

# Rendu TP2

---

Oumaima Talouka, Zineb Slam

30 avril 2017

Dans ce TP2 nous allons travailler sur la méthode de classification automatique qui est une méthode non supervisée, c'est à dire on a aucune connaissance a priori ni sur le nombre ni sur le nom des classes qui peuvent exister. Nous utiliserons en majorité la méthode des **kmeans** dont nous analyserons et critiquerons les résultats

## 1 Exercice 1 : Visualisation des données

Nous disposons de 3 jeux de données : **Iris**, **Crabs** et une jeu de données **Mutations** de dissimilarites des espèces

### 1.1 Iris

Les iris est un jeu de donnes de la librairie *MASS* avec **150 individus** et 4 variables (Sepal.Length, Sepal.Width, Petal.Length et Petal.Width). La variable Z de réponse est l'espèce. Le but est d'identifier l'espèce de chaque individu en fonction des 4 variables : . Pour commencer nous allons centrer et réduire les données puis effectuer une ACP pour pouvoir réduire la dimension du dataset et passer de 4 dimensions et 2 dimensions. Après avoir fait appel a la fonction *pr-comp* nous affichons les données dans le premier plan factoriel sans tenir compte de l'espèce. Nous notons d'abord que les 2 premiers plan factoriels expliquent **95.8%** des données (*inertie expliquée*). Ensuite nous utilisons la fonction *autoplot* pour colorer chaque espèce selon son appartenance a une espèce et ce dans le même plan factoriel.

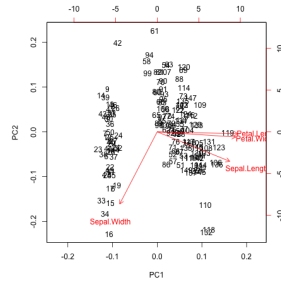


FIGURE 1 – Représentation des données Iris dans les 2 premiers plan factoriel

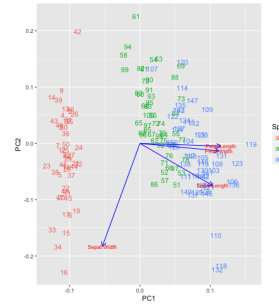


FIGURE 2 – Représentation des données Iris dans les 2 premiers plan factoriel en colorant les espèces

Nous observons donc que selon le graphe qu'il existe au moins 2 classes bien distinctes. Maintenant nous allons faire appel à la fonction *autoplot* en colorant chaque individu selon son espèce. On constate qu'en fait il existe 3 classes : **setosa**, **versicolor** et **virginica**. Dans la première partie nous avons confondu les classes versicolor et virginica qui sont très proches et donc indistinguables à l'œil nu. L'appartenance à la classe setosa est majoritairement expliquée par la variable Sepal.Width, tandis que les 2 autres classes par les 3 autres. Enfin Petal.Length et Petal.Width sont deux variables fortement corrélées. On devrait donc s'attendre à 3 classes pour ce jeu de données.

### 1.1.1 Conclusion

## 1.2 Crabs

Nous allons procéder de la même manière que dans la question précédente mais cette fois pour les données Crabs2. Le jeu de données Crabs2 compte 200 individus et 4 variables (**FL2**, **RW2**, **CL2**, **BD2**) et deux variables de réponses le sexe et l'espèce. Nous utilisons la fonction *interaction* pour merger le sexe et l'espèce et obtenir une seule variable de réponse z. **91.2%** des données sont expliquées par les 2 premières composantes principales. Les 2 figures ci-dessous représentent les données Crabs dans les 2 premiers plans factoriels sans distinguer les espèces (à gauche) puis en les colorant (à droite).

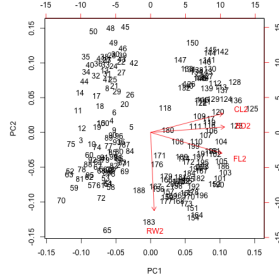


FIGURE 3 – Représentation des données Crabs2 dans les 2 premiers plan factoriel

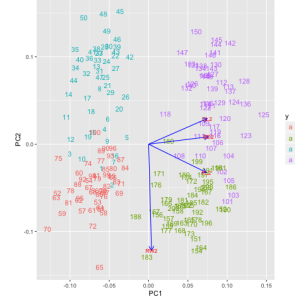


FIGURE 4 – Représentation des données Crabs2 dans les 2 premiers plan factoriel en colorant les espèces

Selon la figure 3 on peut observer qu'il existe au moins 2 classes celle de gauche et celle de droite voir 4 classes si on distingue 2 dans celle de droite et 2 dans celle de gauche. Néanmoins on peut remarquer que ces classes ne sont pas très séparées. La figure 4 montre qu'il existe bien 4 classes B.F, O.F, B.M et O.M. (B et O pour distinguer les 2 espèces alors que , M et F pour le sexe) . La deuxième composante PC2 qui est concentrée dans la variable RW2 permet de distinguer le sexe, tandis que CL2, FL2, BO2 encodées dans PC1 permettent de distinguer l'espèce. On pourrait alors mettre l'hypothèse que le fait que le sexe ne soit pas aussi distinguable que l'espèce est le fait qu'il est expliqué par une seule variable tandis que l'espèce par 3.

### 1.2.1 Conclusion

Pour conclure la l'AFTD nous a permis de passer d'une représentation de dissimilarités de dimension 20 à une autre de dimension 6 avec un pourcentage d'inertie expliquée de 95%. Nous n'avons pas pu choisir 2 dimensions car la représentation n'est pas fiable comme les dissimilarités calculées sont loines des initiales. On peut supposer que le fait que cette méthode n'a pas été aussi performante avec ce jeu de données est parce que nos dissimilarités ne sont pas des distances Euclidiennes mais les différences entre les acides aminés dans les chaînes chromosomiques. chaque différence contribue à la distance de mutation en fonction du nombre minimum de nucléotides qui devrait être changé pour convertir l'un dans l'autre. Fitch & Margoliash ont utilisé ces données pour construire un arbre phylogénétique.

## 1.3 Mutations

Nous disposons dans cette étude d'une matrice de dissimilarités entre 20 individus (Homme, Singe, Kangourou, Cheval...). Grâce à la méthode **AFTD** et la fonction nous allons essayer de réduire la dimension du jeu de données. Nous essaierons de trouver au fur et à mesure le bon nombre de variables à choisir. Nous allons alors commencer par réduire le nombre de dimensions à 2. Pour

évaluer cette représentations nous utiliserons le pourcentage d'inertie expliquée cumulée et le Diagramme de Shepard Notons que dans le Diagramme de Shepard un axe représente la disimilaritee calculée par AFTD alors que l'autre les disimmilarites initiales a partir de la matrice 20x20. Ainsi si la disimilaritee calculée par l'AFTD est exacte le Diagramme de Shepard sera une fonction  $y=x$ .

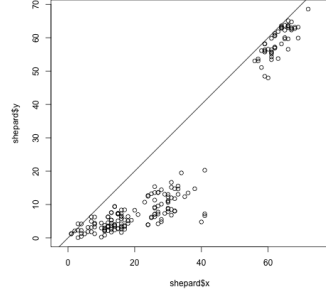


FIGURE 5 – Diagramme de Shepard des Mutations avec 2 variables

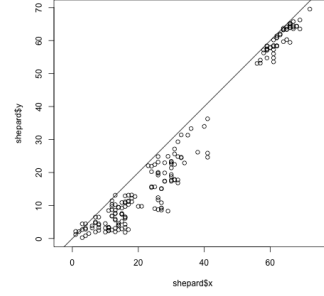


FIGURE 6 – Diagramme de Shepard des Mutations avec 3 variables

On remarque d'après les 2 premières présentations ci dessous que le choix de 2 variables n'est pas très représentatif de nos donnes, en effet les distances calculées par représentations en 2 dimensions restent relativement que celle de 20. Avec 3 variables il y'a plus de données les disimilarites calculées se rapprochent de leur valeur exacte mais il reste toujours d'autres points éloignés Dans la partie suivante nous allons améliorer cette représentation avec 4 puis 5 variables.s

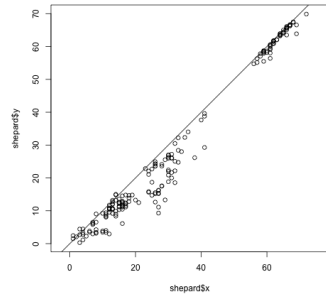


FIGURE 7 – Diagramme de Shepard des Mutations avec 4 variables

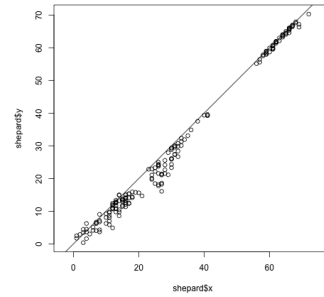


FIGURE 8 – Diagramme de Shepard des Mutations avec 5 variables

Nous remarquons que la représentation avec 4 variables le calcul de certaines disimilarites est exacte mais certaines restent loines des valeurs initiales. Avec 5 variables on se rapproche plus du cas initial ou toutes les disimilarites calculées sont plus proche de la droite  $y=x$ . Ces résultats étaient prévisibles si on calcule le pourcentage d'inertie expliquée cumule en mettant l'option  $eig = TRUE$  dans la fonction *cmscale* et récupérer les valeurs propres.

### 1.3.1 Conclusion

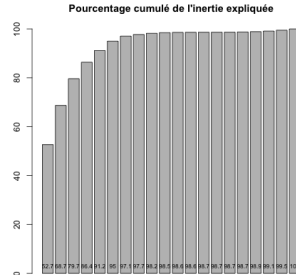


FIGURE 9 – Pourcentage d’inertie cumulee expliquee par AFTD

On remarque bien qu’avec 2 variables seulement 68% et a partir de 4 variables le pourcentage est supérieur a 91%.

## 2 Exercice 2

## 3 Exercice 3 : Méthodes des centres mobiles

Le but de cet exercice est de manipuler la méthode des kmeans sur les trois jeux de données Iris2, Crabs2 et mutations et analyser la qualité de la méthode et ses limites. On utilisera l’index de Rand ajustée ainsi que les représentations graphiques pour juger des performances des kmeans.

### 3.1 Iris2

#### 3.1.1 Question 1

Nous commençons par classifier nos méthodes en 2, 3 puis 4 classes et les représentant dans en fonction des 2 variables **Petal.Length** et **Sepal.Width** car comme on a vu dans lq première partie ces dernières sont bien expliquées par les 2 premières composantes principales de l’ACP. Les résultats sont représentés ci-dessous :

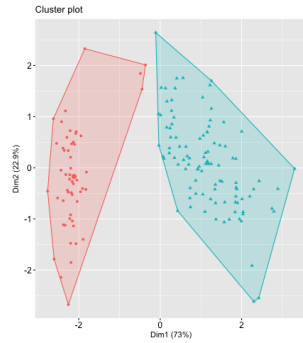


FIGURE 10 – Représentation du jeu de données de Iris en 2 classes



FIGURE 11 – Représentation du jeu de données de Iris en 3 classes

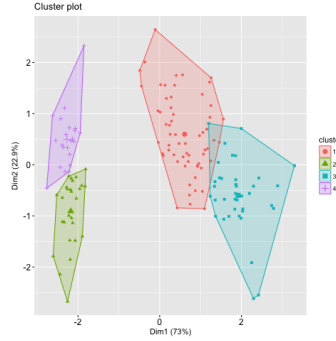


FIGURE 12 – Représentation du jeu de données de Iris en 4 classes

On remarque qu'avec  $k=2$  classes nous obtenons des individus qui appartiennent à la classe 1 alors qu'ils sont plus proches de la classe 2. Tandis qu'avec  $k=3$  on a des individus qui appartiennent à une classe mais qu'ils sont très éloignés. La représentation avec  $k=4$  classes semble plus correcte de plus avec une connaissance a priori des données nous savons que c'est le nombre de classes correct.

### 3.1.2 Question 2

À présent nous avons effectué plusieurs opérations de kmeans avec  $k=3$ , et on remarque qu'à chaque opération nous obtenons des résultats différents. En effet, ceci est normal car à la première itération de l'algorithme, celui-ci choisit au hasard 3 points ( $k=3$ ) appartenant aux individus et va construire à partir de ces points les 3 classes. Pour toujours obtenir le même résultat, on peut faire un appel au préalable à la fonction `set.seed(10)` sur R.

### 3.1.3 Question 3

Dans cette question nous cherchons a déterminer le nombre de classes minimum dans le jeu de données Iris2. Pour cela on utilise la méthode coude. Tout d'abord on effectue 100 opérations kmeans pour chaque  $k = 2, 3 \dots 10$ , on calcule l'inertie intra-classe pour  $k$  et choisissant l'inertie minimum pour représenter l'inertie de chaque  $k$ , ce qui s'écrit comme  $\hat{I}_k = \min_{i=1..100} I_{k_i}$ . Pour cela on utilise une matrice  $10 \times 100$  ou on sauvegarde l'inertie a chaque opération puis on récupère l'inertie intra-classe minimale de chaque ligne. Le code en R détaillé et commente de cet algorithme est a trouver en annexe. La méthode du coude consiste a choisir le point d'inflexion sur la courbe tel que l'inertie intra-classe ne diminue pas de manière aussi rapide en rajoutant des clusters. Comme il est représenté dans la figure 13 et comme on s'attendu aussi, le nombre de classes minimum du jeu de données Iris2 est 3.

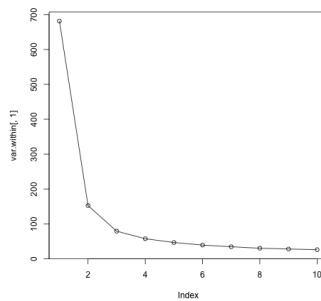


FIGURE 13

### 3.1.4 Question 4 : Analyse des résultats pour $k=3$

On commence par afficher la matrice de confusion Pour analyser la qualité de classification de la méthode des kmeans nous utilisons l'index de Rand ajusté et la représentation graphique. L'index de Rand est calculé en R grâce a la fonction *adjustedRandIndex*. L'index de Rand est compris entre 0 et 1, plus l'index est proche de 1, plus la classification de la méthode correspond a la classification initiale. Nous obtenons un **index = 0.73** ce qui montre que notre classification se rapproche de la classification de la colonne espèces.

Pour mieux analyser les points qui sont mal classés nous avons fait une interaction entre la colonne espèce qui représente la vrai classification et la colonne `iris.kmeans$cluster` puis nous les avons représenté graphiquement.

Espece	setosa	versicolor	virginica
1	50	0	0
2	0	2	36
3	0	48	14

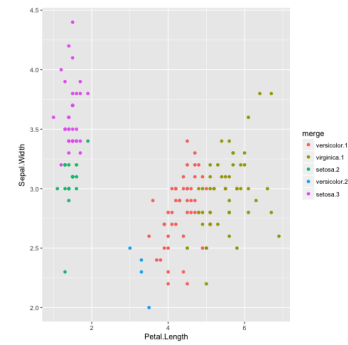


FIGURE 14 – Comparaison de la classification originale et celle des kmeans

### 3.1.5 Conclusion