fait d'une distance dans la vie de tous les jours.

### 2.1.1 Distances sur des espaces vectoriels normés

Il est possible de définir de nombreuses applications vérifiant les propriétés d'une distance. Sur un espace vectoriel normé, on peut utiliser la norme en posant :

$$\forall (x, y) \in E \times E,\ d(x,y) = \|y - x\| .$$

On se place dans le cas d'un espace en dimension finie (auquel on se limitera ici), il est possible de définir plusieurs normes :

**La norme euclidienne (ou norme 2)** $\sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$

**La norme 1** $\sum_{i=1}^{n} |x_i - y_i|$

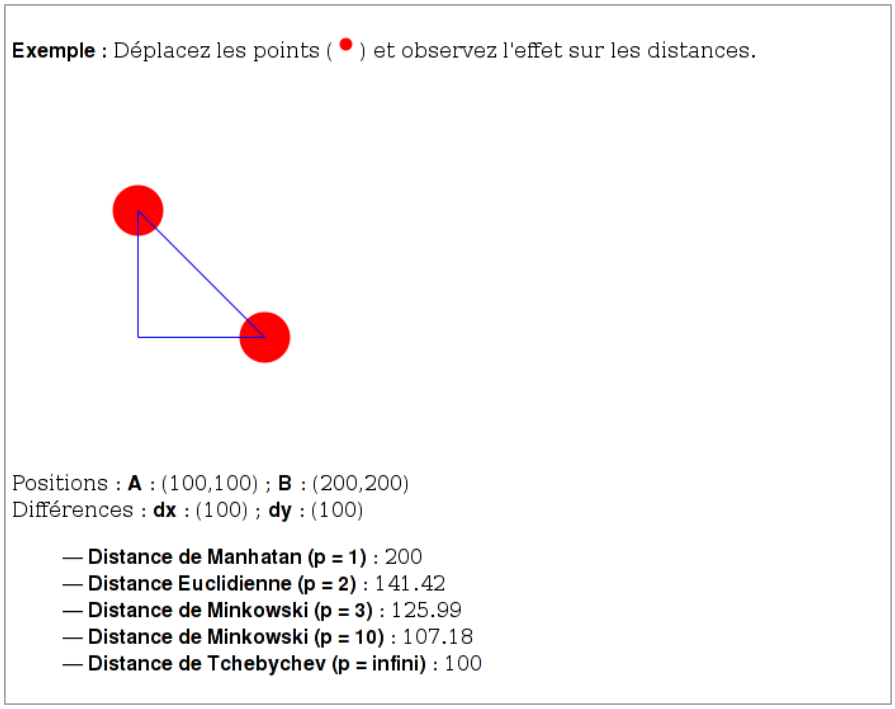
**La norme p** $\sqrt[p]{\sum_{i=1}^{n} |x_i - y_i|^p}$

**La norme infinie** $\sup_{1 \leq i \leq n} |x_i - y_i|$

Bien que ces normes soient équivalentes en un certain sens mathématique (car nous sommes en dimension finie), elle n'ont pas le même sens physique. Il faut donc choisir celle que l'on utilise avec attention en fonction de son jeu de données. D'une manière générale, on remarque que lorsque $p$ augmente, la mesure de la distance donne plus de poids à la dimension selon laquelle la différence de coordonnées est

la plus grande.

De ce fait, la distance de Manhattan ($p = 1$) peut être utilisée pour réduire l'effet de points aberrants dans une des dimensions alors que la distance de Tchebychev ($p = \infty$) peut être utilisée pour déterminer si deux points sont différents sans ce soucier de selon quelle dimension.




Version interactive :
www.gt-mathsbio.biologie.ens.fr/interactif/clustering/distances.html

### 2.1.2 Normaliser et standardiser des données

Les différentes normes ne distinguent pas les différentes dimensions, cependant quand on calcule la distance entre deux points c'est bien la dimension dans laquelle ceux ci sont le plus éloignés qui a le plus de poids dans la somme.

Ce fait souligne l'importance d'avoir des unités cohérentes dans un jeu de données (de façon évidente, si l'une des dimension a ses coordonnées en mètres et l'autre en kilomètres, la mesure de distance n'a pas de sens).

Comment faire lorsque les dimensions représentent des choses différentes ? Par exemple la taille moyenne à l'âge adulte et le nombre moyen de descendants : deux

individus sont ils plus différents s'ils ont un mètre ou un descendant de différence ?

Une première solution à ce problème fait appel à la normalisation et à la standardisation des données.

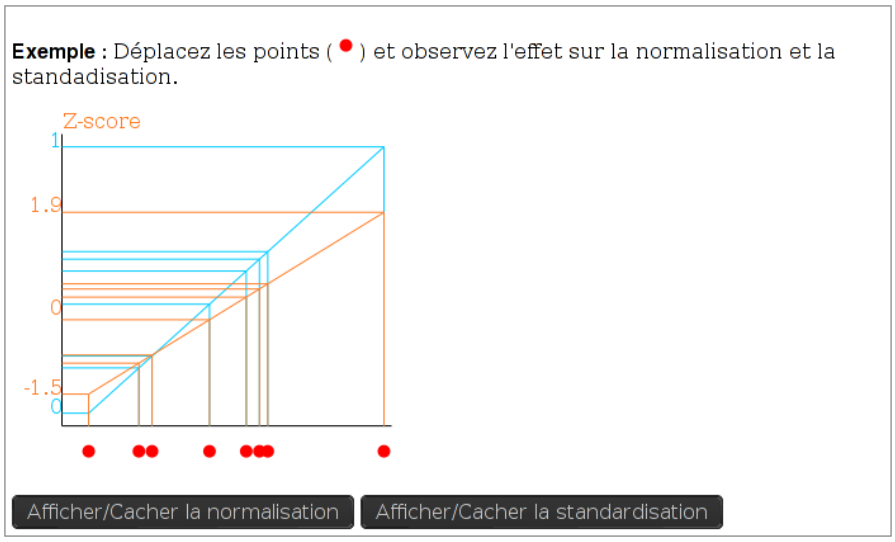
- **Normaliser des données** consiste à ramener toutes les coordonnées dans l'intervalle $[0, 1]$, de façon linéaire, en faisant correspondre le minimum à 0 et le maximum à 1. La valeur normalisée de $x$ est ainsi :

$$N(x) = \frac{x - x_{min}}{x_{max} - x_{min}} .$$

  Si l'on réalise une normalisation, il faut apporter une attention particulière aux valeurs dites aberrantes (*outliers*), en les enlevant par exemple, au risque d'associer l'intervalle de valeurs d'intérêt à un très petit sous-intervalle de $[0, 1]$ (perte de précision).

- **Standardiser des données** consiste à soustraire à toutes les coordonnées la moyenne du jeu de données selon cette dimension et à diviser par l'écart type selon cette dimension. On obtient ce que l'on appelle un *z-score* qui mesure le nombre d'écarts types qui séparent le point considéré de la moyenne. L'avantage de la standardisation est que le *z-score* est directement relié aux pourcentiles si la distribution des données suit une loi normale. La valeur standardisée de $x$ est ainsi :

$$Z(x) = \frac{x - \mu}{\sigma} .$$




Version interactive :

www.gt-mathsbio.biologie.ens.fr/interactif/clustering/normaliser.html

### 2.1.3 Le fléau de la dimension

On peut pratiquement toujours se ramener à des distances sur des espaces vectoriels normés en représentant nos données sous la formes de vecteurs.

- Pour une série temporelle $(X_t)_{0\geq t\geq d}$, on peut prendre une dimension par point d'échantillonnage $(x_0, x_1 \dots x_d)$.
- Pour des images (en noir et blanc) une dimension par pixel (ou 3 dimensions par pixels si l'image est en couleurs).
- Pour des données de micro-array on peut prendre une dimension par gène dont on a la valeur d'expression (entre $10^2$ et $10^4$)
- Pour des données de génotypage on peut prendre une dimension par SNP (typiquement $10^6$).

Cependant, en général on évite de travailler avec des données en grande dimension car on se heurte à un ensemble de problèmes que l'on regroupe habituellement sous le terme de "fléau de la dimension" (*curse of dimensionality* [Verleysen and François, 2005]) et qui proviennent du fait que les espaces géométriques de grande dimension ont parfois des propriétés contre-intuitives :

**Arguments mathématiques :** La majorité des outils sont adaptés à des espaces de faible dimensions. L'exemple phare étant les problèmes d'optimisation (maximiser ou minimiser une fonction en fonction de ses paramètres) qui sont plus compliqués à résoudre en grande dimension. Comme nous le verrons certaines techniques de groupement sont en fait des problèmes d'optimisation ($k$-mean).

De plus, le nombre de points de données nécessaires pour échantillonner un espace avec la même précision augmente exponentiellement avec le nombre de dimensions. Par exemple, imaginons que l'on souhaite échantillonner 10% de points uniformément distribués dans un hypercube unitaire de dimension $d$.

- Si $d = 1$, l'hypercube est un segment unitaire, prendre 10% des points revient à prendre un dixième de ce segment, soit un hypercube de dimension 1 et de coté 0.1.
- Si $d = 2$, l'hypercube est un carré unitaire, prendre 10% des points revient à prendre un hypercube de dimension 2 de coté $\approx 0.31$, cela signifie qu'il faut échantillonner 31% de chaque dimension (alors que 10% de l'unique dimension suffisait plus tôt).

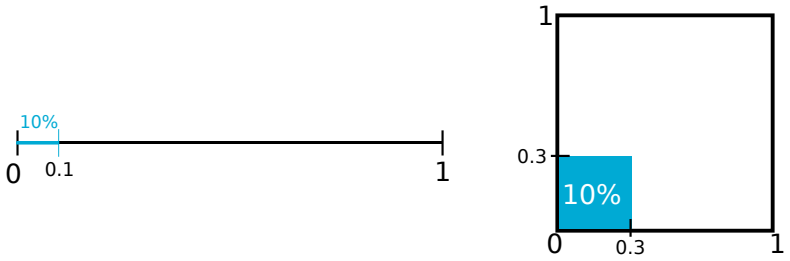



FIGURE 1 – Le fléau de la dimension : pour explorer 10% d'un espace, il faut en échantillonner 10% selon son unique dimension pour un espace de dimension 1, mais 33% selon chaque dimension pour un espace de dimension 2, etc.

De manière générale, si $r$ est la proportion des points uniformément répartis dans l'hypercube unitaire de dimension $d$ que l'on souhaite échantillonner, if faut prendre un hypercube de même dimension et de coté $I$ tel que :

$$I^d = r \Leftrightarrow I = \sqrt[d]{r}\,.$$

Ainsi, en dimension 10, il faut échantillonner 79% de toutes les dimensions pour avoir 10% des points.

Enfin, Il y a un effet de concentration des normes : dans un espace de grande dimension, les distances entre des paires de points tendent à sembler identiques.

Si l'on considère deux points dans l'hypercube unitaire, la distance maximale qui les sépare est donnée par la norme du vecteur $(1, 1, \ldots, 1)$. Pour la distance euclidienne on a :

$$\|(1, 1, \ldots, 1)\|_2 = \sqrt{d}\,.$$

Ainsi, la distance maximale entre deux points n'augmente pas linéairement avec la dimension. On peut aller plus loin : il est possible de montrer [Beyer et al., 1999] que pour tout vecteur aléatoire $x \in \mathbb{R}^d$ entouré d'autres points $y_i$,

$$\lim_{d \to +\infty} \frac{\max_i(d(x, y_i)) - \min_i(d(x, y_i))}{\min_i(d(x, y_i))} = 0\,,$$

ce qui signifie que la distance d'un point à son plus proche et plus lointain voisin tendent à être similaires quand la dimension est très grande.

**Arguments de pertinence scientifique :** Quand le nombre de dimensions est grand, il est difficile de distinguer les variables qui ont un intérêt biologique de celles qui n'apportent que du bruit. D'autant plus que les mesures de distance, nous l'avons vu, ne font pas de hiérarchie entre les dimensions. Ainsi, la première

méthode pour réduire la dimension d'un jeu de données est d'utiliser les connaissance préalable sur le phénomène afin de ne conserver que les dimensions pertinentes. Par exemple en enlevant les gènes constitutifs pour comparer des profils d'expression, éventuellement après avoir vérifié qu'ils n'étaient pas statistiquement différents d'un échantillon à l'autre.

**Argument technique :** Réduire la dimension permet aussi de faciliter la visualisation, de réduire la mémoire nécessaire pour stocker un jeu de données et d'accélérer les analyses.

**Conclusion :** Il est intéressant de réduire le nombre de dimensions d'un échantillon (on parle de *feature extraction*). Cela peut être fait par une Analyse en composante principales (ACP), NMDS, par détection de contours en analyse d'image, en réalisant une analyse spectrale d'une série temporelle (par exemple en prenant la décomposition en série de Fourier).

## 2.2 Similarités

Dans certains cas, on peut utiliser des mesures de similarité qui ne sont pas des distances (au sens de la définition que nous avons donné plus haut). Dans ce cas là, il faut être prudent car ces mesures n'ont pas nécessairement les propriétés de séparation, symétrie et sub-additivité caractéristiques des distances et qui font que ces mesures se comportent de la même façon que l'idée intuitive que l'on se fait d'une distance.

### 2.2.1 Similarité cosinus et coefficient de corrélation

Dans certaines situations, la distance classique entre deux objets ne nous permet pas vraiment de rendre compte de l'idée intuitive que l'on se fait de leur dissimilarité, par exemple dans le cas de deux profils d'expressions de gènes anti-corrélés ou identiques à une translation près : on souhaiterait qu'il soient relativement proches car leurs comportements sont qualitativement proches même si leurs points sont éloignés (en terme de norme).

Pour résoudre ce problème on peut être amené à utiliser la similarité cosinus ou le coefficient de corrélation de Pearson qui donnent une mesure de la corrélation linéaire entre les deux vecteurs.

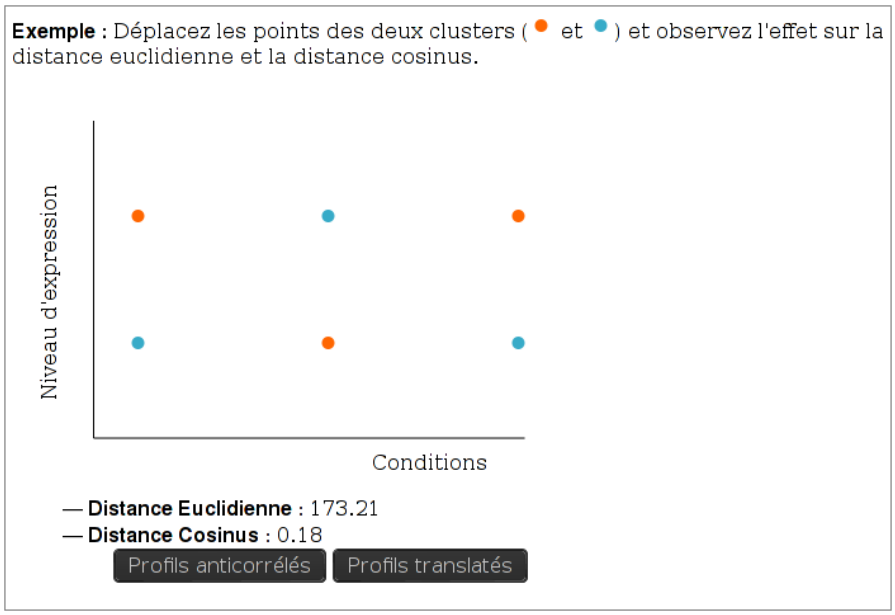
$$\cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2 \times \sum_{i=1}^{n} (B_i)^2}}$$

$$\rho = \frac{cov(A,B)}{\sigma_A \times \sigma_B} = \frac{\sum_{i=1}^{n} (A_i - \mu_A) \times (B_i - \mu_B)}{\sqrt{\sum_{i=1}^{n} (A_i - \mu_A)^2 \times \sum_{i=1}^{n} (B_i - \mu_B)^2}}$$

On remarque que ces deux mesures sont identiques si la variable est centrée (moyenne nulle), par exemple par le truchement de la standardisation !

Ce qui nous intéresse cependant c'est une mesure de la dissimilarité (qui vaut 0 quand les objets sont identiques). Cela se fait simplement en considérant $d_C(A,B) = 1 - \cos(\theta)$ (distance cosinus) ou $d_p(A,B) = 1 - \rho_{A,B}$ (distance de Pearson).

Il faut toutefois faire attention avec ces mesures : celles ci ne possèdent pas les propriétés de séparation (deux vecteurs colinéaires et de même sens auront une similarité cosinus de 1 qu'ils soient égaux ou non) ni de sub-additivité (on peut obtenir la propriété de sub-additivité en utilisant la similarité angulaire plutôt que cosinus).




Version interactive :

www.gt-mathsbio.biologie.ens.fr/interactif/clustering/cosinus.html

### 2.2.2 Scores de Similarité

Entre deux séquences, il est très commun d'utiliser un sore de d'alignement donné par une matrice de substitution (type BLOSUM62).

# 3 Méthodes de partitionnement de données

## 3.1 Méthodes fondées sur la connectivité

### 3.1.1 Principe

Ces méthodes sont fondées sur le postulat que des objets appartenant à un même groupe sont plus proches entre eux que des membres d'autres groupes (les groupes sont plus "connectés" entre eux). Elles sont la base théorique de nombreuses autres méthodes mais ne sont plus très utilisées aujourd'hui.

Le résultat est un dendogramme : un arbre dont les feuilles sont les objets du jeu de données et chaque noeud correspond à un groupe d'objets. On parle ainsi de "classification hiérarchique".

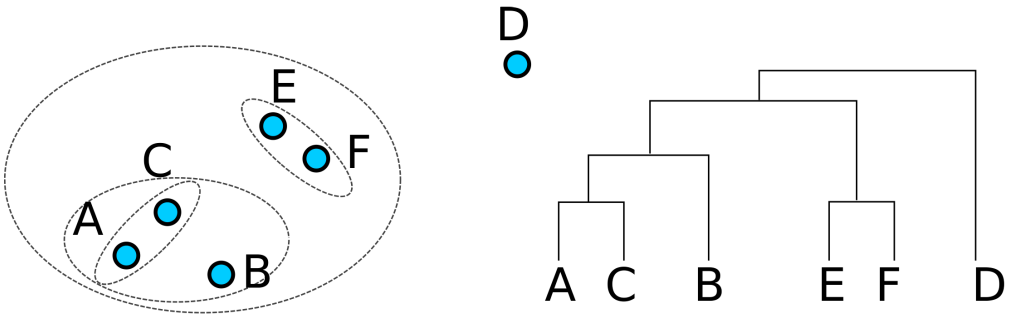
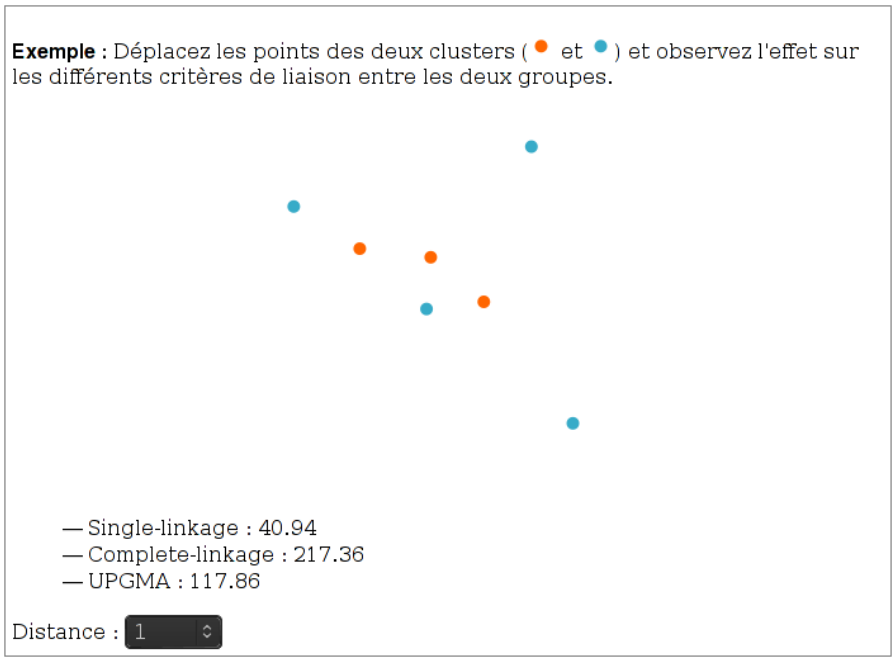



FIGURE 2 – Partitionnement hiérarchique. A gauche, le jeu de données partitionné ; à droite, le dendograme correspondant.

Pour utiliser ces méthodes, il est nécessaire d'avoir défini :

- Une distance entre deux objets,
- Un critère de liaison, c'est à dire un moyen de calculer la distance à un groupe constitué de plus d'un objet. Exemples :
  - Single-linkage clustering (la distance minimale à un objet du groupe)
  - Complete linkage clustering (la distance maximale à un objet du groupe)

– UPGMA (“Unweighted Pair Group Method with Arithmetic Mean” – la distance moyenne aux objets du groupe).




Version interactive :

www.gt-mathsbio.biologie.ens.fr/interactif/clustering/liaison.html

Ces méthodes peuvent être :

**Ascendantes** Chaque observation démarre dans un groupe individuel et on fusionne étape par étape les groupes.

**Descendantes** Toutes les observations démarrent dans un groupe unique que l'on divise. (Non présentées ici)

### 3.1.2 Exemple d'algorithme : classification ascendante hiérarchique (CAH)

Le principe de la classification ascendante hiérarchique est de construire un dendrogramme des données au fur et à mesure que l'on rassemble les objets dans des groupes de plus en plus gros.

```
Initialisation : Tous les individus sont seuls dans un groupe.

On calcule la matrice des distances deux à deux entre les individus.

TANT QU'IL Y A PLUS D'UN GROUPE:
- On fusionne les deux groupes les plus proches en un seul.
- On met à jour la matrice des distance en utilisant le critère de liaison
```

### 3.1.3 Intérêt et limites

- Hiérarchiques : on a un dendogramme avec différents niveaux de partitions emboîtés en un seul passage. Si on veut $k$ groupes il suffit de couper le dendogramme au bon endroit.
- Trop lentes pour des grands jeux de données ($O(n^3)$))
- Non robustes aux *outliers*.
- Possibilité de chaînage des groupes quand on prend le Single-linkage. (Par exemple, deux groupes reliés par une chaîne de points très proches les uns des autres.)

## 3.2 Méthodes fondées sur le centroïde

### 3.2.1 Principe

Dans cette famille d'algorithmes, chaque groupe est caractérisé un vecteur appartenant au même espace que les données qu'on appelle son *centroïde*. Le centroïde peut être un élément des données (par exemple la médiane) ou non (par exemple la moyenne).

Quand on fixe le nombre de groupes à $k$ (comme dans l'algorithme des $k$-moyennes), le but est de trouver les $k$ centroïdes des groupes. Chaque point étant assigné au groupe dont le centroïde est le plus proche.

### 3.2.2 Les algorithmes des $k$-moyennes

```
Associer chaque point de l'échantillon à un groupe dans {1...k}
TANT QUE convergence non atteinte
| Déplacer le centroïde (phase d'actualisation)
| Assigner à chaque point de donnée son centroïde (phase d'assignation)
```

ou

```
Tirer k centroïdes au hasard dans l'espace des données
TANT QUE convergence non atteinte
| Assigner à chaque point de donnée son centroïde (phase d'assignation)
| Déplacer le centroïde (phase d'actualisation)
```

**Convergence** L'algorithme des $k$-moyennes peut être considéré comme un problème d'optimisation où l'on tente de minimiser la somme du carré des distances euclidiennes des points au centroïde de leur groupe. Pour un groupe $\mathcal{C}$ de centre $c$ cela donne :

$$\phi = \sum_{p \in \mathcal{C}} [d(c,p)]^2 \,.$$

Il est assez intuitif que cette quantité $\phi$ diminue à chaque phase d'assignation (parce que l'on choisit les groupes de chaque point de telle sorte que $d(c,p)$ soit minimale). Lors de l'implémentation, le critère de convergence porte généralement sur le différentiel de cette grandeur entre deux étapes (lorsqu'il est plus petit qu'une valeur limite, on considère que l'algorithme a convergé).

D'autre part, on peut prouver que $\phi$ est à un minimum local lorsque le centroïde est placé à la moyenne des coordonnées des points du cluster, ainsi, en dimension 2 :

$$\phi = \sum_{p \in \mathcal{C}} [d(c,p)]^2 = \sum_{p \in \mathcal{C}} (x_p - x_c)^2 + (y_p - y_c)^2 \,.$$

Ainsi, la dérivée de $\phi$ selon l'une des dimensions est :

$$\begin{aligned}
\frac{\partial \phi}{\partial x_c} &= \frac{d}{dx_c} \sum_{p \in \mathcal{C}} (x_p - x_c)^2 \\
&= \frac{d}{dx_c} \sum_{p \in \mathcal{C}} x_p^2 - 2 x_p x_c + x_c^2 \\
&= 2 \sum_{p \in \mathcal{C}} x_c - x_p \,.
\end{aligned}$$

Elle s'annule en :

$$\begin{aligned}
\frac{\partial \phi}{\partial x_c} = 0 &\Leftrightarrow 2 \sum_{p \in \mathcal{C}} x_c - x_p = 0 \\
&\Leftrightarrow \sum_{p \in \mathcal{C}} x_c = \sum_{p \in \mathcal{C}} x_p \\
&\Leftrightarrow x_c \sum_{p \in \mathcal{C}} 1 = \sum_{p \in \mathcal{C}} x_p \\
&\Leftrightarrow x_c = \frac{1}{|\mathcal{C}|} \sum_{p \in \mathcal{C}} x_p \,.
\end{aligned}$$

donc la dérivée partielle de $\phi$ selon n'importe quelle dimension est nulle lorsque le centroïde est la moyenne des coordonnées des points du groupe. De plus, il s'agit

bien d'un minimum local car la dérivée seconde est positive :

$$\frac{\partial \phi^2}{\partial^2 x_c} = 2 \sum_{p \in \mathcal{C}} 1 = 2|\mathcal{C}| \,.$$

Par contre, il n'y a aucune garantie que $\phi$ atteint un minimum global, et pour cause, en fonction de l'initialisation, l'algorithme des $k$-moyennes peut très bien s'arrêter sur un minimum local ne reflétant pas du tout la structure des données (voir Figure 3).

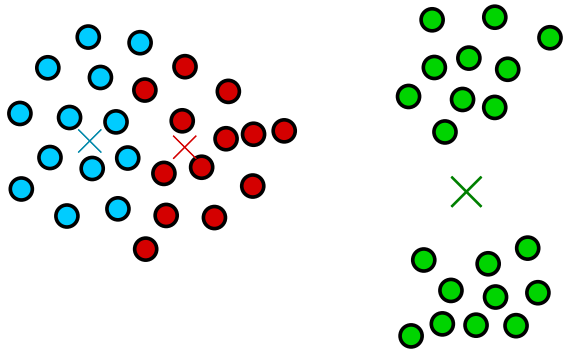


FIGURE 3 – Un exemple de configuration dans lequel un algorithme des $k$-moyennes ne donne pas de bons résultats.

Pour pallier ce problème, on peut effectuer plusieurs réalisations de l'algorithme en variant les conditions initiales.

**Variantes des $k$-moyennes** Il existe de nombreuses variantes de cet algorithme, en voici quelques unes :

- **$k$-médianes** La fonction à optimiser n'est plus la somme des carrés des distances mais la somme des distances. Cela revient à utiliser la norme 1 plutôt que la norme 2.

- **$k$-médioïdes** (Par exemple l'algorithme PAM pour *Partitioning Around Medioïds*) Dans ce cas les centre des groupes (médioïdes) sont toujours des points du jeu de données. La grandeur à optimiser n'est donc plus la somme des distances au centre mais la somme des distance des points du jeu de données à l'un d'entre eux. Cet algorithme est ainsi plus robuste aux *outliers*. En effet, les algorithmes fondés sur le centroïde ont la propriété de grouper tous les points même les plus éloignés, qui peuvent ainsi déplacer le centroïde assez loin de l'idée intuitive que l'on se fait du centre du groupe (Il est donc une pratique courante d'exclure les *outliers* avant l'analyse). Par exemple, cela pourrait éviter le cas de convergence locale illustré par la Figure 3.

• **Soft $k$-mean** Dans ce cas, on associe les points à une probabilité d'appartenir au groupe. Nous y reviendrons.

Pour résumer, ces algorithmes reviennent à résoudre un problème d'optimisation, ils ont la particularité de nécessiter que l'on prédéfinisse le nombre de groupes (par une connaissance préalable du problème ou par une première analyse des données). Ils ont le désavantage d'être sujets à la convergence vers un minimum local. Enfin ils ont tendance à être plus performants quand les groupes cibles sont de même taille et peuvent être assez sensibles aux *outliers*.

## 3.3 Méthodes fondées sur la densité

### 3.3.1 Principe

Plutôt que de chercher des points plus proches les uns des autres en les comparant paire à paire, ces méthodes sont fondées sur la détection de zones possédant une plus forte densité de mesure (une plus grande concentration de points) dans l'espace de l'échantillon.

### 3.3.2 Un premier algorithme assez simple : DBSCAN

L'algorithme DBSCAN pour *Density-Based Spatial Clustering of Applications with Noise* est le représentant le plus connu des méthodes de groupement fondées sur la densité. Au centre de cette méthode se trouve le concept d'accessibilité par densité (*density-reachability*) :

- Un point $p$ est dit **directement $\epsilon$-densité-accessible** d'un point $q$ s'il se trouve à une distance inférieure ou égale à $\epsilon$ de $q$.
- De plus un point $p$ est dit **$\epsilon$-densité-accessible** d'un point $q$ s'il existe un chemin de points directement $\epsilon$-densité-accessible les uns des autres qui permet de relier $p$ à $q$.

Les groupes sont constitués de points accessibles par densité.

Enfin, on définit l'e-voisinage d'un point $p$ comme l'ensemble des points $q$ tel que $d(p, q) \leq e$. Son cardinal est noté `|e-voisinage de P|` ou $\mathcal{N}_e(P)$.

DBSCAN prend deux paramètres :

**e** La distance maximale pour que deux points soient directement densité-accessibles.

**M** Le nombre minimal de points dans un groupe.

```
DBSCAN(e,M):

C = 0
POUR CHAQUE point P non visité:
| P est maintenant visité
| SI |e-voisinage de P| > M:
| | P fait partie du cluster C
| | L = liste des points de l'e-voisinage de P:
| | POUR CHAQUE point P' de L
| | | SI P' non visité:
| | | | P' visité
| | | | SI |e-voisinage de P'| > M:
| | | | | L = Union de L et de l'e-voisinage de P'
| | | SI P' ne fait encore partie d'aucun cluster:
| | | | P' fait partie du cluster C
| C = C + 1
```

On peut résumer cet algorithme par "Je parcours les points de mon jeu de données en les assignant à un groupe s'ils ne sont pas isolés. Quand je visite un point je rajoute tout son e-voisinage à ma liste de points à visiter. Si ma liste de point à visiter est vide, je commence un nouveau groupe et je prend un point non visité au hasard."

Les avantages principaux avantages de cet algorithme sont :

- Il n'est pas nécessaire de préciser un nombre de groupe au préalable,
- Les *outliers* (qui ont moins de $M$ points dans leur e-voisinage) ne sont assignés à aucun groupe.

Cependant, il possède une limitation inhérentes au concept d'accessibilité par densité : si les groupes sont de densités très différentes, l'algorithme ne détectera correctement que ceux dont la densité est proche du paramètre `e`.

### 3.3.3 Un algorithme plus sophistiqué : OPTICS

Il existe une extension de l'algorithme DBSCAN nommée OPTICS pour *Ordering Points To Identify the Clustering Structure* qui lève la limitation vue précédemment de devoir préciser la densité des groupes via le paramètre `e`.

Une réalisation de l'algorithme `OPTICS(e,M)` donne la même information que `DBSCAN(e',M)` pour toutes les valeurs $e' \leq e$. Cependant la sortie d'OPTICS est

plus compliquée à interpréter : il s'agit de l'ordre de visite des points et leur distance d'accessibilité de l'un au suivant.

On définit :
La **distance de coeur** (*core-distance*) du point $p$ :

$$CD(p) = \begin{cases} \text{distance au M-ième point le plus proche de p,} \\ \text{non définie si } \mathcal{N}_e(p) < M. \end{cases} ;$$

La **distance d'accessibilité** (*reachability distance*) entre les points $p$ et $q$ :

$$RD(p,q) = \begin{cases} \max(CD(p), d(p,q)), \\ \text{non définie si } \mathcal{N}_e(p) < M. \end{cases}$$

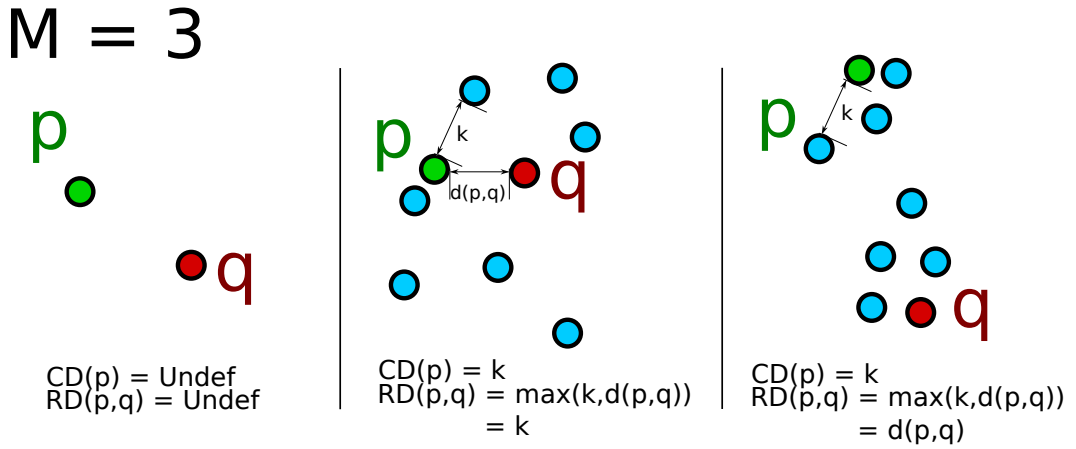



FIGURE 4 – Illustration du fonctionnement d'OPTICS. CD, distance de coeur ; RD, distance d'accessibilité.

On note que la distance de coeur est majorée par $e$ alors que la distance d'accessibilité est minorée par $e$.

On peut résumer cet algorithme par : "Je parcours les points de mon jeu de données en notant l'ordre dans lequel je le fait. De plus, à chaque point visité, j'ajoute les points de son e-voisinage à ma liste de points à visiter mais (contrairement à DBSCAN) cette liste est ordonnée par distance d'accessibilité croissante."

La différence principale avec DBSCAN est donc que l'on utilise une file de priorité et non pas une simple liste pour stocker les points à visiter. Il s'agit d'un type de donnée abstrait permettant de stocker un élément associé à une priorité et

de récupérer efficacement l'élément avec la plus faible priorité. L'implémentation d'une file de priorité n'est pas le sujet de cet atelier mais celle ci se fait usuellement à l'aide d'un arbre binaire.

```
OPTICS(M,e):

C = 0
POUR CHAQUE point P non visité:
| P est maintenant visité (l'ordre dans lequel les points sont visité est stocké)
| SI CD(p) est définie:
| | L = MAJ(P,L)
| POUR TOUT point Q de L dans l'ordre de priorité croissant:
| | Q est maintenant visité.
| | SI CD(Q) est définie:
| | | L = MAJ(Q,L)
| RETOURNER l'ordre de visite et la priorité de chaque point visité

MAJ(P,L):
POUR TOUT point V de l'e-voisinage de P:
| SI V n'est pas déjà dans la file de priorité L:
| | Ajouter V avec la priorité RD(V,P)
| SI NON:
| | SI RD(V,P) est plus petit que la priorité de V
| | | Assigner la priorité de V à RD(V,P)
```

Ainsi la sortie d'OPTICS est un graphe de la distance d'accessibilité en fonction de l'ordre de visite. Les vallées de ce graphe correspondent à des groupes de points plus denses.

On peut retrouver tous les clusters qui auraient été prédits par `DBSCAN(e')` en parcourant la sortie d'OPTICS de la façon suivante :

```
C = 0
POUR CHAQUE point P dans l'ordre de sortie d'OPTICS
| SI RD(P) < e':
| | P est associé au groupe C
| SI NON:
| | SI il y a déjà des points associés au groupe C
| | | C = C+1
```

ce qui revient graphiquement à fixer un seuil et à définir les groupes comme une portion du parcours où $RD < e'$ . En effet, cela signifie que tous ces points sont *e'-accessibles* donc ils forment un groupe au sens de DBSCAN.

On peut aller plus loin et transformer la sortie d'OPTICS en un dendogramme équivalent, si cela vous intéresse je vous redirige vers l'article de Sander et al.

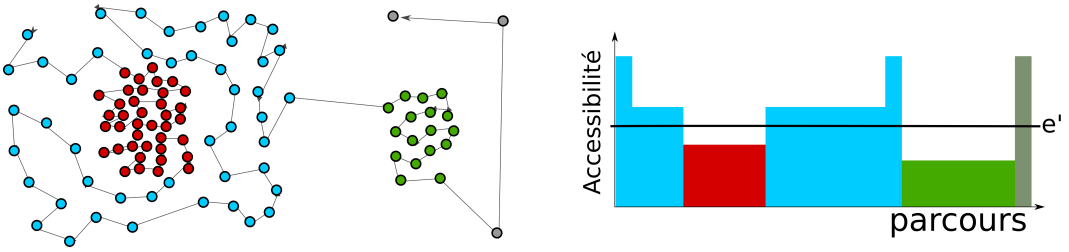



FIGURE 5 – Sortie d'OPTICS : droite, graphe de la distance d'accessibilité en fonction de l'ordre de visite ; gauche, groupes de points correspondants.

[2003]. En ce sens les algorithmes fondés sur la densité peuvent être considérés comme des extensions des algorithmes fondés sur la connectivité.

## 3.4 Méthodes fondées sur des distributions

### 3.4.1 Principe des modèles de mélange

Si les algorithmes de groupement par densité peuvent être considérés comme un raffinement des algorithmes fondés sur la connectivité, les algoritmes fondés sur des distributions peuvent être vus comme une extension des algorithmes de la famille des $k$-moyennes.

En effet, au lieu de représenter chaque groupe par un point représentatif (centroïde) dont les membres sont plus proche que les points représentatifs des autres groupes, on représente ici chaque groupe par une distribution en considérant que les membres du groupes sont les points issus de la distribution caractéristique du groupe.

On parle ainsi d'un modèle de mélange : on considère que la distribution des données est le résultat du mélange de tirages de différentes distributions et notre but est d'associer à chaque point une de ces distributions.

Une fois encore on utilise un algorithme d'optimisation qui alterne deux étapes et qui est dit d'espérance-maximisation (*Expectation-Maximisation* ou EM). Ces deux phases sont analogues aux phases d'Assignation-Mise à jour des algorithmes fondés sur les $k$-moyennes.

| **Centroïdes** | **Assignation**<br>Associer chaque point à un centroïde | **Mise à jour**<br>Déplacer les centroïdes de manière à réduire la distance des points du groupe à leur centroïde |
|---|---|---|
| **Distributions** | **Espérance**<br>Calculer la probabilité conditionnelle qu'un point soit issu d'une distribution donnée | **Maximisation**<br>Modifier les distribution de manière à maximiser la vraisemblance |

### 3.4.2 Un intermédiaire : l'algorithme doux des $k$-moyennes

L'algorithme doux des $k$-moyennes (*Soft k-means*) que nous avons rapidement évoqué plus tôt, peut être considéré comme un intermédiaire entre les méthodes fondées sur centroïdes telles que nous les avons présentés jusqu'ici et les méthodes de distribution telles que nous allons les voir.

La principale différence avec l'algorithme des $k$-moyennes classique provient de la phase d'assignation. On définit l'appartenance du point $x_i$ au groupe $k$ (de centre $c_k$) par $r_{k,i}$. Dans le cas de l'algorithme classique, $r_{k,i}$ prend ses valeurs dans $\{0, 1\}$ :

$$r_{k,i} = \begin{cases} 1 & \text{si } k = \arg\max_j(d(c_j, x_i)), \\ 0 & \text{sinon.} \end{cases}$$

Dans le cas de l'algorithme doux, $r_{k,i}$ prend ses valeurs dans $[0, 1]$ :

$$r_{k,i} = \frac{\exp(-\beta d(c_k, x_i))}{\sum_j \exp(-\beta d(c_j, x_i))}\,.$$

La phase de mise à jour se fait de la même façon pour les deux algorithmes c'est-à-dire que les centroïdes sont déplacés pour minimiser la distance des points d'un groupe à son centroïde en prenant la coordonnée moyenne des points pondérée par leur appartenance :

$$c_k = \frac{\sum_i x_i r_{k,i}}{\sum_i r_{k,i}}\,,$$

la seule différence étant que dans l'algorithme doux l'appartenance à un groupe n'est pas booléenne. La sortie de cet algorithme est donc pour chaque point une proportion d'appartenance à un cluster. Si l'on ne souhaite avoir qu'un groupe par point, il suffit de choisir critère de décision, le plus simple étant de prendre le groupe d'appartenance maximale :

$$k_i = \arg\max_k(r_{k,i})\,.$$

Il est cependant assez courant de prendre un critère de décision seuillé, ce qui permet de ne pas trancher pour les points dont l'appartenance à un groupe ou

l'autre est discutable :

$$k_i = \begin{cases} \arg\max_k(r_{k,i}) & \text{si } \max_k(r_{k,i}) > \alpha, \\ \text{non défini} & \text{sinon.} \end{cases}$$

### 3.4.3 L'algorithme de groupement en mélange gaussien par espérance-maximisation

**Un petit peu de statistiques bayesiennes** Dans un modèle de mélange gaussien, chaque cluster est caractérisé non pas par un centroïde mais par une distribution gaussienne de même dimension $d$ que les données.

On considère que le jeu de données $(x_1, x_2, \ldots, x_n)$ est une réalisation des variables aléatoires $(X_1, X_2, \ldots, X_n)$. L'appartenance d'un point $X_i$ à un groupe est donné par la variable aléatoire $Z_i$. On parle de variable cachée parce qu'elle sert d'intermédiaire de calcul mais on ne la mesure pas vraiment sur le jeu de données.

Ainsi, l'évènement "le point $X_i$ appartient au groupe $k$" s'écrit $Z_i = k$. Avant de regarder les données, on considère que les points ont tous la même probabilité d'appartenir à tous les groupes. Ainsi les $Z_i$ sont i.i.d. et suivent la distribution *a priori* suivante :

$$\mathbb{P}(Z_i = k) = \pi_k = \frac{1}{M}\,.$$

Ces valeurs peuvent être prises différentes si on a des raisons de penser qu'un groupe est plus probable que l'autre, mais en général on se contente d'une distribution *a priori* dite "non informative" comme celle ci.

De même, on suppose que la densité de probabilité des coordonnées de la variable aléatoire $X_i$ sachant qu'elle fait partie du groupe $k$ (i.e. la densitée conditionnée à l'évènement $Z_i = k$) est une gaussienne dont les paramètres ne dépendent que de $k$ :

$$f_{X_i}(x \mid Z_i = k) = \mathcal{G}(x, \mu_k, \Sigma_k)$$

On note $\theta_k = (\pi_k, \mu_k, \Sigma_k)$ le vecteur des paramètres caractérisant le groupe $k$ avec : $\pi_k$ la probabilité a priori du groupe ; $\mu_k \in \mathbb{R}^d$ le vecteur de moyennes et $\Sigma_k \in M_{d,d}(\mathbb{R})$ la matrice de variance-covariance.

De manière analogue à l'algorithme doux des $k$-moyennes, on va construire une valeur d'appartenance d'un point à ce groupe en prenant la probabilité de l'évènement $Z_i = k$ **sachant** la valeur du point $(X_i = x_i)$ et les paramètres

caractérisant la distribution du groupe $k$. C'est l'étape Expectation [1].

$$\begin{aligned} r_{k,i} &= \mathbb{P}(Z_i = k \mid X_i = x_i, \theta_k) \\ &= \frac{\mathbb{P}(Z_i = k \mid \theta_k) f_{X_i}(x_i \mid Z_i = k, \theta_k)}{\sum_j^M \mathbb{P}(Z_i = j \mid \theta_k) f_{X_i}(x_i \mid Z_i = j, \theta_k)}, \text{ (“formule de Bayes”)} \\ &= \frac{\pi_k \mathcal{G}(x_i, \mu_k, \Sigma_k)}{\sum_j \pi_j \mathcal{G}(x_i, \mu_j, \Sigma_j)} \end{aligned}$$

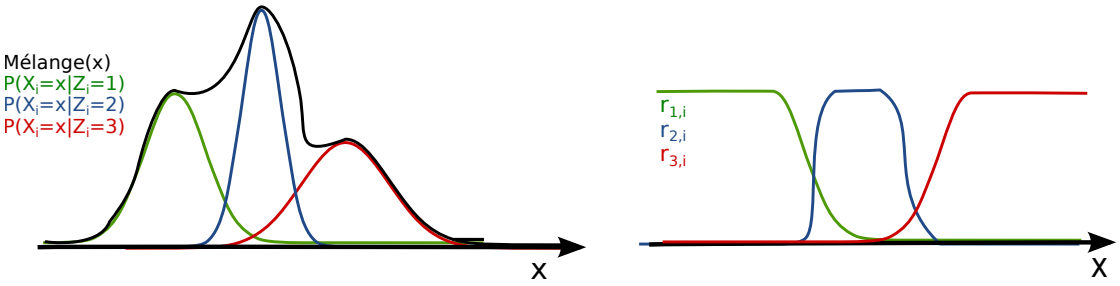



FIGURE 6 – Modèle de mélange Gaussien unidimensionnel. Gauche, densité de probabilité du mélange gaussien (noir). Chaque groupe est caractérisé par une densité de probabilité gaussienne (couleur), qui correspond à la densité conditionnée à l'appartenance au groupe. Pour des soucis de lisibilité, l'échelle verticale n'est pas la même pour les différentes densités (l'aire sous la courbe de chaque densité doit être égale à 1) ; droite, appartenance d'un point aux groupes 1 à 3 en fonction de sa valeur x.

On met ensuite à jour les paramètres des distribution caractéristiques des groupes par l'estimation du maximum de vraisemblance : c'est l'étape Maximisation.

La vraisemblance de la valeur d'un paramètre $\theta$ sachant les données $(X = x)$ est définie comme “la probabilité de $(X = x)$ sachant $\theta$”. Dans le cas d'une variable $X$ à densité, on la définit donc par :

$$L(\theta \mid X = x) = f_X(x \mid \theta)\,.$$

Le but d'une estimation du maximum de vraisemblance est de trouver la valeur de paramètre qui fait que les données observées sont les plus “vraisemblables”, c'est à dire qui maximisent $L$.

1. NdE : pour un traitement plus formel de ce modèle, voir les notes d'Alexis Jacq.

Pour un cas très simple il est possible d'avoir une formule analytique de la vraisemblance, mais dans la majorité des cas pratiques (comme ici), cela n'est pas possible – c'est la raison pour laquelle on utilise des algorithmes d'optimisation comme l'algorithme EM. Pour rendre les choses plus faciles on procède à certaines transformations :

- On enlève tout coefficient multiplicateur qui ne dépend pas de $\theta$
- On travaille avec le logarithme de la vraisemblance (log-vraisemblance) noté $\mathcal{L}$

Dans le cas du problème du mélange gaussien, on retrouve une version pondérée par les appartenances du classique estimateur du maximum de vraisemblance :

$$\begin{aligned}
\hat{\theta}_k &= \arg\max_{\theta_k} \; L(\theta_k \mid x_1, \ldots, x_n) \\
&= \arg\max_{\theta_k} \; \mathcal{L}(\theta_k \mid x_1, \ldots, x_n) \\
&= \arg\max_{\theta_k} \; \log\left(\prod_i f_{X_i}(x_i \mid \theta_k)\right) \quad (X_i \text{ iid}) \\
&= \arg\max_{\theta_k} \sum_i \log\left(f_{X_i}(x_i \mid \theta_k)\right) \\
&= \arg\max_{\theta_k} \sum_i \log\left(\sum_k f_{X_i}(x_i | Z_i = k, \theta_k)\, \mathbb{P}(Z_i = k \mid \theta_k)\right) \\
&= \arg\max_{\theta_k} \sum_i \log\left(\sum_k \mathcal{G}(x_i, \mu_k, \Sigma_K)\pi_k\right)
\end{aligned}$$

On alterne ensuite ces deux phases : calcul des critères d'appartenances à partir des paramètres ; mise à jour des paramètres, etc.

### 3.4.4 Résumé

| | $k$-moyennes | Soft K-means | Modèle de Mélange et EM |
|---|---|---|---|
| Le point $X_i$ appartient au groupe $k$ | $k_i = \arg\max_k(d(c_k, x_i))$ | | $Z_i = k$ ; $\mathbb{P}(Z_i = k) = \pi_k$ |
| Appartenance de i à k | $r_{k,i} \in \{0, 1\}$ | $r_{k,i} = \frac{\exp(-\beta d(c_k, x_i))}{\sum_j \exp(-\beta d(c_j, x_i))}$ | $r_{k,i} = \frac{\pi_k \mathcal{G}(x_i, \mu_k, \Sigma_k)}{\text{mélange}(x)}$ |
| Phase de Mise à jour | $c_k = \frac{\sum_i x_i r_{k,i}}{\sum_i r_{k,i}}$ | (idem) | $\hat{\theta} = \arg\max_\theta(L(\theta))$ |

# 4 Évaluation des partitionnements de données

Nous l'avons vu, il existe un grand nombre de méthodes d'analyse en groupement, utilisant différentes méthodes de mesures de similarité, et requérant des paramètres (nombre ou taille minimale d'un groupe, densité minimale d'un groupe...). L'enjeu est donc de savoir comment évaluer la qualité d'un partitionnement pour pouvoir choisir une méthode et des valeurs de paramètres.

Pour cela il existe deux grandes approches : la famille des évaluations internes se propose d'évaluer le partitionnement de données vis à vis des propriétés que l'on souhaite avoir pour les groupes (séparation, cohérence) alors que la famille des évaluation externes passe par la comparaison d'un partitionnement avec une référence – par exemple un partitionnement manuel.

## 4.1 Méthodes d'évaluation internes

Généralement lorsque l'on regroupe des données, on cherche à avoir :

- Une **grande similarité intra-groupe** : les éléments d'un même groupe se ressemblent fortement
- Une **faible similarité inter-groupe** : les éléments de deux groupes différents sont différents.

Ainsi, on peut imaginer un indice qui mesure l'adéquation du partitionnement avec ces propriétés désirées, et que l'on peut utiliser pour comparer deux partitionnements.

Soit un partitionnement des points $(X_i)_{i=1,\dots,N}$, en groupes de centroïdes $(c_k)_{k=1,\dots,M}$. On définit $\sigma_k$, la dispersion (selon la norme $q$) du groupe $k$ par :

$$\sigma_k = \sqrt[q]{\frac{1}{T}\sum_{i \in k}(x_i - c_k)^q}\,.$$

Plus cette dispersion est grande, plus les points sont éloignés de leur centroïde en moyenne.

Pour deux groupes $i$ et $j$ le rapport des similarités intra-groupe sur inter-groupes peut alors être défini ainsi :

$$R_{i,j} = \frac{\sigma_i + \sigma_j}{d(c_i, c_j)}\,.$$

Plus cette mesure est grande plus la dispersion intra-groupe est importante par rapport à la distance inter-groupes, et donc moins les groupes sont bien définis. Elle a aussi l'avantage d'être :

- Positive,
- Symétrique,
- Si $\sigma_j \geq \sigma_k$ et $d(c_i, c_j) = d(c_i, c_k)$, alors $R_{i,j} > R_{i,k}$,
- Si $\sigma_j = \sigma_k$ et $d(c_i, c_j) \leq d(c_i, c_k)$, alors $R_{ij} > R_{ik}$.

La valeur moyenne du rapport de similarité maximale pour chaque groupe est appelée indice de Davies-Bouldin du partitionnement :

$$DB = \frac{1}{M} \sum_{i=1}^{M} \max_{i \neq j} R_{i,j} \, .$$

Un partitionnement est de qualité d'autant meilleure – au sens de Davies-Bouldin – que cette quantité est petite.

## 4.2 Méthodes d'évaluation externes

Quand on possède un partitionnement de référence d'un jeu de données (qui peut être par exemple une classification manuelle d'un sous-ensemble du jeu de données que l'on veut analyser) on peut vouloir mesurer la concordance entre cette référence et le partitionnement obtenu via l'analyse en groupement.

Une possibilité est d'utiliser une matrice de confusion (la terminologie nous vient de l'apprentissage supervisé, on s'en sert pour caractériser les erreurs d'un algorithme entrainé). Pour la construire, on mesure le nombre de paires de points qui sont :

- Dans le même groupe dans le partitionnement automatique ET le partitionnement de référence (Vrai positif, TP)
- Dans des groupes différents dans le partitionnement automatique ET le partitionnement de référence (Vrai négatifs, TN)
- Dans le même groupe dans le partitionnement automatique MAIS pas dans le partitionnement de référence (Faux positifs, FP).
- Dans des groupes différents dans le partitionnement automatique MAIS pas dans le partitionnement de référence (Faux négatifs, FN).

Cela permet de calculer la mesure de Rand du partitionnement, qui est la proportion de paires correctement classées :

$$R = \frac{TP + TN}{TP + TN + FN + TN} \in [0, 1] \,.$$

La limite de cette mesure est qu'elle donne le même poids aux faux positifs et faux négatifs (ce qui n'est pas toujours souhaitable en fonction de l'interprétation que l'on fait du partitionnement).

# 5 Conclusion

S'il existe de nombreuses méthodes d'analyse en groupement, il n'y a pas de solution miracle. Chaque famille de méthode a sa propre façon de définir ce qu'est un groupe, et il en découle des propriétés particulières mais aussi des limitations intrinsèques qu'il convient d'avoir en tête et de mettre en relation avec nos connaissance biologiques du système afin de choisir la plus pertinente. D'une manière générale il est assez sain de considérer l'analyse de données comme une science expérimentale : il n'y a pas de protocole qui permette de répondre à toutes les questions, mais plutôt différentes façons d'observer un jeu de données.

Enfin, le domaine de l'apprentissage automatique (*Machine Learning*) est bien plus vaste que les quelques problèmes d'apprentissage non supervisés que nous venons de survoler. Un autre domaine qui pourrait intéresser le biologiste est la classification (ou apprentissage supervisé) où l'on souhaite assigner de nouveaux points de données à une catégorie après observation d'un jeu d'apprentissage. Ce qui peut servir par exemple en oncologie (pour classer les profils d'expression susceptibles d'être cancéreux après observation de nombreux patients) ou en écologie (pour réaliser une taxonomie automatique d'individus dans un échantillon après avoir réalisé un entraînement de l'algorithme par un naturaliste). D'autre part, la sélection de variable (*feature selection*) est une branche de l'apprentissage automatique qui s'intéresse à trouver les sous-ensembles de variables qui sont les plus informatifs pour les opérations de partitionnement ou de classification.

# Références

E. Alpaydin. *Introduction to Machine Learning.* MIT press, 2nd edition, 2010.

M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander. Optics : Ordering points to identify the clustering structure. *ACM SIGMOD international conference on Management of data*, pages 49–60, 1999.

M. Aubry. De l'analyse de groupement [clustering] tentative d'en dégager les principaux concepts et de les appliquer à l'analyse des puces à ADN. URL ouestgenopuces.univ-rennes1.fr/formations/analyse_partie2c.pdf.

K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft. When is "nearest neighbor" meaningful ? In *Database Theory — ICDT'99*, volume 1540 of *Lecture Notes in Computer Science*, pages 217–235. Springer Berlin Heidelberg, 1999.

V. Estivill-Castro. Why so many clustering algorithms : a position paper. *ACM SIGKDD Explorations Newsletter*, 4(1) :65–75, 2002.

I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3 :1157–1182, 2003.

M. Mohri, Rostamizadeh A., and Talwalkar A. *Foundations of Machine Learning*. MIT press, 2012.

J. Sander, X. Qin, Z. Lu, N. Niu, and A. Kovarsky. Automatic extraction of clusters from hierarchical clustering representations. In *Advances in knowledge discovery and data mining*, pages 75–87. Springer, 2003.

M. Verleysen and D. François. The curse of dimensionality in data mining and time series prediction. In *Computational Intelligence and Bioinspired Systems*, pages 758–770. Springer, 2005.

R. B. Zadeh and S. Ben-David. A uniqueness theorem for clustering. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*, pages 639–646. AUAI Press, 2009.