Classification automatique de données temporelles en classes ordonnées

MOHAMED BEN HAMDOUNE[1] et LUCAS ISCOVICI[1]

Université Paris-Descartes, 12 Rue de l'École de Médecine, 75006 Paris, France,
mohamed.ben_hamdoune@etu.parisdescartes.fr
lucas.iscovici@etu.parisdescartes.fr
https://www.univ-paris5.fr/

Résumé La classification automatique ou *clustering* consiste à partionner un ensemble d'objets (instances) décrit par un ensemble de variables en groupes (classes) homogènes. Il en ressort que pour certaines données, ce sont une suite de valeurs numériques représentant l'évolution d'une quantité spécifique au cours du temps, d'où la notion de série temporelle. Le but sera de proposer un systéme de classification provenant de divers jeux de données en utilisant l'algorithme dynamique de Fisher et de constater son efficacité en la comparant à d'autres méthodes.

Keywords : Apprentissage Non Supervisé, CAH, K-means, R, Programmation dynamique, Algorithme de Fisher, Séries temporelles.

Table des matières

1 Introduction

1.1 Contexte et motivations

Dans le cadre de notre cursus scolaire, nous sommes amenés à réaliser un projet concernant l'apprentissage non supervisé. L'objectif de ce projet est d'étudier une méthode de clustering qui recherche des classes ordonnées suivant une chronologie. **La méthode de programmation dynamique de Fisher** est utilisée pour la segmentation de signaux temporels, ou pour la détection de changement dans une séquence de données.
Dans les systèmes de reconnaissance de la parole par exemple, on exploite ce type de méthode pour partitionner des signaux audio en plages temporelles associées à des locuteurs différents qui sont ensuite identifiés. De manière plus générale, l'organisation d'une **séquence de données en segments homogènes** est un processus qui aide à mieux les organiser.

1.2 Contribution et organisation du rapport

Dans ce projet, nous avons commencé par l'implémentation de l'algorithme de programmation dynamique de Fisher dans le logiciel de statistique **R**[1]. Ensuite nous appliquons l'algorithme à des données simulées joint avec le rapport. Il sera ici l'objet de déterminer le nombre de segment (le nombre de classes) à l'aide de l'algorithme de Fisher. Nous comparerons ces résultats avec d'autres algorithmes de clustering comme K-means et la méthode CAH en utilisant le critère de Ward.

2 Implémentation de l'algorithme de Fisher

2.1 Séries temporelles

De nos jours, de nombreuses applications réelles [2] génèrent et stockent des données de séries chronologiques.
Cependant, comme il ressort des précédentes méthodes de regroupement générales, telles que K-means et autres, ne sont pas conçues pour les données de séries chronologiques et donc peut ne pas bien fonctionner. Ceci est principalement dû au fait que la plupart des regroupements méthodes sont construites autour de la distance Euclidienne (ou bien la distance de Minkowski), ce qui ne semble pas être une bonne mesure pour les données de séries chronologiques.
Le clustering peut être considéré comme le problème d'apprentissage non supervisé le plus important. Ainsi, comme pour tout autre problème de ce type, il s'agit de trouver une structure dans une collection de données non étiquetées.
Un cluster est donc une collection d'objets qui sont **similaires** entre eux et **dissemblables** aux objets appartenant à d'autres clusters.

2.2 Programmation dynamique

La programmation dynamique est une méthode permettant de résoudre un problème complexe en le décomposant en un ensemble de sous-problèmes plus simples [3], en résolvant chacun de ces sous-problèmes une seule fois et en stockant leurs solutions à l'aide d'une structure de données basée sur la mémoire (tableau, carte, etc.).
Chacune des solutions du sous-problème est indexée dune manière ou dune autre, généralement en fonction des valeurs de ses paramètres dentrée, afin de faciliter sa recherche. Ainsi, la prochaine fois que le même sous-problème se produit, au lieu de recalculer sa solution, on se contente de consulter la solution précédemment calculée, ce qui permet de gagner du temps de calcul.
Cette technique consistant à stocker des solutions aux sous-problèmes au lieu de les recalculer est appelée mémorisation.
Dans le cadre du projet, notre pré-calcul se fait lors que l'on crée le tableau contenant le **diamètre des classes** comme nous allons le voir dans la prochaine section.

2.2.1 Mémorisation de la matrice des diamètres des classes

Nous créons une matrice D appelé souvent **matrice des diamètres des classes**. Nous utilisons le package *matrixStats* très utile pour les jeux de données contenant beaucoup de variables puisque nous avons eu un très grand gain de vitesse.

```
diam <- function(donnees){
  D <- matrix(data = 0, nrow = nrow(donnees), ncol = nrow(
    donnees))
  for (a in 1:(n-1)){
    for (b in (a+1):n){
      D[a,b] <- var(donnees[a:b,])*(b-a+1)
    }
  }
  D
}
```

Nous avons rencontrer des problèmes pour le calcul du diamètre pour le jeu de données Aiguillage, et nous avons implémenter une version multidimensionnelle permettant d'avoir des résultats en un temps raisonnable. (voir benchmarks).

```
diamND <- function(donnees){
    D <- matrix(0,nrow(donnees),nrow(donnees))
    foreach(a=1:(n-1)) %do% {
        foreach(b=(a+1):n) %do% {
          D[a,b]= sum(matrixStats::colVars(donnees[a:b,])
        }
    }
    D
}
```

3 Clustering

Le partitionnement de données (ou clustering) est une des méthodes d'analyse des données qui vise à diviser un ensemble de données en différents **groupes homogènes**, c'est-à-dire que les données de chaque sous-ensemble partagent des caractéristiques communes, qui correspondent le plus souvent à des critères de proximité.

3.1 K-means

K-means est une méthode de partitionnement de données qui, à partir d'un ensemble de points, va pouvoir déterminer pour un nombre de classes fixé une répartition des points qui minimise un critère appelé *inertie* ou variance *intra-classe*.
Plus formellement étant donnée k, nous allons donc chercher à répartir les points $x_1, x_2, .., x_n$ en k groupes $C_1, C_2, , C_K$ de telle sorte que :

$$\sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{x \in C_k} ||x - (\frac{1}{|C_k|} \sum_{x \in C_k} x)||^2$$

soit le plus minimale.

3.2 Classification Ascendante Hiérarchique

La classification hiérarchique ascendante est un algorithme qui regroupe des objets similaires en groupes appelés clusters. Le noeud final est un ensemble de clusters dans lequel chaque cluster est distinct, et les objets de chaque cluster sont globalement similaires.
La classification hiérarchique peut être réalisée avec une matrice de distance ou des données brutes. Lorsque des données brutes sont fournies, le logiciel (dans notre cas le langage R) calcule automatiquement une matrice de distance en arrière-plan. La classification hiérarchique commence par traiter chaque observation comme une grappe distincte. Ensuite, il exécute à plusieurs reprises les deux étapes suivantes :

1. Identifier les deux clusters les plus proches l'un de l'autre.
2. Fusion des deux clusters les plus similaires.

Cela continue jusqu'à ce que tous les clusters soient fusionnés.
Comme pour les métriques de distance, le choix des critères de couplage doit être effectué sur la base de considérations théoriques du domaine dapplication.
La méthode de Ward consiste à regrouper les classes de façon que l'augmentation de l'inertie intraclasse soit minimum.
Une question théorique clé concerne les causes de la variation.
En l'absence de justifications théoriques claires pour le choix des critères de

couplage, la méthode de Ward est la méthode par défaut raisonnable. Cette méthode calcule les observations à regrouper en réduisant la somme des distances au carré de chaque observation par rapport à la moyenne des observations d'un groupe.

4 Jeux de données

4.1 Simulation

Le jeux de données Simulation [4] est un vecteur de données possédant 210 valeurs distinctes. La valeur minimum est de -2.6972 alors que son maximum est de 4.4076 . Pour un écart type assez petit puisqu'il est de seulement 1.3965. Voici un graphique ci-dessous :

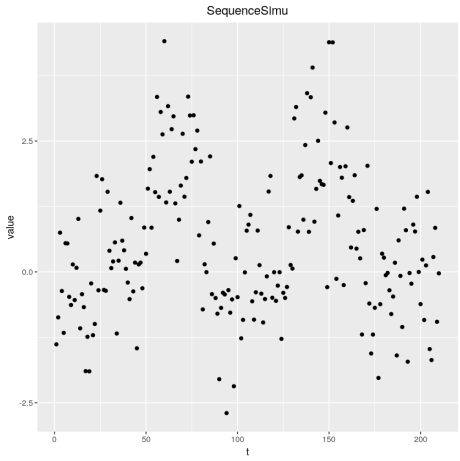


Figure 1. Nuage de points des données Simulation.

4.2 Aiguillage

Ce jeu de données contient 140 entrées décrit par 552 variables, la dernière colonne colonnes correspond à la classe. C'est un jeu de données labelisé et la répartition des classes au sein du jeu de données n'est pas uniforme.

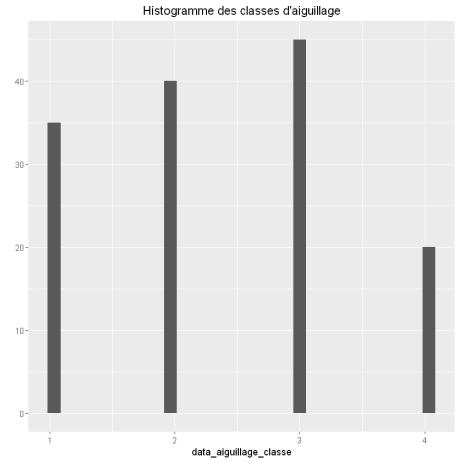


Figure 2. Histogramme des classes d'aiguillage.

Pour le calcul des diamètres de classes, pour le jeux de données aiguillage contenant plus de 552 variables nous avons pu constater un temps de calcul exponentielle. Voici un graphique d'un benchmark effectuer entre la fonction **var** de base sur R et **matrixStats** :

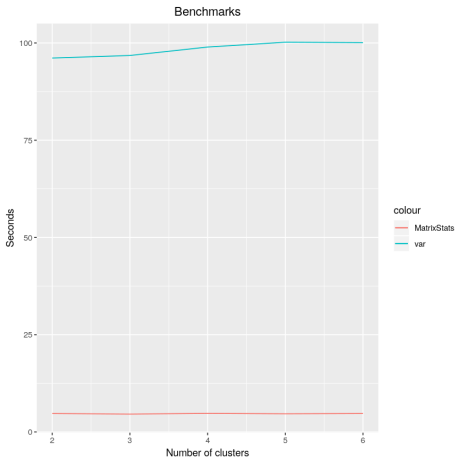


Figure 3. Différence de compléxité temporelle.

Le gain est énorme, puisqu'en utilisant la librairie **matrixStats**, nous avons une rapidité multiplier par 20.

5 Étude comparative

5.1 Déterminer le nombre de cluster *k*

Déterminer le bon nombre de cluster est une tâche qui peut s'avérer très compliquée. Pour cela nous avons choisi 2 critères afin d'évaluer la qualité du partitionnement : la méthode **du Coude** et le coefficient de **Silhouette** (Des packages comme *"facto extra"* et *"Nbclust"* facilitent la possibilité d'effectuer un graphique sur les méthodes permettant d'avoir une idée sur le nombre de classe avec R .

- La méthode du Coude : L'idée derrière cette méthode est d'exécuter K-Means sur l'ensemble des données pour un nombre de cluster de valeurs k (par exemple $k = 1$ à 10) et de calculer pour chaque rang k la somme des erreurs au carré (SSE). Ainsi on peut choisir un nombre de cluster k de telle sorte que l'ajout d'un autre cluster ne donne pas une meilleure modélisation des données.
Cette méthode s'appelle aussi la méthode du *"coude"* car si on affiche les informations sur un graphe, nous pouvons choisir le rang k au niveau où la courbe forme un *"coude"*.

- Le coefficient de Silhouette : Cette méthode est une mesure de la similitude entre un objet et son propre cluster (cohésion) par rapport aux autres clusters (séparations). Autrement dit, le coefficient de silhouette $s(x)$ permet d'évaluer, pour un point x donné, si ce point appartient au "bon" cluster : est-il proche des points du cluster auquel il appartient ? Est-il loin des autres points ?
La valeur de la silhouette est compris entre -1 et 1 où plus la valeur est proche de 1 et plus l'assignation de x à son cluster est satisfaisante.

6 Résultats

Nous allons commenter dans cette section les différents résultats obetenus en utilisant les 3 méthodes (Fisher, K-means, CAH). Notre critère principale sera la méthode du Coude pour déterminer la nombre de cluster, ensuite nous afficherons le plot des nuages et voir ainsi les intersections entre les différentes classes. Nous aurions pu labelisé le jeux de données Simulation comme nous allons le voir plus tard mais nous avons décider l'utilisation la matrice de confusion uniquement pour le jeux de données concernant l'aiguillage.

6.1 Simulation

Nous commençons par regarder le graphique en utilisant l'implémentation de l'algorithme de Fisher :

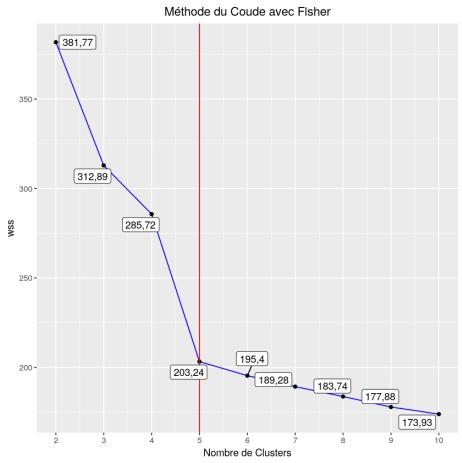


Figure 4. Différences de la variance pour chaque cluster (données Simulation)

En utilisant la méthode du coude, il en ressort que le nombre de clusters est à 5. La forme de décroissance ne laisse place à aucune doute alors que pour Kmeans et la CAH, ils existent des nuances. Ce que nous voyons auparavant grâce aux nuage de points ce confirme, nous avons bien 5 classes homogènes [5] distinctes :

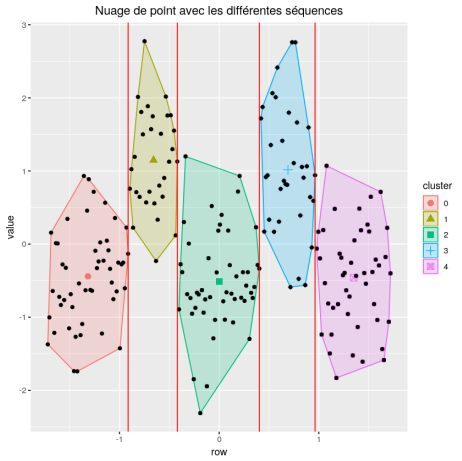


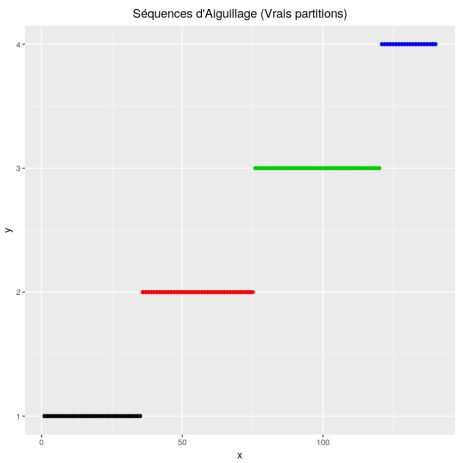
Figure 5. Nuage des points avec son cluster (données Simulation)

Même si nous n'avons pas la classe de chaque entrée, le nuage de point ne semblent grouper les points sans prendre en compte la chronologie. Pour la suite

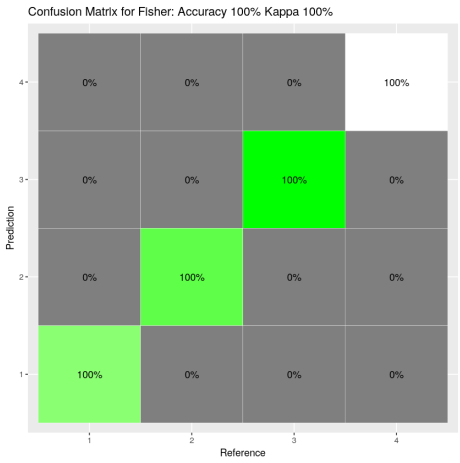
nous avons effectué une comparaison avec K-Means et CAH (Ward), ces deux méthodes (en annexe) nous ont indiqués 4 (voir 3) clusters ainsi que la même forme (approximativement) de groupement. Ces résultats sont sans surprise puisque K-Means et CAH ne tiennent pas en compte la notion de chronologie. Les résultats s'explique puisque la classification hiérarchique doit être utiliser lorsque les données sous-jacentes ont une structure hiérarchique (comme les corrélations sur les marchés financiers) et que nous souhaitons récupérer la hiérarchie or ce n'est pas le cas ici.

6.2 Aiguillage

Notre jeux de données est labalisé, ce qui nous permettra de calculer la précision des différentes méthodes que nous utiliserons, voici le graphique des partitions :



Maintenant nous appliquons les 3 méthodes et comparerons leur précisions et rappels respectifs :



Avec l'algortihme de Fisher, les 4 séquences sont parfaitement bien classer et la CAH possèdent tout de même 94.3% et un rappel de 92.2% alors que Kmeans n'arrivent pas à faire mieux que l'aléatoire : une précision de 38.6% et un rappel de 16.3% seulement. Le retour issu de l'algorithme de programmation dynamique de Fisher sur les données Aiguillage, a mis en évidence que la classification a été bien réalisé à laide de cet algorithme avec un taux derreur null. En ce qui concerne la CAH et K-means nous retiendrons que : K-means possèdent les faiblesses suivantes :

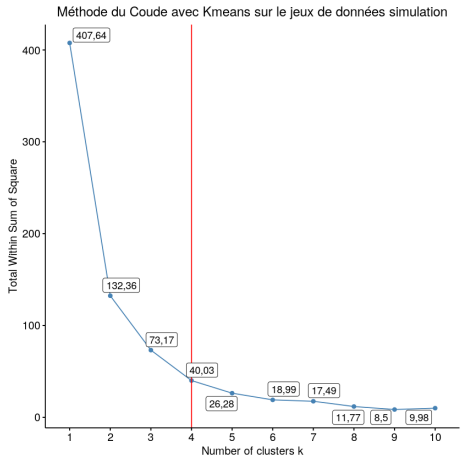
1. **Malédiction de la dimensionnalité** : L'utilisation de K-means signifie principalement un travail sur la distance euclidienne avec une augmentation des dimensions, les distances euclidiennes deviennent inefficaces. Pour une donnée 100 dimensions ou plus chaque points est beaucoup trop éloignés de entre eux.
2. **Clusters sphériques** : K-means réalise des formes uniquement sphériques. Donc, cela échoue si les données ressemblent à quelque chose comme ci-dessus, et généralement où les données ne sont pas sphériques.

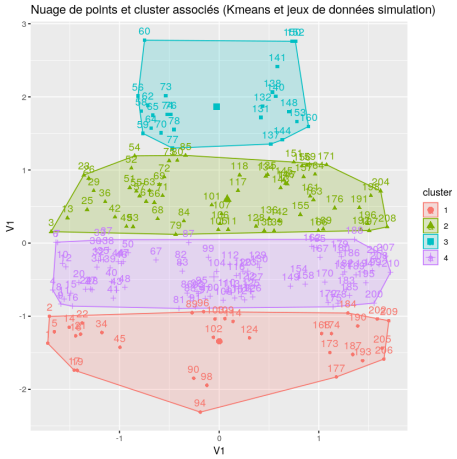
La classification hiérarchique avec le critère de Ward fonctionne très bien puisqu'il y 'a une hiérarchie sur l'ensemble du jeux de données (on remarque une ascendance des classes au fur et à mesure des observations sur le graphique des vraies partitions).

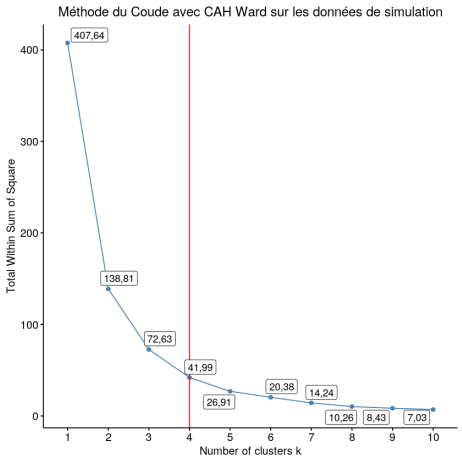
7 Conclusion

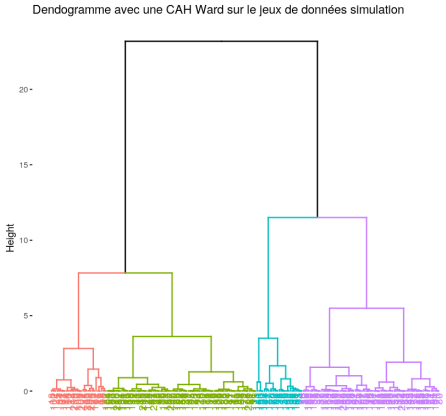
Ce projet d'apprentissage non-superivisé nous a permis d'implémenté l'algorithme de programmation dynamique de Fisher et de le comparé à la CAH et sur K-means. Nous avons pu voir que l'algortihme de Fisher est clairement approprié dans le cadre de séries temporelles puisqu'il tiens compte de la temporalité des données alors que K-Means et CAH Ward ont une approche spaciale. En effet ces deux autres méthodes regardent les jeux de données comme un ensemble de points dans un espace à un ou plusieurs dimensions sans prendre en compte le critère d'ordre temporelle. Même si nous avons effectuer quelques optimisations grâce à l'utilisation de package approprier, nous avons comme perspectives de voir l'efficacité de l'algorithme sur un GPU (composant plus approprié pour le calcul matriciel intensif) car lorsque nous ferons façe à un ensembles de données plus volumineux il est très fort probable que la complexité non linéaire de l'algorithme pose un grand soucis.

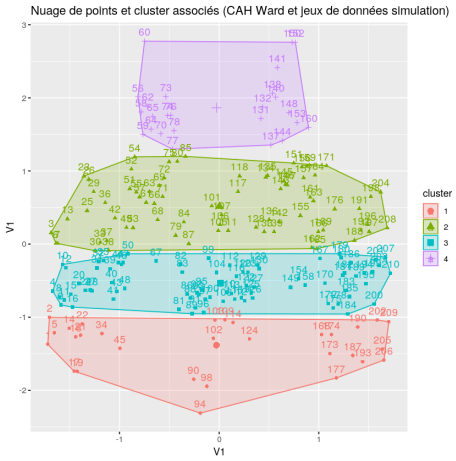
A Annexe

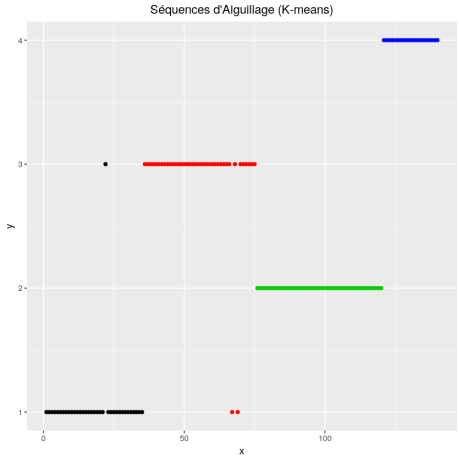


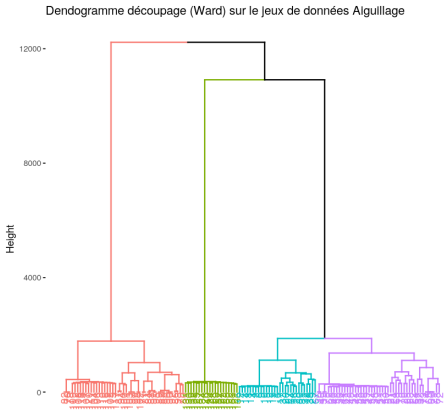

Nuage de points et cluster associés (Kmeans et jeux de données simulation)
V1
V1
cluster
1
2
3
4

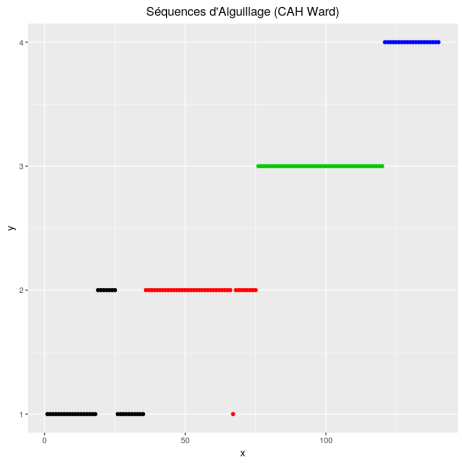

Méthode du Coude avec CAH Ward sur les données de simulation
Total Within Sum of Square
Number of clusters k
407,64
138,81
72,63
41,99
26,91
20,38
14,24
10,26
8,43
7,03

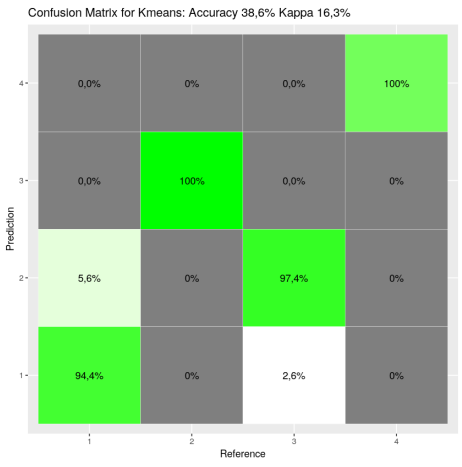

Dendogramme avec une CAH Ward sur le jeux de données simulation
Height
0
5
10
15
20

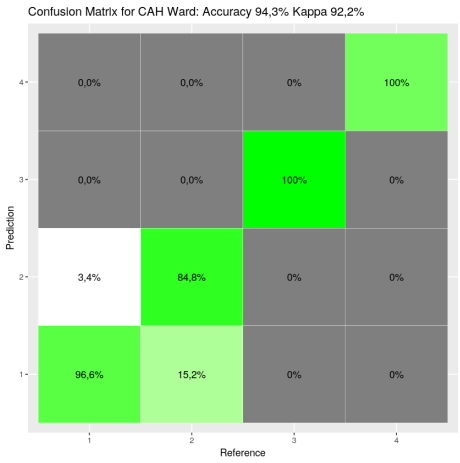

Nuage de points et cluster associés (CAH Ward et jeux de données simulation)
V1
V1
cluster
1
2
3
4


Séquences d'Algulllage (K-means)
4
3
2
1
y
0
50
100
x


Dendogramme découpage (Ward) sur le jeux de données Algulllage
12000
8000
4000
0
Height


Séquences d'Aiguillage (CAH Ward)
y
x
1
2
3
4
0
50
100


Confusion Matrix for Kmeans: Accuracy 38,6% Kappa 16,3%
Prediction
Reference
0,0%
0%
0,0%
100%
0,0%
100%
0,0%
0%
5,6%
0%
97,4%
0%
94,4%
0%
2,6%
0%
1
2
3
4



Références

1. R Core Team, *R : A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2014.
2. S.-E. Benkabou, *Outlier detection for time series data : application to tyre data*. Theses, Université de Lyon, Mar. 2018.
3. B. Hugueney, G. Hébrail, and Y. Lechevallier, "Reduction de series temporelles par classification et segmentation," 01 2006.
4. A. S. Chamroukhi Faicel, G. Gerard, and A. Patrice, "Classification automatique de donnees temporelles en classes ordonnees.," 2013.
5. W. D. Fisher, "On grouping for maximum homogeneity," *Journal of The American Statistical Association - J AMER STATIST ASSN*, vol. 53, pp. 789–798, 12 1958.