

# STT3030 - Cours #9

Arthur Charpentier

Automne 2024

## Apprentissage supervisé

- ▶  $\mathbf{X} = \{X_1, \dots, X_p\}$  forment une collection de prédicteurs.
- ▶ On dit que  $\mathbf{X} \in \mathcal{X}_1 \times \dots \times \mathcal{X}_p = \mathcal{X}$ , où  $\mathcal{X}_j$  est le domaine de  $X_j$ .
- ▶ Ce sont les variables que nous utilisons pour prédire.
- ▶  $Y \in \mathcal{Y}$  est une réponse.
- ▶ On veut prédire la réponse  $Y$  avec les prédicteurs  $X$ .
- ▶ On suppose l'existence d'une fonction  $f: \mathcal{X} \rightarrow \mathcal{Y}$
- ▶ Souvent de la forme la vraie fonction  $Y = f(X) + \varepsilon$

Où  $\varepsilon$  est une supposée variabilité inexpliquée.

En apprentissage supervisé, on veut estimer  $f$ .

# Introduction

En apprentissage non-supervisé, il n'y pas de concept de réponse ou de prédicteurs.

Il y a un ensemble de variables  $\mathbf{X} \in \mathcal{X}$ , c'est tout.

On n'impose pas de structure et on ne suppose pas l'existence d'une *vraie fonction f*.

L'objectif est d'apprendre *quelque chose* par rapport à la distribution de ces données.

On parle souvent de capturer des caractéristiques de la distribution des observations.

Peut-on apprendre cette distribution ?

Peut-on apprendre des caractéristiques importantes de celle-ci ?

Peut-on extraire l'information importante de ces données ?

Peut-on déterminer des observations inattendues ?

# Introduction

Deux problèmes standards en apprentissage non-supervisé:

- ▶ La mise-en-grappe/le regroupement : Méthodes et techniques afin de créer des groupes de variables **similaires**
  - ▶ La réduction de dimension: Méthodes et techniques pour **projeter** les données  $\mathbf{X} \in \mathcal{X}$  sur  $\mathbf{Z} \in \mathcal{Z}$  où la dimension de  $\mathcal{Z}$  est **beaucoup** plus petite que celle de  $\mathcal{X}$ .
- 
- ▶ Dans la mise-en-grappe, on veut regrouper des observations similaires.
  - ▶ On forme des groupes à l'aide des variables **X**.
  - ▶ On forme des groupes d'observations qui se ressemblent (groupes pour lesquelles les variables sont similaires)
  - ▶ Beaucoup de modèles supervisés sont moins performants et lents à optimiser lorsque  $p$  est trop grand. (exemple: modèles linéaires si  $p > n$ )
  - ▶ Il peut parfois être difficile de visualiser des données  $\mathbf{X} \in \mathcal{X}$  lorsqu'elles sont de très grande dimension.

## Introduction : réduction de dimension

- ▶ Dans la réduction de dimension on vise à apprendre une fonction  $f: \mathcal{X} \subset \mathbb{R}^p \rightarrow \mathcal{Z} \subset \mathbb{R}^d$  où  $d \ll p$ .
- ▶ On veut souvent conserver le maximum d'information pour distinguer les observations malgré la réduction de dimension.
- ▶ On doit aussi penser à la forme de  $f$ , il de pouvoir calculer  $f^{-1}$  aussi pour récupérer des observations dans le domaine d'origine  $\mathcal{X}$ .

Pour ces deux problèmes on n'apprend pas l'équivalent de  $\hat{f}: \mathcal{X} \rightarrow \mathcal{Y}$  qui *imité* une vraie fonction  $f$ .

Bien que l'on apprend un modèle ou une fonction pour former des groupes ou réduire la dimension des données, il n'y existe pas de vérité que l'on approxime.

C'est ce qui en fait de l'apprentissage non-supervisé.

## Pas de mesure de succès

Parce qu'on n'imité pas une vérité ou un *vrai modèle* il est plus difficile d'évaluer le succès du modèle appris (des groupes formés par exemple).

- ▶ Un défi de l'apprentissage non-supervisé est le manque d'une bonne mesure de succès.
- ▶ Comme il n'y a pas de réponse, nous ne pouvons pas utiliser l'erreur de prédiction/classification.
- ▶ Il n'y a pas de vrais groupes ni de vraie représentation de petite dimension non plus.
- ▶ Un problème est donc qu'il est difficile d'établir si nous avons raisonnablement accompli notre tâche d'apprentissage non-supervisé.

## Pas de mesure de succès

- ▶ Le succès du modèle est plus subjectif.
- ▶ L'évaluation du succès d'une technique d'apprentissage non-supervisé est un problème encore d'actualité.
- ▶ Ce sera un problème important pour nous; on va devoir se satisfaire d'apprendre sans savoir si on a réellement bien appris.

# Norm

A **norm**  $\|\cdot\|$ , in  $\mathbb{R}^n$ , satisfies

- ▶ homogeneity,  $\|a\vec{u}\| = |a| \cdot \|\vec{u}\|, \forall a$
- ▶ triangle inequality,  $\|\vec{u} + \vec{v}\| \leq \|\vec{u}\| + \|\vec{v}\|$
- ▶ positivity,  $\|\vec{u}\| \geq 0$
- ▶ definiteness,  $\|\vec{u}\| = 0 \iff \vec{u} = \vec{0}$

$\ell_1$  norm:  $\|\mathbf{x}\|_{\ell_1} = |x_1| + \cdots + |x_n|$ ,

see **taxicab geometry**

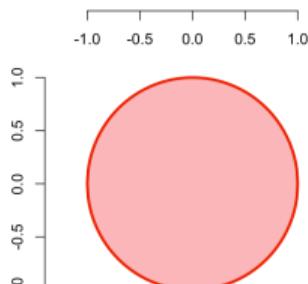
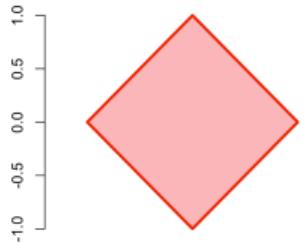
$\ell_2$  norm:  $\|\mathbf{x}\|_{\ell_2} = \sqrt{x_1^2 + \cdots + x_n^2}$ ,

$\ell_p$  norm: with  $p \geq 1$ ,

$$\|\mathbf{x}\|_{\ell_p} = (|x_1|^p + \cdots + |x_n|^p)^{1/p}$$

e.g.  $\|\mathbf{x}\|_{\ell_\infty} = \max\{|x_i|\}$

Unit balls ( $\|\mathbf{x}\| \leq 1$ ) are convex sets



# Hilbert Space and Inner Products

An **inner product**  $\langle \cdot, \cdot \rangle$ , in  $\mathbb{R}^n$ , satisfies

- ▶ symmetry,  $\langle \vec{u}, \vec{v} \rangle = \langle \vec{v}, \vec{u} \rangle$
- ▶ linearity,  $\langle a\vec{u} + b\vec{v}, \vec{w} \rangle = a\langle \vec{u}, \vec{w} \rangle + b\langle \vec{v}, \vec{w} \rangle$
- ▶ positivity,  $\langle \vec{u}, \vec{u} \rangle \geq 0$
- ▶ definiteness,  $\langle \vec{u}, \vec{u} \rangle = 0 \iff \vec{u} = \vec{0}$

**Example:** On the set of  $\mathbb{R}^n$  vectors,  $\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x}^\top \mathbf{y}$

Furthermore

- ▶  $\|\mathbf{x}\| = \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle}$  defines a norm
- ▶  $d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|$  defines a distance

**Example:** On the set of  $m \times n$  matrices,  $\langle \mathbf{A}, \mathbf{B} \rangle = \text{trace}(\mathbf{A}\mathbf{B}^\top)$

**Example:** On the set of random variables,  $\langle X, Y \rangle = \mathbb{E}(XY)$

# Distance

$d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|$  defines a distance.

$$d(\mathbf{a}, \mathbf{b}) + d(\mathbf{b}, \mathbf{c}) \geq d(\mathbf{a}, \mathbf{c})$$

**Example** Let  $\mathbf{y}, \mathbf{x}_1, \dots, \mathbf{x}_n$ . The nearest neighbor of  $\mathbf{y}$ , among the  $\mathbf{x}_i$ 's is

$$\mathbf{x}_{i^*} = \underset{\mathbf{x} \in \{\mathbf{x}_1, \dots, \mathbf{x}_n\}}{\operatorname{argmin}} \{d(\mathbf{x}, \mathbf{y})\}$$

```
1 > X = matrix(runif(8*3), 8, 3)
2 > dist(X, method = "euclidean")
3      1       2       3       4       5       6       7
4 2 0.832
5 3 0.698 0.455
6 4 0.539 1.054 0.716
7 5 0.834 1.074 0.785 0.921
8 6 0.755 0.977 0.554 0.354 0.871
9 7 0.153 0.739 0.691 0.666 0.885 0.849
10 8 0.734 0.990 0.804 0.977 0.263 0.989 0.752
```

## Cauchy-Schwarz Inequality

$$|\langle \mathbf{x}, \mathbf{y} \rangle| \leq \|\mathbf{x}\| \cdot \|\mathbf{y}\| \text{ with equality only when } \mathbf{x} = \lambda \mathbf{y}$$

**Application:**  $x_i \leftarrow x_i - \bar{x}$  and  $y_i \leftarrow y_i - \bar{y}$ ,

$$\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x}^\top \mathbf{y} = \sum_{i=1}^n x_i y_i$$

$$\|\mathbf{x}\| \cdot \|\mathbf{y}\| = \sqrt{\mathbf{x}^\top \mathbf{x}} \cdot \sqrt{\mathbf{y}^\top \mathbf{y}} = \sqrt{\sum_{i=1}^n x_i^2} \cdot \sqrt{\sum_{i=1}^n y_i^2}$$

$$\text{corr}(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^n x_i y_i}{\sqrt{\sum_{i=1}^n x_i^2} \cdot \sqrt{\sum_{i=1}^n y_i^2}} = \frac{\langle \mathbf{x}, \mathbf{y} \rangle}{\|\mathbf{x}\| \cdot \|\mathbf{y}\|} \in [-1, +1]$$

and  $\text{corr}(\mathbf{x}, \mathbf{y}) = \pm 1$  only when  $\mathbf{x} = \lambda \mathbf{y}$ .

## Angles

Since  $|\langle \mathbf{x}, \mathbf{y} \rangle| \leq \|\mathbf{x}\| \cdot \|\mathbf{y}\|$ , define  $\theta$  as the unique number in  $[0, \pi]$  that satisfies

$$\langle \mathbf{x}, \mathbf{y} \rangle = \|\mathbf{x}\| \cdot \|\mathbf{y}\| \cos(\theta), \text{ or } \theta = \arccos \frac{\langle \mathbf{x}, \mathbf{y} \rangle}{\|\mathbf{x}\| \cdot \|\mathbf{y}\|}$$

## Mahalanobis distance

A  $n \times n$  symmetric matrix  $\mathbf{M}$  is positive definite if  $\mathbf{x}^\top \mathbf{M} \mathbf{x} > 0$  for any  $\mathbf{x} \in \mathbb{R}^n \setminus \{\mathbf{0}\}$ .

**Proposition:** If  $\mathbf{M}$  is a positive definite (symmetric) matrix, then  $\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x}^\top \mathbf{M} \mathbf{y}$  defines an inner product.

(furthermore, conversely, if  $\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x}^\top \mathbf{M} \mathbf{y}$  defines an inner product, then  $\mathbf{M}$  is definite positive)

- ▶  $\langle \mathbf{x}, \mathbf{y} \rangle_M = \mathbf{x}^\top \mathbf{M} \mathbf{y}$  defines an inner product,
- ▶  $\|\mathbf{x}\|_M = \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle_M}$  defines a norm
- ▶  $d_M(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_M$  defines a distance

Given  $\Sigma$  some  $n \times n$  definite positive matrix, define

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{(\mathbf{x} - \mathbf{y})^\top \Sigma^{-1} (\mathbf{x} - \mathbf{y})}$$

Given  $\mu \in \mathbb{R}^n$ , define the Mahalanobis “norm” as

$$\|\mathbf{x}\| = d(\mathbf{x}, \mu) = \sqrt{(\mathbf{x} - \mu)^\top \Sigma^{-1} (\mathbf{x} - \mu)}$$

If  $\Sigma$  is diagonal, it is also called standardized Euclidean distance.

See [on the generalised distance in statistics](#), 1936.

# Introduction

Dans la mise-en-grappe, il faut déterminer des groupes différents d'observations similaires.

La première approche de mise-en-grappe que nous voyons est le *k-mean clustering*, appelé ici partitionnement en *k*-moyennes.

Il s'agit d'une approche relativement simple conceptuellement de mise-en-grappe, qui vise à

- ▶ (1) Déterminer des centres de masse (barycentres, ou moyennes) pour des groupes et
- ▶ (2) Déterminer ensuite le groupe auquel appartient une observation en fonction de la distance entre l'observation et les centres de masse.

## Fonction objective

On définit les groupes par leurs centre de masse.

Intuitivement on associe une observation au groupe pour lequel le centre de masse est le plus proche.

Pour faire un partitionnement en  $k$ -moyennes, on veut donc identifier  $k$  centres de masse ( $k$  moyennes) et assigner les points au centre de masse le plus proche.

L'objectif est donc d'établir les groupes (les centres) ainsi que les assignments (à quel groupe appartiennent les observations) afin de minimiser la distance entre les points et le centre de leur groupe.

## Fonction objective

L'objectif est de réduire la somme des distances entre les points et la moyenne du groupe auquel ils appartiennent:

$$Obj = \sum_{i=1}^n \sum_{j=1}^k r_{i,j} d(\mathbf{x}_i, \mu_j) \text{ ou } \sum_{i=1}^n \sum_{j=1}^k r_{i,j} \|\mathbf{x}_i - \mu_j\|, \quad (1)$$

ou des fonctions de ces distances, e.g.,

$$Obj = \sum_{i=1}^n \sum_{j=1}^k r_{i,j} d(\mathbf{x}_i, \mu_j)^2 \text{ ou } \sum_{i=1}^n \sum_{j=1}^k r_{i,j} \|\mathbf{x}_i - \mu_j\|^2, \quad (2)$$

pour une distance  $d$  (ou une norme  $\|\cdot\|$ ) sur  $\mathcal{X}$ , et où  $r_{i,j} = 1$  si l'observation  $i$  est assignée au groupe  $j$  et 0 sinon et  $\mu_j$  est la moyenne du groupe  $j$ .

Cette fonction objective représente donc, comme voulu, la somme des distances euclidiennes entre les observations et la moyenne de leur groupe.

Dessin au tableau!

## Fonction objective

En quelque sorte on veut trouver des groupes *qui font du sens*. Mais on ne connaît ni les centres de groupes, ni les groupes en tant que tels (les appartenances).

En plus, il n'y existe pas de groupes! On se souvient, il n'y a pas de réponse catégorielle groupe dans ce problème.

S'il y avait des groupes et que ceux-ci étaient connus, il serait facile de calculer leur moyenne.

Si on connaissait le centre des groupes, il serait facile de déterminer de quels groupes les observations sont les plus proches.

## Fonction objective

Il est impossible de minimiser la fonction objective en fonction des assignements  $r_{i,j}$  et des moyennes  $\mu_j$  conjointement.

On ne peut pas calculer la dérivée de la fonction objective simplement et il y a trop de combinaison d'assignements possibles ( $k^n$ ) pour tout tester.

Par contre, si les moyennes sont connues, nous pouvons facilement minimiser cette fonction pour les assignements.

Pareillement, si les assignements sont connus, nous pouvons facilement déterminer les valeurs des moyennes qui minimise la fonction objective.

Supposons les moyennes connues, pour minimiser la somme des distances il suffit d'assigner les observations à la moyenne la plus proche:

$$r_{i,j} = \begin{cases} 1 & \text{si } j = \operatorname{argmin}_l \|\mathbf{x}_i - \mu_l\|^2 \\ 0 & \text{sinon} \end{cases}$$

## Algorithme

Supposons les assignements connus, afin d'obtenir les moyennes qui minimisent la fonction objective il suffit de calculer les moyennes des points assignés à chaque groupe:

$$\mu_j = \frac{\sum_{i=1}^n r_{i,j} \mathbf{x}_i}{\sum_{i=1}^n r_{i,j}}$$

On peut donc en quelque sorte minimiser la fonction objective en alternance.

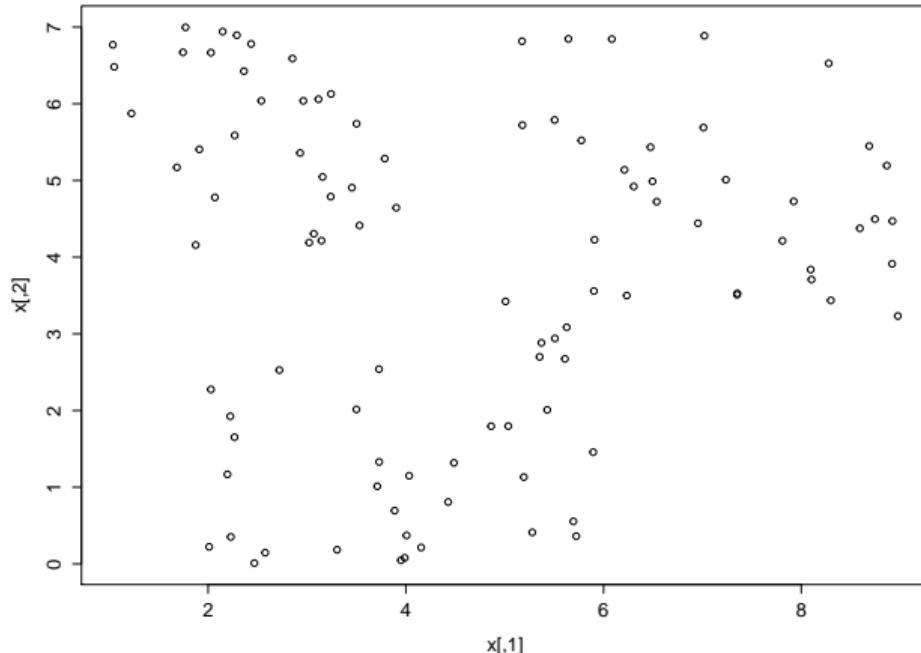
## Algorithme

Le partitionnement en  $k$ -moyenne est fondé sur un algorithme itératif qui exécute deux opérations jusqu'à atteindre une stabilité:

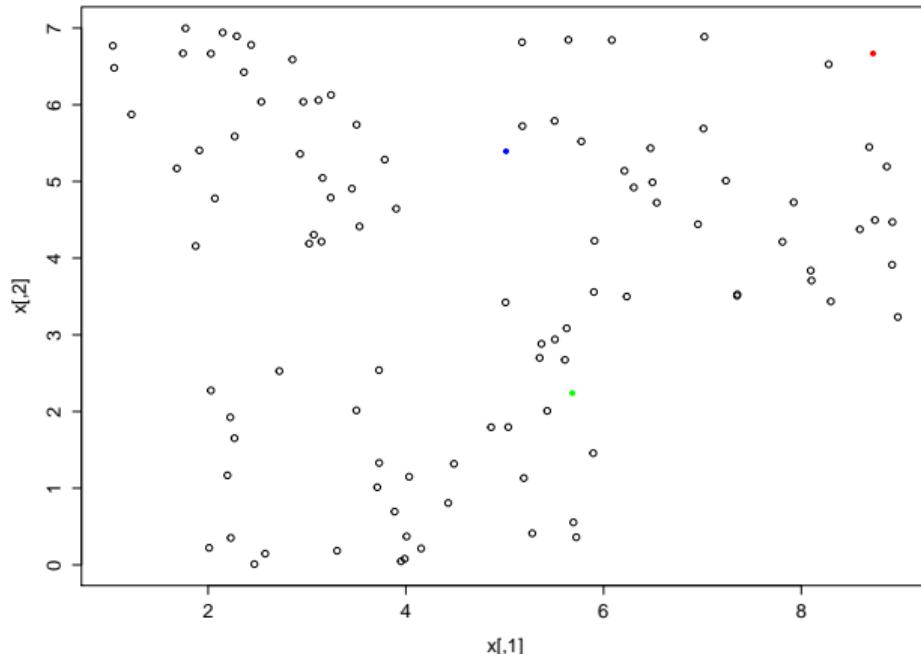
1. Étant donné des groupes d'observations, on estime les moyennes des groupes.
2. Étant données des moyennes de groupe, on assigne les observations aux groupes.

Après avoir exécuté ces deux opérations quelques fois, les moyennes cessent de bouger et les observations cessent de changer de groupes; le processus se stabilise.

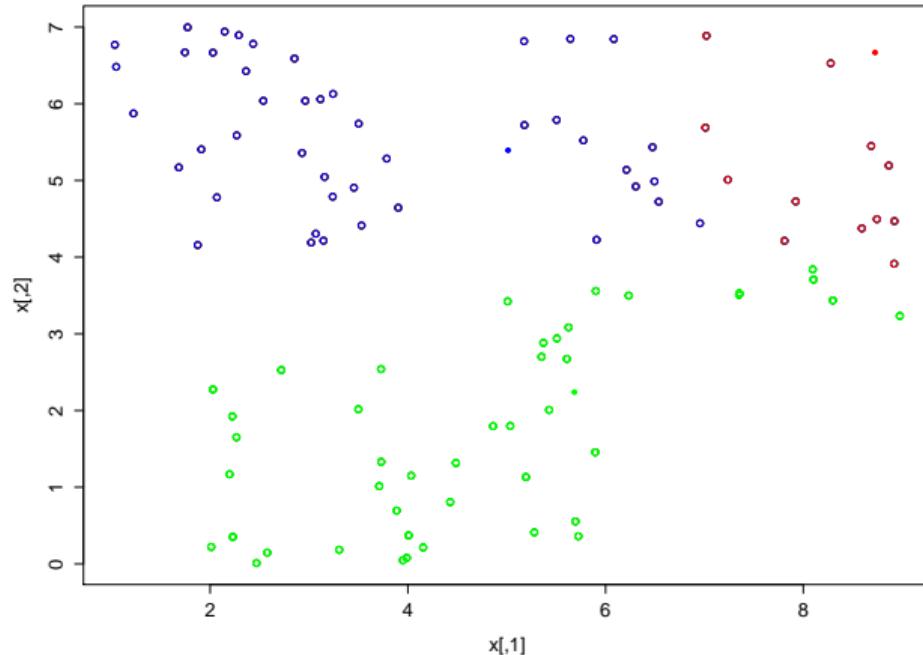
## Algorithme: exemple



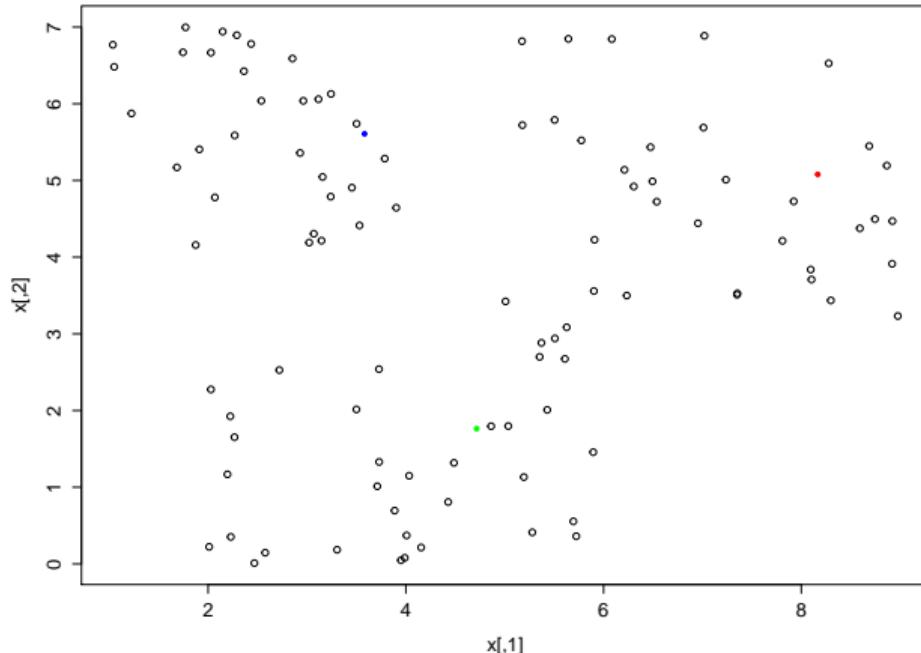
## Algorithme: exemple



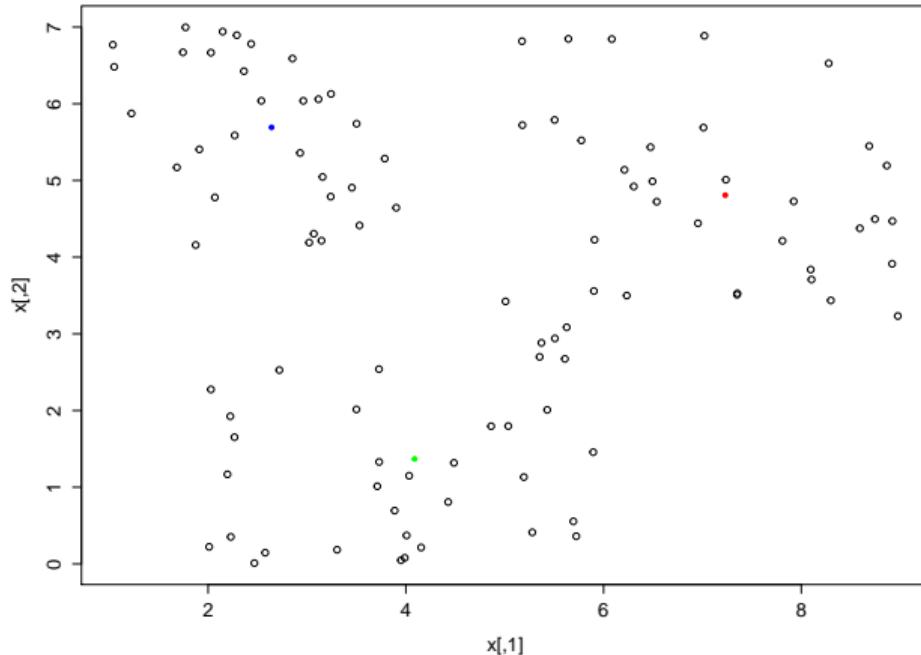
## Algorithme: exemple



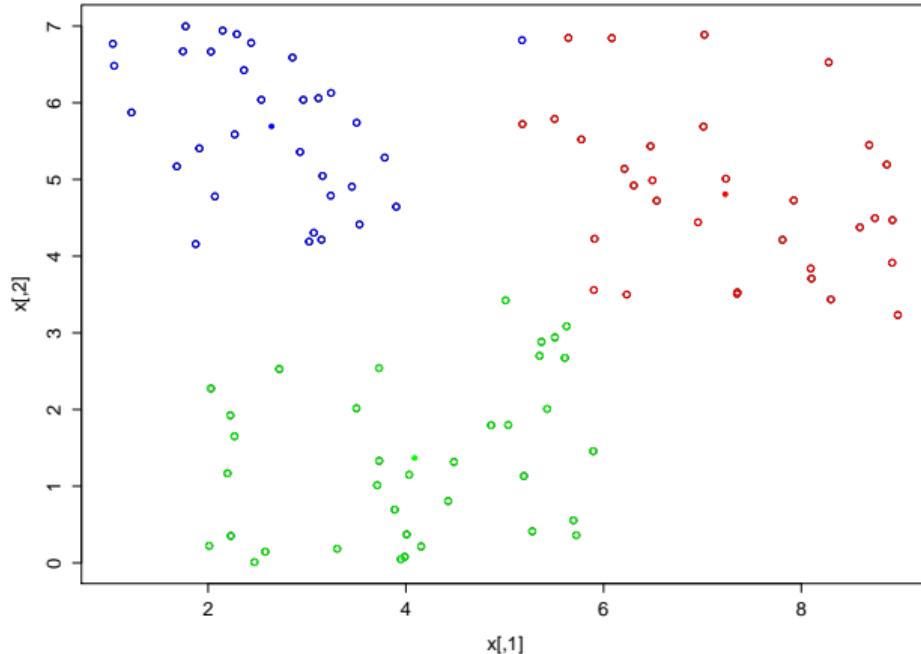
## Algorithme: exemple



## Algorithme: exemple



## Algorithme: exemple



## Algorithme

- ▶ Il s'agit donc d'un algorithme itératif qui à chaque étape met à jour les centres des groupes puis les assignements.
- ▶ Bonne nouvelle: le moment où l'on ne change plus les assignements, la moyenne demeure la même et donc le processus itératif peut arrêter puisque les centres/assignements ne bougeront plus aussi.

## Algorithme: problèmes

Ça prend un point de départ!

On va souvent prendre un point de départ aléatoire:

- ▶ des assignements aléatoires OU
- ▶ des moyennes aléatoires.

# Algorithme: exemple avec assignements aléatoires

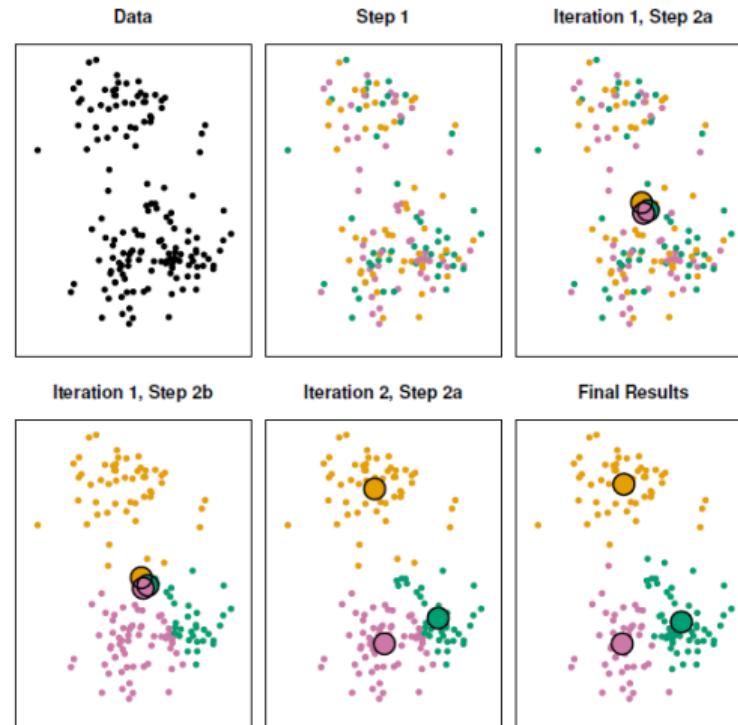


Figure 1: Figure 12.8 de James (2013)

---

## Algorithm 12.2 *K*-Means Clustering

---

1. Randomly assign a number, from 1 to  $K$ , to each of the observations. These serve as initial cluster assignments for the observations.
  2. Iterate until the cluster assignments stop changing:
    - (a) For each of the  $K$  clusters, compute the cluster *centroid*. The  $k$ th cluster centroid is the vector of the  $p$  feature means for the observations in the  $k$ th cluster.
    - (b) Assign each observation to the cluster whose centroid is closest (where *closest* is defined using Euclidean distance).
- 

Figure 2: Extrait de James (2013)

## Algorithme: problèmes

- ▶ Comme le point de départ est aléatoire, on peut obtenir différentes solutions.
- ▶ Parfois on peut restez pris dans un minimum local et non global.

## Algorithme: problèmes



Figure 3: Figure 12.9 de James (2013)

## Algorithme: problèmes

- ▶ Afin d'utiliser l'algorithme, il faut avoir préalablement déterminé le nombre de groupes  $k$  parce que  $k$  est ...
- ▶ un hyperparamètre.
- ▶ Des fois, on peut déterminer  $k$  en fonction du problème.
- ▶ Sinon on fait quoi ?

## Déterminer le nombre de groupe $k$

Dans un problème d'apprentissage supervisé, on utilise la validation croisée ou un ensemble de validation pour déterminer les hyperparamètres.

C'est-à-dire que l'on met au point une collection de modèles (pour différents  $k$ ,  $d$  ou  $\lambda$ ) et que l'on teste chacun des modèles appris sur de nouvelles données (validation).  
Ensuite, on choisit la valeur de l'hyperparamètre qui maximise la performance sur les données de validation.

Par contre, en apprentissage non-supervisé, il n'y existe pas de mesure de performance.  
Alors comment peut-on choisir la valeur de l'hyperparamètre qui maximise la performance ?

## Déterminer le nombre de groupe $k$

- ▶ On pourrait utiliser la fonction objective comme mesure de performance.
- ▶ Mais comme avant, plus on augmente la complexité plus la performance sur les données teste s'améliore: ici plus  $k$  est grand plus la somme des distances est petites.
- ▶ Par exemple si  $k = n$  on a donc  $Obj = 0$ .
- ▶ Il n'y existe pas de solution parfaite.

## Déterminer le nombre de groupe $k$

- Une solution standard est de calculer une fonction objective pénalisée!

Par exemple:

$$Obj_{\text{pen}} = \sum_{i=1}^n \sum_{j=1}^k r_{i,j} \|\mathbf{x}_i - \mu_j\|^2 + \lambda k, \quad (3)$$

À un certain moment, l'ajout d'une composante  $k$  réduira la somme des distances moins que  $\lambda$  et nous aurons trouvé un équilibre.

## Déterminer le nombre de groupe $k$

- ▶ Comment choisir  $\lambda$  ? Pas facile.
- ▶ Des techniques bien établis pour des modèles de vraisemblance (AIC,BIC) suggèrent  $\lambda = n$  ou  $\lambda = \sqrt{n}$ , mais ça fait peu de sens ici, on parlera surtout au prochains cours.
- ▶ Disons que pour le partitionnement en  $k$ -moyenne, il n'y pas de vraie solution.

## Déterminer le nombre de groupe $k$

Souvent le nombre de groupe  $k$  sera déterminé par

- ▶ l'objectif du projet de recherche,
- ▶ par les collaborateurs,
- ▶ suite à une visualisation des données.

# Clustering with R & References

La référence la plus simple est [Husson et al. \(2011\)](#), qui détaille la mise en pratique sous R,

```
1 > FactoMineR::HCPC()  
2 > cluster::agnes()  
3 > stats::hclust()  
4 > stats::kmeans()
```

# Objectif

- ▶ On a  $n$  observations dans un espace de dimension  $p$ :

$$\mathbf{x}_i = (x_{i1}, x_{i2}, \dots, x_{ip}) \in \mathcal{X} \subset \mathbb{R}^p, \quad i = 1, \dots, n$$

et aucune variable réponse.

- ▶ Chacune des observations est caractérisée par un poids  $p_i$ ,  $i = 1, \dots, n$  tels que
$$\sum_{i=1}^n p_i = 1.$$
- ▶ Généralement, on a  $p_i = 1/n$ ,  $i = 1, \dots, n$ .
- ▶ On veut déterminer  $K$  groupes  $C_1, \dots, C_K$ .

## Indice de similarité

Si on note  $N$  l'ensemble des observations  $\{1, \dots, n\}$ , alors l'**indice de similarité** est une application  $s : N \times N \rightarrow \mathbb{R}^+$  telle que

$$\begin{aligned}s(i, j) &= s(j, i), & \forall (i, j) \in N \times N \\s(i, i) &= \bar{s} > 0, & \forall i \in N \\s(i, j) &\leq \bar{s}, & \forall (i, j) \in N \times N.\end{aligned}$$

On peut également définir une version normée de cet indice:

$$s^*(i, j) = \frac{1}{\bar{s}} s(i, j), \quad \forall (i, j) \in N \times N.$$

## Indice de dissimilarité

Si on note  $N$  l'ensemble des observations, alors l'**indice de dissimilarité** est une application  $d : N \times N \rightarrow \mathbb{R}^+$  telle que

$$\begin{aligned} d(i, j) &= d(j, i), & \forall (i, j) \in N \times N \\ d(i, j) &= 0 \Leftrightarrow i = j. \end{aligned}$$

On peut également définir une version normée de cet indice:

$$d^*(i, j) = \frac{1}{d} d(i, j), \quad \forall (i, j) \in N \times N.$$

On a alors

$$d^*(i, j) = 1 - s^*(i, j) \text{ et } s^*(i, j) = 1 - d^*(i, j).$$

## Distance

Une distance est un indice de dissimilarité qui vérifie, en plus, la propriété suivante (inégalité triangulaire):

$$d(i, j) \leq d(i, k) + d(k, j), \quad \forall (i, j, k) \in N \times N \times N.$$

Pour  $\mathbf{x}, \mathbf{y} \in \mathbb{R}^p$ , on peut définir plusieurs mesures de distance (présentées dans  $\mathbb{R}^2$ )

- ▶ la mesure **euclidienne** ( $\ell_2$ ):

$$d^2(x, y) = (x_1 - y_1)^2 + (x_2 - y_2)^2;$$

- ▶ la mesure normalisée de **Mahalanobis**:  $Q = D_{s^2}^{-1} = \text{diag}(s_1^{-1}, \dots, s_p^{-1})$ , où  $s_j^2$  est la variance de la  $j$ ème composante de  $x$ ; et la mesure euclidienne généralisée avec mesure  $\mathbf{Q} > 0$  par

$$d^2(x, y) = (\mathbf{x} - \mathbf{y})^\top \mathbf{Q}(\mathbf{x} - \mathbf{y}).$$

# Distance

- ▶ la distance de **Manhattan** ( $\ell_1$ ):

$$d(x, y) = |x_1 - y_1| + |x_2 - y_2|;$$

- ▶ la distance **euclidienne** ( $\ell_2$ ):

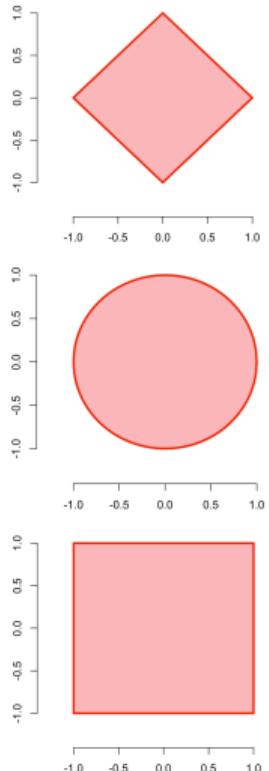
$$d^2(x, y) = (x_1 - y_1)^2 + (x_2 - y_2)^2;$$

- ▶ la distance de **Mahalanobis** ( $\ell_2$ ):

$$d^2(x, y) = \frac{(x_1 - y_1)^2}{\sigma_1^2} + \frac{(x_2 - y_2)^2}{\sigma_2^2};$$

- ▶ la distance **maximale** ( $\ell_\infty$ ):

$$d(x, y) = (\max(|x_1 - y_1|, |x_2 - y_2|));$$



## Distance

- ▶ On peut définir la distance de **Minkowski d'ordre  $p$**  ( $\ell_p$ ):

$$d(x, y) = ((x_1 - y_1)^p + (x_2 - y_2)^p)^{1/p};$$

- ▶ On utilise généralement la distance **euclidienne** si toutes les variables sont mesurées sur la même échelle.
- ▶ Si toutes les variables ne sont pas mesurées sur la même échelle, on utilise plutôt la mesure normalisée (Mahalanobis).
- ▶ De façon équivalente, on peut également centrer et réduire les données avant d'utiliser la distance euclidienne.

# Euclidean Distance

Consider  $d^2(\mathbf{u}, \mathbf{v}) = (\mathbf{u} - \mathbf{v})^\top (\mathbf{u} - \mathbf{v}) = \|\mathbf{u} - \mathbf{v}\|^2$

Set of points  $\mathbf{u}$  such that  $d(\mathbf{0}, \mathbf{u}) = \|\mathbf{u}\| = 1$

- ▶ circle (or sphere in higher dimension)

Set of points  $\mathbf{x}$  such that  $d(\mathbf{x}, \mathbf{u}) = d(\mathbf{x}, \mathbf{v})$  ?

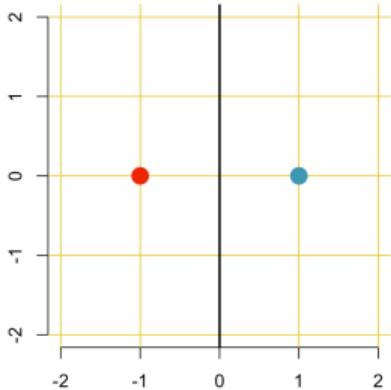
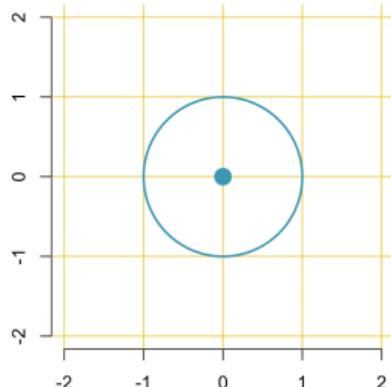
- ▶ straight line (or plane in higher dimension)

(orthogonal to  $\mathbf{u} - \mathbf{v}$ )



KEY
1 YOSEMITE
2 KINGS CANYON
3 SEQUOIA
4 DEATH VALLEY
5 ZION
6 BRYCE CANYON
7 CAPITOL REEF
8 ARCHES
9 BLACK CANYON OF THE GULCH
10 CANTONLANDS
11 MESA VERDE
12 PECOS
13 PETRIFIED FOREST
14 GUADALUPE
15 MONTANARDO
16 EVERGLADES

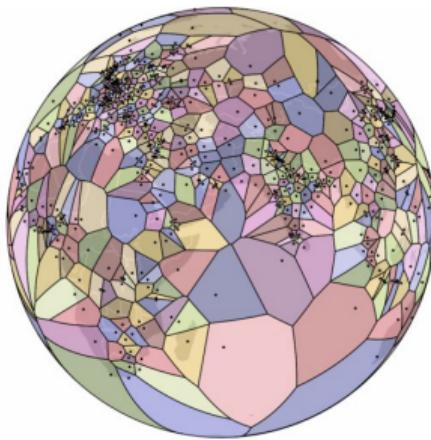
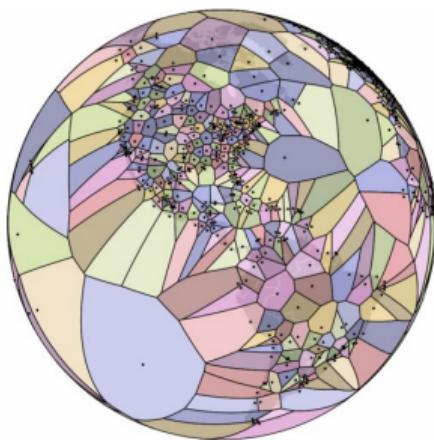
The United States  
partitioned by  
closest  
national park



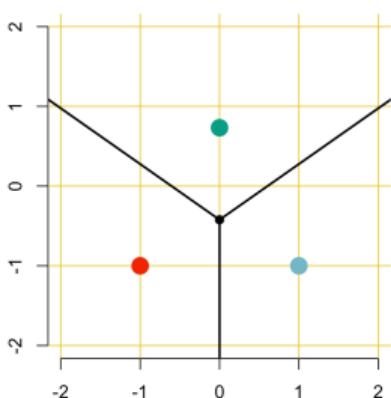
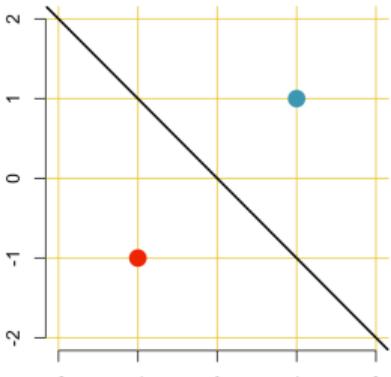
# Euclidean Distance & Voronoi Sets

Given a set of points  $u_1, u_2, \dots, u_n$

Set of points  $x$  the closest to  $u_i$ ?  
(see also Delaunay triangulation)



(intersections are straight lines or planes)

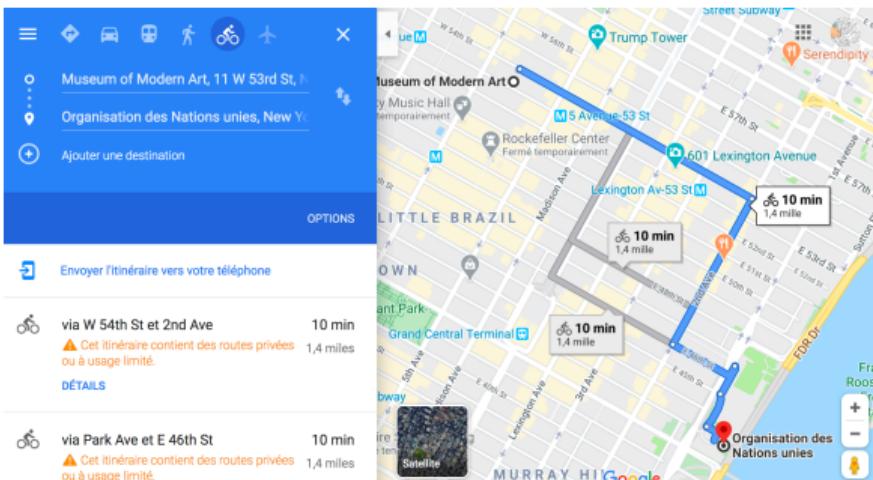


# Manhattan ( $\ell_1$ ) Distance

$\|\mathbf{u}\| = |u_1| + |u_2|$  is also a norm (Manhattan)

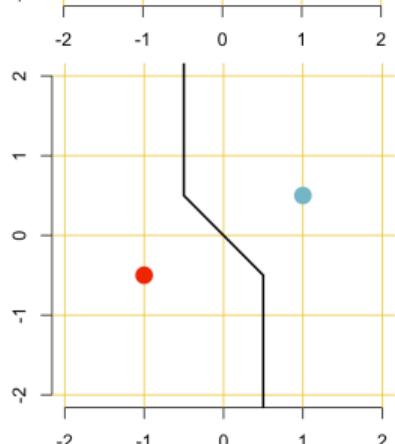
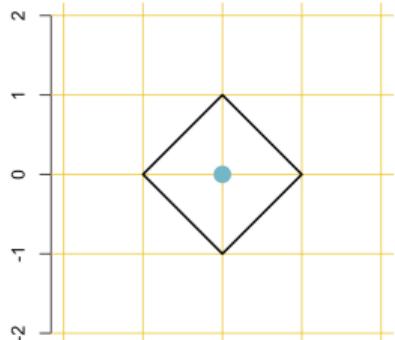
Set of points  $\mathbf{u}$  such that  $d(\mathbf{0}, \mathbf{v}) = \|\mathbf{u}\| = 1$

- ▶ square (or cube in higher dimension)



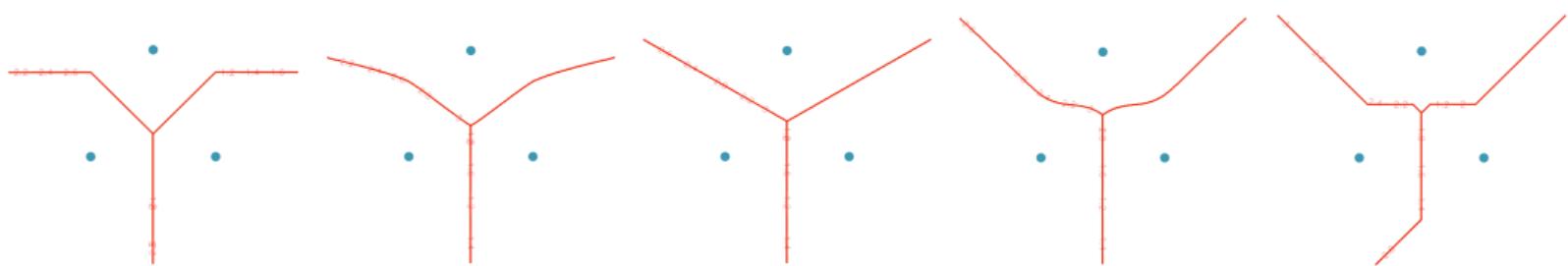
Set of points  $\mathbf{x}$  such that  $d(\mathbf{x}, \mathbf{u}) = d(\mathbf{x}, \mathbf{v})$  ?

- ▶ portions of straight lines (or planes)



# $\ell_p$ Distance & Voronoi Sets

$$p \in \{1, 1.5, 2, 3, \infty\}, d(\mathbf{x}, \mathbf{y}) = \left( \sum_{i=1}^d |x_i - y_i|^p \right)^{1/p}$$



## Partition

- ▶ Une **partition**  $\mathcal{P}_K$  de  $N$  en  $K$  groupes est un ensemble  $\{C_1, \dots, C_K\}$  de groupes non-vides, d'intersections nulles et dont l'union permet de retrouver  $N$ , c'est-à-dire
  - ▶  $C_k \neq \emptyset, \forall k \in \{1, \dots, K\};$
  - ▶  $C_k \cap C_j = \emptyset, \forall k \neq j;$  et
  - ▶  $C_1 \cup \dots \cup C_K = N.$

**Example:** On considère  $N = \{1, 2, 3, 4, 5, 6, 7\}$  et

$$\mathcal{P}_3 = \{\{7\}, \{5, 4, 6\}, \{1, 2, 3\}\}.$$

Il s'agit d'une partition.

## *k*-means

Given  $n$  points  $\mathbf{x}_i$  in  $\mathbb{R}^d$ , we want to find  $\mu_1^*, \dots, \mu_k^*$ , solutions of

$$V_{k\text{-means}} = \min \left\{ \sum_{j=1}^k \sum_{i \in S_j} \|\mathbf{x}_i - \mu_j\|^2 \right\}$$

that we can write

$$V_{k\text{-means}} = \min \left\{ \sum_{i=1}^n \min_{j=1, \dots, k} \underbrace{\|\mathbf{x}_i - \mu_j\|^2}_{\text{within inertia}} \right\}$$

In  $k$ -means, clusters are defined as Voronoi sets obtained from means  $\mu_1, \dots, \mu_k$

## *k*-means

Finding the optimal solution to the  $k$ -means clustering problem for  $n$  observations in  $d$  dimensions can be (exactly) solved in time  $O(n^{dk+1} \log n)$ ... which is very long.

Given an initial set of  $k$ -means,  $\mu_1, \dots, \mu_k$ , alternate between the two following steps (also called **Lloyd's algorithm**, from [Lloyd \(1982\)](#))

- ▶ Assignment step: we partition observations according to the Voronoi diagram generated by  $\mu_1, \dots, \mu_k$ 's,

$$S_i^{(t)} = \{x_j : \|x_j - \mu_i^{(t)}\|^2 \leq \|x_j - \mu_{i'}^{(t)}\|^2 \forall i' \in \{1, \dots, k\}\},$$

where each  $x_j$  is assigned to exactly one class  $S_i^{(t)}$ , at step  $t$ .

- ▶ Update step: we compute the new centroids of the clusters,

$$\mu_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \sum_{x_j \in S_i^{(t)}} x_j$$

## *k*-means

Key why this might work :

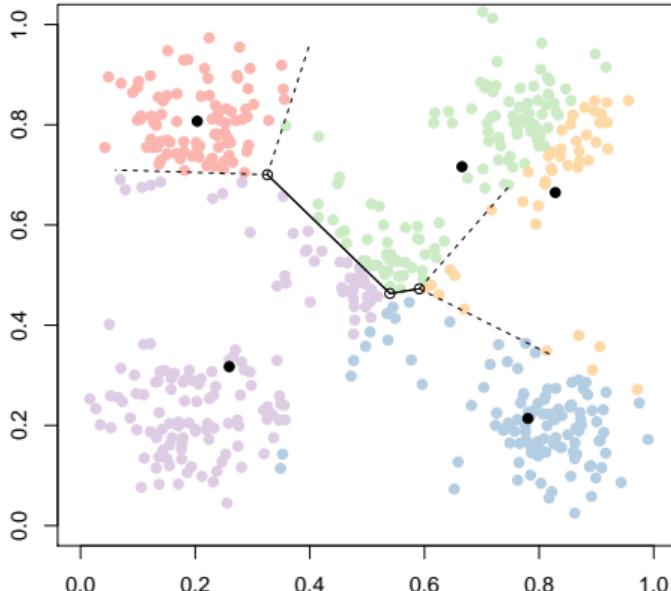
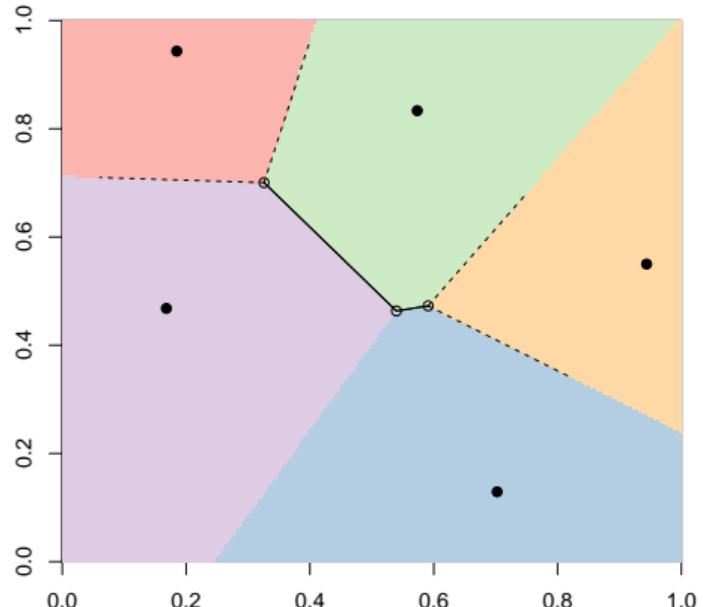
$$\frac{1}{\#I_j} \sum_{i,i' \in I} \sum_{j=1}^k [x_{il} - x_{i'l}]^2 = 2 \sum_{i \in I} \sum_{j=1}^k [x_{il} - \bar{x}_{jl}]^2 \text{ where } \bar{x}_{jl} = \frac{1}{\#I_j} \sum_{i \in I_j} x_{i,l}$$

The value of

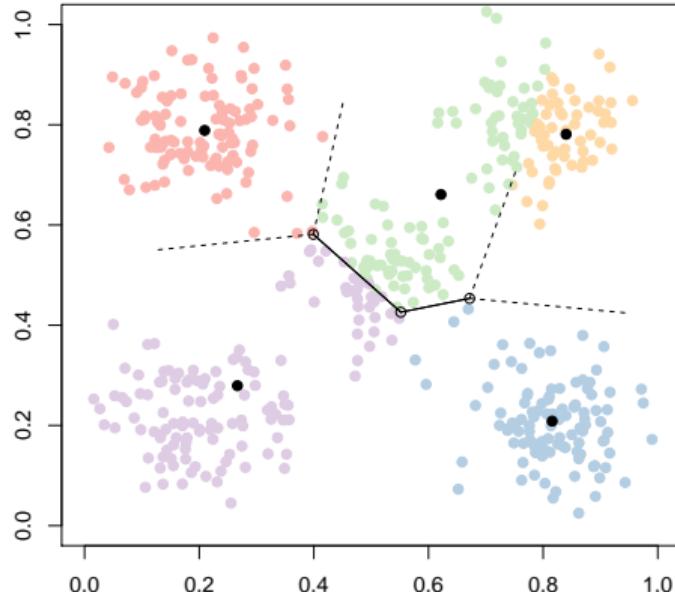
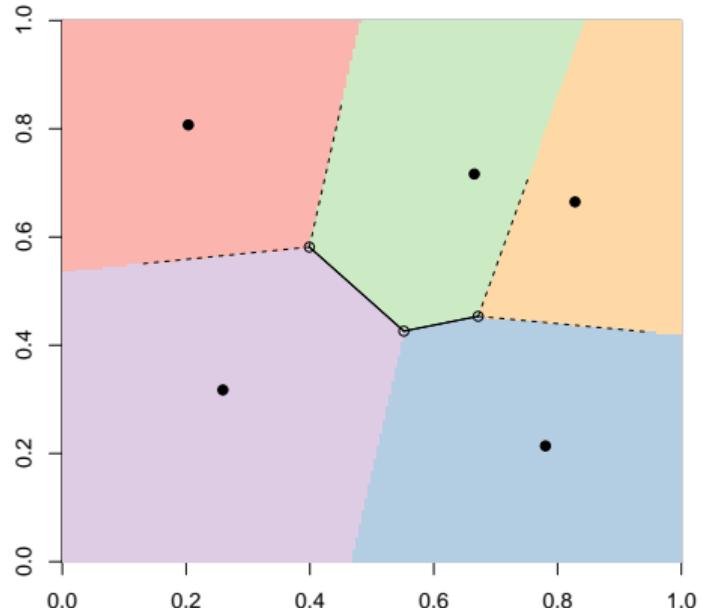
$$\min_{I_1, \dots, I_K} \left\{ \sum_{j=1}^K \frac{1}{\#I_j} \sum_{i,i' \in I_j} \sum_{l=1}^k [x_{il} - x_{i'l}]^2 \right\}$$

will necessarily decrease (but no guarantee to reach the minimum, since the function is not convex).

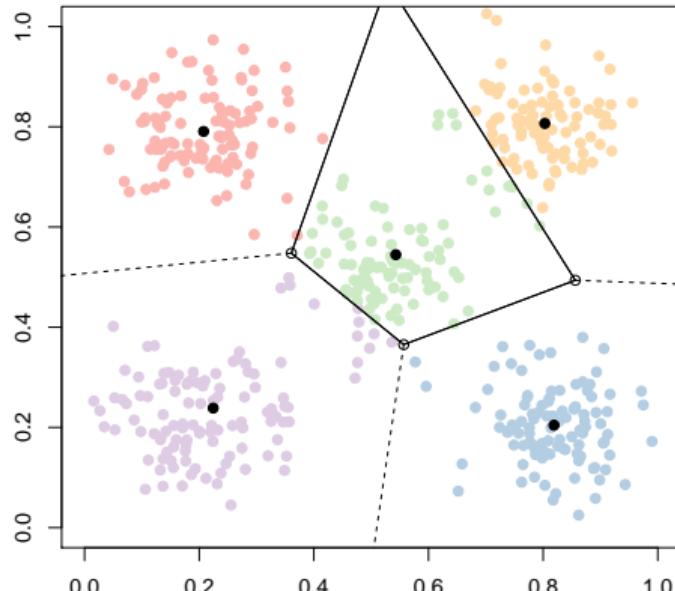
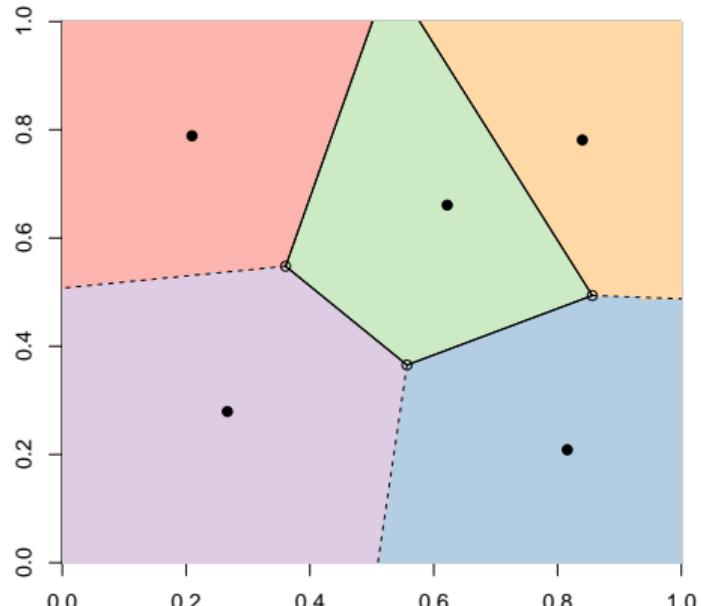
## *k*-means



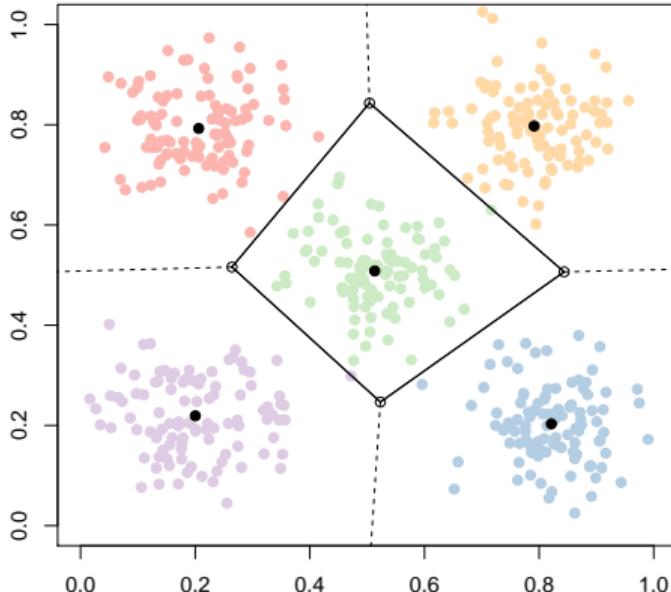
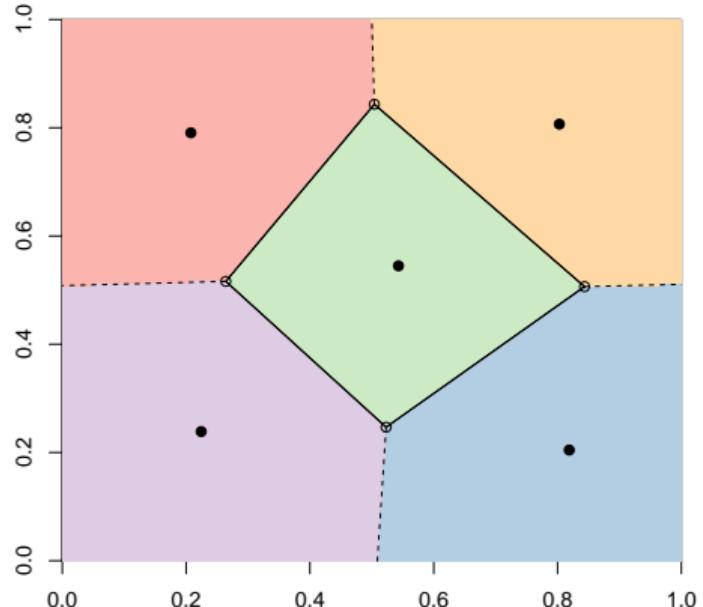
## *k*-means



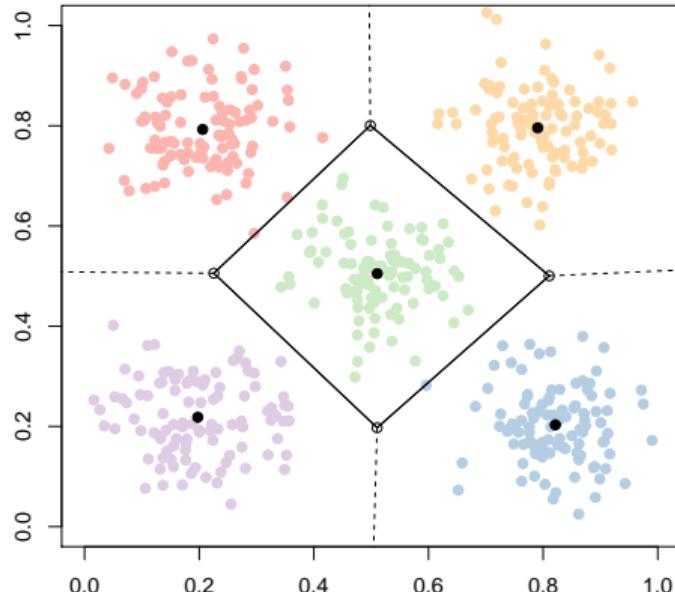
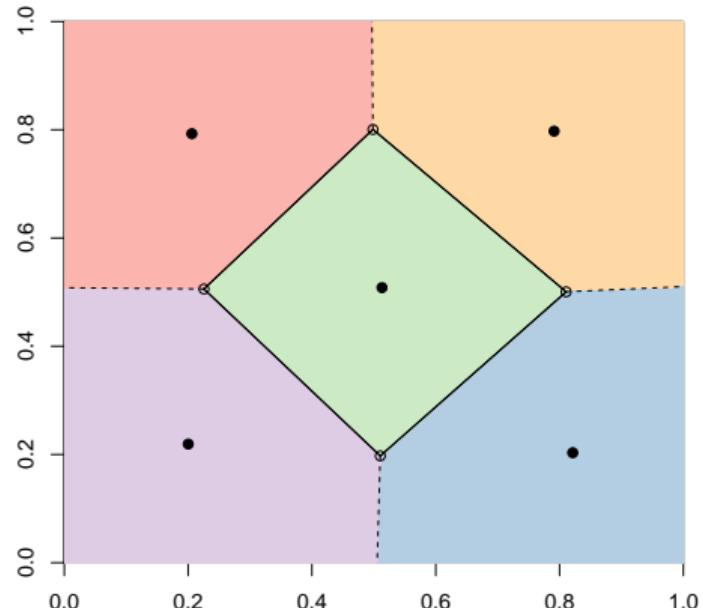
## *k*-means



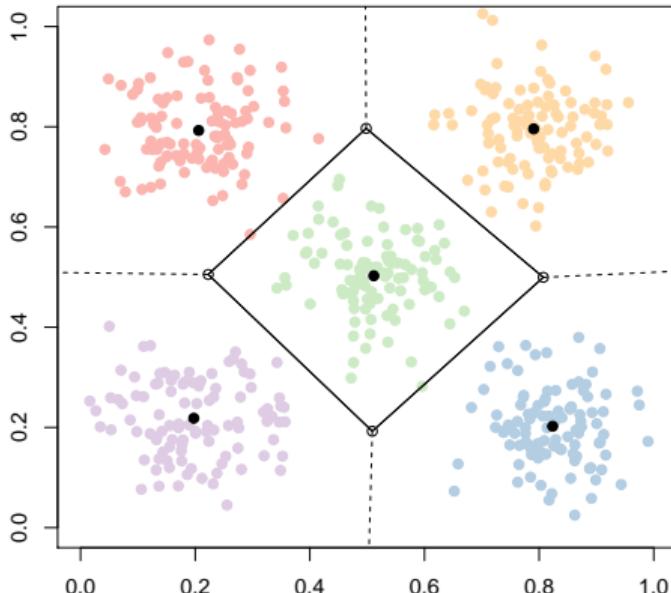
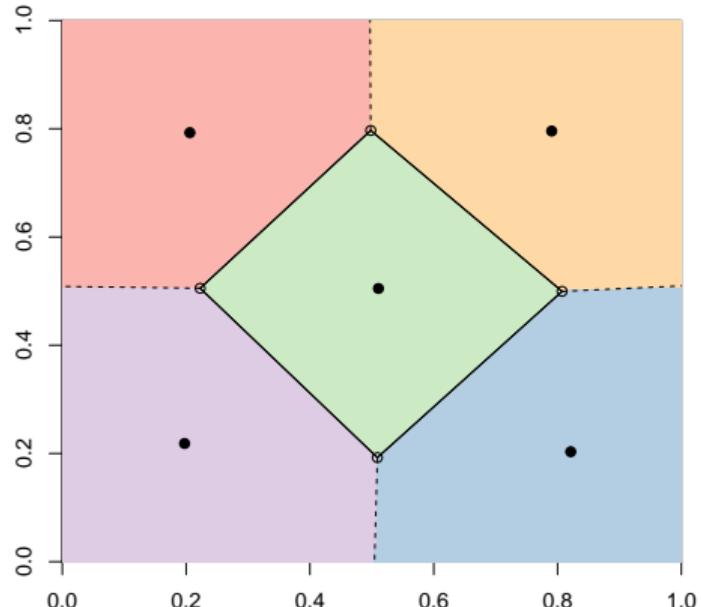
## *k*-means



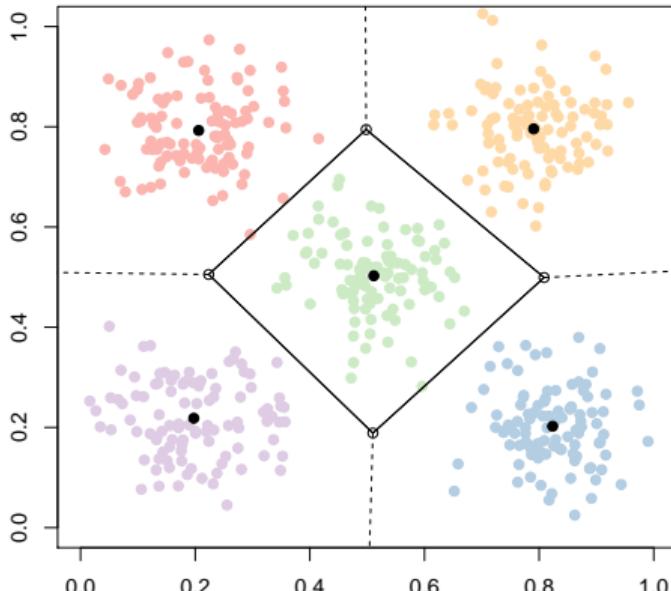
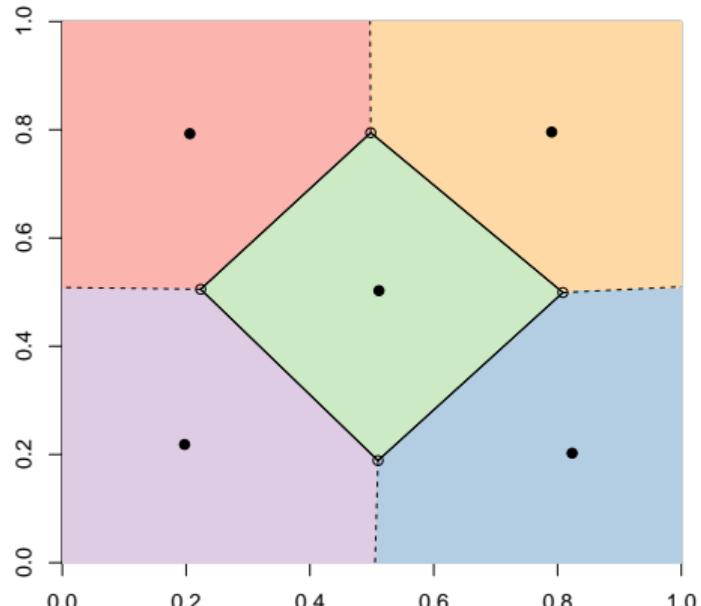
## *k*-means



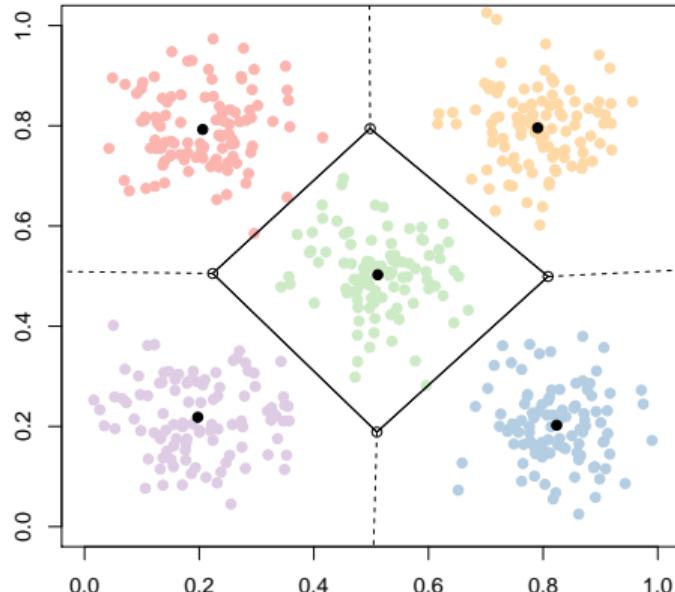
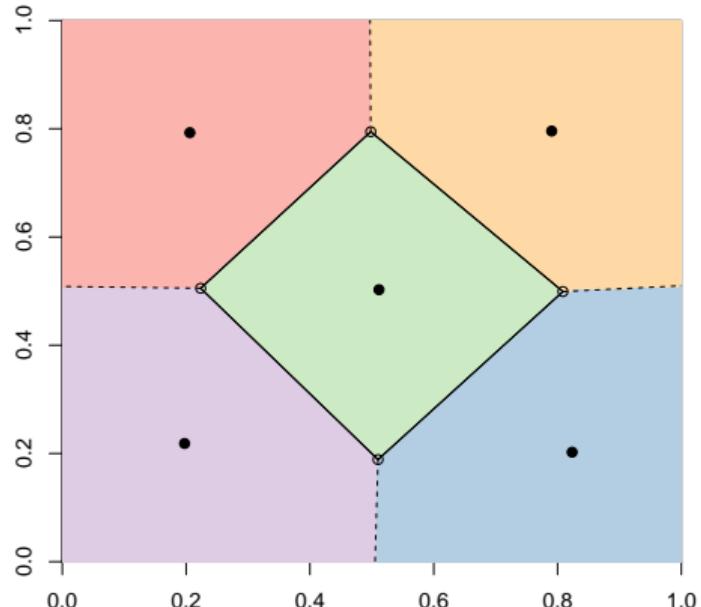
## *k*-means



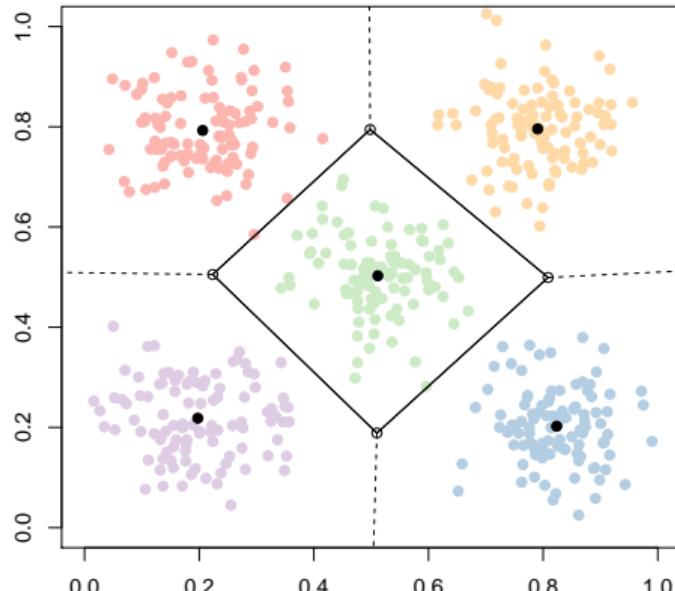
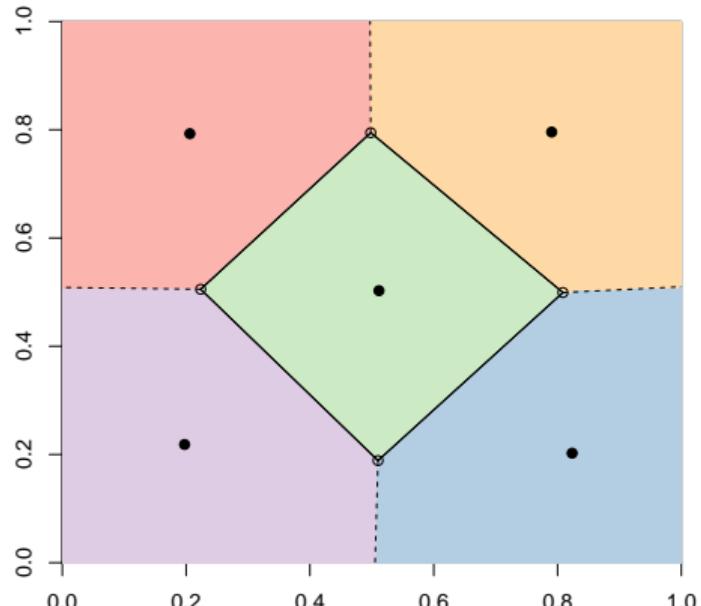
## *k*-means



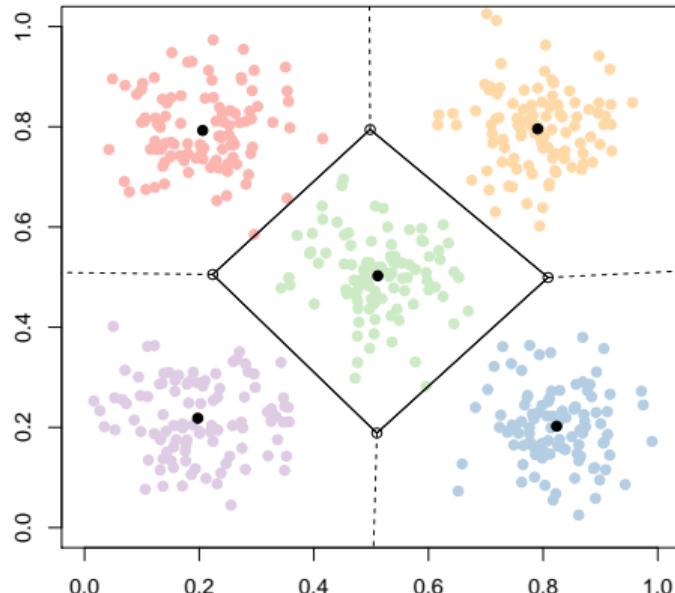
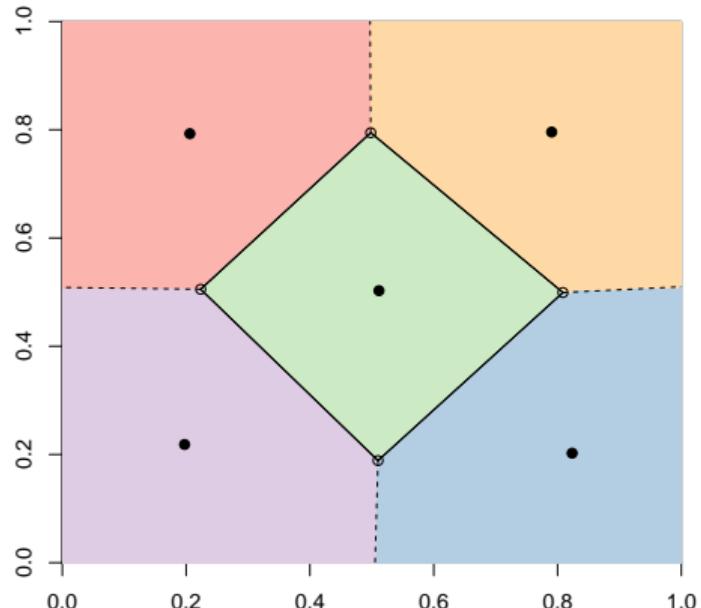
## *k*-means



## *k*-means



## *k*-means



## *k*-means

On peut visualiser des clusters d'individus sur la sortie d'une ACP\*  
(par exemple les secteurs d'activité en Europe)

```
1 > jobs = read.csv(header=TRUE,  
2 "http://user.math.uzh.ch/furrer/download/sta121/europejobs.csv"  
3 )  
4 > res = PCA(jobs)  
5 > ind = get_pca_ind(res)  
6 > res.km = kmeans(ind$coord, centers = 3, nstart = 15)  
7 > grp = as.factor(res.km$cluster)  
8 > fviz_pca_ind(res, col.ind = grp)
```

En faisant une ACP, on a des observations dans  $\mathbb{R}^5$  (par défaut)

\* ACP, Analyse en Composantes Principales, discuté la semaine prochaine...

# *k*-means



## *k*-means Interpretation

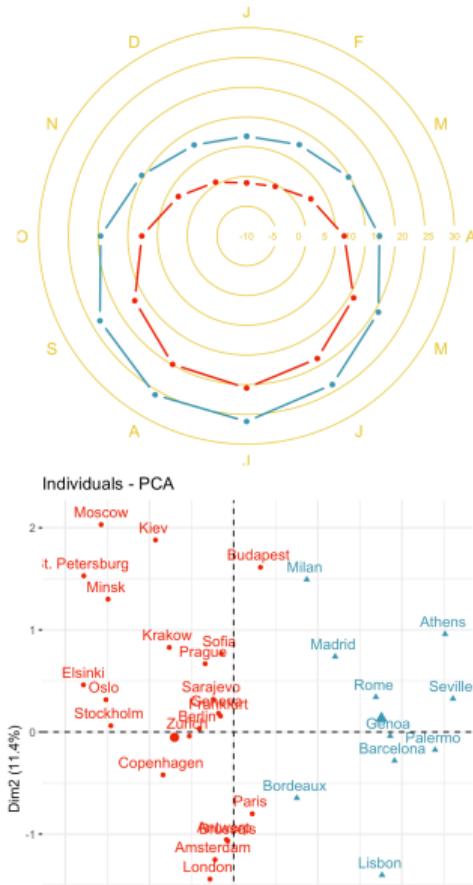
As it typically converges to a **local minimum**, it is optimal to run *k*-means algorithm several times so as to empirically determine good clusters.

*k*-means work well when (true) clusters are compact and convex subsets (linear decision boundaries). If not, we cannot expect to converge.

**Remark:** This is not what we do with hierarchical clustering.

# Clusters Interpretation

```
1 > res = PCA(temperature[,1:12])
2 > ind = get_pca_ind(res)
3 > res.km = kmeans(ind$coord, centers = 2,
      nstart = 15)
4 > grp = as.factor(res.km$cluster)
5 > df = data.frame(grp, temperature[,1:12])
6 > dsc = aggregate(df[,2:13], by=list(df$grp),
      ,mean)
7 > dsc
     Jan   Feb   Mar   Apr   May   Jun
1 -1.2 -0.4  2.8  7.3 12.3 15.8
2  7.7  8.8 11.2 14.1 17.8 21.5
     Jul   Aug   Sep   Oct   Nov   Dec
1 17.8 17.1 13.4  8.7  3.7  0.5
2 24.2 23.8 21.1 16.7 11.9  8.8
```



## Hiérarchie

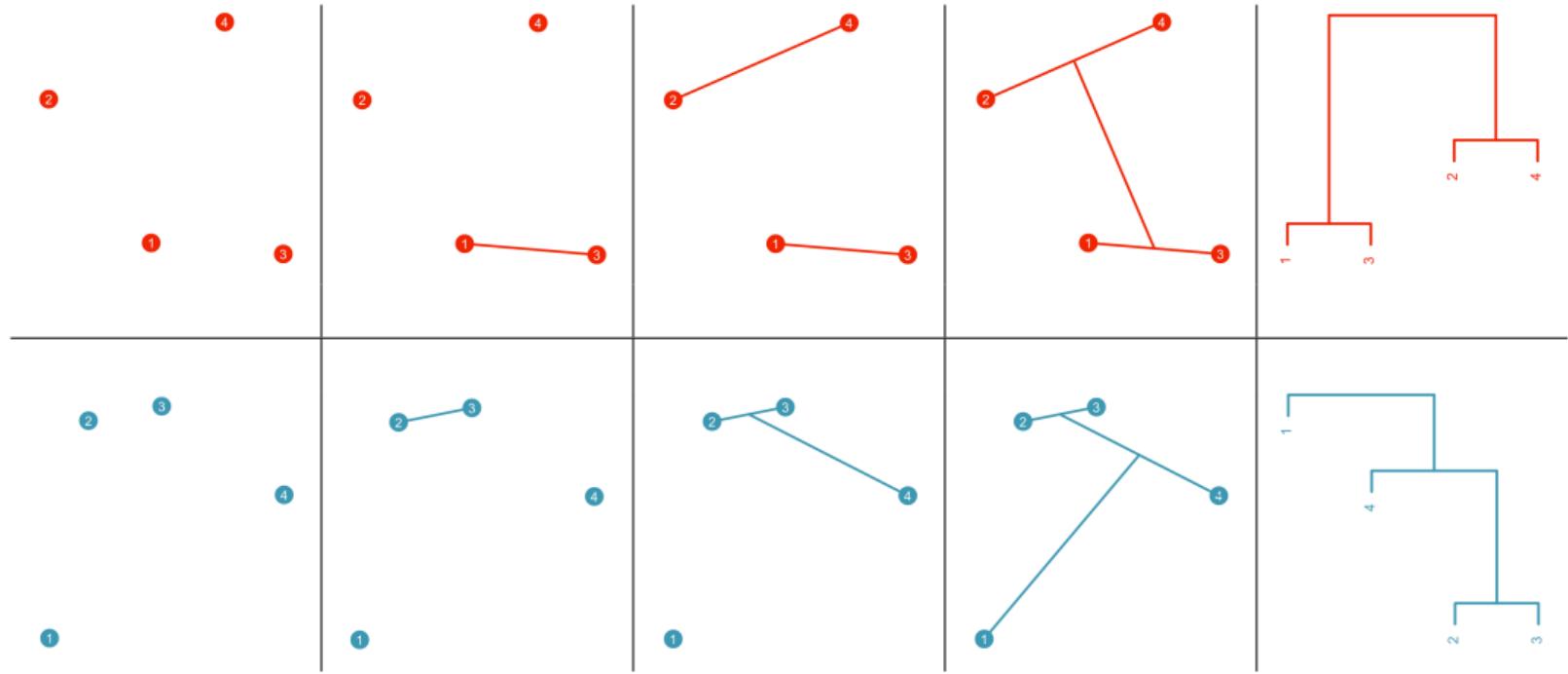
- ▶ Une **hiérarchie**  $\mathcal{H}$  de  $N$  est un ensemble  $\{C_1, \dots, C_K\}$  de groupes non-vides tels que
  - ▶  $N \in \mathcal{H}$ ;
  - ▶  $\forall i \in N, \{i\} \in \mathcal{H}$ ; et
  - ▶  $\forall A, B \in N, A \cap B \in \{A, B, \emptyset\}$ .

**Example:** On considère  $N = \{1, 2, 3, 4, 5, 6, 7\}$  et

$$\mathcal{H} = \{\{1\}, \dots, \{7\}, \{4, 5\}, \{2, 3\}, \{4, 5, 6\}, \{1, 2, 3\}, \{4, 5, 6, 7\}, N\}.$$

Vérifier qu'il s'agit d'une hiérarchie.

# Regroupement Hiérarchique



## Inertie

- ▶ L'inertie permet de décrire à l'aide d'un seul nombre l'hétérogénéité d'un groupe de points.
- ▶ Soit  $\mathbf{x} = \begin{bmatrix} x_1 & x_2 & \cdots & x_p \end{bmatrix} \in \mathbb{R}^p$  un nuage dont chacun des points est caractérisé par un poids  $p_i$ ,  $i = 1, \dots, n$  tels que  $\sum_{i=1}^n p_i = 1$ .
- ▶ Généralement, on a  $p_i = 1/n$ ,  $i = 1, \dots, n$ .

## Inertie

- Le centre de gravité d'un nuage de points est défini par

$$g = \sum_{i=1}^n p_i \mathbf{x}_i = \bar{\mathbf{x}}.$$

- L'inertie totale d'un nuage de point est donnée par

$$I_T = \sum_{i=1}^n p_i d^2(\mathbf{x}_i, g),$$

où  $d()$  est la distance euclidienne.

## Inertie

- ▶ Dans une procédure de classification, le nuage de points sera partitionné en  $K$  groupes  $C_1, \dots, C_K$  dont les poids respectifs sont donnés par  $P_j$  tel que

$$P_j = \sum_{i: \mathbf{x}_i \in C_j} p_i.$$

- ▶ Le centre de gravité de chacun des groupes est donné par

$$g_j = \frac{1}{P_j} \sum_{i: \mathbf{x}_i \in C_j} p_i \mathbf{x}_i.$$

## Inertie

- ▶ L'inertie entre les groupes (**between clusters inertia**) est donnée par

$$I_B = \sum_{j=1}^K P_j d^2(g_j, g).$$

- ▶ L'inertie interne des groupes (**within clusters inertia**) est donnée par

$$I_W = \sum_{j=1}^K P_j I_{C_j}, \quad I_{C_j} = \frac{1}{P_j} \sum_{i: \mathbf{x}_i \in C_j} p_i d^2(\mathbf{x}_i, g_j).$$

- ▶ Le théorème de König-Huygens indique que

$$I_T = I_B + I_W,$$

c'est-à-dire que l'inertie total d'un jeu de données est constante.

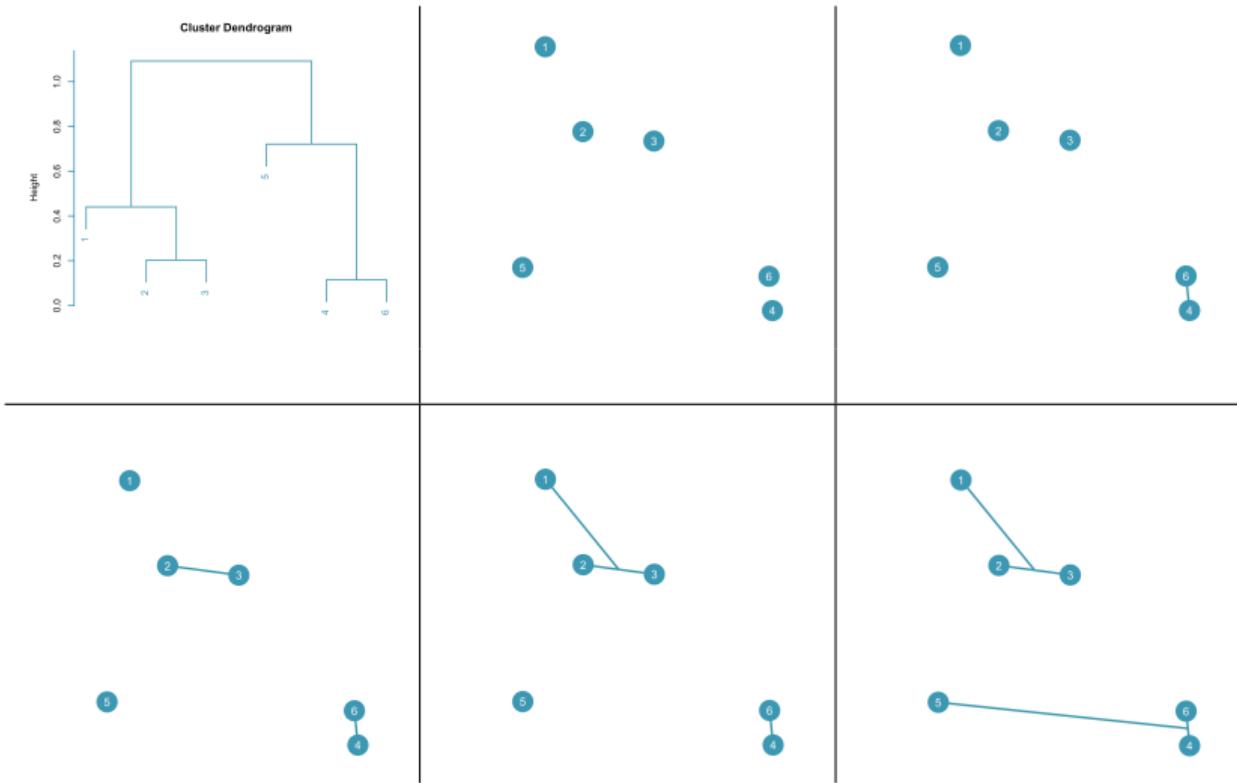
## Inertie

- ▶ Une “bonne” classification aura ainsi une inertie inter-groupe élevée et une inertie intra-groupe faible.
- ▶ Minimiser l'intertie intra-groupe est équivalent à maximiser l'inertie inter-groupe (puisque l'inertie totale est constante).
- ▶ La proportion de l'inertie totale expliquée par la partition  $\mathcal{P}_K$  est

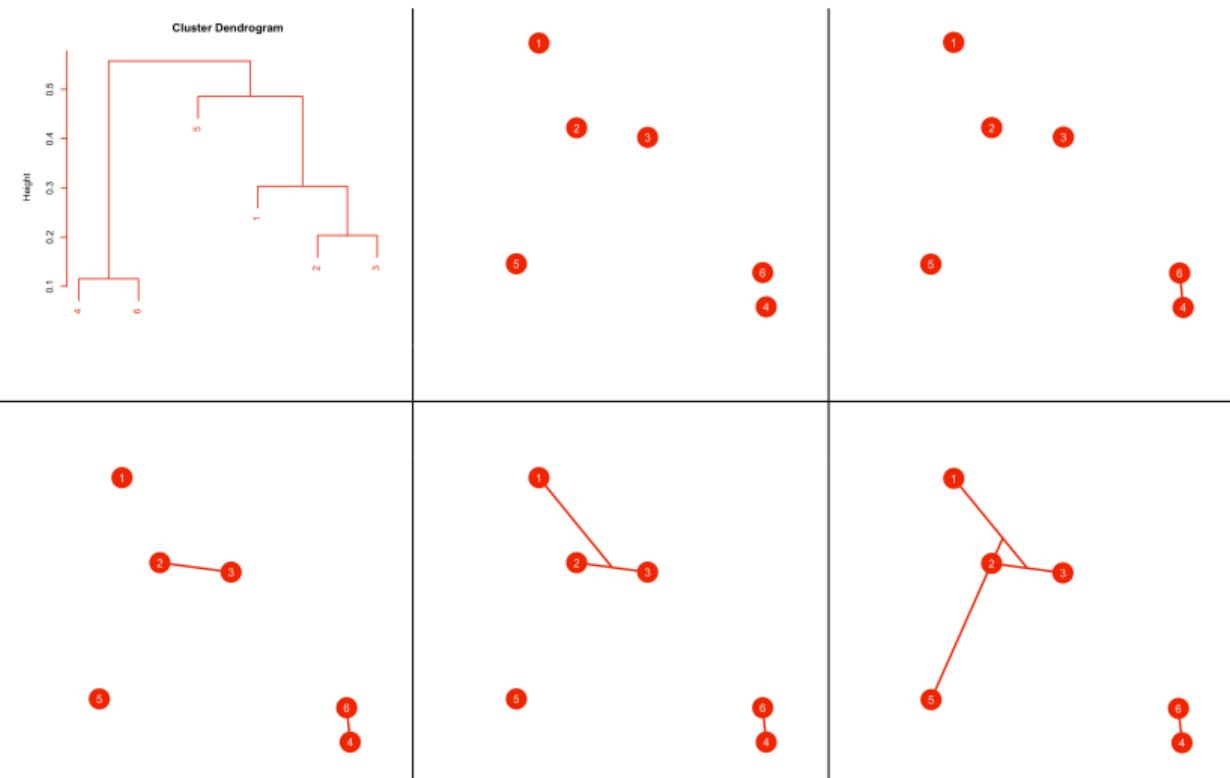
$$\frac{I_B}{I_T} = \left(1 - \frac{I_W}{I_T}\right).$$

- ▶ Cette valeur augmente lorsque le nombre de groupe(s)  $K$  augmente: elle ne peut être utilisée que pour comparer des partitions avec le même nombre de groupes.

# Exemple (complete)



# Exemple (single)



## Algorithmes hiérarchiques ascendants

- ▶ La procédure hiérarchique ascendante implique de débuter l'analyse avec le plus grand nombre possible de groupes (**cluster**) (généralement en prenant les données individuellement).
  - ▶ À chacune des étapes, on fusionne les deux groupes les plus semblables (les plus proches).
  - ▶ On poursuite la procédure jusqu'à terminer l'analyse avec le moins de groupes possibles (généralement un seul groupe contenant toutes les données).
  - ▶ On analyse l'arbre de classification (ou dendrogramme) ainsi obtenu et on choisit le nombre de classes  $K$  à conserver.
- Demande de prendre une décision supplémentaire: comment définit-on la similitude entre deux groupes?

## Algorithmes hiérarchiques ascendants

La similitude entre deux groupes peut être déterminée de plusieurs façons.

- ▶ Lien simple (**simple linkage**): la proximité entre deux groupes ( $A$  et  $B$ ) est donnée par

$$\delta(A, B) = \min_{a_i \in A, b_j \in B} (d(a_i, b_j)),$$

c'est-à-dire qu'on utilise le minimum de la distance entre chacun des points du groupe  $A$  et chacun des points du groupe  $B$ .

## Algorithmes hiérarchiques ascendants

- ▶ Lien complet (**complete linkage**): la proximité entre deux groupes ( $A$  et  $B$ ) est donnée par

$$\delta(A, B) = \max_{a_i \in A, b_j \in B} (d(a_i, b_j)),$$

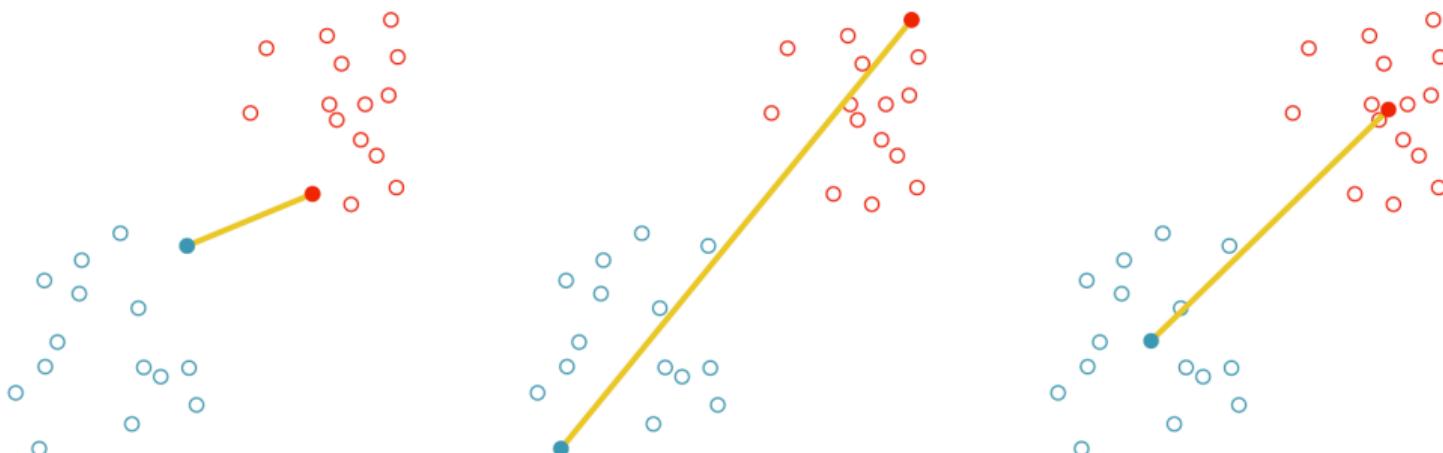
c'est-à-dire qu'on utilise le maximum de la distance entre chacun des points du groupe  $A$  et chacun des points du groupe  $B$ .

## Algorithmes hiérarchiques ascendants

- Lien moyen (**average linkage**): la proximité entre deux groupes ( $A$  et  $B$ ) est donnée par

$$\delta(A, B) = \frac{1}{\text{card}(A)\text{card}(B)} \sum_{a_i \in A} \sum_{b_j \in B} (d(a_i, b_j)).$$

Cette fois, on utilise la moyenne des distances entre chacun des points du groupe  $A$  et chacun des points du groupe  $B$ .



## Algorithmes hiérarchiques ascendants

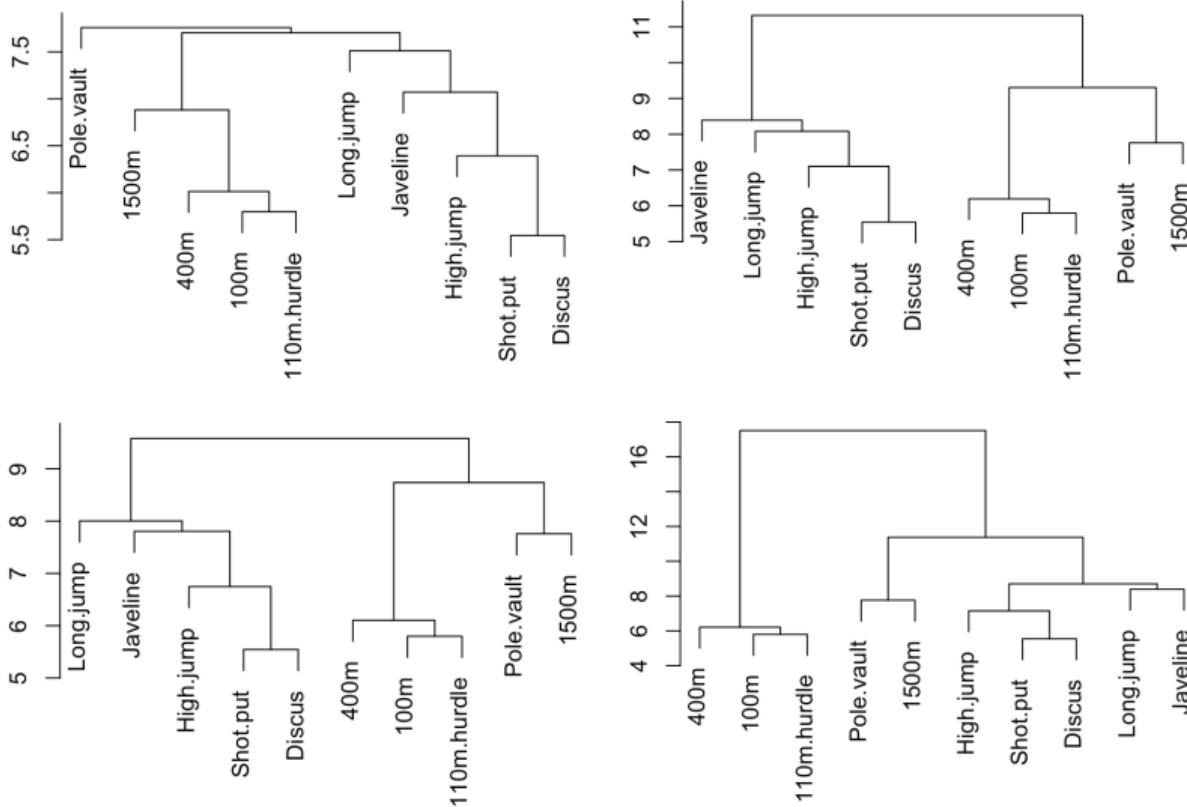
- ▶ Algorithme de **Ward**: on décompose l'inertie du nuage de points et minimise la perte d'information (perte d'inertie entre les groupes  $I_B$ ) à chaque étape: on optimise donc le même critère que pour un algorithme de classification par ré-allocation dynamique.
- ▶ À chacune des étapes, la perte d'information due au regroupement des groupes  $A$  et  $B$  est donnée par

$$\delta(A, B) = \frac{P_A P_B}{P_A + P_B} d^2(g_A, g_B).$$

Également, on définit la perte d'information due au regroupement des groupes  $C = A \cup B$  et  $D$  par

$$\delta(C, D) = \frac{(P_A + P_D)\delta(A, D) + (P_B + P_D)\delta(B, D) - P_D\delta(A, B)}{P_A + P_B + P_D}.$$

# Algorithmes hiérarchiques ascendants



# Algorithmes hiérarchiques ascendants

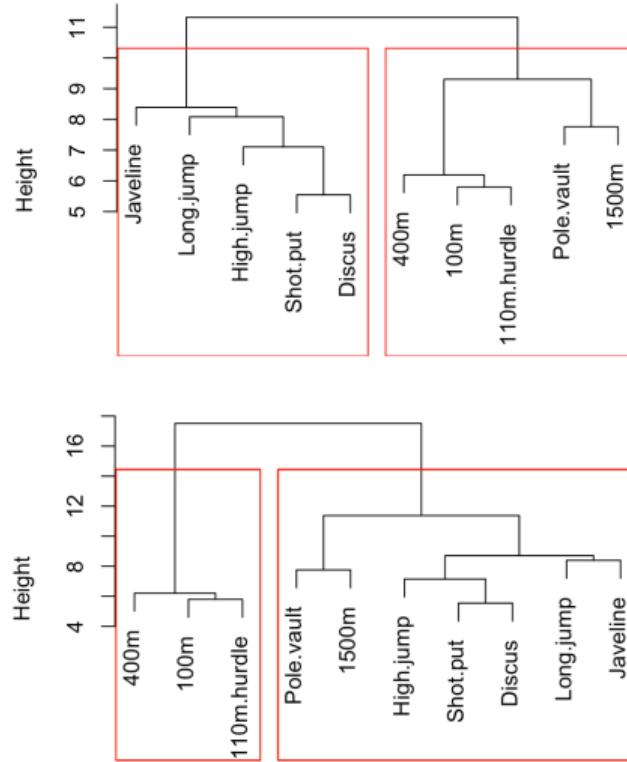
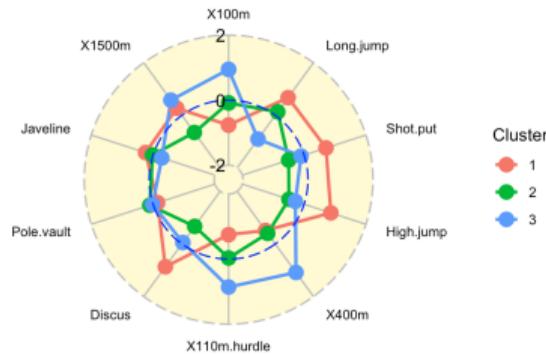
Ainsi, avant de pouvoir utiliser un algorithme hiérarchique ascendant, il faut répondre aux deux questions suivantes:

- ▶ Quelle règle de ressemblance, de proximité, entre les observations choisir? → **Choix de la mesure de distance**
- ▶ Comment définit-on la similitude entre deux groupes? → **Choix de la méthode d'aggrégation.**

# Decathlon

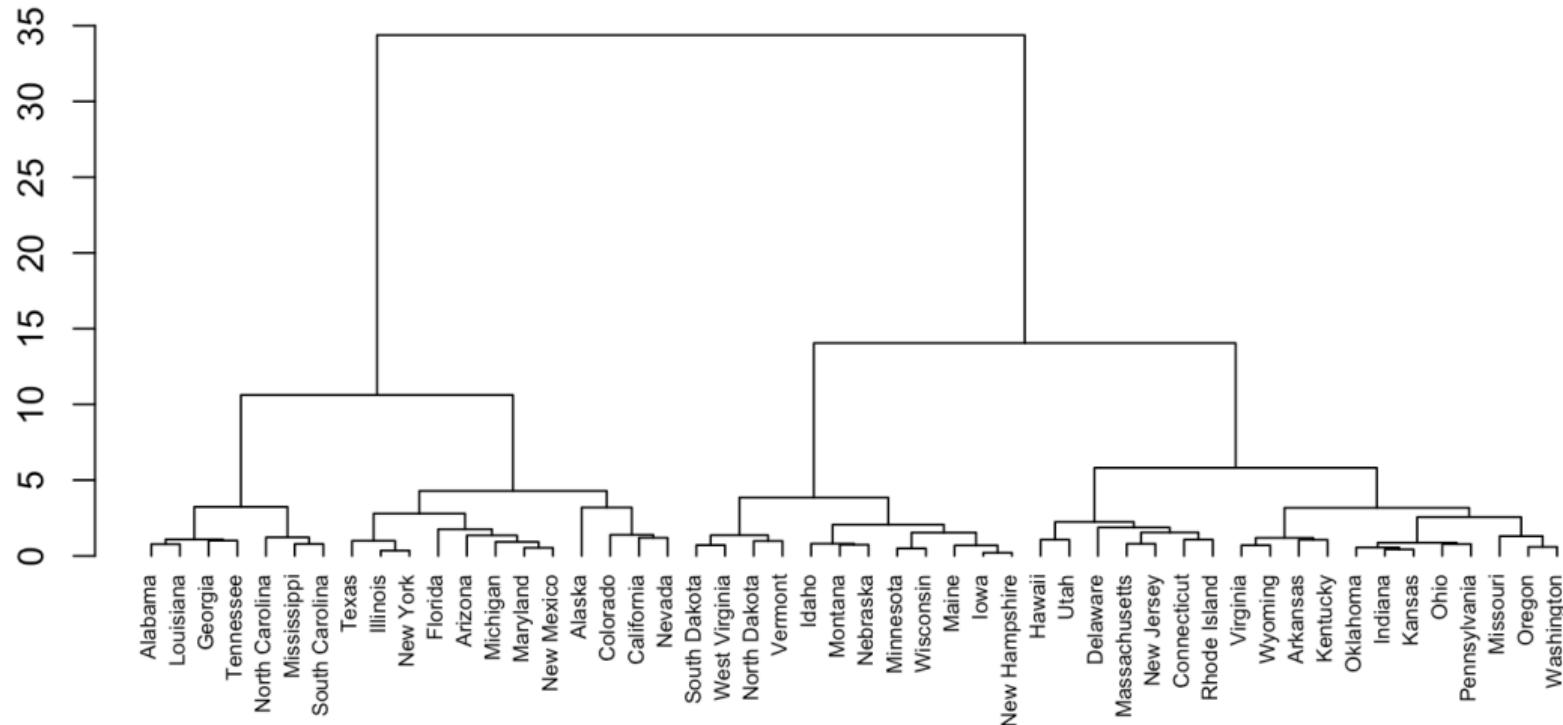
One can look at clusters of sports  
(Ward and complete)

or clusters of sportmen



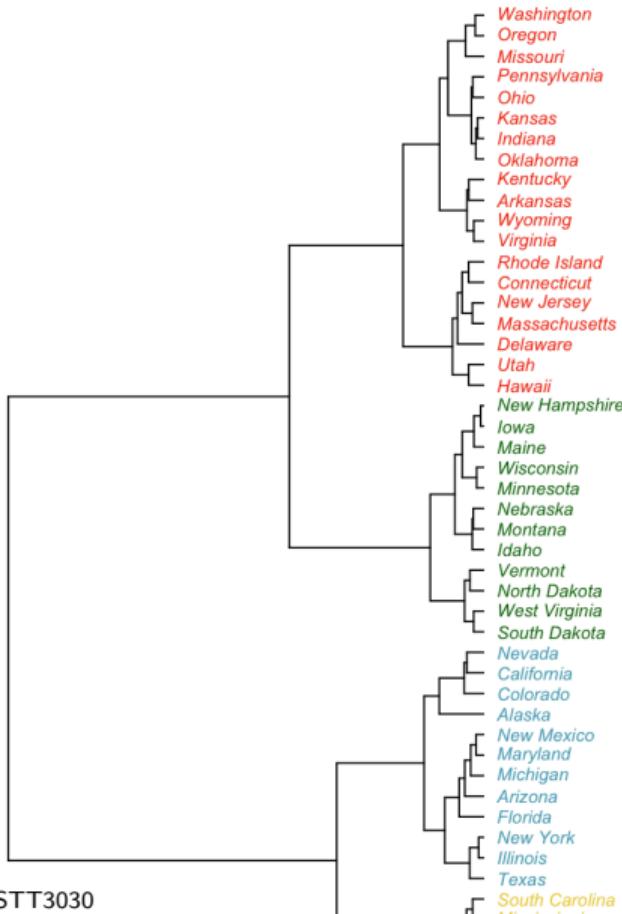
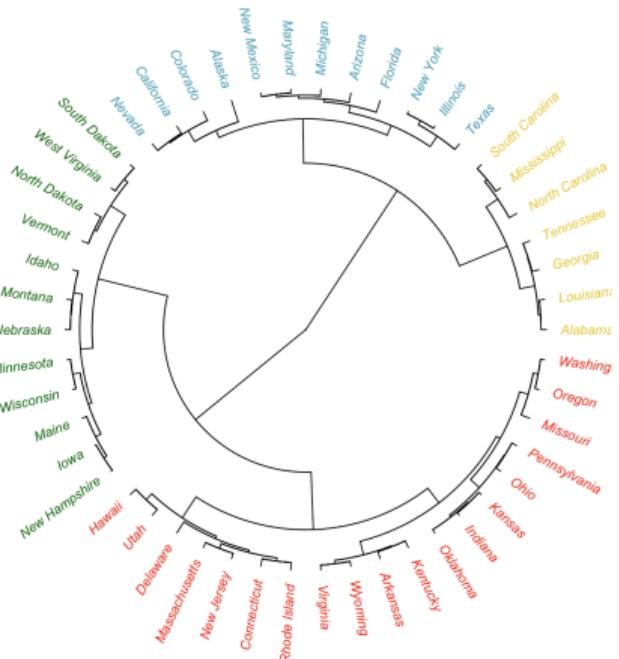
# Comparing US States

Using Ward's method,



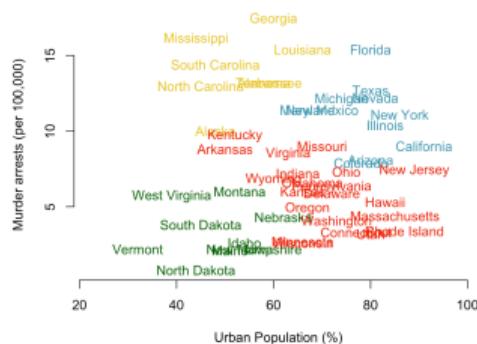
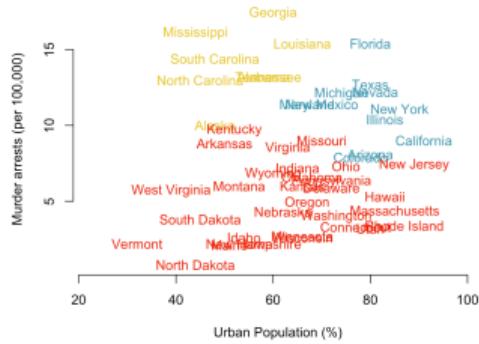
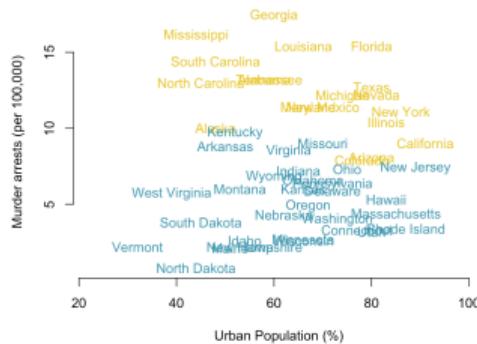
# Comparing US States

One can use a **phylogenetic tree**



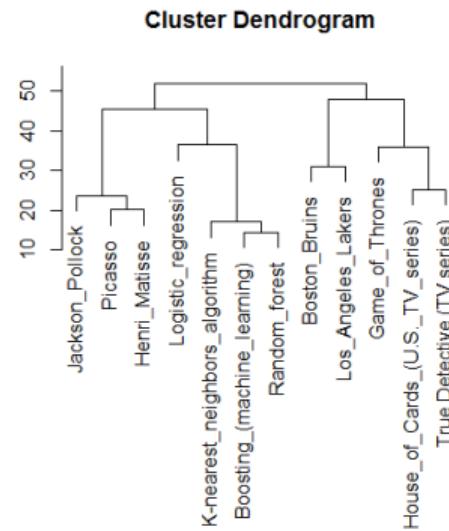
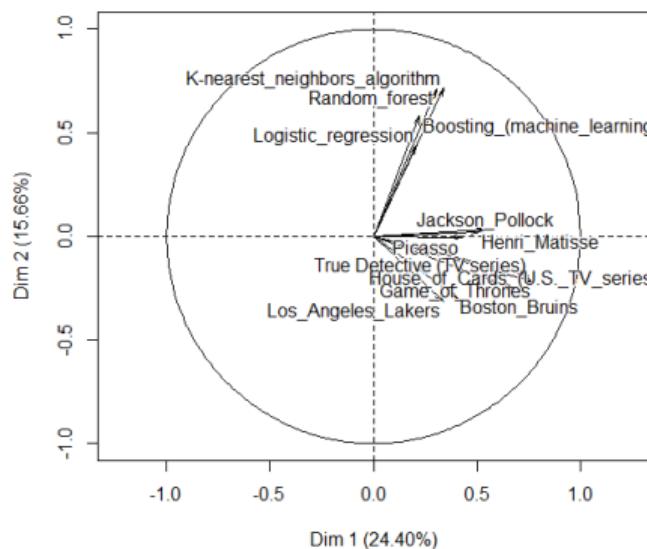
# Comparing US States

On the two dimensional representation (urban population, murder rate), when obtain, as **optimal** 2, 3 or 4 classes

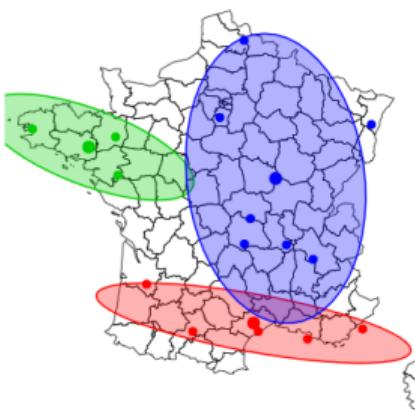
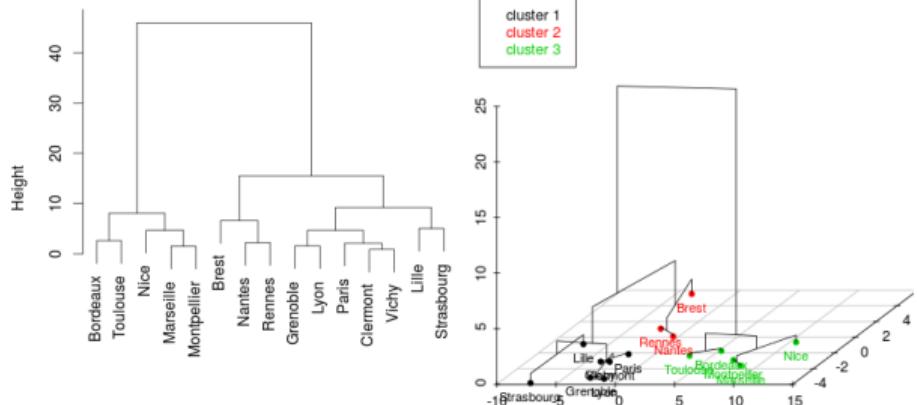
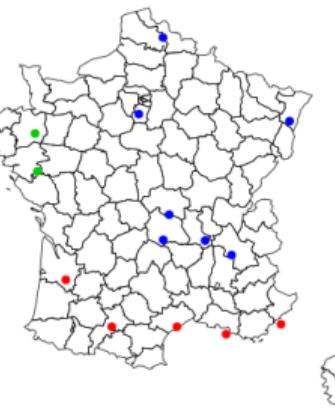
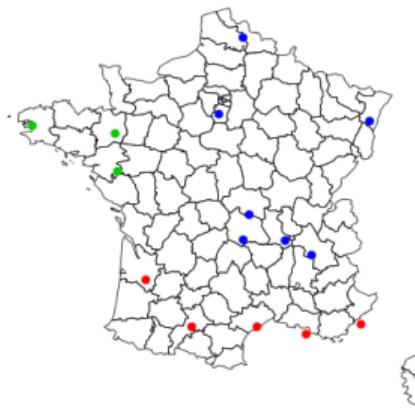
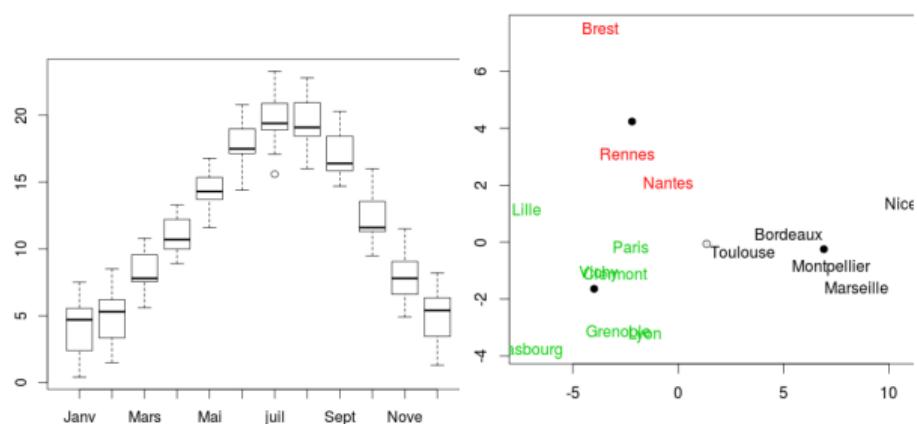


# Comparing Wikipedia Pages

Bag of works from some pages `Boosting_(machine_learning)`, `Random_forest`, `K-nearest_neighbors`, `Logistic_regression`, `Boston_Bruins`, `Los_Angeles_Lakers`, `Game_of_Thrones`, `House_of_Cards`, `True_Detective`, `Picasso`, `Henri_Matisse`, `Jackson_Pollock`.



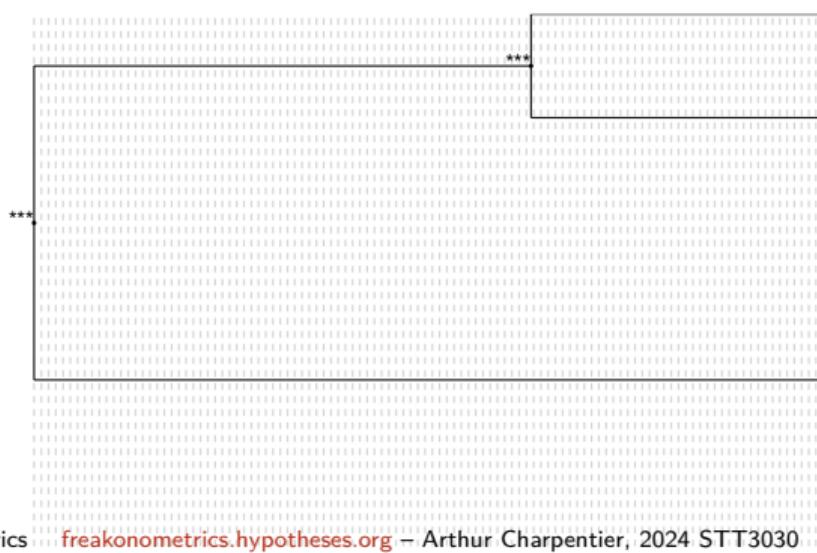
# Comparing 15 Cities Temperatures (in France)



# Clusters of Factors

```
1 > library(factorMerger)
2 > library(DALEX)
3 > data(Apartments)
4 > MF = mergeFactors(response = Apartments$m2.price,
5 + factor = Apartments$district, family = "gaussian")
6 > plot(MF)
```

Factor Merger Tree

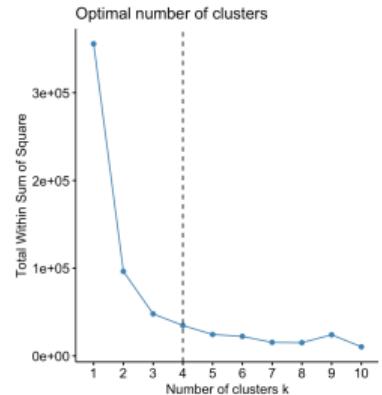


Group means  
with 95% confidence intervals

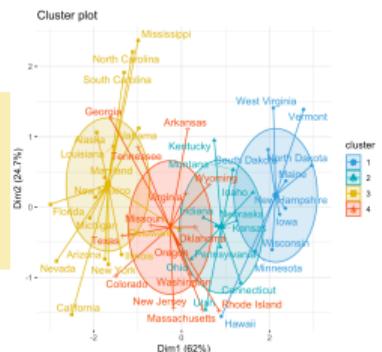


# Clusters with R

```
1 > library(factoextra)
2 > fviz_nbclust(USArrests, kmeans, method = "wss") +
3   geom_vline(xintercept = 4, linetype = 2)
```



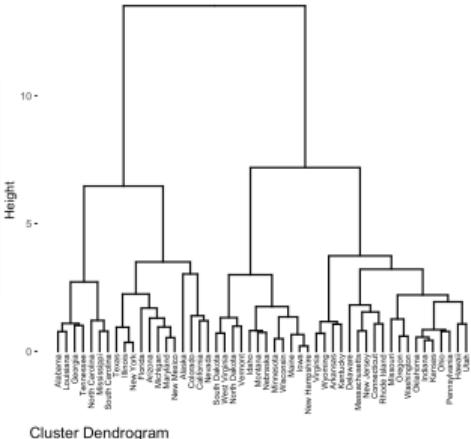
```
1 > km.res = kmeans(df, 4, nstart = 25)
2 > fviz_cluster(km.res, data = USArrests, ellipse.type = "euclid", star.plot = TRUE, repel = TRUE, ggtheme = theme_minimal())
```



# Clusters with R

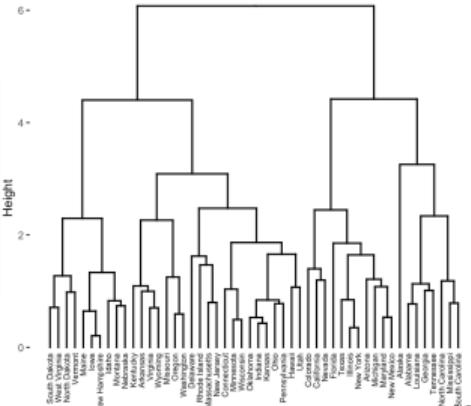
```
1 > df = scale(USArrests)
2 > res.dist = dist(df, method = "euclidean")
3 > res.hc = hclust(d = res.dist, method = "ward.D2")
4 > library("factoextra")
5 > fviz_dend(res.hc, cex = 0.5)
```

Cluster Dendrogram



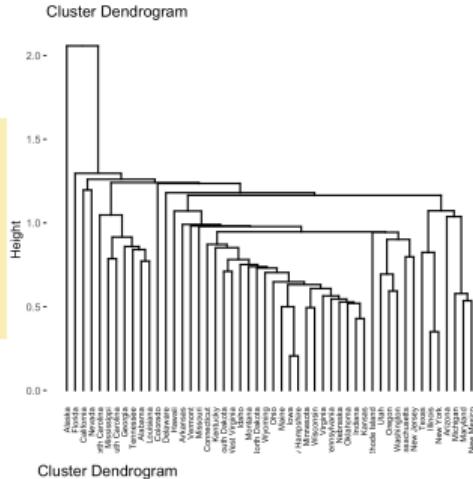
```
1 > df = scale(USArrests)
2 > res.dist = dist(df, method = "euclidean")
3 > res.hc = hclust(d = res.dist, method = "complete")
4 > library("factoextra")
5 > fviz_dend(res.hc, cex = 0.5)
```

Cluster Dendrogram

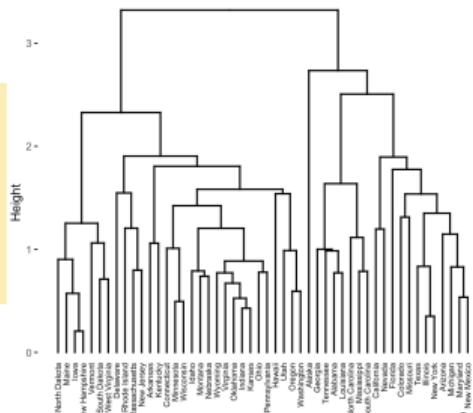


# Clusters with R

```
1 > df = scale(USArrests)
2 > res.dist = dist(df, method = "euclidean")
3 > res.hc = hclust(d = res.dist, method = "single")
4 > library("factoextra")
5 > fviz_dend(res.hc, cex = 0.5)
```

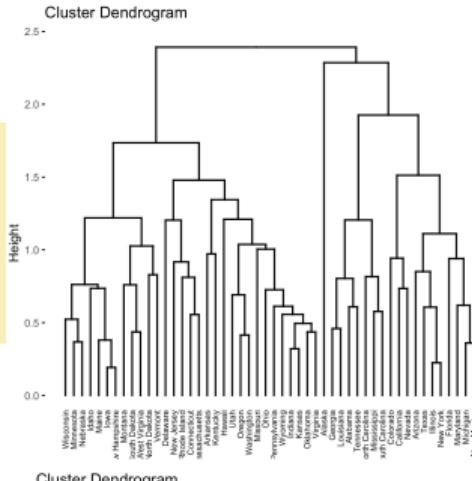


```
1 > df = scale(USArrests)
2 > res.dist = dist(df, method = "euclidean")
3 > res.hc = hclust(d = res.dist, method = "average")
4 > library("factoextra")
5 > fviz_dend(res.hc, cex = 0.5)
```

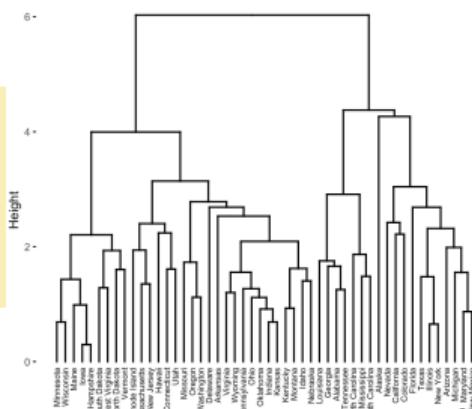


# Clusters with R

```
1 > df = scale(USArrests)
2 > res.dist = dist(df, method = "manhattan")
3 > res.hc = hclust(d = res.dist, method = "average")
4 > library("factoextra")
5 > fviz_dend(res.hc, cex = 0.5)
```

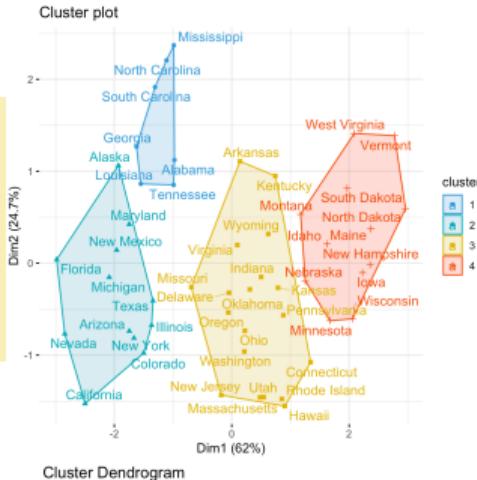


```
1 > df = scale(USArrests)
2 > res.dist = dist(df, method = "maximum")
3 > res.hc = hclust(d = res.dist, method = "average")
4 > library("factoextra")
5 > fviz_dend(res.hc, cex = 0.5)
```



# Clusters with R

```
1 > res.dist = dist(df, method = "euclidean")
2 > res.hc = hclust(d = res.dist, method = "ward.D2")
3 > grp = cutree(res.hc, k = 4)
4 > fviz_cluster(list(data = df, cluster = grp),
   ellipse.type = "convex", show.clust.cent = FALSE
, ggtheme = theme_minimal())
```

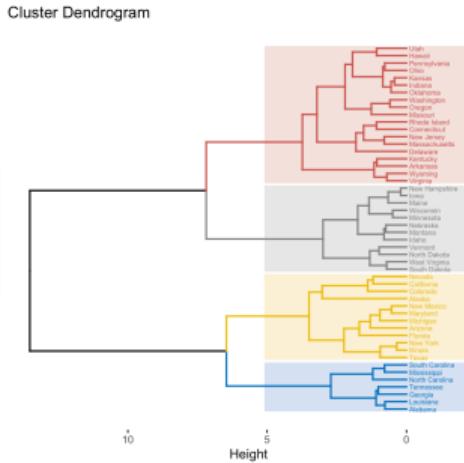


```
1 > fviz_dend(res.hc, k = 4, cex = 0.5,
  color_labels_by_k = TRUE, rect = TRUE, rect_fill
  = TRUE)
```

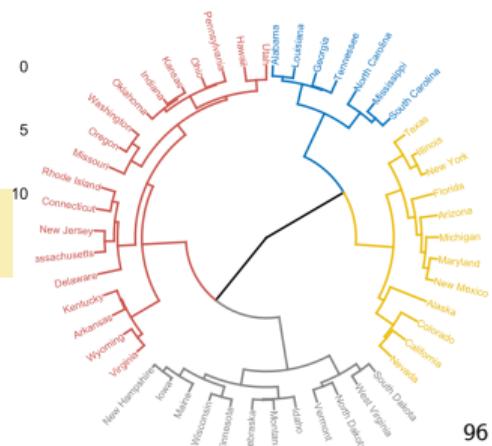


# Clusters with R

```
1 > fviz_dend(res.hc, k = 4, cex = 0.4, horiz = TRUE,  
   k_colors = "jco", rect = TRUE, rect_border = "  
   jco", rect_fill = TRUE)
```

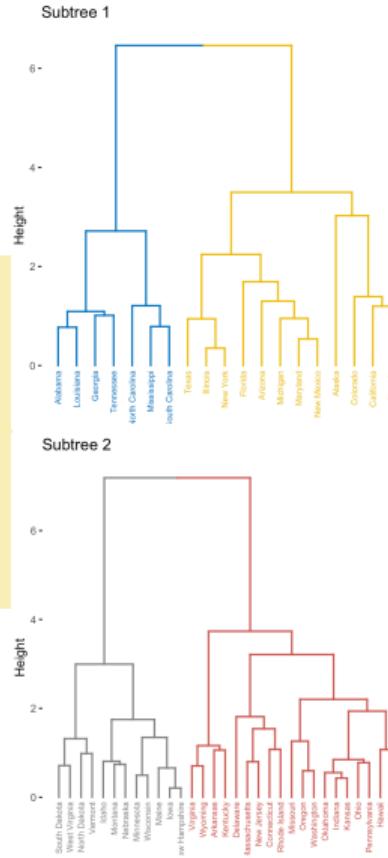


```
1 > fviz_dend(hc, cex = 0.5, k = 4, k_colors = "jco",
   type = "circular")
```



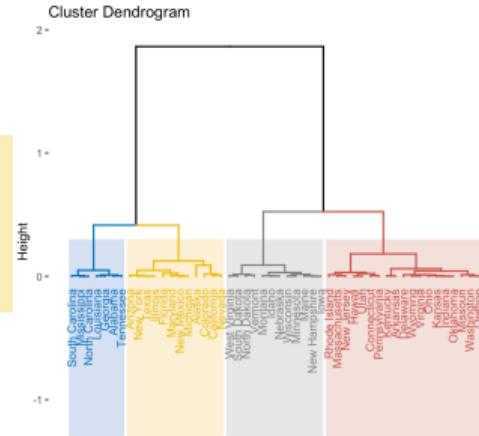
# Clusters with R

```
1 > dend_plot = fviz_dend(hc, k = 4, cex = 0.5,
   +   k_colors = "jco")
2 > dend_data = attr(dend_plot, "dendrogram")
3 > dend_cuts = cut(dend_data, h = 10)
4 > fviz_dend(dend_cuts$lower[[1]], main = "Subtree
   +   1")
5 > fviz_dend(dend_cuts$lower[[2]], main = "Subtree
   +   2")
```



# Clusters with R

```
1 > res.hcpc = HCPC(res.pca, graph = FALSE)
2 > fviz_dend(res.hcpc, cex = 0.7, palette = "jco",
  rect = TRUE, rect_fill = TRUE, rect_border = "
  jco", labels_track_height = 0.8)
```



## Pour le prochain cours:

- ▶ Lecture: 12.1, 12.4.1 et 9.1 ([Bishop and Nasrabadi \(2006\)](#))
- ▶ Préparation: 9.2 ([Bishop and Nasrabadi \(2006\)](#)), 9.3 ([Bishop and Nasrabadi \(2006\)](#))
- ▶ Exercices: 12.6.1,
- ▶ Lab : 12.5.3 (K-means)

## Références

- Bishop, C. M. and Nasrabadi, N. M. (2006). *Pattern recognition and machine learning*, volume 4. Springer.
- Husson, F., Lê, S., and Pagès, J. (2011). *Exploratory multivariate analysis by example using R*, volume 15. CRC press.
- James, G. (2013). *An introduction to statistical learning*. springer.
- Lloyd, S. (1982). Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137.