
Université Paris 1 Panthéon-Sorbonne

École Doctorale Sciences Mathématiques de Paris Centre (ED 386)

*Laboratoire :* Statistiques, Analyse et Modélisation Multidisciplinaire, EA 4543


# THÈSE DE DOCTORAT DE MATHÉMATIQUES APPLIQUÉES


**Me MYSELF**


Imp : A New

Dirigée par
Madalina Olteanu, Marie Chavent et Jérôme Lacaille

*Date de soutenance :* Jour Mois Année

*Après avis des rapporteurs :* Prénom NOM (Institution)
Prénom NOM (Institution)

*Jury de soutenance :*

| | | |
|---|---|---|
| Prénom NOM | (Institution) | Rapporteur |
| Prénom NOM | (Institution) | Codirecteur de thèse |
| Prénom NOM | (Institution) | Examinateur |
| Prénom NOM | (Institution) | Examinateur |
| Prénom NOM | (Institution) | Codirecteur de thèse |
| Prénom NOM | (Institution) | Invité |

**NNT :**

# Table des matières

# 0

# Règles et notations



## 0.1 Règles

1. un mot utilisé en anglais apparaît la première fois en italique et normalement par la suite : cela évitera de se demander quels mots en anglais doit-on toujours mettre en italique (cmd : textit).
2. si le mot utilisé en anglais désigne un nom, d'algorithme par exemple, faut-il le mettre en italique ? Cela n'a pas de sens de le traduire et c'est un nom propre donc l'utilisation de l'italique est-il inutile ici ? (cmd : textit)
3. Les termes que nous introduisons et qui ne sont pas familiers dans la littérature seront "emphasés" (cmd : emph) ce qui, avec le style du document est équivalent à mettre en italique (cmd : textit). Ex : la *vraie partition*, un *bon clustering*. (importance de variables ?)
4. que faire si le nom de l'approche est en fait pas un nom propre : stepwise forward-backward / hill climbing /
5. citations entre des parenthèses lorsque la citation n'est pas sujet (cmd : citep), citations sans les parenthèses sinon (cmd : cite).
6. des citations précises d'articles peuvent être faites à l'aide de la commande say. Par ailleurs, nous ne faisons pas correspondre les notations de notre article avec celles de citations : la norme $\ell_1$ - "...the l1 norm...".
7. utilisation des chiffres/nombres : mettre en toutes lettres seulement lorsque l'on ne désigne pas une valeur ? Ex : un ensemble de données contenant une variable indicatrice constituée de 0 et de 1.
8. nous notons Équation, Figure, Table avec des majuscules.
9. Nous faisons référence aux équations avec des parenthèses et sans parenthèses pour les tables et les figures : Équation (5) (cmd : eqref), Figure 5 (cmd : ref).
10. Les équations auxquelles on ne fait pas référence ne sont pas indexées.
11. Les descriptions (captions) apparaissent en dessous pour les figures et au dessus pour les tables.

12. les noms d'algorithmes, notamment en anglais, prennent une majuscule : Sparse $K$-means - Gap Statistic - Regularized Weighted $K$-means.
13. Les mots anglais sont écrits suivant le dictionnaire UK (non US), sauf pour les noms propres qui sont déjà définis : Regularized $K$-means - $K$-means regularised.

## 0.2 Sémantique

1. classes : un partitionnement des données.
2. clusters : des classes séparées forment des clusters. Des classes ne représentent pas des clusters lorsqu'elles ne sont pas bien séparées.
3. Les $K$-means - au pluriel.
4. le mot sparse peut être utilisé comme adjectif et donc il s'accorde (sparses) ou comme verbe (sparsifier).
5. ensemble de données - supprimer base de données et jeu de données.
6. observations - supprimer individus et objets.
7. s.c. signifie "sous contrainte" dans un problème d'optimisation.
8. matrice de covariance : pour designer la matrice de variance-covariance.
9. inter classes / intra classes : et non pas inter-classes / intra-classes
10. normaliser : nous utilisons le mot normaliser uniquement pour désigner le fait de centrer une variable et de la réduire à variance unitaire. Pour éviter toute confusion, on indiquera le plus souvent "normaliser à moyenne 0 et variance unitaire". Si une autre normalisation est utilisée, elle sera explicitée en toutes lettres à chaque fois (ex : normalisation par la somme des corrélations). Le mot "standardisation" n'est jamais utilisé.
11. utilisation du "nous" : utilisé la plupart du temps pour décrire nos contributions, contrairement au "on" plus général.

## 0.3 Notations

1. $\mathbf{X}$ est la matrice $n \times p$ des $n$ observations décrites par $p$ variables numériques.
2. $\mathbf{x}^j \in \mathbb{R}^n$ désigne la $j$-ième variable.
3. $\mathbf{x}_i \in \mathbb{R}^p$ représente la $i$-ème observation.
4. $\mathbf{X}_q \in \mathbb{R}^{n \times q}$ est une matrice de $n$ observations et où $q$ indique précisément le nombre de colonnes de la matrice avec généralement $\mathbf{X}_q \in \mathbf{X}$.
5. $\{\mathbf{X}^1, ..., \mathbf{X}^d\}$ est un ensemble de matrices $n \times p$ issues de la même distribution, indexées de 1 à $d$.
6. $K \geq 1$ le nombre de clusters.
7. $P_K = \{C_1, \ldots, C_K\}$ une partition des données en $K$ clusters.
8. $P_K \in \mathcal{P}_K$, $\mathcal{P}_K$ l'ensemble des partitions possibles en $K$ clusters.
9. $\bar{\mathbf{x}} \in \mathbb{R}^p$ est le centre des données.
10. $\bar{x}_k^j$ la moyenne de la variable $j$ dans le cluster $k$.
11. $n_k$ le nombre d'individus dans le cluster $k$.
12. La norme $\ell_p$ : $\|.\|_p$
13. $\mathbf{w}^\intercal$ : transposé
14. $\sigma_1$ : représente l'écart type de la variable $\mathbf{x}^1$ et donc $\sigma_1^2$ sa variance.

15. diag($\mathbf{x}$), $\mathbf{x} \in \mathbb{R}^p$ : représente une matrice diagonale de taille $p \times p$ et dont les termes diagonaux sont ceux du vecteur $\mathbf{x}$.

16. les matrices sont en majuscules en gras, les vecteurs en minuscules en gras, les vecteurs aléatoires en majuscule, les scalaires en minuscules. Les règles s'appliquent aussi au symboles. Ex : $\mathbf{\Sigma}$ pour la matrice de covariance (et non pas $\Sigma$).

## 0.4 À définir

1. modèle sous-jacent.
2. performance du modèle.
3. sparse.
4. sélection de variables.
5. importance de variables.
6. pondérations des variables.
7. variables importantes et variables de bruit.
8. l'algorithme Sparse $K$-means (Witten and Tibshirani, 2010) $\neq$ la famille des algorithmes de $K$-means sparses, c'est-à-dire tous les algorithmes sparses se basant sur le $K$-means.
9. la famille des GMM sparse.
10. données mixtes.
11. les inerties intra et inter classes.
12. la vraie partition.
13. La redondance et variables redondantes.
14. Complexité du modèle.
15. Schéma et scénario.

## 0.5 Questions

1. Dans le grand tableau des packages, faut-il mettre les packages GitHub en notes de bas de page dans le tableau ?
2. Dans les simulations, doit-on donner uniquement "le vrai modèle" aux GMM sparses ?
3. doit-on laisser $\mu_{k,j}$ ou doit-on écrire $\mu_k^j$ pour les centres ? (Section 2.5)
4. Normaliser les variables dans les simulations, pour les méthodes GMM sparses ?
5. Un mot sur les packages python ?

# 1

# Rappels sur l'apprentissage supervisé et non supervisé



## 1.1 Attention

Ceci n'est pas une introduction à proprement parler. Ce chapitre rappelle juste des concepts de base, des notations et des notions utiles pour le chapitre 1. Il n'est pas entièrement rédigé et sera rédigé complètement lorsque la thèse arrivera à son terme. Les sections ne sont pas liées entre elles et le chapitre n'est pas construit de manière uniforme.

Mais les notations et les concepts sont en adéquation avec le chapitre 1.

## 1.2 Définition du clustering et présentation des algorithmes

Le clustering est l'une des tâches les plus courantes en apprentissage non supervisé. Elle consiste à mettre en évidence des groupes d'observations (également appelés *clusters* ou *classes*) dans un jeu de données. Pour l'analyse exploratoire de données, cela permet de mieux comprendre la structure d'un jeu de données et peut également être utilisé pour la classification. Le clustering peut être défini comme le partitionnement des données en groupes (ou clusters) de sorte que les éléments similaires (ou au sens d'une fonction de distance sous-jacente) partagent le même cluster et que tous les membres de chaque cluster soient similaires (ou, de manière équivalente, que les éléments dissimilaires soient séparés en clusters différents) (Ben-David, 2018). C'est un objectif très difficile notamment en raison de la non-transitivité de la notion de similarité : si $A$ est similaire à $B$, et $B$ est similaire à $C$, $A$ n'est pas nécessairement similaire à $C$. Nous verrons plus en détail dans les chapitres suivant les défis qui se posent en clustering.

Pour définir une similarité entre deux observations, on peut les décrire par un ensemble de variables et utiliser une distance appropriée telle que la distance euclidienne. C'est le point de vue qui sera principalement utilisé dans cette thèse mais d'autres distances et des types de données plus complexes peuvent être utilisés.

### 1.2.1 Définition

En clustering le choix de la distance est primordiale et est souvent défini par l'application et les données. Ainsi la littérature sur le clustering inclut généralement une longue liste de définitions pour les distances entre objets. Ce ne sera pas le cas ici, nos applications se basent sur la distance euclidienne :

$$d(\mathbf{x}_i, \mathbf{x}_{i'}) := ||\mathbf{x}_i - \mathbf{x}_{i'}||_2 = \sqrt{\sum_{j=1}^{p} \left(x_i^j - x_{i'}^j\right)^2}.$$

## 1.3 Évaluation des méthodes de clustering et des méthodes de sélection de modèles en clustering

### 1.3.1 Comment évaluer les méthodes de clustering

Pour évaluer une méthode de clustering, il est d'usage dans la littérature de comparer la partition obtenue avec une partition sous-jacente que l'on appelle la *vraie partition*. Ainsi, évaluer la qualité d'un clustering est équivalent à mesurer le taux d'erreur en apprentissage supervisé. Néanmoins, cela nécessite d'avoir connaissance de la vraie partition, ce qui n'est généralement pas le cas pour les données réelles. En fait, les méthodes de clustering ne peuvent être évaluées que sur des jeux de données simulées. Pour illustrer certains aspects d'une méthode, les jeux de données réelles sont indispensables, mais les ensembles de données simulées sont les seuls qui garantissent une évaluation des performances significative et fiable. Ce point sera discuté en profondeur dans les sections suivantes.

Généralement en clustering, le nombre de clusters à trouver est inconnu et il faut l'estimer conjointement à la partition. Certains algorithmes tels que les algorithmes de clustering hiérarchiques Ward (1963), permettent d'estimer le nombre de clusters, d'autres utilisent des critères de pénalité fondés statistiquement tels que le BIC Schwarz (1978) ; Lebarbier and Mary-Huard (2006) pour les modèles de mélanges Gaussiens (GMM). Enfin pour une grande partie des algorithmes dont l'algorithme des $K$-means, le choix du nombre de clusters reste une question difficile et ouverte. C'est pourquoi, lorsqu'un algorithme de clustering est couplé à une méthode fournissant une estimation du nombre de classes, ce qui sera majoritairement le cas pour les algorithmes que nous verrons dans le chapitre suivant, il est possible de les évaluer dans le même temps.

Néanmoins, cela ne permettra pas d'étudier les performances de l'algorithme de clustering seul. Une bonne pratique dans ce cas est de fixer le nombre de clusters, pour évaluer l'algorithme de clustering de manière indépendante. Une deuxième évaluation suivra pour évaluer la méthodologie globale.

Supposons alors que nous connaissions la vraie partition, ce qui est le cas lorsque l'on simule des données, reste à définir une mesure pour comparer deux partitions. Si le nombre de classes des deux partitions est le même, le taux de bon classement peut être utilisé (cite). Si le nombre de classes est grand ou si les classes sont déséquilibrées, d'autre mesures plus adéquates ont été proposées (cite). Dans le cas où le nombre de classes n'est pas le même, plusieurs mesures ont été définies pour évaluer la performance de l'algorithme. Par exemple, l'Adjusted Rand Index (ARI) défini par Hubert and Arabie (1985) est selon Romano et al. (2015) un choix recommandé lorsque les clusters sont majoritairement équilibrés.

Cependant, si l'on cherche à évaluer des méthodes sur différents ensembles de données, calculer un ARI moyenné peut être trompeur. En effet, les ensembles de données peuvent être de difficultés différentes et, par conséquent, une simple moyenne peut induire en erreur Demšar (2006). La moyenne fait sens pour agréger les résultats d'une même simulation qui a été répétée, mais pas pour agréger des résultats sur différents ensembles de données.

Afin de comparer les méthodes sur plusieurs bases de données de complexité et de difficulté différentes, il est possible de faire des tests de rangs avec le test de rangs de Wilcoxon (Demšar, 2006)) ou de calculer une performance normalisée par la difficulté de l'ensemble des données (Hofmeyr, 2018), pour autant qu'on puisse l'estimer.

### 1.3.2 Le problème de l'évaluation sur des données réelles

Pour utiliser une vraie partition , il est nécessaire de s'assurer qu'elle représente bien exactement les clusters présents dans les données. Cela nécessite une analyse approfondie du jeu de données et, quand bien même celle-ci est faite, cela ne garantit pas que partition des données soit conforme à la réalité. En effet, il peut toujours exister des variables latentes non observées qui induisent des structures de clusters dans les données, provoquant un décalage entre la variable désignant la vraie partition et les vrais clusters sous-jacents. Pour autant que nous le sachions, aucune méthodologie ou analyse n'a été suggérée pour garantir avec certitude

qu'un jeu de données réelles soit partitionné en $K$ classes. La seule étude que nous avons trouvée dans ce domaine est celle de Bouveyron and Brunet-Saumard (2014b), qui ont étudié la base de données de Campbell and Mahon (1974). ils ont constaté, en utilisant l'ACP, que deux variables binaires indépendantes déterminent en fait quatre clusters que l'on peut découvrir sur les deuxième et troisième composantes principales. Par conséquent, il semble que sur cet ensemble de données, il existe quatre groupes distincts.

En général, les algorithmes de clustering tels que les $K$-means sont souvent testés sur des ensembles de données réelles et les clusters obtenus comparés à une variable catégorielle désignée comme la vraie partition. Il se peut que l'algorithme réussisse à retrouver les clusters définis par la vraie partition et que l'ARI soit élevé. Par exemple, c'est le cas pour la base de données sur les maladies cardiaques HDdata*, pour lequel l'algorithme des K-means exécuté sur les variables numériques retrouvent les clusters définis par la variable qui code la présence ou l'absence de maladie cardiaque. On sait que les classes sont quasiment linéairement séparables sur la première composante principale. Cependant, comme on peut le voir sur la Figure 1.1(a), la frontière entre les classes est très dense et les classes peuvent représenter une distribution unimodale plutôt que bimodale, ce qui soulève des questions quant à l'existence de 2 clusters distincts.

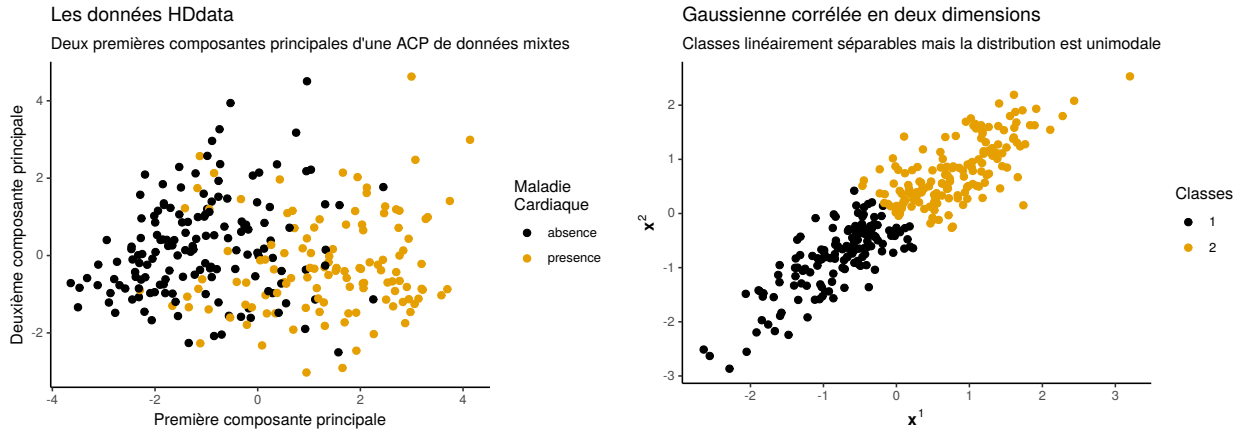
Pour résumer, sur des données réelles, rien ne peut garantir qu'une variable donnée ou définie par l'utilisateur représente la vraie partition si toutefois celle-ci existe et en aucun cas le fait qu'un algorithme retrouve un partitionnement défini arbitrairement est la preuve que celui-ci représente les clusters, les seuls clusters, la vraie partition.

### 1.3.3 Comment évaluer les méthodes de sélection de modèle pour le clustering

On doit distinguer l'évaluation d'une méthode de clustering pour un nombre de classes fixé, de l'évaluation de la qualité du modèle, c'est-à-dire de l'adéquation du nombre de classes choisi avec la réalité.

Par exemple, une distribution Gaussienne unimodale non sphérique peut être divisée en deux en son centre pour former artificiellement deux classes comme c'est le cas sur la Figure 1.1(b). Dans ce cas, la partition trouvée par les 2-means sera la meilleure solution au sens de l'ARI, mais ne sera pas un *bon* modèle, puisqu'en réalité il n'y a pas lieu de considérer deux clusters

Il faut donc être très prudent lors de la simulation de données destinées à évaluer les méthodes de sélection de modèles. Dans les expériences, nous éviterons au maximum le chevauchement entre les clusters en générant des clusters séparés, ce qui donnera des scénarios plus simples mais fiables.




(a) (b)

**Figure 1.1 :** Le graphique (a) représente les classes définies par la variable indiquant la présence ou l'absence d'une maladie cardiaque sur la première et deuxième composante principale d'une ACP mixte. Le graphique (b) illustre des classes artificielles définies arbitrairement sur un jeu de données Gaussien de dimension deux. Sur ces deux exemples les $K$-means ont des très bons résultats ($K$-means sur les composantes principales pour HDdata).

### 1.3.4 Résumé

Pour résumer, trois points sont importants à retenir :

1. Pour évaluer des méthodes de clustering ou des méthodes de sélection de modèles en clustering, il ne suffit pas de savoir que l'on a trouvé le « bon » nombre de clusters, il faut comparer les partitions

*https ://archive.ics.uci.edu/ml/datasets/statlog+(heart)

obtenues avec une mesure de dissimilarité Pour des simulations répétées de données de complexités comparables, les résultats peuvent être moyennés, sinon il faudra utiliser par exemple des tests de rangs.

2. Sur des données réelles, rien ne ne peut garantir qu'une variable donnée ou définie par un utilisateur représente la vraie partition si toutefois celle-ci existe et en aucun cas le fait qu'un algorithme retrouve un partitionnement défini arbitrairement n'est la preuve que ce partitionnement représente la vraie partition.

3. L'évaluation des méthodes de clustering ou des méthodes de sélection de modèle nécessite de disposer de bases de données qui sont effectivement structurées en clusters. Même si cela parait évident, ce n'est pas si simple, caril n'y a pas de définition concrète de ce qu'est un cluster. C'est pourquoi dans la suite nous utiliserons des jeux de données simulées et des vérifications visuelles de clusters simulés.

## 1.4 Définitions sur la sélection et l'importance de variables

SÉLECTION DE VARIABLES En statistique, la *sélection de variables* est le processus de sélection d'un sous-ensemble de variables pertinentes à utiliser dans la construction d'un modèle. Les techniques de sélection de variables sont utilisées principalement pour deux raisons :

1. améliorer les performances du modèle.

2. simplifier les modèles pour les rendre plus faciles à interpréter.

On peut également noter quelques raisons secondaires :

- temps d'apprentissage plus court, par exemple pour de la mise en production.
- éviter la malédiction de la dimension.
- réduire les coûts d'acquisition et de stockage des données.

L'hypothèse centrale de l'utilisation d'une technique de sélection de variables est que les données contiennent des variables qui sont redondantes et/ou non pertinentes et qui peuvent donc être supprimées sans entraîner de perte d'information. La redondance et la pertinence sont deux notions distinctes, car une variable pertinente peut être redondante en présence d'une autre variable pertinente avec laquelle elle est fortement corrélée. Les techniques de sélection de caractéristiques doivent être distinguées des méthodes de transformations de l'espace comme l'ACP, créent de nouvelles variables à partir de fonctions des variables d'origine.

MÉTHODES SPARSES Dans ce manuscrit, les méthodes de sélection de variables seront appelées des méthodes *sparses* car elles permettent de trouver des modèles plus petits que le modèle complet, des modèles parcimonieux. De plus, nous nous intéressons à l'utilisation des méthodes sparses dans un but d'interprétabilité plus que d'efficacité, c'est-à-dire que nous cherchons en priorité construite à des modèles explicatifs plutôt que des modèles performants. En revanche, avoir un modèle performant reste indispensable, car un mauvais modèle ne donnera pas de bonnes interprétations. Ce compromis entre interprétabilité et performance forme l'épine dorsale de notre travail et il a guidé la plupart de nos réflexions. ! [1]( Nous considérons deux types d'algorithmes sparses :

- *pondération binaire* ; l'algorithme catégorise les variables en deux groupes dont l'un est inclus dans le modèle et l'autre est exclu en attribuant respectivement des poids de 1 ou 0 aux variables.

- *pondération positive ou nulle* ; l'algorithme attribue un poids positif ou nul aux variables indiquant la contribution de la variable au modèle. Un poids nul indique que la variable est exclue du modèle.

- *sans pondération* ; Il se peut que le poids des variables ne soient pas explicitement introduits dans le modèle et dans ce cas il faut faire appel à des méthodes dites *post-hoc* pour calculer la contribution des variables.

Les méthodes sparses permettent de déterminer les variables pertinentes, qui sont importantes pour le modèle, en les incluant dans le modèle. Nous faisons un choix sémantique d'appeler *variables importantes* les variables qui sont liées au modèle et *variables non importantes* ou *variables de bruit*, les variables indépendantes du modèle. Lorsque la contribution d'une variable n'est pas directement donnée par le modèle, des méthodes de calcul de l'importance de variables peuvent être utilisées. )[1]!

IMPORTANCE DE VARIABLES Le domaine de l'importance de variables est très lié à celui de la sélection de variables. D'un point de vue statistique, le calcul de l'importance des variables vise à satisfaire deux objectifs :

i) estimer la contribution de chaque variable d'entrée à la performance du modèle.

ii) mesurer la dépendance entre les variables d'entrée et le modèle sous-jacent.

Cependant, il faut noter que ces objectifs peuvent être contradictoires, par exemple lorsque les variables sont corrélées. En effet, si deux variables sont fortement corrélées, l'une d'entre elles peut être écartée sans dégrader les performances du modèle. Par ailleurs, elle reste liée à des variables du modèle sous-jacent, elle détient de l'information sur le modèle et est donc dépendante de celui-ci.

Dans ce travail de thèse, nous visons à atteindre l'objectif ii). Ce cadre est différent de celui qui est habituellement fixé dans la littérature. En effet, la majeure partie des travaux en sélection et importance de variables vise à atteindre l'objectif i). Par conséquent, dans la littérature, une variable hors du modèle sous-jacent, mais corrélée à une variable du modèle sous-jacent est définie comme redondante et non importante. Nous n'adoptons pas ce point de vue car le modèle sous-jacent n'est jamais disponible. On n'est alors jamais assuré de ne sélectionner uniquement que des variables du modèle notamment dans le cas de données fortement corrélées. Par ailleurs cette représentation théorique (existence d'un modèle sous-jacent) est une simplification qui ne permet pas de répondre aux besoins des applications sur des données réelles. En effet, si deux variables sont très corrélées et l'une est considérée comme importante, alors l'autre l'est aussi puisqu'elle partage la même information et elles doivent être toutes deux intégrées dans l'analyse de données.

Parler de classement.

REDONDANCE D'INFORMATION Dans le cadre de l'objectif ii), deux variables corrélées liées au modèle sont redondantes, car elles apportent la même information mais nous les définissons toutes les deux comme importantes. La *redondance* est donc une mesure de dépendance entre les variables importantes et la corrélation implique donc de la redondance dans les données. En revanche, des variables de bruit ne sont pas redondantes en ce sens qu'elles sont indépendantes du modèle. Cela n'empêche pas que des variables de bruit peuvent être dépendantes entre elles. Par contre, des variables de bruit ne peuvent pas être dépendantes de variables importantes.

### 1.4.1 Évaluer les méthodes d'importances de variables

# 2

# État de l'art : Analyses, comparaisons et critiques des algorithmes de clustering sparse

## 2.1 Introduction

Le clustering est ![2]( à ce jour )[2] ! une tâche compliquée à résoudre, alors le clustering sparse peut paraître quant à lui inabordable. Déjà il y a plus de 20 ans, Gnanadesikan et al. (1995) écrivaient "One of the thorniest aspects of cluster analysis continue to be the weighting and selection of variables" et plus récemment Raftery and Dean (2006) commentaient que "Less work has been done on variable selection for clustering than for classification, perhaps reflecting the fact that the former is a harder problem. In particular, variable selection and dimension reduction in the context of model-based clustering have not received much attention". Les difficultés tant dans la formulation que dans la mise en oeuvre des algorithmes font que les algorithmes de clustering sparses restent encore peu utilisés comparativement aux algorithmes de clustering standards et cela est encore plus surprenant au vue de la popularité des méthodes sparses dans le contexte supervisé. Néanmoins nous allons voir que de nombreuses méthodes déjà exploitables existent et offrent des solutions pertinentes.

### 2.1.1 Du clustering au clustering sparse : motivations

Il est légitime de penser que s'il existe une structure de clustering dans des ensembles de données réelles, les clusters sous-jacents ne diffèrent que selon certaines variables. Les ensembles de données réelles contiennent souvent des variables indépendantes et non informatives pour le clustering, que nous appelons variables de bruit, et celles-ci peuvent être en très grand nombre. Considérer explicitement ces variables comme du bruit dans le modèle va permettre tout d'abord d'améliorer les performances de l'algorithme et ensuite de réduire le nombre de variables du modèle, qui sera ainsi plus facile à interpréter.

### 2.1.2 Contributions de ce chapitre

La contribution essentielle de ce chapitre est de proposer une étude détaillée et critique d'une grande partie des algorithmes de clustering permettant de faire de la sélection de variables, ! [3]( en mettant l'accent notamment sur les méthodes basées sur les $K$-means ou les GMM )[3] !. De plus, une analyse comparative des méthodes, s'appuyant sur des schémas de simulations structurés et des indicateurs précis, est proposée.

Il existe deux excellentes études comparatives dans ce domaine : celles de Bouveyron and Brunet-Saumard (2014a) et de Fop and Murphy (2018). Ces études sont orientées plus spécifiquement sur des méthodes basées sur les modèles de mélanges Gaussiens. Dans Bouveyron and Brunet-Saumard (2014a) on trouve une description d'algorithmes non sparses pour des données de grande dimension et les algorithmes sont illustrés sur des données réelles. Dans Fop and Murphy (2018), les auteurs ajoutent une partie sur l'analyse des classes latentes qui sont des modèles de mélanges permettant de faire du clustering sur des observations décrites le plus souvent par des variables catégorielles. Ils illustrent les méthodes sur des ensembles de données réelles et font une analyse comparative des temps de calcul des algorithmes.

Notre apport réside donc dans le fait d'étendre ce type d'étude aux algorithmes basés sur les K-means et de proposer une analyse comparative numérique détaillée des algorithmes dont le code est disponible librement.

De plus nous insistons sur l'état de l'art des algorithmes de clustering sparse spécifiquement des familles des $K$-means sparses et GMM sparses. Une critique des méthodes est exposée et un résumé des résultats de simulations obtenus dans la littérature est donné. L'étude est complétée par une analyse complète de ces simulations lorsque le code est disponible. Enfin, nous menons notre propre analyse numérique dans la Section 2.9.

## 2.2 Rappels

Dans cette section, nous allons faire des rappels sur les critères d'inertie et sur l'algorithme des $K$-means. C'est points sont nécessaires tant pour la compréhension que pour la formulation des méthodes décrites dans ce chapitre et dans ce manuscrit en général.

### 2.2.1 Critères d'inertie

En clustering, lorsque l'on utilise la distance euclidienne, on peut naturellement se baser sur des critères de variance qui sont équivalents à une normalisation près aux critères d'inertie. Rappelons que l'inertie totale est la somme des distances euclidiennes au carré des individus au centre des données. On sait d'après le théorème de König-Huygens que la inertie totale se décompose en la somme de l'inertie inter-classes et de l'inertie

intra-classes et cette décomposition s'étend à la variance totale :

$$\underbrace{\sum_{j=1}^{p} \frac{1}{n} \sum_{i=1}^{n} (x_i^j - \bar{x}^j)^2}_{t^j} = \underbrace{\sum_{j=1}^{p} \frac{1}{n} \sum_{k=1}^{K} \sum_{i \in C_k} (x_i^j - \bar{x}_k^j)^2}_{v^j} + \underbrace{\sum_{j=1}^{p} \frac{1}{n} \sum_{k=1}^{K} n_k (x_k^j - \bar{x}^j)^2}_{b^j}, \tag{2.1}$$

où $\bar{\mathbf{x}}_k = (\bar{x}_k^1, ..., \bar{x}_k^p)^\intercal$ est le centre du cluster $k$, $\bar{\mathbf{x}}$ est le centre des données et $n_k$ le nombre d'individus dans le cluster $k$ avec $\bar{x}_k^j = \frac{1}{n_k} \sum_{i \in C_k} x_i^j$ et $\bar{x}^j$ la moyenne de la variable $j$ et $t^j, v^j, b^j$ sont respectivement la variance, la variance intra-classes et la variance inter-classes de la variable $j$. Les critères d'inertie permettent de rendre compte simultanément de l'homogénéité de de la séparabilité des classes et ceci est illustré par la Figure 2.1. L'inertie totale étant une quantité fixe indépendante des classes, minimiser l'inertie intra, c'est-à-dire rendre homogène les classes, est équivalent à maximiser l'inertie inter, c'est-à-dire séparer des classes. L'algorithme

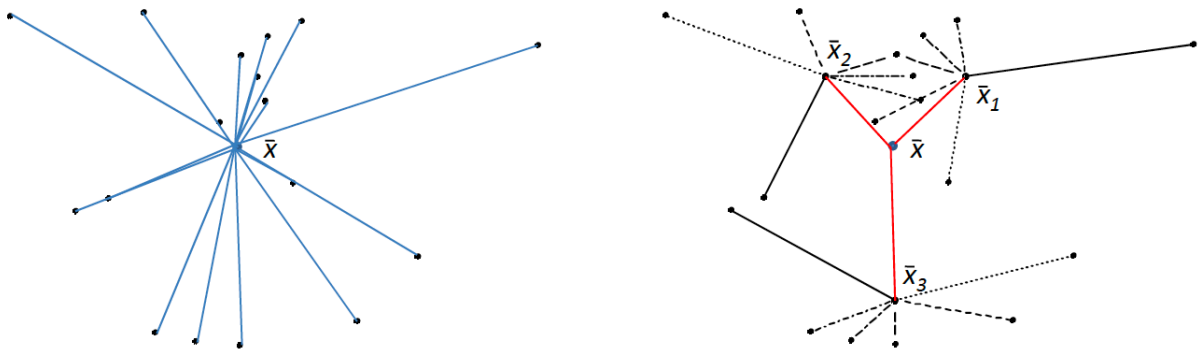



**Figure 2.1 :** Le graphique illustre l'Équation 2.1.

le plus utilisé en clustering se base sur ces relations car il minimise l'inertie intra-classes et donc maximise l'inertie inter-classes. Il est connu sous le nom de $K$-means ($K$-moyennes ou algorithme des centres mobiles).

### 2.2.2 $K$-means

La méthode $K$-means MacQueen et al. (1967) ; Hartigan and Wong (1979) ; Lloyd (1982) est sans aucun doute l'algorithme de clustering le plus connu et le plus utilisé. Comme son nom l'indique, il se définit en choisissant un paramètre $K$ correspondant au nombre de clusters à trouver. Le but est de minimiser la variance intra-classes ou de manière équivalente maximiser la variance inter-classes :

$$\underset{P_K \in \mathcal{P}_K}{\text{minimiser}} \sum_{j=1}^{p} v^j \iff \underset{P_K \in \mathcal{P}_K}{\text{maximiser}} \sum_{j=1}^{p} b^j. \tag{2.2}$$

L'Équation 2.2 définit un problème non convexe qui est donc difficile à optimiser. La stratégie mise en place est un algorithme itératif, fonctionnant par étapes successives Lloyd (1982). L'algorithme commence par initialiser les $K$ centres notés $\mathbf{m}_1, ..., \mathbf{m}_K$, en choisissant par exemple des observations tirées aléatoirement et alterne entre les deux étapes suivantes jusqu'à la convergence de l'algorithme.

1. **Affectation** : chaque observation est assigné au centre le plus proche,

$$C_k \leftarrow \{i \mid \underset{k'=1,...,K}{\text{argmin}} \; ||\mathbf{x}_i - \mathbf{m}_{k'}||_2^2 = k\}.$$

2. **Minimisation** : chaque centre est mis à jour pour devenir la moyenne de son cluster,

$$\mathbf{m}_k \leftarrow \frac{1}{n_k} \sum_{i \in C_k} \mathbf{x}_i.$$

De fait de sa non convexité, on ne peut empêcher que l'algorithme soit souvent piégé dans des minimums locaux. C'est pour cela qu'il est recommandé de relancer plusieurs fois l'algorithme avec des initialisations différentes et de choisir la meilleure solution au sens de L'Équation 2.2. D'autres types d'initialisation sont aussi

possibles, tel que le $K$-means++ Arthur and Vassilvitskii (2007). L'intuition derrière cette approche est qu'il faut choisir les centres initiaux les plus distants possible : le premier centre de cluster est choisi uniformément au hasard parmi les observations, après quoi chaque centre est choisi parmi les points de données restants avec une probabilité proportionnelle à sa distance au carré du centre de cluster le plus proche.

## 2.3 Versions pondérées non sparses

Ces méthodes sont à l'origine des méthodes sparses. À voir le niveau de détails que l'on veut - mais il faudra sûrement dire au moins quelques mots.

- Friedman and Meulman (2004)
- Huang et al. (2005)
- Jing et al. (2007)
- Domeniconi et al. (2007)
- Chen et al. (2012)

## 2.4 Méthodes sparses basées sur les $K$-means

### 2.4.1 WT-$K$-means (1)

Dans cette sous-section, nous introduisons un algorithme qui a une place centrale dans ce manuscrit. Cet algorithme a été présenté dans un article de Daniella Witten et Robert Tibishirani paru en 2010 (Witten and Tibshirani, 2010). L'idée générale est de réécrire le problème d'optimisation des $K$-means en introduisant des poids et une contrainte basée sur la norme $\ell_1$ de ces poids pour obtenir de la sparsité en forçant des poids à valoir 0, excluant les variables du modèle. On nomme cet algorithme WT-$K$-means et il fait partie de la famille des algorithmes $K$-means sparse.

L'algorithme repose sur l'introduction d'un nouveau critère de variance, la variance pondérée et pénalisée. Pour cela il faut repartir de L'Équation (2.2) et ajouter des poids aux variances inter-classes par variable ainsi qu'une pénalité dépendant de ces poids. Cela donne le critère suivant :

$$\sum_{j=1}^{p} w_j b_j - \lambda h(\mathbf{w}) = \mathbf{w}^\intercal \mathbf{b} - \lambda h(\mathbf{w}) \ , \tag{2.3}$$

où $\mathbf{w}^\intercal = (w_1, ..., w_p)$, est le vecteur des poids et $\mathbf{b}^\intercal = (b_1, ..., b_p)$ sont les variances inter-classes par variable, pour une partition donnée $C_1, ..., C_K$, et $\lambda \geqslant 0$ est l'hyperparamètre déterminant l'intensité de la pénalisation. Dans ce qui suit, $K$, le nombre de clusters, ainsi que $\lambda$ sont supposés être fixés à l'avance. Dans l'algorithme WT-$K$-means introduit dans Witten and Tibshirani (2010), le terme de régularisation est choisi par analogie avec le cadre *lasso* (least absolute shrinkage and selection operator) (Tibshirani, 1996), donc $h(\mathbf{w}) = \|\mathbf{w}\|_1$. Ainsi, lorsque l'optimisation de l'Équation (2.3) se fait par rapport à $\mathbf{w}$, d'une part les poids traduisent directement la contribution des variables au clustering et d'autre part l'on obtient une représentation sparse des variables. En effet, la pénalité $\ell_1$ permet la mise à 0 de certains poids ce qui implique que les variables correspondantes n'influent plus sur l'inertie inter-classes et de fait sont exclues du modèle.

On peut justifier le bien fondé d'un tel type de critère. En effet, nous avons décrit dans l'introduction que le but du clustering était de regrouper des individus similaires ce qui se traduit par la construction des classes homogènes. Or, on sait que les $K$-means minimisent la variance intra-classes, c'est à dire qu'ils cherchent des classes homogènes et par la même occasion ils maximisent la variance inter-classes, variance inter-classes qui est une mesure de la séparabilité des classes. Par conséquent, la variance inter-classes par variable rend compte du pouvoir discriminatif de chaque variable. Il est donc astucieux de faire de la sélection de variables en se basant sur un tel critère.

Optimiser la version pondérée de la variance inter-classes revient à trouver des poids et des clusters. Le problème peut alors se réécrire sous la forme d'une double optimisation :

$$\underset{C_1,...,C_K,\mathbf{w}}{\text{maximiser}} \ \mathbf{w}^\intercal \mathbf{b} - \lambda \|\mathbf{w}\|_1 \ \text{ s.c. } \ \|\mathbf{w}\|_2^2 \leqslant 1, w_j \geqslant 0 \ \forall j. \tag{2.4}$$

Le paramètre $\lambda$ peut être choisi à l'aide de critères d'évaluation de qualité du modèle. Par ailleurs, en très grande majorité, les critères d'évaluation en clustering sont destinés à choisir le nombre de clusters $K$.

Ici, $\lambda$ détermine le sous-espace dans lequel les clusters résident. Les deux tâches sont très différentes et les outils qui fonctionnent pour l'une ne vont pas obligatoirement fonctionner pour l'autre. Dans leurs travaux, Witten and Tibshirani (2010) utilisent la Gap Statistic (Tibshirani et al., 2001). Le Gap Statistic est une méthode bien connue, elle est utilisée pour déterminer le nombre de clusters dans un ensemble de données. Elle consiste à comparer le logarithme de la variance inter-classes à son espérance sous une distribution de référence obtenue par exemple, ! [8](en tirant selon une loi uniforme des observations )[9] !. Dans leur article, elle est donc étendue au chois du paramètre $\lambda$, mais ils en nuancent les performances "the performance of the gap statistic of Section 3.2 for selecting the tuning parameter is mixed. This is not surprising, since tuning parameter selection in the unsupervised setting is known to be a very difficult problem. This is an area in which more work is needed.". Le choix de $K$ n'est pas abordé par les auteurs.

Nous aurons une ample discussion dans les sections suivantes et dans les chapitres suivants sur le choix des paramètres, l'initialisation, la normalisation, la convergence ; qui font l'ipséité stricte de l'algorithme mais qui sont aussi communs à toute la famille des $K$-means sparse.

### 2.4.2 Extensions du WT-$K$-means

WT-$K$-MEANS ET DONNÉES ABERRANTES (2) Plusieurs extensions du WT-$K$-means ont été proposées. Historiquement, l'une des premières proposée consiste à rendre l'algorithme robuste aux données extrêmes ou aberrantes (Kondo et al., 2012). Pour cela, les auteurs se basent sur une version différente des $K$-means, appelée Trimmed $K$-means (Cuesta-Albertos et al., 1997). L'idée principale est de remplacer l'étape d'affectation des observations dans les $K$-means par une nouvelle étape :

1. **Affectation**$^*$ : Après avoir affecté les observations aux clusters, on les ordonne selon leur distance au centre de leur cluster, on retire (*we trim*) un pourcentage $\alpha$ des observations les plus éloignées de leur centre, et on met à jour les centres des clusters sans tenir compte des observations retirées.

À chaque étape les observations retirées peuvent être différentes. Une première approche naïve pour rendre robuste le WT-$K$-means consisterait à utiliser l'algorithme Trimmed $K$-means à la place de l'algorithme des $K$-means.

Selon les auteurs, le problème avec cette approche serait que si l'ensemble des données possède une observation considérée comme aberrante, alors le WT-$K$-means avec $K = 2$ pourra éventuellement former deux clusters dont l'un contiendra uniquement cette valeur aberrante. et ? Kondo et al. (2012) proposent donc de modifier l'algorithme pour résoudre ce problème. L'idée est qu'à partir de la partition trouvée à chaque itération par le WT-$K$-means, il faut déterminer deux ensembles d'observations à retirer. Le premier à partir du sous-espace pondéré par les poids du WT-$K$-means et le second à partir de l'espace original. En procédant ainsi, les auteurs assurent que les poids nuls ne masqueront pas nécessairement les valeurs aberrantes dans les variables de bruit.

Une deuxième extension est proposée par Brodinová et al. (2019), dont la procédure algorithmique est complexe. Toujours dans l'idée de faire du clustering sparse en présence de données aberrantes, les auteurs reprennent le schéma méthodologique de Kondo et al. (2012) et proposent de remplacer le Trimmed $K$-means par l'algorithme ROBIN (3) (ROBust INitialization) de Al Hasan et al. (2009). Cet algorithme impose des contraintes sur les observations choisies à l'initialisation, contraintes qui ont pour but de rendre l'initialisation insensible aux valeurs aberrantes. En plus de l'utilisation de ROBIN, la méthodologie expliquée précédemment est utilisée. Par conséquent, deux ensembles d'observations à retirer sont calculés sur les données pondérées et non pondérées, mais en identifiant les valeurs aberrantes à l'aide de l'algorithme Local Outlier Factor (LOF) (Breunig et al., 2000), où LOF est appliqué indépendamment sur chaque cluster. Les auteurs indiquent que la procédure proposée est conçue de manière à ce que les paramètres requis soient sélectionnés automatiquement et que notamment cela permettrait de se défaire de l'hyperparamètre $\alpha$. Il faut noter que malheureusement cela se fait au prix de nouveaux paramètres puisque LOF requiert deux nouveaux hyperparamètres dont le nombre de plus proches voisins. Ces nouveaux hyperparamètres sont fixés par défaut suivant les recommandations des articles de références (Breunig et al., 2000 ; Al Hasan et al., 2009).

WT-$K$-MEANS ET GROUPES DE VARIABLES (4) Un autre travail sur les $K$-means sparses a été proposé dans Huo and Tseng (2017) et cette approche étend l'algorithme WT-$K$-means avec une pénalité dite *overlapping group*. L'approche permet de sparsifier au sein des groupes et la fonction de pénalité permet le chevauchement des groupes (Jacob et al., 2009). La combinaison de la pénalité lasso de norme $\ell_1$ et de la pénalité overlapping group sert à faire de la sélection de variables et encourage également (mais ne force pas) les variables du même groupe à être sélectionnées ensemble. Cela rejoint donc dans sa finalité la pénalité sparse group lasso combinant group lasso et norme $\ell_1$ Friedman et al. (2010a), mais avec une contrainte de structure de groupe

différente. Formellement, Le problème d'optimisation s'écrit comme suit :

$$\underset{C_1,...,C_K,\mathbf{w}}{\text{maximiser}}\ \mathbf{w}^\intercal\mathbf{b} - \lambda\Big[\alpha\left\|\mathbf{w}\right\|_1 + (1-\alpha)h(\mathbf{w})\Big]\ \text{s.c.}\ \left\|\mathbf{w}\right\|_2^2 \leqslant 1, w_j \geqslant 0\ \forall j,$$

où est le paramètre de réglage de la pénalité contrôlant le nombre de poids non nuls, $\alpha \in [0,1]$ est un terme contrôlant l'équilibre entre la pénalité intra-groupes ($\left\|\mathbf{w}\right\|_1$) et la pénalité inter-groupes ($h(\mathbf{w})$). Notamment, si $\alpha = 1$, il n'y a pas de terme de pénalité de groupes et la fonction objectif est équivalente à celle du WT-$K$-means et si $\alpha = 0$, seule la pénalité overlapping group subsiste. En outre, la pénalité overlapping group $h(\mathbf{w})$ est définie comme $h(\mathbf{w}) = \sum_{l=1}^{L} \upsilon_l \left\|\mathbf{s}_l \circ \mathbf{w}\right\|_2$ avec $\circ$ le produit d'Hadamard, $L$ le nombre total de groupes, $\mathbf{s}_l = (\mathbf{s}_{l,1}, ..., \mathbf{s}_{l,p})^\intercal$ est le vecteur d'appartenance du group $l$ où chaque variable est au moins dans un groupe ou seule dans sons groupe pour équilibrer les pénalisations et $\upsilon_l \geqslant 0$ est un poids associé à chaque groupe et les auteurs préconisent de fixer $\upsilon_l$ à la taille du groupe comme cela est fait dans Yuan and Lin (2006) pour pénaliser par la taille du groupe, sauf pour les variables étant dans plusieurs groupes où dans ce cas la variable de compte que pour $\frac{1}{q}$ dans $\upsilon_l$ où $q$ est le nombre de groupes auxquels elle appartient.

La difficulté avec la pénalisation overlapping group est le problème d'optimisation correspondant. En effet, ce type de pénalisation ne permet pas en général d'avoir une solution directement dérivable ou optimisable. Ainsi, il est nécessaire d'utiliser la méthode Alternating Direction Method of Multipliers (ADMM) pour résoudre ce problème (Boyd et al., 2011) et une optimisation de ce type est extrêmement compliquée comme le soulignent les auteurs : "we discuss how to use ADMM to optimize the weight term, which is critical and a difficult problem since it involves both the l1 norm penalty and overlapping group lasso penalty". De plus, l'ADMM a besoin d'un hyperparamètre supplémentaire, en plus de $K$ et des deux paramètres de pénalisation $\alpha$ et $\lambda$, pour contrôler sa convergence.

En résumé, c'est un algorithme qui offre une grande liberté sur la modélisation de la structure des données en raison de la forme de la pénalité, mais en contrepartie il nécessite une optimisation beaucoup plus complexe. Les solutions sur le choix des hyperparamètres et le critère d'arrêt sont celles qui ont été proposées par Witten and Tibshirani (2010).

STRUCTURED WT-$K$-MEANS AVEC PÉNALISATION PAR LE LAPLACIEN Dans un article récent de Gong et al. (2018), une nouvelle pénalité $h(\mathbf{w})$ est proposée pour accompagner le problème d'optimisation 2.3. La méthode exploite les informations de corrélation entre les variables via la technique du Laplacien sparse, notamment en fixant $h(\mathbf{w}) = \sum_{j,l\in\{1...,p\}} \text{cor}(\mathbf{x}^j, \mathbf{x}^l) \times (w_j - w_l)^2$. Ainsi, le problème peut s'écrire sous la forme d'une double optimisation :

$$\underset{C_1,...,C_K,\mathbf{w}}{\text{maximiser}}\ \mathbf{w}^\intercal\mathbf{b} - \lambda \sum_{j,l\in\{1...,p\}} \text{cor}(\mathbf{x}^j, \mathbf{x}^l) \times (w_j - w_l)^2\ \text{s.c.}\ \left\|\mathbf{w}\right\|_2^2 \leqslant 1, w_j \geqslant 0\ \forall j.$$

Ainsi, pour un $\lambda$ fixé, une forte corrélation entre les variables $\mathbf{x}^j$ et $\mathbf{x}^l$ conduit à une petite différence entre les poids $w_j$ et $w_l$. Par conséquent, si des variables corrélées sont considérées comme importantes pour le clustering, elles auront tendance à être sélectionnées ensemble. De même, si elles ne sont pas importantes pour le clustering, elles auront tendance à être écartées ensemble. Les variables pertinentes sélectionnées par la méthode partagent donc une même structure et les résultats sont plus faciles à interpréter.

[*](On peut regretter le fait que la pénalité ne prenne pas en compte les variables ayant des corrélations négatives en utilisant par exemple $\text{cor}(\mathbf{x}^j, \mathbf{x}^l)^2$)[*]. Mais les auteurs suppose dans l'article que $\text{cor}(\mathbf{x}^j, \mathbf{x}^l) \geqslant 0$ et étant donné que la méthode a été développée pour des données d'expression de gênes et des images, cela ne pose pas nécessairement de problème pour ce type d'application. D'autre part en régression, la pénalité lasso a tendance à ne sélectionner qu'une seule variable dans le groupe de variables corrélées même si un grand nombre de ces variables, voire toutes, font partie du modèle sous-jacent (Bühlmann et al., 2013). En revanche, cette difficulté n'a pas encore été observée en clustering et nous verrons dans les chapitres suivants que le WT-$K$-means a plutôt tendance à produire le phénomène inverse qui est de sélectionner les variables corrélées ensemble (Chapitre 4).

SUBSPACE WT-$K$-MEANS Une dernière extension a été proposée cette année par Diallo et al. (2021). Les auteurs proposent de pondérer la variance inter-classes par variable et par cluster ce qui donne le critère suivant :

$$\underset{\mathbf{w},P_K}{\text{maximiser}} \sum_{k=1}^{K}\sum_{j=1}^{p} w_{k,j} b_{k,j} - \lambda\left\|\mathbf{w}\right\|_1,\ \text{s.c.}\ \left\|\mathbf{w}\right\|_2^2 \leqslant 1, w_{k,j} \geqslant 0\ \forall j,k,$$

où $\mathbf{w}^\intercal = (w_{1,1}, ..., w_{K,p})$ les poids par variable et par cluster. Cette approche est évidemment intéressante car elle apporte une information supplémentaire qui est l'importance des variables par cluster, ce qui permet de

savoir si une variable est discriminante uniquement pour certains clusters. Malheureusement cette formulation a un défaut notable que les auteurs ont omis de mentionner. Imaginons un ensemble de données avec une seule variable centrée qui suit un mélange de trois Gaussiennes tel que les clusters sont séparés, équidistants, de même taille, de même variance. La variable est donc centrée en 0 et son centre se confond avec le milieu d'un des trois clusters, disons le deuxième. Or par définition, $b_{2,1} = 0$ car le centre du cluster se confond avec le centre des données et ainsi $w_{2,1} = 0$. Cet exemple met en évidence le fait que l'algorithme supprime, entre autres, des clusters proches du centre des données. Ce type de comportement n'est pas recherché à priori et peut même être délétère pour le clustering. Pour incorporer l'importance des variables par cluster, il faudrait considérer les clusters deux à deux et non dans leur ensemble.

### 2.4.3 $K$-means sparse par hill climbing (5)

Dans Arias-Castro and Pu (2017), les auteurs proposent ce qu'ils appellent "a simple approach to sparse clustering by hill climbing". Dans les faits, cela revient à utiliser le même schéma que Daniella Witten et Robert Tibshirani dans Witten and Tibshirani (2010), sauf qu'au lieu de proposer un critère pénalisé, les auteurs cherchent un sous-ensemble de variables qui maximise la variance inter-classes à l'aide d'un algorithme itératif de type *hill climbing*. Le hill climbing s'apparente à un algorithme de type stepwise forward-backard en introduisant un paramètre à optimiser, indiquant le nombre de variables à sélectionner. La taille de cet ensemble est choisi à l'aide du Gap-statistic comme dans Witten and Tibshirani (2010).

La grande différence entre les deux méthodes est que pour différents niveaux de pénalisation, le WT-$K$-means peut trouver une même partition et un même sous-ensemble de variables. Au contraire pour que la méthode décrite dans Arias-Castro and Pu (2017) conduise au meilleur modèle il faut que le nombre de variables importantes (pour un vrai modèle sous-jacent) soit présent dans les sous-ensembles testés. Par exemple, si un ensemble de données possède 1000 variables avec 12 variables importantes, il faudra tester l'algorithme avec la taille du sous-ensemble de variables importantes égale à 12, ce qui paraît irréaliste en pratique.

Lors de leurs simulations ! [9](le vrai paramètre (le cardinal du sous-ensemble de variable) apparaît toujours dans la liste des paramètres testés, ce qui donne un clair avantage à l'algorithme testé )[9] ! mais finalement les résultats ne sont pas forcément très différents de ceux obtenus avec le WT-$K$-means. Une analyse du temps de calcul a été faite et l'algorithme semble plus rapide mais encore une fois il faut connaître le bon nombre de variables importantes contrairement au cas du WT-$K$-means qui sélectionne généralement d'avantage de variables sur ces exemples ce qui fausse les résultats.

### 2.4.4 Regularized $K$-means

L'idée principale du Regularized $K$-means est d'étendre l'algorithme $K$-means en ajoutant un terme de pénalité group lasso sur les centres (Sun et al., 2012). Plus précisément, le Regularized $K$-means est formulé comme suit :

$$\underset{C_1,\dots,C_K,\mathbf{C}}{\text{minimiser}} \frac{1}{n}\sum_{k=1}^{K}\sum_{i\in C_k}\sum_{j=1}^{p}(x_i^j - \bar{x}_k^j)^2 + \sum_{j=1}^{p}\lambda_j \left\|\mathbf{c}^j\right\|_2, \tag{2.5}$$

où $\mathbf{c}^j = (\bar{\mathbf{x}}_1^j, ..., \bar{\mathbf{x}}_K^j)^\intercal$ et $\mathbf{C} = [\mathbf{c}^1, ..., \mathbf{c}^p]$ et pour $j = 1, ..., p$ on un terme de régularisation $\lambda_j$ par variable, ce qui est donc l'équivalent du group lasso adaptatif en supervisé (Wang and Leng, 2008). Cette extension du group lasso dit adaptatif a été proposée en raison des problèmes théoriques que posait le *group lasso* (Yuan and Lin, 2006). Ici la version adaptative donne à l'utilisateur la possibilité de pénaliser chaque variable différemment. Néanmoins, l'avantage pratique d'ajouter $p-1$ paramètres à estimer, qui plus est en clustering, est discutable. Les questions de sélection de modèles sont des questions difficiles à résoudre et elles seront abordées en détail dans la suite et il semble plus judicieux de fixer $\lambda_1 = ... = \lambda_p$ pour ce modèle.

Il est impossible de résoudre tel quel le double problème d'optimisation (2.5) où l'on optimiserait par rapport aux centres et la partition itérativement à l'aide de l'algorithme des $K$-means standard. L'astuce est donc algorithmique et elle consiste à réécrire l'algorithme heuristique de Loyd en y modifiant une étape. En effet, la pénalisation s'effectue uniquement par les centres, or si on laisse converger les $K$-means standards entre chaque mise à jour de la pénalisation, on aboutit chaque fois à la même solution ! [10]((en supposant qu'ils trouvent les même centres à chaque itération))[10] !.

Ils montrent dans leurs travaux que l'Équation (2.5) équivaut à

$$\underset{C_1,\dots,C_K,\mathbf{C}}{\text{minimiser}} \sum_{j=1}^{p}\left[\frac{1}{n}(\mathbf{x}^j - \mathbf{Z}\mathbf{c}^j)^\intercal(\mathbf{x}^j - \mathbf{Z}\mathbf{c}^j) + \lambda_j \left\|\mathbf{c}^j\right\|_2\right], \tag{2.6}$$

où $\mathbf{Z} \in \mathbb{R}^{n \times K}$ est la matrice d'appartenance aux clusters. Malheureusement, la solution de l'Équation (2.6) n'est pas donnée dans l'article. On remarque que malheureusement l'Équation (2.6) n'a pas de solution dans le cas général (Hastie et al., 2019). Une solution existe seulement lorsque la matrice d'appartenance aux clusters $\mathbf{Z}$ est orthonormale. La matrice $\mathbf{Z}$ n'est pas orthonormale par définition car elle n'est pas de rang plein et la somme de ses colonnes n'est pas de norme égale à 1, ce qui pose des difficultés lors de la résolution du problème d'optimisation qui ne sont pas discutées dans l'article. Malgré tout, si nous supposons que la solution du problème d'optimisation (2.5) suivant $\mathbf{c}^j$ existe, on peut résoudre ce dernier avec la procédure proposée par Sun et al. (2012) : **Initialisation** : les centres $\mathbf{C}$ sont initialisés à l'aide du $K$-means. Ensuite on répète jusqu'à ce que l'on obtienne convergence ou ce que l'on satisfasse à un critère d'arrêt :

1. **Affectation** : chaque observation est assignée au centre le plus proche.

$$C_k \leftarrow \{i \mid \underset{k'=1,\dots,K}{\operatorname{argmin}} ||\mathbf{x} - \mathbf{c}_{k'}||_2^2 = k\}$$

2. **Minimisation** : chaque centre est mis à jour à l'aide de l'Équation ( **??**).

où $\mathbf{c}_{k'} = (\mathbf{c}_{k'}^1, ..., \mathbf{c}_{k'}^p)^\intercal$. C'est-à-dire que $P_K$ et $\mathbf{C}$ sont mis à jour séparément à chaque itération en supposant que l'autre est fixé. Comme critère d'arrêt, les auteurs proposent "the iteration stops when $\mathbf{Z}$ does not change any more.". Plusieurs choix sont possibles et comme aucun détail n'est donné, on ne peut pas savoir ce que les auteurs ont implémenté. En effet, faire la différence entre des matrices d'appartenance serait une solution atypique. Il serait aussi possible d'utiliser un indice de dissimilarité entre partitions tel quel l'Adjusted Rand Index (ARI) (Hubert and Arabie, 1985).

### 2.4.5 Lasso-Weighted $K$-means

Cette méthode de $K$-means pondérés avec pénalisation lasso développée dans Chakraborty and Das (2019) aussi appelée Lasso-Weighted $K$-means (LW-$K$-means) est différente du WT-$K$-means car elle se base sur le W-$K$-means de Huang et al. (2005) (à noter que le WT-$K$-means est aussi une méthode des $K$-means pondérés avec pénalisation lasso). Le problème d'optimisation correspondant s'écrit comme suit :

$$\underset{C_1,\dots,C_K,\mathbf{w}}{\text{maximiser}} \; g(\mathbf{w}, \alpha, \beta)\mathbf{b} - \lambda \|\mathbf{w}\|_1 \;\text{ s.c. }\; \|\mathbf{w}\|_2^2 \leqslant 1, w_j \geqslant 0 \; \forall j.$$

où $g(\mathbf{w}, \alpha, \beta) = (w_1^\beta + \frac{\alpha}{p^2}|w_1|, ..., w_p^\beta + \frac{\alpha}{p^2}|w_p|)^\intercal$ est une fonction de poids bénéficiant de deux nouveaux paramètres $\alpha$ et $\beta$.

Cette nouvelle méthode bénéficie des bons résultats du W-$K$-means tout en proposant une solution sparse. De plus, l'article s'appuie sur la preuve de consistance du W-$K$-means pour prouver celle du LW-$K$-means. La grande différence se trouve dans la réécriture du poids associé à une variable, ce qui implique que la fonction objectif associe deux nouveaux hyperparamètres à optimiser. Ces deux paramètres sont fixés de manière arbitraire et les auteurs ne discutent pas leur influence. Pour le paramètre de régularisation lasso $\lambda$, les auteurs indiquent "The value of $\lambda$ was chosen by performing some hand-tuned experiments.".

Les auteurs concluent sur des données simulées que les performances de leur algorithme sont proches de celles du WT-$K$-means en terme de partitionnement, mais que le WT-$K$-means a tendance à sélectionner un grand nombre de variables de bruit, en leur attribuant un poids très faible mais non nul. De plus, ils insistent sur le fait que le WT-$K$-means est plus lent et que cela est dû à l'utilisation du Gap Statistic qui nécessite de multiples exécutions de l'algorithme.

Les auteurs du LW-$K$-means Chakraborty and Das (2019), en collaboration avec deux autres auteurs ont proposé l'Entropy Weighted Power $K$-means (EWP-$K$-means) (Chakraborty et al., 2020). Ce travail est différent des autres tant par sa méthode d'optimisation que par son étape d'affectation. En effet, l'algorithme emploie un schéma de maximisation-minimisation (Lange, 2016) utilisant le Power $K$-means (Xu and Lange, 2019) ce qui est donc très différent des algorithmes précédents. Le Power $K$-means est un algorithme de clustering très récent se basant sur la maximisation-minimisation mais il a déjà reçu beaucoup d'attention notamment grâce à ses bonnes performances, son faible temps de calcul et sa capacité à éviter les minima locaux. À différents niveaux il semble plus compétitif que la version Loyd de $K$-means. Les auteurs de Chakraborty et al. (2020) ont donc voulu l'étendre en proposant une version pondérée : l'EWP-$K$-means. Il faut noter que cette version ne permet pas de faire de la sélection de variables. Les résultats de leurs simulations sont remarquables et ils semblent montrer la supériorité de leur méthode face au WT-$K$-means. Malheureusement, encore une fois, les auteurs font l'impasse sur le choix du paramètre $\lambda$ ce qui donne une vision incomplète des simulations. En effet, l'EWP-$K$-means utilise trois paramètres en plus de $K$. Deux d'entre eux sont fixés : "we fix them at $s_0 = -1$ and $\eta = 1.05$ across all real and simulated settings considered in this paper". Aucune

étude ou détail spécifique n'est donné. Pour le paramètre $\lambda$, c'est encore plus problématique car ils déclarent "We require the tuning parameter $\lambda > 0$ to be specified, typically chosen via cross-validation detailed in Section 4.1." mais il n'est plus mentionné nulle part ailleurs dans la suite de l'article.

### 2.4.6 Résumé des travaux sur les $K$-means sparse

À partir de ce parcours de la littérature sur le $K$-means sparse, quatre points intéressants peuvent être notés.

1. Performances : nombreux sont les articles comparant les performances du WT-$K$-means en termes de clustering sur des données simulées, en calculant l'ARI entre la vraie partition et celle trouvée par l'algorithme par exemple. Sur ce point, les performances sont souvent similaires et à part des cas très atypiques, il n'y a pas de schéma précis ou de type de données (par exemple des images ou des données très corrélées) pour lesquelles le WT-$K$-means se comporterait moins bien que les autres algorithmes.

2. Sélection de variables : les articles développant des méthodes de sélection de variables s'intéressent souvent à des méthodes interprétables, un des objectifs étant de sélectionner uniquement les variables importantes et toutes les variables importantes, c'est-à-dire celles qui sont liées au vrai clustering sous-jacent. Les travaux soulignent que le WT-$K$-means sélectionne généralement trop de variables, attribuant un poids faible mais non nul à certaines variables de bruit. Il nous semble que cela est majoritairement dû au choix du paramètre de pénalisation $\lambda$. Nos expériences confirment dans la suite cette assertion.

3. Temps de calcul : certains articles comparent le temps de calcul du WT-$K$-means avec ceux des autres algorithmes, et ils montrent que celui-ci est globalement plus lent. Là encore, en nous fondant sur plusieurs études, il nous semble que c'est dû à l'usage du Gap Statistic pour choisir le paramètre $\lambda$. Ce dernier requiert de nombreuses exécutions de l'algorithme pour trouver le $\lambda$ optimal.

4. Le choix du $\lambda$ optimal : la version originale (Witten and Tibshirani, 2010) et quasiment toutes les extensions et travaux basés dérivés (à l'exception de Sun et al. (2012) ; Chakraborty and Das (2019)) utilisent le Gap Statistic pour déterminer le paramètre de pénalisation optimal $\lambda$. Comme on l'a remarqué précédemment, cette méthode est coûteuse en temps de calcul et elle semble mettre l'accent sur les performances plutôt que sur l'interprétabilité. De plus Daniella Witten et Robert Tibshirani eux-mêmes portent un avis critique en affirmant que "the performance of the gap statistic of Section 3.2 for selecting the tuning parameter is mixed. This is not surprising, since tuning parameter selection in the unsupervised setting is known to be a very difficult problem. This is an area in which more work is needed.". Nous tenterons d'expliquer plus en détail par la suite les difficultés et les possibles solutions de ce problème.

## 2.5 Méthodes sparses basées sur les GMM

Avant de discuter des différentes méthodes de GMM sparses, nous allons faire quelques rappels sur les GMM en clustering.

### 2.5.1 Les modèles de mélanges Gaussien

DÉFINITIONS Pour le clustering basé sur les modèles de Fraley and Raftery (2002), on considère que les données proviennent d'une population distribuée comme un mélange de groupes et que chaque composante de ce mélange est modélisée par sa distribution de probabilité conditionnelle. Dans ce contexte, les observations $\mathbf{X}$ sont supposées être des réalisations indépendantes d'un vecteur aléatoire de dimension $X \in \mathbb{R}^p$. On considère une famille de $K$ densités paramétriques $p(.,\theta)$ où $\theta$ est le paramètre définissant la distribution de $X$. Si on définit des a des poids positifs $\pi_1, ..., \pi_K$ tels que $\sum_{k=1}^{K} \pi_k = 1$ qui sont les proportions de mélange, alors un modèle de mélange Gaussien est défini par :

$$p(\mathbf{x}_i, \theta) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_i | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k),$$

où $\mathcal{N}(\mathbf{x}_i | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ est une densité multivariée Gaussienne de moyenne $\mu_k$ et de matrice de covariance $\Sigma_k$. Par la suite, nous appelerons GMM ce type de modèle (pour Gaussian mixture models) Il existe des modèles de mélanges non Gaussien et on introduire des fonctions de densités paramétriques différentes pour

un même mélange, mais ils ne seront pas utilisés dans ce manuscrit. Pour un ensemble d'observations $\mathbf{X}$, la log-vraisemblance de ce modèle de mélange est alors :

$$\mathcal{L}(\theta) = \sum_{i=1}^{n} \log\Big( \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_i | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \Big).$$

Cependant, l'estimation de ce modèle ne peut se faire directement par la maximisation de la vraisemblance puisque les classes des observations sont inconnues. Une solution pour optimiser cette équation repose sur un algorithme itératif. Bien qu'il ne soit pas spécifiquement dédié aux modèles de mélange, l'algorithme EM (Expectation-Maximization), proposé par Dempster et al. (1977) est certainement la technique la plus populaire pour estimer des modèles de mélange et son fonctionnement repose sur la vraisemblance des données complétées. Notons $\mathbf{Z} \in \{0,1\}^{n \times K}$ la matrice indicatrice d'appartenance aux clusters dont les variables sont définies comme $\mathbf{z}^k = (z_1^k, ..., z_i^K)^\intercal$ avec $z_i^k = 1$ si $i \in C_k$ et 0 sinon, pour tout $k = 1, ..., K$ que l'on peut aussi écrire sous sa forme vectorielle sans indice $\mathbf{z} = (\mathbf{z}^1, ..., \mathbf{z}^K)$. Ainsi, l'algorithme EM maximise itérativement l'espérance conditionnelle de la log-vraisemblance des données complétées :

$$\mathcal{L}(\theta, \mathbf{z})_c = \sum_{k=1}^{K} \sum_{i=1}^{n} z_i^k \log\Big( \pi_k \mathcal{N}(\mathbf{x}_i | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \Big).$$

Différentes formes des modèles de mélange Généralement lorsque l'on parle des modèles de mélange Gaussien (GMM), on fait le plus souvent référence à un modèle avec de nombreux paramètres notamment car le modèle autorise les les matrices de covariance $\Sigma_k$ à dépendre du cluster. Comme le font remarquer Bouveyron and Brunet-Saumard (2014b), ce type de modèle avec $p = 100$ et $K = 4$ requiert l'estimation de 20603 paramètres alors qu'un $K$-means ne nécessitera l'estimation que de 400 paramètres, soit les moyennes des 4 clusters sur les 100 variables.

Il est possible de mettre des contraintes sur les matrice de covariance, notamment en la supposant égale dans chaque cluster $\forall k, k' = 1, ..., K, \boldsymbol{\Sigma}_{k'} = \boldsymbol{\Sigma}_k \neq \boldsymbol{\Sigma}_X$ avec $\boldsymbol{\Sigma}_X$ la matrice de covariance de $X$ qui est forcément différente de celles des clusters.

Dans cet esprit, Banfield and Raftery (1993) ainsi que Celeux and Govaert (1995) ont proposé, presque simultanément, une forme de modèles de mélange Gaussien. Pour ce faire, on commence à décomposer en valeurs propres les matrices de covariance :

$$\boldsymbol{\Sigma}_k = s_k \mathbf{V}_k \mathbf{A}_k \mathbf{V}_k^\intercal, \tag{2.7}$$

où $\mathbf{V}_k \in \mathbb{R}^{p \times p}$ est la matrice des vecteurs propres qui donnent l'orientation du cluster $k$, $\mathbf{A}_k \in \mathbb{R}^{p \times p}$ est une matrice diagonale proportionnelle aux valeurs propres qui donnent la forme du cluster et $s_k \in \mathbb{R}$ donne le volume du cluster $k$. Ainsi, on peut fixer $\mathbf{V}_k = \mathbf{V}, \forall k$ pour fixer l'orientation par cluster, et la forme et le volume peuvent être aussi fixés de la même manière. Différentes combinaisons sont proposées pour obtenir finalement 14 formes de modèles différents.

Bayesian Information Criterion le problème du choix entre plusieurs modèles est particulièrement bien posé dans le paradigme bayésien. Le cœur de la modélisation statistique bayésienne est l'appréhension de l'incertitude par le biais de distributions préalables sur les paramètres $p(\theta)$, maintenant traités comme des variables latentes.

Comme nous l'avons déjà dit brièvement dans ce chapitre, de nombreux critères ont été proposés pour choisir entre différents modèles dans le cadre non-supervisé. Parmi ces modèles, les critères bayésiens qui choisissent le modèle pour lequel la probabilité des observations est la plus grande sont largement utilisés dans le cas des GMM. Le critère Bayesian Information Criterion (BIC) (Schwarz, 1978) est certainement l'un des plus connus et des plus utilisés. Le critère BIC est constitué de deux termes : le terme de vraisemblance qui favorise la sélection d'un modèle complexe et un terme de pénalité, fonction croissante du nombre de paramètres, qui favorise la sélection d'un modèle parcimonieux. Le principe du critère BIC est donc de choisir le modèle qui minimise la quantité suivante :

$$\text{BIC}(\mathcal{M}) = -2\mathcal{L} + \eta(m)\log(n)$$

où $\eta(\mathcal{M})$ est le nombre de paramètres du modèle $\mathcal{M}$.

En pratique, le critère BIC est à préférer au critère Akaike Information Criterion (AIC) qui ne pénalise pas suffisamment la complexité des modèles et a tendance à surestimer le nombre de paramètres à estimer.

Dans la suite de cette section, nous présentons les méthodes de clustering sparses basées sur les GMM. D'une part, des critères de clustering pénalisés ont été proposés pour traiter le problème de la sélection des

variables dans le clustering. Dans le contexte des GMM, plusieurs travaux ont introduit un terme de pénalité dans la fonction de log-vraisemblance afin d'obtenir la sparsité. D'autre part, certains auteurs abordent le problème de la sélection des variables en clustering comme un problème de sélection de modèle en déterminant le rôle de chaque variable.

### 2.5.2 Méthodes par pénalisation de la vraisemblance

FORME GÉNÉRALE ! [9](Il s'avère que les modèles de mélanges en clustering (Section **??**) se prête facilement à la sélection de variables)[9] !. Plutôt que de chercher $\boldsymbol{\mu}_k$ et $\boldsymbol{\Sigma}_k$ qui maximisent la log-vraisemblance, la log-vraisemblance pénalisée peut être maximisée, où la pénalité est choisie pour obtenir une sélection de variables plus ou moins sparse. Une forme générale de la fonction de log-vraisemblance pénalisée est la suivante :

$$\mathcal{L}_h(\theta) = \mathcal{L}(\theta) - h(\theta)$$

où $\mathcal{L}(\theta)$ représente la fonction de log-vraisemblance et $h(\theta)$ est la fonction de pénalité (pénalité qui est diminue avec le nombre de variables du modèle). Les modèles suivants se caractérisent par le choix de la fonction de pénalité $h$.

PÉNALISATION PAR LES CENTRES En supposant que les données sont centrées et que le modèle de mélange est Gaussien avec des matrices de covariance diagonales communes, Pan and Shen (2007) proposent la pénalité suivante :

$$h(\theta) = \lambda \sum_{k=1}^{K} \sum_{j=1}^{p} |\mu_{k,j}|$$

où $\mu_{k,j}$ désigne la moyenne de la $j$-ième variable dans le cluster $k$ et $\lambda$ un hyperparamètre qui représente le niveau de sparsité souhaité. $\lambda$ est sélectionné par un critère BIC modifié qui prend en compte le niveau de sparsité dans le terme de ! [11](complexité du modèle)[11] !. Lorsque le paramètre $\lambda$ est grand, certains des éléments $\mu_{k,j}$ sont exactement égaux à zéro. Si, pour une certaine variable $j$, et $\forall k = 1, ..., K, \mu_{k,j} = 0$, alors la variable $j$ est considérée comme exclue du modèle final, car elle ne permet en aucune façon de discriminer les clusters. Il faut noter qu'il est nécessaire que les variables soient centrées et de variance unitaire, comme l'indiquent les auteurs. Cette méthode est similaire dans sa formulation au Regularized $K$-means car la pénalité est fonction des centres. On peut remarquer que Wang and Zhu (2008) ont étendu la pénalisation à la norme infini, ce qui a pour conséquence de discriminer tous les centres d'une même variable ensemble. Cela a l'avantage d'imposer une sélection de variables globalement importantes, mais en même temps une information précieuse est perdue : une variable peut être importante que sur certaines composantes du mélange, certains clusters.

PÉNALISATION PAR LES CENTRES ET LES VARIANCES Xie et al. (2008) ont étendu le modèle de Pan and Shen (2007) en relaxant la contrainte d'égalité sur les matrices de covariance. En effet, ils considèrent le cas des matrices de covariance diagonales, mais différentes par clusters. En posant $\forall k \in \{1, ..., K\}$, $\boldsymbol{\Sigma}_k = \text{diag}(\sigma^2_{k,1}, ..., \sigma^2_{k,p})$ cela conduit à la fonction de pénalité suivante :

$$h(\theta) = \lambda \sum_{k=1}^{K} \sum_{j=1}^{p} |\mu_{k,j}| + \gamma \sum_{k=1}^{K} \sum_{j=1}^{p} |\sigma^2_{k,j} - 1|.$$

Dans ce cas, un second terme régularisé est ajouté et porte sur la variance de la variable $j$ de la $k$-ième composante $\sigma^2_{k,j}$ qui peut être réduit à 0. Comme précédemment, les hyperparamètres $\lambda$ et $\gamma$ sont sélectionnés par un critère BIC modifié. De plus, il est là encore indispensable de normaliser les variables à moyenne 0 et variance 1. La justification de la normalisation est triviale lorsque l'on observe la forme de la deuxième partie de la pénalité : $\sigma^2_{k,j} - 1$, où le 1 devrait être remplacé par $\sigma^2_j$ dans le cas où $\sigma_j \neq \sigma_l, \forall j, l$.

PÉNALISATION PAR LES CENTRES ET LES COVARIANCES Enfin quelques autres versions, peut-être moins connues, ont été proposées mais nous n'en introduirons qu'une seule qui a été récemment utilisée dans un travail qui nous sera utile par la suite. Cette pénalisation est la plus générale qui soit et est énoncée dans le travail de Zhou et al. (2009) où des hypothèses de moins en moins fortes sont imposées à la matrice de covariance. En effet, la pénalité est écrite comme suit :

$$h(\theta) = \lambda \sum_{k=1}^{K} \sum_{j=1}^{p} |\mu_{k,j}| + \gamma \sum_{k=1}^{K} \sum_{j=1}^{p} \sum_{l \neq j} |(\boldsymbol{\Sigma}_k^{-1})_{j,l}| \tag{2.8}$$

où $(\Sigma_k^{-1})_{j,l}$ est l'élément de la $j$-ième colonne et de la $l$-ième ligne de l'inverse de la matrice de covariance du cluster $k$. En particulier , l'algorithme *graphical lasso* Friedman et al. (2008), qui utilise une procédure de *coordinate descent* pour le lasso, est utilisé pour estimer la matrice inverse sparse $\mathbf{\Sigma}_k^{-1}$ pour tout $k = 1, ..., K$. Malheureusement l'estimation de telles matrices sur des clusters en grande dimension avec peu d'observations peut s'avérer être une tâche ardue cite.

### 2.5.3 Méthodes par pénalisation des loadings (6)

Dans cette section sur les méthodes GMM pénalisées, introduisons une dernière méthode sparse qui se base sur l'algorithme Fisher-EM (Bouveyron and Brunet, 2012). Charles Bouveyron et Camille Brunet proposent une famille de modèles de mélange qui ajustent les données dans un sous-espace discriminant commun. Ce modèle de mélange, appelé Discriminative Latent Mixture (DLM), se caractérise par un sous-espace latent commun à tous les groupes et est supposé être le sous-espace le plus discriminant pour une dimension fixée, en maximisant la séparation entre les groupes. L'inférence des modèles DLM ne peut pas se faire avec l'algorithme EM en raison des caractéristiques spécifiques de son sous-espace latent. Pour surmonter ce problème, Bouveyron and Brunet (2012) ont proposé un algorithme, nommé Fisher-EM, pour estimer à la fois le sous-espace discriminant et les paramètres du modèle de mélange. Cet algorithme est basé sur l'algorithme EM auquel une étape supplémentaire est ajoutée entre l'étape E et l'étape M. Cette étape supplémentaire, appelée étape F, vise à calculer la matrice de projection $\mathbf{U}$ dont les colonnes couvrent l'espace latent discriminant et cela revient à chercher, à chaque itération, une estimation $\hat{\mathbf{U}}$ telle que :

$$\hat{\mathbf{U}} = \underset{\mathbf{U}}{\text{maximiser}}\ \text{trace}\left(\mathbf{U}^\intercal \hat{\mathbf{\Sigma}}_\mathbf{X} \mathbf{U}\right)^{-1} \left(\mathbf{U}^\intercal \hat{\mathbf{\Sigma}}_\mathbf{b} \mathbf{U}\right) \text{ s.c. } \mathbf{U}^\intercal \mathbf{U} = \mathbf{I}_d \tag{2.9}$$

où $\hat{\mathbf{\Sigma}}_\mathbf{X}$ est la matrice de covariance des données $\mathbf{X}$ (estimée) et $\hat{\mathbf{\Sigma}}_\mathbf{b}$ est la matrice de covariance inter-classes estimée à partir des clusters et donc différentes à chaque itération si la partition change et $d \leqslant K - 1$ la dimension de l'espace latent.

Dans le contexte de la sélection de variables, les auteurs proposent une extension de l'algorithme Fisher-EM pour réduire le nombre de variables dans l'espace latent discriminant (Bouveyron and Brunet-Saumard, 2014a). Ainsi, ils décident de pénaliser directement la matrice de projection $\mathbf{U}$, aussi appelée matrice des *loadings*. Pour réaliser la pénalisation de $\mathbf{U}$, trois solutions sont proposées dans Bouveyron and Brunet-Saumard (2014a). Les trois solutions ont un point commun, elles modifient l'étape F de l'algorithme Fisher-EM pour introduire de la sparsité dans $\mathbf{U}$.

Dans la première approche, la matrice $\hat{\mathbf{U}}$ est approximée par une matrice sparse à l'aide d'une régression pénalisée qui peut être calculée par l'algorithme Least-Angle Regression (Efron et al., 2004).

Dans la deuxième approche, le critère de Fisher utilisé dans l'étape F de l'algorithme Fisher-EM est reformulé comme un problème de régression qui peut être résolu en effectuant une décomposition en valeurs singulières (SVD) de la matrice de covariance inter-classes. Une approximation de la solution de ce problème de régression est obtenue en effectuant une SVD pénalisée (Witten et al., 2009).

Pour la dernière approche, les auteurs reformulent la maximisation du critère de Fisher comme un problème de régression et ils obtiennent une matrice loadings sparse en résolvant le problème lasso associé à ce problème de régression. Cependant, la résolution de ce problème lasso n'est pas directe dans ce cas et nécessite l'utilisation d'un algorithme itératif.

En ce qui concerne les détails de l'implémentation, Bouveyron and Brunet-Saumard (2014a) ont proposé d'initialiser l'algorithme Sparse Fisher-EM avec le résultat de l'algorithme Fisher-EM et de déterminer la valeur des deux paramètres de pénalisation à l'aide du critère BIC, pour lequel la complexité du modèle dépend du nombre de valeurs non nulles clair ?.

L'un des grands avantages des algorithmes Fisher-EM et Sparse Fisher-EM est qu'ils permettent de visualiser les classes dans l'espace latent. En effet, offrir un outil de visualisation conjointement à celui du clustering est très intéressant, qui plus est en clustering car la vérification visuelle des classes obtenues est encore à ce jour un des seuls moyens de s'assurer de la qualité de la partition obtenue. Ce point sera abordée plus en détail dans la Section 2.8.

Nous faisons remarquer que nous voulions aussi bénéficier de cet outil de visualisation en développant l'équivalent du Sparse Fisher-EM mais en se basant sur les $K$-means. Pour cela, le schéma de l'algorithme LDA-$K$-means aurait été repris (Ding and Li, 2007). Cet algorithme itère entre deux étapes, où l'espace latent est celui défini par les axes discriminants d'une LDA qui discrimine les clusters qui sont fixés, puis les clusters sont trouvées à l'aide des $K$-means opérant dans l'espace latent fixé, trouvé par ces axes. L'étape de modélisation de la LDA revient à maximiser le même critère 2.9 pour trouver les axes $\mathbf{U}$ en remplaçant la matrice de covariance des données par la matrice de covariance intra-classes au dénominateur $\hat{\mathbf{\Sigma}}_\mathbf{w}$, mais garder $\hat{\mathbf{\Sigma}}_\mathbf{X}$ au dénominateur est une alternative possible (et le problème est mieux posé) Wang et al. (2019). Ensuite

différentes possibilités s'offrent à nous : soit pénaliser les axes discriminant suivant la méthode de Trendafilov and Jolliffe (2007) mais donc qui ne considère pas le cas $n < p$, soit considérer le cas $n < p$ en régularisant la matrice de covariance $\hat{\mathbf{\Sigma}}_{\mathbf{X}}$ ou en utilisant la pseudo inverse de Moore-Penrose pour la rendre inversible (Bickel and Levina, 2004) mais dans ce cas là il n'y a pas de sélection de variables, soit en considérant le problème comme un problème d'*optimal scoring* pénalisé avec l'algorithme de Clemmensen et al. (2011) mais cet algorithme est plus compliqué et coûteux en temps de calcul, soit utilisé la version pénalisé du problème de Fisher (Witten and Tibshirani, 2011) pour obtenir des axes discriminant sparse mais dans ce modèle fait l'hypothèse que $\hat{\mathbf{\Sigma}}_{\mathbf{w}}$ est diagonale. Aucune solution n'était donc complètement satisfaisante.

Parler du cas n<p

### 2.5.4 Sélection de variables via la sélection de modèles

L'idée sous-jacente des modèles présentés dans cette section est de déterminer pour chaque variable son rôle dans le clustering. Ainsi, une variable peut être déclarée comme importante, ou plus exactement dépendante du clustering, ou indépendante (Raftery and Dean, 2006). Certains travaux vont même plus loin (Maugis et al., 2009a,b) en décrivant deux sous-ensembles de variables non-importantes : les variables redondantes et les variables totalement indépendantes du clustering.

L'APPROCHE DE RAFTERY ET DEAN Les auteurs définissent la matrice des données $\mathbf{X}$ comme étant partitionnée en trois parties :

- $\mathbf{X}^C$ les variables importantes pour le clustering.
- $\mathbf{X}^P$ les variables à tester pour le clustering. Le modèle devra définir si elles sont importantes ou s'il vaut mieux les supprimer.
- $\mathbf{X}^N$ les variables indépendantes du clustering, non importantes ou aussi appelées variables de bruit. Les variables non importantes sont définies comme indépendantes du clustering mais dépendantes des variables importantes selon une relation linéaire, relation qui est explicitée ci-dessous.

Ensuite, la décision d'inclusion ou d'exclusion de $\mathbf{X}^P$ est prise en comparant les modèles :

$$\mathcal{M}_1 : p(\mathbf{X}|\mathbf{z}) = p(\mathbf{X}^C, \mathbf{X}^P|\mathbf{z})p(\mathbf{X}^{NC}|\mathbf{X}^C, \mathbf{X}^P),$$
$$\mathcal{M}_2 : p(\mathbf{X}|\mathbf{z}) = p(\mathbf{X}^C|\mathbf{z})p(\mathbf{X}^C|\mathbf{X}^P)p(\mathbf{X}^{NC}|\mathbf{X}^C, \mathbf{X}^P).$$

Dans le modèle $\mathcal{M}_1$, $\mathbf{X}^P$ est utile pour le regroupement et la distribution conjointe $p(\mathbf{X}^C, \mathbf{X}^P|\mathbf{z})$ correspond à une distribution de mélange Gaussien ; d'autre part, le modèle $\mathcal{M}_2$ indique que $\mathbf{X}^P$ ne dépend pas du clustering $\mathbf{z}$ et la distribution conditionnelle $p(\mathbf{X}^P|\mathbf{X}^C)$ correspond à une régression linéaire. Une caractéristique importante de la formulation est que, dans $\mathcal{M}_2$, il n'est pas nécessaire que les variables non importantes d'être indépendantes des variables de clustering. Cela permet d'écarter les variables redondantes redondantes liées aux variables de clustering mais pas au clustering lui-même. Encore une fois, nous ne partageons pas ce point de vue dans ce manuscrit. Il est pour nous difficile d'imaginer deux variables identiques, dont l'une serait importantes et l'autre considérée comme non importante et redondante.

Les modèles en compétition sont comparés en utilisant l'approximation BIC de leurs vraisemblances marginales :

$$\mathrm{BIC}_1 = \mathrm{BIC}_{clust}(\mathbf{X}^C, \mathbf{X}^P),$$
$$\mathrm{BIC}_2 = \mathrm{BIC}_{noclust}(\mathbf{X}^C) + \mathrm{BIC}_{reg}(\mathbf{X}^P|\mathbf{X}^C),$$

! [12](où $\mathrm{BIC}_{clust}(\mathbf{X}^C, \mathbf{X}^P)$ est le BIC d'un GMM où les variables de $\mathbf{X}^P$ apparaissent comme des variables supplémentaires importantes pour le modèle, $\mathrm{BIC}_{noclust}(\mathbf{X}^C)$ est le BIC d'un GMM pour les variables déjà définies comme importantes et $\mathrm{BIC}_{reg}(\mathbf{X}^P|\mathbf{X}^C)$ est le BIC d'un modèle de régression des variables $\mathbf{X}^P$ sur $\mathbf{X}^C$. Ensuite si $\mathrm{BIC}_1 - \mathrm{BIC}_2 > 0$ alors $\mathbf{X}^P$ est ajoutée à $\mathbf{X}^C$. )[12] !

Comme décrit par Raftery and Dean (2006), la mise en œuvre pratique de la méthodologie ci-dessus nécessite de pouvoir vérifier l'inclusion et l'exclusion des variables. Deux méthodologies sont proposées : un algorithme de recherche stepwise forward-backward, et un algorithme *headlong*. À chaque étape d'inclusion, l'algorithme stepwise forward-backward considère tour à tour chaque variable qui ne fait pas partie des variables déjà sélectionnées et évalue s'il y est pertinent de l'ajouter. De même, à chaque étape d'exclusion, on supprime la variable pour laquelle la probabilité de la suppression estimée par la méthode est la plus élevée. L'algorithme stepwise forward-backward peut être mis en œuvre en commençant par l'ensemble vide ou en commençant par l'ensemble complet.

D'autre part l'algorithme de recherche headlong ne vérifie pas toutes les variables une par une. L'algorithme parcourt les variables jusqu'à ce qu'il trouve une variable pour laquelle la probabilité d'inclusion ou de l'exclusion dépasse un seuil prédéfini. L'algorithme headlong est moins coûteux en temps de calcul que l'algorithme stepwise forward-backward, mais la recherche est non-exhaustive et elle donne lieu à de moins bonnes solutions.

EXTENSION DU MODÈLE L'hypothèse d'indépendance des groupes est discutable. En effet, d'une part, considérer uniquement le cas où les variables non importantes sont indépendantes à la fois du clustering et des variables importantes, comme cela a été considéré dans le travail de Law et al. (2004), semble irréaliste, selon certains auteurs (ci-après ; ce qui n'est pas notre cas). D'autre part, considérer que les variables non importantes dépendent des variables importantes par une relation linéaire est une hypothèse forte qui peut ne pas être valide dans certains cas pratiques (dans le cas où l'on considère que des variables peuvent être dépendantes de variables importantes et être considérées comme non importantes). Concrètement, $\mathbf{X}^P$ ne peut être lié qu'à un sous-ensemble $\mathbf{X}^R \subseteq \mathbf{X}^C$ des variables importantes. De cette manière, les auteurs évitent l'inclusion de paramètres inutiles qui pénaliseraient trop la log-vraisemblance. Une autre limitation de la procédure de Raftery et Dean est liée à leur algorithme de sélection des variables de type forward stepwise. Ce type de procédure a des défauts bien connus, comme celui de construire des modèles emboîtés, or les modèles plus petits ne sont pas forcément des sous-ensembles des modèles plus grands (Judd et al., 2011). Pour surmonter ces limitations, Maugis et al. (2009a) relâchent les hypothèses en proposant une nouvelle formulation du modèle :

$$\mathcal{M}_3 : p(\mathbf{X}|\mathbf{z}) = p(\mathbf{X}^C|\mathbf{z})p(\mathbf{X}^C|\mathbf{X}^R \subseteq \mathbf{X}^C)p(\mathbf{X}^{NC}|\mathbf{X}^C, \mathbf{X}^P)$$

avec $p(\mathbf{X}^C|\mathbf{X}^R \subseteq \mathbf{X}^C)$ le terme de régression de $\mathbf{X}^P$ sur les variables importantes $\mathbf{X}^R$. Pour ce modèle, le BIC est donné par :

$$\text{BIC}_3 = \text{BIC}_{noclust}(\mathbf{X}^C) + \text{BIC}_{reg}(\mathbf{X}^P|\mathbf{X}^R \subseteq \mathbf{X}^C),$$

avec $\text{BIC}_{reg}(\mathbf{X}^P|\mathbf{X}^R \subseteq \mathbf{X}^C)$ le BIC de la régression de $\mathbf{X}^P$ sur $\mathbf{X}^R$ après la sélection de l'ensemble optimal de prédicteurs $\mathbf{X}^P$. En outre, si $\text{BIC}_1 - \text{BIC}_3 > 0$, $\mathbf{X}^P$ peut être ajoutée. Le modèle offre plus de possibilités lors de la modélisation mais un problème apparaît : $\mathbf{X}^R$ doit également être déterminée par un algorithme stepwise.

MODÈLE SRUW (7) Les auteurs vont encore plus loin dans Maugis et al. (2009b) en considérant un quatrième modèle où des variables peuvent être complètement indépendantes des variables importantes et cela se traduit par :

$$\mathcal{M}_4 : p(\mathbf{X}|\mathbf{z}) = p(\mathbf{X}^C|\mathbf{z})p(\mathbf{X}^P)p(\mathbf{X}^{NC}|\mathbf{X}^C, \mathbf{X}^P)$$

avec $p(\mathbf{X}^P)$ la densité d'une distribution Gaussienne. Alors, le BIC correspondant est donné par :

$$\text{BIC}_4 = \text{BIC}_{noclust}(\mathbf{X}^C) + \text{BIC}_{reg}(\mathbf{X}^P|\tilde{\mathbf{X}}^R \subseteq \mathbf{X}^C).$$

Cela correspond au modèle $\text{BIC}_3$ où le modèle de régression avec $\mathbf{X}^R = \emptyset$ est autorisé et donc le but est de résoudre le problème $\text{BIC}_1 - \text{BIC}_4 > 0$ en cherchant à chaque étape les variables à inclure ou exclure avec une procédure stepwise en autorisant toutes les variables de bruit à être indépendantes des variables importantes. Une procédure de type stepwise backward est mis en oeuvre pour le modèle de clustering, plus efficace que la version forward, mais aussi beaucoup plus lente car elle requiert de multiples évaluations de modèles incluant la quasi-totalité des variables, ce qui devient impossible pour des ensembles de données à partir de quelques dizaines de variables (Bouveyron et al., 2019).

EXTENSION DU MODÈLE SRUW (8) Les procédures de sélection de variables avec une, deux ou plus de procédures stepwise intégrées restent coûteuses en temps de calcul et des procédures alternatives sont souhaitables. Ainsi, Gilles Celeux, Cathy Maugis-Rabusseau et Mohammed Sedki proposent dans un travail récent (Celeux et al., 2019) de combiner le modèle SRUW avec la méthode de Zhou et al. (2009). La procédure stepwise est remplacée par un classement des variables à l'aide du modèle décrit par l'Équation (2.8). À proprement parler, le modèle avec pénalité lasso ne donne pas de classement des variables. Celeux et al. (2019) proposent de compter pour chaque valeur de $\lambda$ et de $\gamma$ le nombre de fois où une variable est sélectionnée, c'est-à-dire le nombre de fois où au moins une moyenne de cluster est différent de 0.

Formellement, les paramètres de régularisation $(\lambda, \gamma)$ varient dans une grille de paramètres notée $\mathcal{G}_\lambda \times \mathcal{G}_\gamma$ et l'importance est définie pour chaque variable $j \in \{1, ..., p\}$ et pour un $K$ fixé tel que :

$$\mathcal{O}_K(j) = \sum_{(\lambda,\gamma)\in\mathcal{G}_\lambda\times\mathcal{G}_\gamma} \mathcal{I}_{(K,\lambda,\gamma)}(j) \tag{2.10}$$

où

$$\mathcal{I}_{(K,\lambda,\gamma)}(j) = \begin{cases} 0 \text{ si } \forall k = 1, ..., K, \mu_{k,j} = 0, \\ 1 \text{ sinon.} \end{cases}$$

Plus $\mathcal{O}_K(j)$ est grand, plus la variable $j$ est censée être liée au clustering et donc importante. Les variables sont ensuite classées par valeurs décroissantes sur $\mathcal{O}_K(j)$. Il est écrit dans Bouveyron et al. (2019) que "It is hoped that using this lasso-like ranking of the variables instead of stepwise algorithms would not degrade the identification of the sets S, R, U and W". Même s'il n'est pas certain que cette méthodologie soit plus performante, elle reste néanmoins plus rapide.

Deux autres défauts de cette méthode, non soulignés dans la littérature peuvent être relevés. Premièrement, pour différentes valeurs de $(\lambda, \gamma)$, des clustering complètement différents peuvent être obtenus. Par exemple, un groupe de variables $\mathbf{X}^1$ peut être considéré comme important par l'algorithme pour des petites valeurs de $\lambda$ alors qu'un deuxième groupe $\mathbf{X}^2$ indépendant du premier peut être considéré comme important par l'algorithme pour des grandes valeurs de $\lambda$. Or, ici, l'importance d'une variable est définie sur l'ensemble de la grille de valeur. En d'autres termes, pour certaines valeurs de $(\lambda, \gamma)$, des modèles très bruités peuvent être obtenus et peuvent modifier l'estimation de $\mathcal{O}_K$. Dissocier la mesure d'importance de la partition à laquelle elle se rattache est risqué. Deuxièmement, l'importance d'une variable définie par l'Équation (2.10) dépend de l'échelle de la grille de $(\lambda, \gamma)$. L'échelle de la grille influence les résultats en mettant l'accent sur certains intervalles de la grille, intervalles pouvant correspondre à des modèles bruités et à une représentation erronée. Par ailleurs, en apprentissage supervisé pour la régression lasso, il est d'usage de fixer une grille d'échelle logarithmique pour $\lambda$ (Friedman et al., 2010b). Ici, l'échelle est linéaire ou fixée par l'utilisateur, mais son influence sur $\mathcal{O}_K$ est non négligeable.

Sélection de variables par maximisation du MICL (9) Dans les approches ci-dessus, à chaque étape de l'algorithme de sélection des variables, la vraisemblance d'un GMM doit être optimisée plusieurs fois. Pour éviter de telles répétitions, Marbac and Sedki (2017) proposent une procédure qui repose sur le critère ICL (Biernacki et al., 2010) et celle-ci ne nécessite pas de multiples appels de l'algorithme EM. Dans ce cadre, une variable est déclarée comme non importante pour le clustering si ses distributions marginales unidimensionnelles sont égales entre toutes les composantes du mélange. Ainsi, l'hypothèse d'indépendance des variables dans les clusters est faite.

Les auteurs introduisent une variable indicatrice de poids $\mathbf{w} = (w_1, ..., w_p)$ telle que $w_j = 1$ si la variable $\mathbf{x}^j$ est importante, 0 sinon. Dans ce contexte, $\mathbf{X}^C = \{\mathbf{x}^j \in \mathbf{X} : w_j = 1, \forall j\}$ et $\mathbf{X}^{NC} = \{\mathbf{x}^j \in \mathbf{X} : w_j = 0, \forall j\}$ et les différents modèles sont spécifiés par $\mathbf{w}$. Ces modèles sont comparés à l'aide d'un critère basé sur la vraisemblance intégrée des données complètes :

$$p(\mathbf{X}, \mathbf{z}|\mathbf{w}) = \int p(\mathbf{X}, \mathbf{z}|\theta, \mathbf{w}) p(\theta|\mathbf{w}) d\theta,$$

où $\theta$ et $\mathbf{z}$ sont définis dans la Section 2.5.1. Après avoir supposé l'indépendance locale et globale de toutes les variables, l'intégrale ci-dessus se réduit à :

$$p(\mathbf{X}, \mathbf{z}|\mathbf{w}) = p(\mathbf{z}) \prod_{j=1}^{J} p(\mathbf{x}_j|w_j, \mathbf{z}).$$

Pour une variable non importante, $p(\mathbf{x}_j|w_j, \mathbf{z}) = p(\mathbf{x}_j|w_j)$ puisque cette quantité ne dépend pas du clustering, alors que pour une variable importante, cette quantité prend des valeurs différentes sur les composantes du mélange. Par conséquent, la sélection des variables est effectuée en déterminant le vecteur $\mathbf{w}$ qui maximise le critère MICL (Maximum Integrated Complete-data Likelihood), donné par :

$$\text{MICL}(\mathbf{w}) = \text{argmax}_{\mathbf{z}} \log p(\mathbf{X}, \mathbf{z}|\mathbf{w}).$$

La maximisation du MICL($\mathbf{w}$) est effectuée de manière itérative en utilisant un algorithme qui alterne entre deux étapes d'optimisation de l'ICL : optimisation sur la classification $\mathbf{z}$ étant donné les données et $\mathbf{w}$, et

maximisation sur $\mathbf{w}$ étant donné les données et la classification $\mathbf{z}$. Selon les auteurs, cette approche est censée être rapide et adaptable aux problèmes comportant un grand nombre de variables. Néanmoins, nous verrons dans nos simulations que tout ceci est relatif. De plus, l'optimisation sur $\mathbf{z}$ peut être coûteuse en temps de calcul pour échantillons de grande taille. Les auteurs suggèrent que pour les tailles d'échantillon inférieures à $10^4$, cette optimisation est encore possible.

Les auteurs étendent leur modèle aux données mixtes Marbac et al. (2020). Ils montrent que le MICL peut s'écrire exactement sans approximation sous une forme optimisable pour les modèles de mélange avec des données mixtes. Par conséquent, la sélection du modèle est effectuée par une procédure simple et rapide qui alterne entre deux maximisations présentées dans Marbac and Sedki (2017).

L'applicabilité du modèle aux données mixtes et le fait que l'algorithme semble avoir un faible temps d'exécution sont des avantages non négligeables de la méthode. En revanche, elle repose sur des hypothèses fortes. En effet, dans une situation où les variables sont corrélées et où de nombreuses variables redondantes sont présentes, les hypothèses d'indépendances locale et globale pourraient être trop restrictives. Il faudra attentivement observer les avantages d'un tel modèle par rapport à des modèles qui ne prennent pas non plus en compte la covariance comme par exemple le WT-$K$-means.

### 2.5.5 Autres méthodes

VSCC : La méthode de Jeffrey L. Andrews et Paul D. McNicholas (10) La méthode VSCC (Variable selection for clustering and classification ) de Andrews and McNicholas (2014) a pour but de trouver les variables qui minimisent la variance intra-classes et donc maximisent la variance inter-classes dans un modèle GMM. En ce sens, le but est similaire à celui du WT-$K$-means. Les auteurs remarquent qu'il faut évidemment normaliser les données pour que les variables aient les mêmes variances. Dans l'article, les variables ont été normalisées à variance un et moyenne zéro.

La méthode VSCC procède par étapes, la première consistant à calculer les variances intra-classes, où les classes sont calculées à l'aide d'un GMM. La première variable sélectionnée est celle qui a la plus petite variance inter-classes $\frac{v^j}{n}$. Ensuite les variables restantes sont sélectionnées suivant un seuil automatiquement défini grâce aux données. En effet, les auteurs ne veulent pas sélectionner des variables redondantes, c'est-à-dire des variables qui partagent une structure commune. Ainsi ils proposent d'appliquer un filtre aux variables à sélectionner : la variable $j$ est sélectionnée si $\text{cor}(\mathbf{x}^j, \mathbf{x}^l) < 1 - \frac{v^j}{n}$ est vrai pour toutes variables $l$ déjà sélectionnées. Enfin, la méthodologie prévoit d'itérer jusqu'à cinq fois la sélection de variables et l'obtention des classes à l'aide des GMM.

Nous pouvons faire plusieurs remarques au sujet de cet algorithme. Premièrement il est très simple et est donc facile à implémenter, surtout si l'on dispose déjà d'un code implémentant les GMM, car la sur-couche de sélection de variables est très peu coûteuse en temps de calcul. Deuxièmement, l'algorithme n'introduit pas de paramètres supplémentaires. En revanche, le choix de la contrainte $\text{cor}(\mathbf{x}^j, \mathbf{x}^l) < 1 - \frac{v^j}{n}$ semble assez arbitraire tout comme l'est le choix du nombre d'itérations (cinq). De plus, si deux variables sont identiques, c'est-à-dire si $\text{cor}(\mathbf{x}^j, \mathbf{x}^l) = 1$, au plus une des deux pourra être sélectionnée ce qui peut être problématique d'un point de vue de l'interpretabilité et de l'importance des variables.

### 2.5.6 Résumé

Comme nous l'avons vu, deux approches principales sont aujourd'hui utilisées pour effectuer de la sélection de variables dans le contexte du model based clustering. D'une part, les approches basées sur la sélection de modèles présentent l'avantage évident d'être entièrement automatiques puisqu'elles sélectionnent automatiquement les variables à retenir. Leurs implémentations actuelles, basées sur des stratégies forward-backward, peuvent cependant empêcher leur application à des données de très haute dimension. D'autre part, les approches basées sur des pénalisations de type lasso sont généralement rapides et ont de bons résultats même sur des données de très haute dimension. Cependant, la sélection du paramètre de sparsité $\lambda$ reste une question ouverte.

## 2.6 Caractéristiques principales des algorithmes de clustering sparse

Dans cette section, des points importants nécessaires au bon fonctionnement des algorithmes sparses sont présentés. (À étendre aux algorithmes GMM - Partie à rédiger).

### 2.6.1 L'importance de la normalisation pour les algorithmes de clustering sparse

Il est d'usage de normaliser les variables à variance unitaire et moyenne zéro pour les algorithmes de clustering basés sur la distance euclidienne. En effet, sans normalisation les algorithmes vont trouver des classes à partir de la structure en variance des données plutôt que de la structure en clusters. La procédure itérative des algorithmes de clustering pondérés et sparses amplifie le phénomène. Par exemple, si sur des données normalisées, $\mathbf{x}^1$ a une variance inter-classes deux fois plus grande que $\mathbf{x}^2$ avec le $K$-means, alors avec le WT-$K$-means au fil des itérations le poids de $\mathbf{x}^2$ va tendre vers zéro. Toutefois si $\mathbf{x}^2$ à une variance trois fois plus grande, alors l'effet inverse va se produire si on ne normalise pas.

Pour toutes les méthodes décrites dans l'état de l'art il faut, au minimum, normaliser la variance des données pour que les variables aient toutes des variances égales et, au mieux, aussi centrer les variables en 0. Étonnamment cela n'est pratiquement jamais recommandé dans la littérature sur le $K$-means sparse. Seuls Arias-Castro and Pu (2017) indiquent "Throughout this section, we standardize the data coordinate-wisely, we assume that the number of clusters is given" ce qui supposerait qu'il normalise les données, Chakraborty and Das (2019) remarque "We standardized the datasets prior to applying the algorithms for all the five algorithms" et Huo and Tseng (2017) normalise aussi. Nous pouvons noter que pour les GMM il n'est généralement pas nécessaire de normaliser car ceux-ci modélisent la variance des données, sauf dans des cas précis tels que le modèle de Pan et Shen et ces extensions "We assume throughout this article that, prior to clustering analysis, we have standardized data so that each attribute has sample mean 0 and sample variance 1" Pan and Shen (2007) ; Xie et al. (2008) ; Zhou et al. (2009). Nous pouvons aussi remarquer que certains modèles GMM pour la sélection de variables, l'hypothèse des variables de bruit Gaussiennes de matrice de covariance diagonales est faite (Bouveyron and Brunet-Saumard, 2014a ; Marbac and Sedki, 2017).

### 2.6.2 L'importance de l'initialisation

Comme pour la majorité des algorithmes de clustering itératif, une bonne initialisation est cruciale. Dans la version originale du WT-$K$-means, l'algorithme est initialisé avec tous les poids égaux $w_j = 1/\sqrt{p}$, ensuite un premier $K$-means est exécuté et on obtient des clusters et donc des nouveaux poids. Dans cette étape, le $K$-means standard est appliqué sur l'espace de départ et donc mécaniquement ce résultat aura beaucoup d'impact pour la suite de l'algorithme. En effet, si à cette étape les variables importantes ont un poids proche de zéro il sera difficile voire impossible pour le WT-$K$-means d'échapper à ce minimum local. Plusieurs solutions ont été évoquées mais pas implémentées et nous ne ferons pas d'analyses comparatives dans ce manuscrit

### 2.6.3 La convergence des algorithmes

Partie à rédiger.

### 2.6.4 Le choix du paramètre $\lambda$ à l'aide de la détection de rupture

Partie à rédiger. Références ? Marie/Madalina.

## 2.7 Description des packages `R` existants

Une partie des méthodes évoquées dans ce chapitre disposent d'un code disponible gratuitement et publiquement. Elles sont pour la quasi-totalité codées en `R` sous forme de package disponible sur le `CRAN`*. Certaines implémentations ont même été publiées dans le Journal of Statistical Software† pour plupart. Quelques une ne sont pas disponibles sur le `CRAN` mais on peut trouver leurs codes sur `GitHub`‡.

La Table 2.1 décrit et compare l'ensemble des packages `R` permettant de faire de l'importance de variables en clustering ou permettant de faire du clustering sur des données mixtes. Plusieurs critères sont comparés, pour savoir notamment si la fonction est adaptée au clustering de données mixtes, si l'algorithme implémenté est utilisable en grande dimension c'est-à-dire lorsque $p \gg n$, si la fonction pondère les variables selon leur importance pour partitionner les données, si une variable est importante pour discriminer seulement certains clusters et enfin si la fonction permet de faire de la sélection de variables. Des indications sur le coût algorithmique des méthodes et leur temps de calcul auraient été les bienvenues mais malheureusement cela dépend de

*https ://cran.r-project.org/
†https ://www.jstatsoft.org/index
‡https ://github.com/

beaucoup de paramètres, tels que le nombre d'observations et de variables et autres hyperparamètres de la méthode, qui interagissent les uns avec les autres et donc une simple évaluation unidimensionnelle peut s'avérer compliquée et réducteur. Néanmoins une appréciation est donnée dans la partie présentant les simulations.

Dans le Chapitre 3, nous proposons une nouvelle méthode de clustering sparse pour des données ayant une structure de groupe et nous l'appliquons dans le cadre de données mixtes. Cette méthode est codée dans le package R vimpclust et est présentée dans le tableau ci-dessous par souci de complétude.

Dans les fonctions implémentées, quatre permettent de faire du clustering de données mixtes et seulement deux d'entre elles offrent la possibilité de faire de la sélection de variables : `VarSelCluster` (Marbac and Sedki, 2017 ; Marbac et al., 2020), qui correspond à la méthode décrite dans la section 2.5.4 dans le dernier paragraphe et `sparswkm` qui est la fonction implémentant notre solution.

Différentes simulations pour comprendre le comportement des différentes méthodes seront proposées dans les sections suivantes.

| Méthode | Famille | code | nom package | Fonction | Données | $p \gg n$ | Importance | Imp par $C_k$ | Sparse |
|---|---|---|---|---|---|---|---|---|---|
| (1) Witten and Tibshirani (2010) | KM | R-package | sparcl | KMeansSparseCluster | num | oui | oui | non | oui |
| (2) Kondo et al. (2012) | KM | R-package | RSKC Kondo et al. (2016) | RSKC | num | oui | oui | non | oui |
| (3) Brodinová et al. (2019) | KM | GitHub | brodsa/wrsk | ROBIN | num | oui | non | non | oui |
| Chavent et al. (2020) | KM | R-package | vimpclust | sparsewkm | mix | oui | oui | non | oui |
| Chavent et al. (2020) | KM | R-package | vimpclust | groupsparsewkm | num | oui | oui | non | oui |
| (4) Huo and Tseng (2017) | KM | GitHub | Caleb-Huo/IS-Kmeans | ISKmeans | num | oui | oui | non | oui |
| (5) Arias-Castro and Pu (2017) | KM | GitHub | victorpu/SAS_Hill_Climb | hill_climb | num | oui | non | non | oui |
| (6) Bouveyron and Brunet-Saumard (2014a) | GMM | R-package | FisherEM | sfem | num | oui | oui | non | oui |
| (7) Maugis et al. (2009b) | GMM | R-package | clustvarsel Scrucca and Raftery (2018) | clustvarsel | num | oui | non | non | oui |
| (8) Celeux et al. (2019) | GMM | R-package | SelvarMix Sedki et al. (2014) | SelvarClustLasso | num | oui | non | non | oui |
| (9) Marbac and Sedki (2017) | GMM | R-package | VarSelLCM | VarSelCluster | mix | oui | non | non | oui |
| (10) Andrews and McNicholas (2014) | GMM | R-package | vscc | vscc | num | oui | oui | non | oui |
| Fop et al. (2017) | GMM | R-package | LCAvarsel | LCAvarsel | cat | oui | non | non | oui |

**Table 2.1 :** Liste de packages codés en R implémentant des méthodes qui effectuent du clustering sparse dont certaines permettent le clustering de données mixtes, du clustering pondérés, du clustering en grande dimension, du clustering sur des sous-espaces. Presque tous les algorithmes sont basés sur l'algorithme des $K$-means, noté KM, ou sur les modèles de mélange Gaussien, noté GMM. La colonne "Données" indique si l'algorithme traite uniquement les données numériques ("num"), uniquement les données catégorielles ("cat") ou le cas des données mixtes ("mix"). La colonne "$p \gg n$" indique si l'algorithme traite des données en grande dimension avec $p \gg n$. La colonne "Importance" indique si l'algorithme pondère les variables en fonction de leur importance. La colonne "Imp par $C_k$" indique si l'algorithme pondère les variables en fonction de chaque cluster. L'information "R-package" est indiquée dans la colonne "code" si le package est disponible sur le CRAN, sinon le code est disponible sur GitHub. L'article scientifique décrivant la méthode est indiqué dans la colonne "Méthode" et si le package est décrit dans un article, celui-ci sera cité dans la colonne "nom package". Pour accéder à la page web d'un package CRAN, suivez le lien : "https ://cran.r-project.org/web/packages/nom-du-package" et remplacez "nom-du-package" par le nom du package présent dans la colonne "nom package". Par exemple, "https ://cran.r-project.org/web/packages/vimpclust" vous donne la page web du package vimpclust. Pour accéder à la page web d'un package GitHub, suivez le lien : "https ://github.com/nom-du-package" et remplacez "nom-du-package" par le nom du package présent dans la colonne "nom package".

## 2.8 Analyse des schémas et des résultats de simulations

### 2.8.1 Analyse des schémas de simulations

Dans cette section, nous comparons et analysons les schémas de simulations proposés dans les articles évoqués dans ce chapitre portant sur l'importance et la sélection de variables. Certains articles partagent intentionnellement les mêmes schémas de simulations ce qui facilite la comparaison entre les méthodes et surtout cela permet d'éviter la défiance quant au choix du schéma de simulation. Le schéma de simulation proposé ici est le plus utilisé dans la littérature et il a été proposé dans Witten and Tibshirani (2010).

![13](LE MODÈLE GLOBAL)[13]! Les simulations ont pour but d'évaluer et de comparer des algorithmes de clustering avec sélection de variables. Ainsi, dans les schémas de simulations proposés, on retrouve généralement deux groupes de variables :

- le groupe des variables importantes qui ont une structure de clustering et dans notre cas, c'est un mélange de Gaussiennes. On note $p_K$ le nombre de variables importantes.
- le groupe des variables de bruit qui sont indépendantes du clustering, indépendantes des variables importantes et indépendantes entre elles. Elles suivent le plus souvent une distribution Gaussienne (sphérique) et plus rarement une distribution uniforme. On note $d$ le nombre de variables de bruit indépendantes.

Certains articles ajoutent aussi des variables dites redondantes comme ceux de Maugis et al. (2009a,b) ; Celeux et al. (2019), mais ce type de schéma ne sera pas abordé et par conséquent dans la suite $p = p_K + d$. Le modèle de mélange sous-jacent global s'écrit :

$$\sum_{k=1}^{K} \frac{1}{K} \mathcal{N}(\mu_k, \mathbf{I}_p) \text{ avec } \mu_k = (\mathbf{m}_k, 0, ..., 0)^\intercal \in \mathbb{R}^{p_K+d} \text{ et } \mathbf{m}_k \in \mathbb{R}^{p_K}. \tag{2.11}$$

Ce mélange Gaussien décrivant les variables peut être de deux types :

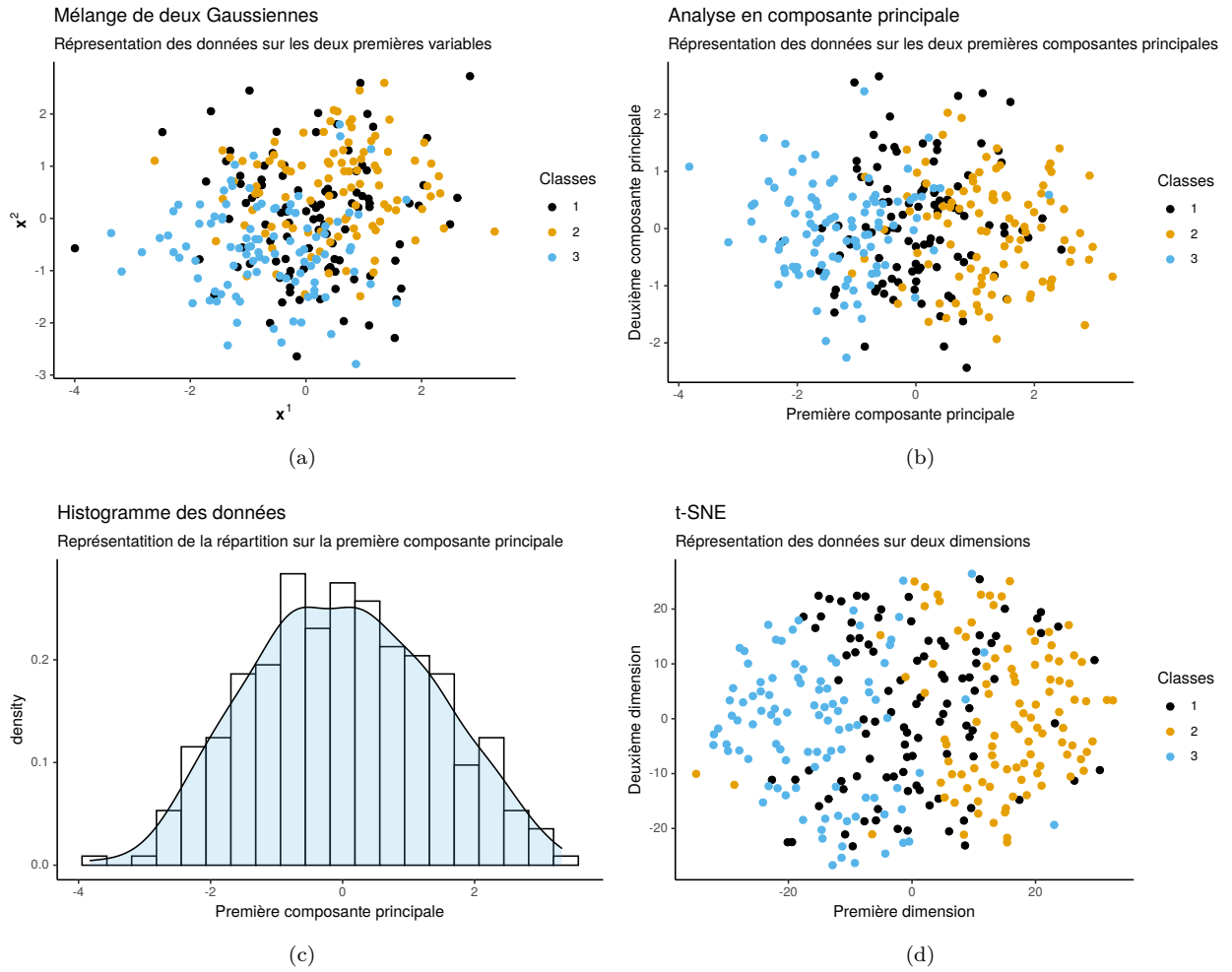
1. un mélange de deux ou trois Gaussiennes sphériques équiprobables, $K = 3$ $\sum_{k=1}^{K} \frac{1}{K} \mathcal{N}(\mu_k, \mathbf{I}_p)$ avec $\mu_1 = -\mu_2 = (\mathbf{m}_1, 0, ..., 0)_p^\intercal$ et $\mathbf{m}_1 = -\mathbf{m}_2 = (m, ..., m)_{p_K}^\intercal$ et si il y a un troisème cluster alors $\mu_3 = \mathbf{0}_p$. Les clusters sont donc alignés sur un axe.
2. un mélange de quatre Gaussiennes sphériques équiprobables, $K = 4$, $\sum_{k=1}^{K} \frac{1}{K} \mathcal{N}(\mu_k, \mathbf{I}_p)$ avec $\mu_1$ et $\mu_2$ décrit comme précédemment et $\mu_3 = -\mu_4 = (\mathbf{m}_3, 0, ..., 0)_p^\intercal$ et $\mathbf{m}_3 = -\mathbf{m}_4 = (-m, ..., -m, m, ..., m, 0, ..., 0)_p^\intercal$ donc $\mathbf{m}_3 = (-m, ..., -m, m, ..., m)_{p_K}^\intercal$ avec $p_K$ un nombre paire et donc les $p_K/2$ premières variables ont une moyenne de $-m$. Les clusters sont donc en carré en dimension deux et sur le plan diagonal d'un hypercube en plus grande dimension.

Les articles ne débattent pas de l'intérêt d'utiliser un schéma plutôt que l'autre. On se restreint ici au premier cas, ce qui est le schéma de simulations proposé par Witten and Tibshirani (2010) (avec $K = 3$) et c'est aussi celui qui est le plus repris dans la littérature (Arias-Castro and Pu, 2017 ; Bouveyron and Brunet-Saumard, 2014a ; Marbac and Sedki, 2017 ; Celeux et al., 2014) tandis que d'autres schémas s'en rapprochent beaucoup (Chakraborty and Das, 2019 ; Chakraborty et al., 2020). Dans Pan and Shen (2007), le schéma de simulation est similaire en tous points, sauf que les clusters n'ont pas le même nombre d'observations.

Par la suite, nous nommerons *schéma de simulation* (ou simplement schéma) un modèle global par exemple décrit par l'Équation 2.11 de type 1 (trois Gaussiennes sphériques équiprobables). Par ailleurs, un schéma permet de simuler plusieurs *scénarios* en fixant différents paramètres comme par exemple $p_K, d, m$. Enlever le schéma 2 : confusion.

À partir du schéma 1, plusieurs scénarios ont été proposés avec par exemple $(p_K, d) = (5, 25)$, $(p_K, d) = (5, 100)$, $(p_K, d) = (50, 100)$ et $n = 30, 300$ (Bouveyron and Brunet-Saumard, 2014a ; Marbac and Sedki, 2017 ; Celeux et al., 2014) , ou encore $(p_K, d) = (50, 200)$, $(p_K, d) = (50, 500)$, (Arias-Castro and Pu, 2017) avec $n = 60, 90$.

APPRÉCIATION DU SCHÉMA DE SIMULATION Toutefois un problème demeure dans ces scénarios et il est dû au choix de la constante $m$. Dans la version originale (Witten and Tibshirani, 2010) $m = 0.6, 0.7, 0.8, 0.9, 1, 1.7$ puis certains auteurs ont testé des valeurs entre ces deux bornes (Arias-Castro and Pu, 2017). Regardons plus précisément un des scénarios proposés avec $n = 300, m = 0.6, p_K = 5$ mais fixons $d = 0$. Nous illustrons ce scénario sur la Figure 2.2 avec des données simulées. Il y a quatre graphiques illustrant les données. Le




(a) (b) (c) (d)

**Figure 2.2 :** Le graphique (a) représente les deux premières variables importantes des données, avec indication des classes. Le graphique (b) les deux premières composantes principales avec indication des classes. Le graphique (c) représente la répartition des données sur la première composante principale. Le graphique (d) représente les classes dans le sous-espace trouvé pour l'algorithme t-SNE. Objectivement, les classes ne correspondent pas à des clusters.

premier graphique (a) représente les deux premières variables importantes des données et il n'y a clairement aucune structure en clusters. Le deuxième graphique (b) représente les deux premières composantes principales d'une ACP effectuées sur les variables importantes. En effet, ! [14](les classes étant alignées sur les variables importantes)[14] !, l'axe traversant les centres des trois classes sur les cinq variables importantes est la direction la plus discriminante et c'est aussi la seule. Théoriquement, c'est aussi l'axe de variance maximale car les classes induisent une corrélation entre les variables importantes en raison de l'alignement de leurs centres. Ainsi, en regardant la première composante principale, on a toute l'information disponible sur la séparabilité des classes. Nous pouvons constater que les classes se superposent et en regardant le troisième graphique (c), qui représente la distribution des données sur la première composante principale, nous ne pouvons pas distinguer les classes et la distribution semble unimodale. Le dernier graphique (d) est obtenu à l'aide de l'algorithme t-SNE (t-distributed stochastic neighbor embedding) (Van der Maaten and Hinton, 2008), qui est une technique de réduction de dimension pour la visualisation de données développée par Geoffrey Hinton et Laurens van der Maaten. Il s'agit d'une méthode non linéaire permettant de représenter un ensemble de points d'un espace à grande dimension dans un espace de deux ou trois dimensions. C'est une méthode très utilisée notamment en clustering pour voir si les données ont une structure en clusters ou même pour valider des résultats (Chakraborty and Das, 2019 ; Chakraborty et al., 2020). Encore une fois les classes se chevauchent et ne sont pas séparables. Somme toute, les données semblent ne pas avoir de structure en clusters et il s'avère que ce scénario est similaire à celui présenté Figure 1.1 (b).

Observons maintenant des scénarios plus simples où $n = 300, d = 0$ avec $p_K = 5, 50$ et $m = 0.6, 1, 1.7$. Nous illustrons ces scénarios sur la Figure 2.3 avec des données simulées. Nous pouvons constater que les scénarios avec $p_K = 5$ variables induisent une structure ambiguë même pour le cas où les classes sont les plus séparées ($m = 1.7$). Les classes ne sont pas linéairement séparables, les frontières entre les classes sont denses

et donc la présence de clusters est contestable. Cela rappelle une nouvelle fois le cas exposé Figure 1.1 (b). Cela n'empêche pas qu'avec $d = 0$ les 3-means obtiendraient de très bons résultats au sens de l'ARI. Lorsque $p_K = 50$, l'information sur les classes est beaucoup plus grande et même dans le cas le plus difficile $m = 0.6$ nous pouvons distinguer des clusters, en tout cas de manière plus précise que pour $p_K = 5$.

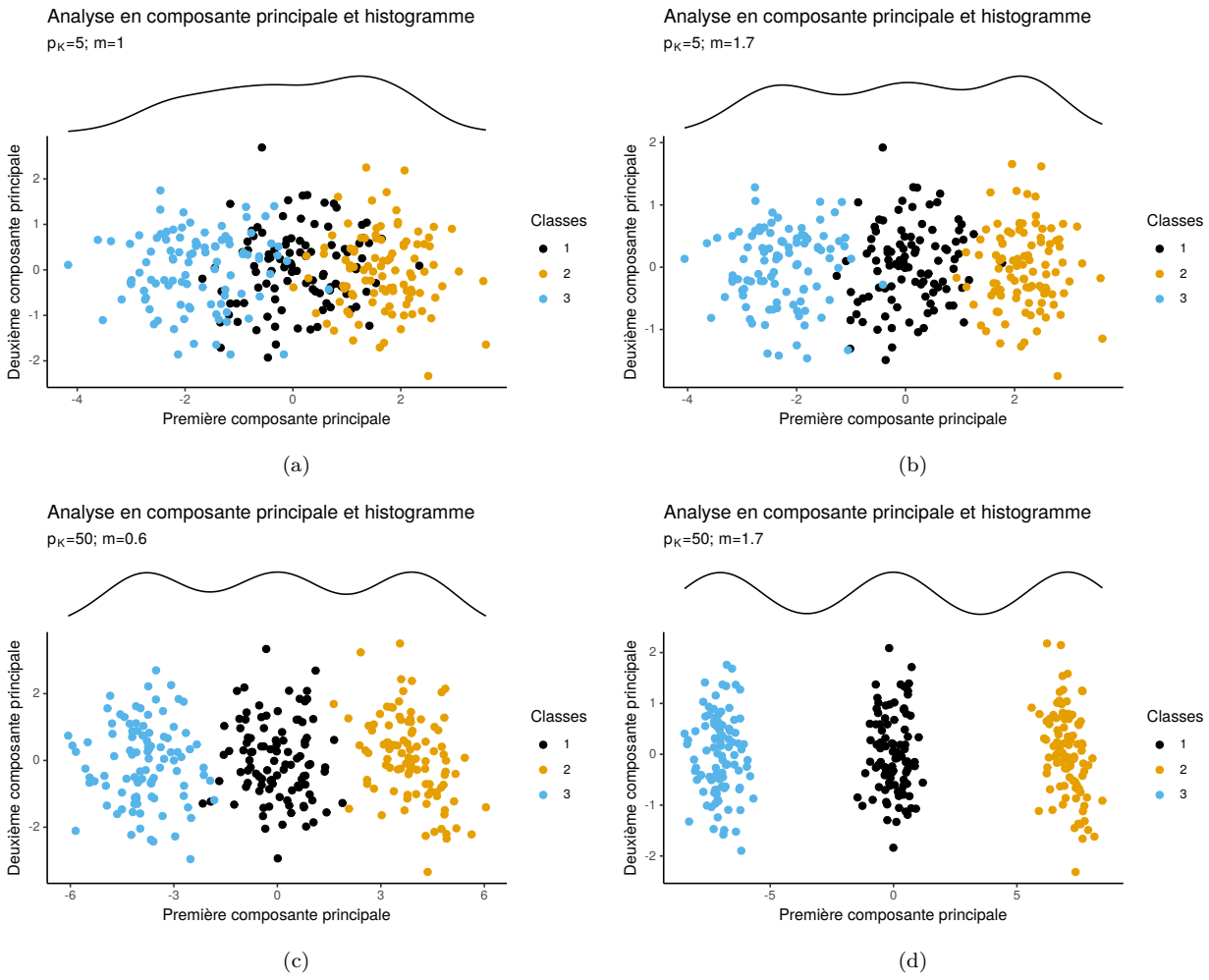



**Figure 2.3 :** Les quatres graphiques représentent les classes sur la première composante principale et la densité des distributions. Dans le cas (a) on a $p_k = 5; m = 1$, pour (b) $p_k = 5; m = 1.7$, pour (c) $p_k = 50; m = 0.6$, pour (d) $p_k = 50; m = 1.7$.

La vérification de la structure en clusters des données peut être jugée quelque peu subjective et n'est le fruit que d'un résultat de simulation. Néanmoins, elle a le mérite d'être faite et toutes les informations sont données au lecteur pour qu'il puisse juger de la qualité du schéma de simulation.

LE PROBLÈME DE LA NORMALISATION Il reste un dernier point à aborder sur le schéma de simulation et il concerne la variance des variables simulées. La variance des variables de bruit est par définition égale à 1 mais il n'en est pas de même pour les variables importantes. La Table 2.2 indique les variances des cinq premières variables importantes. Elles ont toutes des variances supérieures à un ce qui pose problème. En effet, Daniella Witten et Robert Tibshirani n'indiquent pas dans leur article qu'il faut normaliser les variables pour les ramener à une même variance et aucune fonction de leur package `R` n'est dédiée à cette tâche. De surcroît, ce n'est pas la seule méthode qui nécessite une normalisation des données et il faut être extrêmement vigilant lors de l'utilisation des méthodes et ce pour deux raisons. Premièrement, des variances plus élevées pour les variables importantes comparativement aux variables de bruit donne un clair avantage aux méthodes nécessitant une normalisation, notamment le WT-$K$-means. Les auteurs auraient pu décider de simuler des données dont les variances des variables importantes seraient beaucoup plus petites que celles des variables de bruit provoquant l'effet inverse : pénaliser les performances du WT-$K$-means (et des autres méthodes similaires). Deuxièmement, les articles se comparant à la méthode de WT-$K$-means suivent les préconisations des auteurs et donc ne normalisent pas non plus. Cela introduit un biais dans les simulations, où les méthodes nécessitant d'être normalisées à variance unitaire vont révéler la structure en variance des données plutôt que

**Table 2.2 :** Table indiquant les variances des cinq premières variables importantes.

| | $\mathbf{x}^1$ | $\mathbf{x}^2$ | $\mathbf{x}^3$ | $\mathbf{x}^4$ | $\mathbf{x}^5$ |
|---|---|---|---|---|---|
| $m = 0.6$ | 1.32 | 1.33 | 1.16 | 1.23 | 1.25 |
| $m = 1$ | 1.82 | 1.56 | 1.76 | 1.63 | 1.99 |
| $m = 1.7$ | 2.72 | 2.99 | 3.30 | 2.70 | 2.98 |

la structure en clusters.

### 2.8.2 Analyse des résultats de simulations dans la littérature

- Dans Witten and Tibshirani (2010), la méthode WT-$K$-means est comparée à deux méthodes sparses, la méthode de Raftery et Dean (Raftery and Dean, 2006) et la méthode de Pan et Shen (Pan and Shen, 2007). Le WT-$K$-means obtient de meilleurs résultats que les deux méthodes et en particulier la méthode de Raftery et Dean rencontre beaucoup de difficultés “The method of Raftery and Dean also did quite poorly”.

  Dans les simulations effectuées, la méthode Sparse K-means donne de bons résultats, dans la mesure où elle permet d'obtenir de bonnes performances et un clustering sparse. Les auteurs soulignent que le $\lambda$ est choisi pour maximiser le Gap Statistic, cependant une meilleure discrimination des variables de bruit aurait pu être obtenue en choisissant le paramètre $\lambda$ selon une utilisation du Gap Statistic plus spécifique : “however, greater sparsity could have been achieved by choosing the smallest tuning parameter value within one standard deviation of the maximal gap statistic”. Par ailleurs, dans les résultats de simulations, seul le nombre de coefficients non nuls est affiché. Par conséquent, il est difficile de juger de la qualité de la sélection des variables importantes et de la discrimination des variables de bruit.

- Sun et al. (2012) considèrent deux scénarios, l'un où $K$ est fixé et l'autre où $K$ doit être estimé. Les seules méthodes sparses comparées sont la méthode proposée est le WT-$K$-means. Les auteurs affirment “Evidently, our proposed method deliver superior results against their competitors in terms of both clustering error and variable selection”, ce qui n'est pas évident du tout au vu des tableaux et les méthodes sont équivalentes en termes de performances car les intervalles de confiances se chevauchent. Pour ce qui est de la sélection de variables, c'est sûrement le choix du paramètre $\lambda$ qui ne permet pas une bonne discrimination de la part du WT-$K$-means. De plus, ils n'affichent que le nombre total de variables sélectionnées donc il est difficile de porter un jugement précis sur la sélection de variables importantes et la discrimination de variables de bruit.

  Pour le deuxième scénario, les auteurs suggèrent d'utiliser le Gap Statistic pour sélectionner sur une grille bidimensionnelle le couple $(K, \lambda)$ pour le WT-$K$-means. Ils comparent avec leur méthodologie et concluent que le WT-$K$-means a de mauvais résultats et le justifient par “The difficulty of gap statistic in selecting number of clusters is also pointed out in Witten and Tibshirani (2010)”. Or, ces auteurs assurent dans la partie de la sélection des paramètres que “We assume that $K$, the number of clusters, is fixed. The problem of selecting $K$ is outside of the scope of this paper”. Choisir le couple $(K, \lambda)$ simultanément avec le Gap Statistic ne semble donc pas une bonne piste et c'est ce que semble montrer les résultats de Sun et al. (2012).

- Arias-Castro and Pu (2017) comparent leur méthode avec le WT-$K$-means. Ils concluent que “Our method performs at least as well as Sparse K-means” et que “Sparse K-means generally results in more features with non-zero weights than the truth. These extraneous features, even with small weights, may negatively impact the clustering results”. Finalement le WT-$K$-means a des résultats très similaires à leur méthode, mais sélectionne plus de variables en moyenne sans différencier les variables de bruit des variables importantes.

- Chakraborty and Das (2019) comparent le WT-$K$-means avec le $K$-means, le W-$K$-means et leur méthode (LW-$K$-means). Ils concluent sur des données simulées que le WT-$K$-means et le LW-$K$-means ont des résultats équivalents tandis que les autres méthodes ($K$-means et W-$K$-means) ne réussissent pas à avoir de si bonnes performances. Pour la sélection de variables, ils se donnent un vecteur binaire qui indique si une variable est importante ou non pour chaque simulation et comparent avec les variables sélectionnées par le WT-$K$-means et le LW-$K$-means à l'aide du coefficient de corrélation de Matthew (Matthews, 1975). Ils observent que le WT-$K$-means n'offre pas des résultats parfaits. Il semble que le choix du paramètre $\lambda$ en soit la cause, forçant l'algorithme à sélectionner des variables de bruit supplémentaires sans que cela nuise à la découverte des clusters sous-jacents. Par ailleurs, l'algorithme

LW-$K$-means obtient à chaque fois un score parfait en ce qui concerne la sélection de variables, mais on rappelle que les auteurs eux mêmes remarquent que “The value of $\lambda$ was chosen by performing some hand-tuned experiments”. Des explications sur la sélection de modèles manquent, celles sur le choix du paramètre $\lambda$ non explicitées ce qui rend les résultats peu clairs.

- Celeux et al. (2014) comparent deux méthodes de sélection sparses selon le schéma de simulations formulé par l'Équation (2.11). Les simulations et les résultats sont crédibles : on voit que le WT-$K$-means est meilleur pour le partitionnement mais que pour la sélection de variables, il a de moins bons résultats (sûrement encore dû au choix du $\lambda$ avec le Gap Statistic). Cette fois-ci, la sélection de variables a été évaluée à l'aide du taux d'erreur de sélection de variables importantes et du nombre moyen de variables sélectionnées, ce qui est suffisamment informatif.

- Dans l'étude de Bouveyron and Brunet-Saumard (2014a), les conclusions sont très similaires à celles de Celeux et al. (2014). Premièrement, ils reprennent le même schéma de simulations, ensuite ils comparent la méthode de Raftery et Dean, le WT-$K$-means, la méthode décrite Maugis et al. (2009a,b) et leur méthode le Sparse Fisher-EM. Dans l'ensemble le WT-$K$-means a les meilleurs résultats en termes d'erreur clustering avec la méthode de Raftery et Dean suivit de très près par le Sparse Fisher-EM, par contre les auteurs notent “The methods differ however in the number of variables they retain to perform the clustering”. Le Sparse Fisher-EM et la méthode de Maugis et al. (2009a,b) ont de bons résultats de sélection tandis que ce n'est pas le cas pour la méthode de Raftery et Dean et le WT-$K$-means, ce qui est finalement une conclusion partagée par tous les articles, même si ici seul le nombre total de variables sélectionnées est observé.

CONCLUSION SUR LES RÉSULTATS DE SIMULATIONS DANS LA LITTÉRATURE Les méthodes sparses, au moins toutes celles que nous avons pu étudier, ont été comparées soit aux $K$-means, soit aux GMM dans leurs versions classiques. Les performances des méthodes sparses étaient toujours soit équivalentes soit supérieures à celles des méthodes classiques. En outre, plus le nombre de variables de bruit augmente, notamment par rapport au nombre de variables importantes, plus les méthodes sparses sont intéressantes.

Ensuite, la plupart des méthodes étudiées ne sont comparées qu'au WT-$K$-means et dans l'ensemble obtiennent des résultats similaires en termes de partitionnement. En revanche, le WT-$K$-means a de mauvais résultats en termes de sélection de variables car il sélectionne généralement trop de variables de bruit et c'est sans doute dû au choix du paramètre $\lambda$. Toutefois les résultats de simulations gagneraient en clarté si des mesures d'évaluation de sélection de variables plus fines que le seul nombre total de variables sélectionnées étaient utilisées.

## 2.9 Simulations : comparaison des méthodes de clustering sparse

Dans cette section nous allons comparer sept différentes méthodes de clustering sparse. En dehors de ces sept méthodes, cinq méthodes qui ont été décrites dans la Section 2.4 et la Section 2.5 n'ont malheureusement pas pu être évaluées notamment la méthode de Pan and Shen (2007) car il n'y a pas de code disponible implémentant la méthode, la méthode de Arias-Castro and Pu (2017) car le code fourni sur GitHub ne fonctionne pas. Par ailleurs, les codes des méthodes EWP-$K$-means et LW-$K$-means sont disponibles, mais il n'y a aucune méthodologie pour choisir les hyperparamètres et particulièrement le paramètre $\lambda$ et donc il est impossible d'automatiser l'analyse comparative. Enfin, il n'y a pas non plus de code disponible pour le Regularized $K$-means (Sun et al., 2012) et même si l'algorithme paraît facile à implémenter, la solution du problème d'optimisation n'est pas donnée dans l'article et il nous semble qu'elle est mal définie (voir 2.4.4). Les sept méthodes comparées sont les suivantes :

1. WT-$K$-means (Sparcl) - méthode (1) : c'est l'algorithme décrit dans Witten and Tibshirani (2010) et implémenté dans `sparcl`. Le paramètre $\lambda$ est choisi au moyen de la méthode du Gap Statistic. Le nombre d'itérations pour la convergence du $K$-means est fixé à 100, le nombre de valeurs de $\lambda$ à tester est 10 et le nombre de permutations pour le Gap-Statistic est pris par défaut à 25. Le nombre d'initialisations à tester pour le $K$-means est fixé à 10.

2. WT-$K$-means (SKM_chgpt) : c'est l'algorithme décrit dans Witten and Tibshirani (2010) ![15]( où le paramètre $\lambda$ est choisi par une méthode de détection de rupture )[15] !. Le nombre d'itérations pour la convergence du $K$-means est fixé à 100 et le nombre de valeurs de $\lambda$ à tester est 10. Le nombre de d'initialisation à tester pour le $K$-means est fixé à 10.

3. Sparse Fisher-EM (SFEM) - méthode (6) : c'est l'algorithme décrit dans Bouveyron and Brunet (2012) et implémenté dans `FisherEM`. Toutes les formes de *discriminative latent mixture* sont testées. La méthode de régression est utilisée pour modéliser la matrice de projection. 10 valeurs de pénalité lasso sont testées sur une grille entre 0 et 1 (le minimum et le maximum) par pas de 0.1. La valeur de la pénalité $L_2$ (elastic-net) est laissée à 0. Le $K$-means initialise la méthode et le nombre d'initialisations à tester pour le $K$-means est fixé à 10. Le reste des paramètres est pris par défaut.

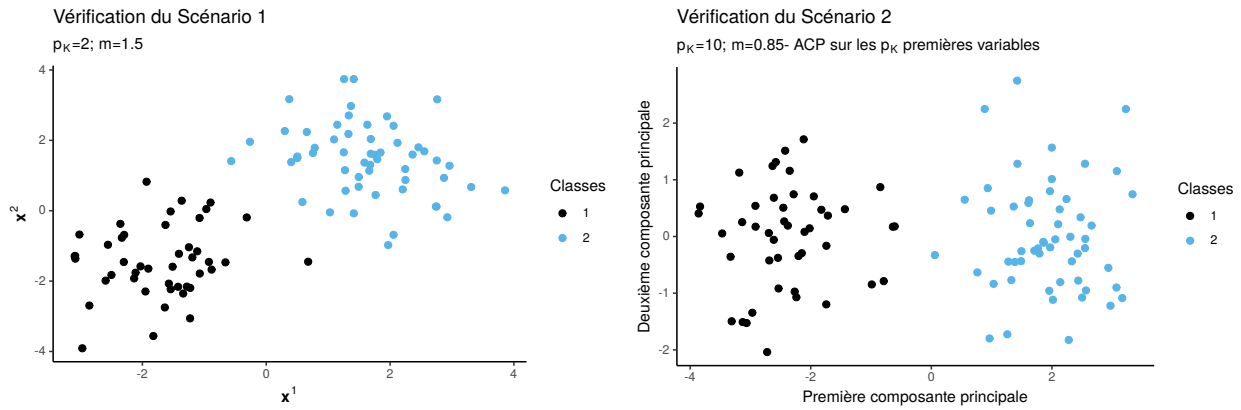
4. Raftery et Dean (Clustvarsel) - méthode (7) : c'est l'algorithme décrit dans Raftery and Dean (2006) et implémenté dans `clustvarsel` Scrucca and Raftery (2018). La méthode *greedy* (forward) est utilisée lorsque cela est possible et sinon on utilise la méthode headlong. Pour le reste on utilisera les paramètres par défaut.

5. Modèle SRUW (SelvarMix) - méthode (8) : c'est l'algorithme décrit dans Celeux et al. (2019) et implémenté dans `SelvarMix` Sedki et al. (2014). La méthode lasso est donc utilisée avec une grille de valeurs de $\lambda$ et $\gamma$ définies automatiquement par l'algorithme. Toutes les formes de GMM sont testées, des plus contraintes aux plus générales. Les trois formes possibles de la matrice de covariance pour le modèle de régression sont testées (sphérique, diagonale et générale). Les deux formes possibles de la matrice de covariance pour les variables de bruit sont testées (sphérique et diagonale). Le critère de sélection est le BIC.

6. Sélection de variables par maximisation du MICL (VarSelLCM) - méthode (9) : c'est l'algorithme décrit dans Marbac and Sedki (2017) et implémenté dans `VarSelLCM`. Le critère de sélection est le BIC.

7. VSCC (Vscc) - méthode (10) : c'est l'algorithme décrit dans Andrews and McNicholas (2014) et implémenté dans `vscc`. Celui-ci ne requiert aucun paramètre spécifique.

Pour toutes les méthodes le nombre de clusters est donné à l'avance, égal au nombre de clusters de la partition simulée. Le schéma de simulation présenté dans la section précédente dans l'Équation (2.11) est utilisé avec $K = 2$ clusters et $n = 100$ observations mais pour $p_K, d$ et $m$ on se restreint à des paramètres différents. En outre, deux scénarios sont testés, le premier avec $p_K = 2, d = 20, m = 1.5$, qui est un scénario avec peu de variables ce qui permet de tester les versions les plus performantes des algorithmes, très coûteuses en temps de calcul notamment la version greedy de Clustvarsel. Mais $p_K = 2$ permet aussi de visualiser directement les clusters pour s'assurer de la structure des données simulées. Pour le deuxième scénario, $p_K = 10; d = 100; m = 0.85$ et c'est donc scénario avec plus de variables. En revanche, les scénarios exacts tels que définis par Daniella Witten et Robert Tibshinari dans leur article n'ont pas pu être testés car les méthodes était trop coûteuses en temps de calcul. Nous représentons les scénarios simulés à l'aide de la Figure 2.4 et les classes représentent bien des clusters séparés.




(a) (b)

**Figure 2.4 :** Le graphique (a) représente les classes définies sur les deux premières variables importantes. Le graphique (b) représente les classes sur les deux premières composantes principales. Les classes représentent bien des clusters séparés.

Plusieurs mesures ont été utilisées pour comparer les méthodes : l'ARI, le ratio de variables importantes sélectionnées qui est, on le rappelle, le nombre de variables importantes sélectionnées notons $p_s$ sur le nombre de variables importantes $p_K$ soit $\frac{p_s}{p_K}$, le ratio de variables de bruit sélectionnées calculer de la même façon et le temps de calcul des variables en secondes. 20 simulations des scénarios seront testées et agrégées et on calcul la moyenne et l'écart type des résultats.

### 2.9.1 Résultats scénario 1 : $K = 2; n = 100; p_K = 2; d = 25; m = 1.5$

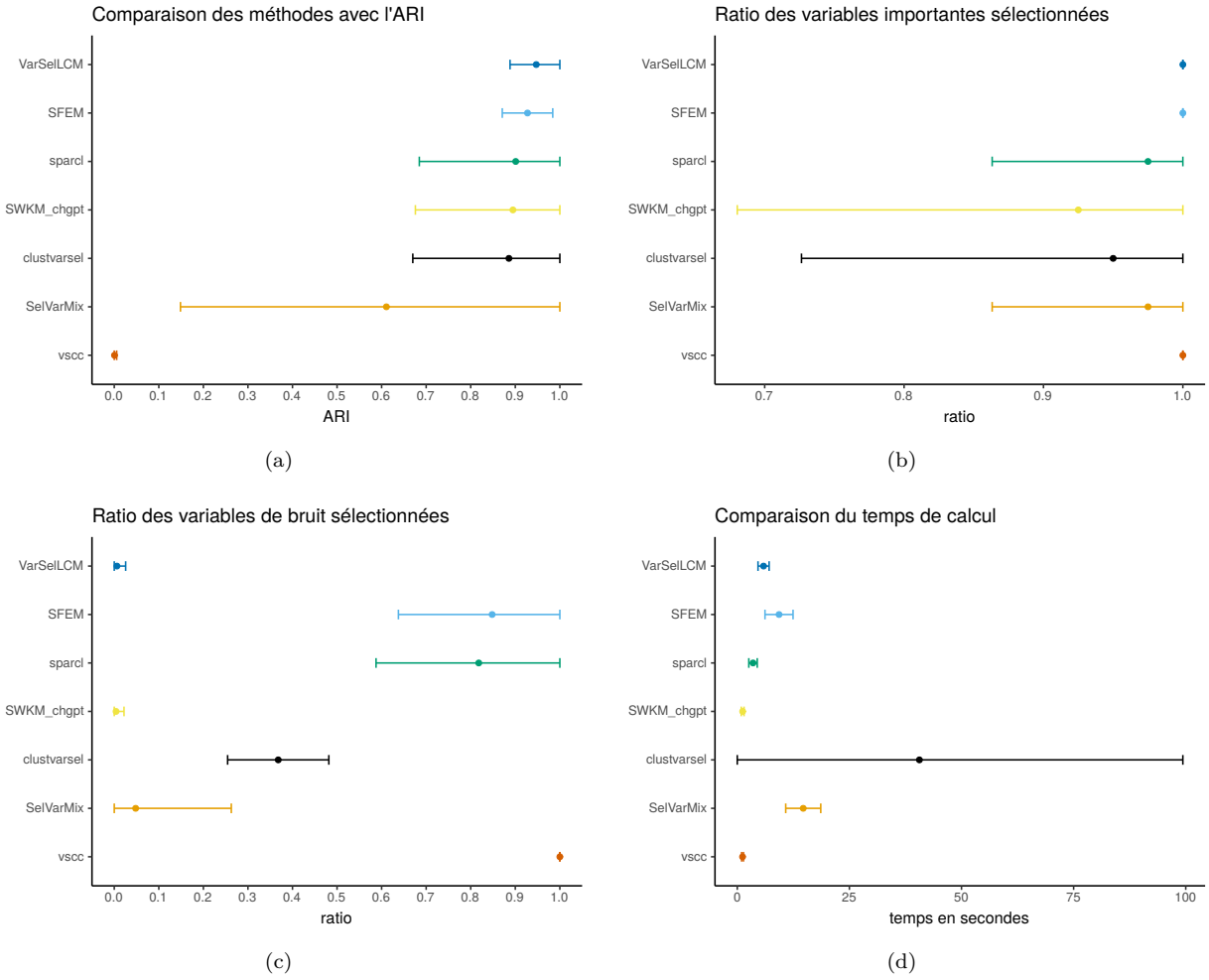



**Figure 2.5 :** Les quatre graphiques représentent les moyennes et écarts-types par méthode pour le scénario $K = 2, n = 100, p_K = 2, d = 20, m = 1.5$ sur 20 simulations de l'ARI (a), le ratio de variables importantes (b) et de bruit (c) sélectionnées et le temps de calcul (d).

Pour résumer, la Figure 2.5 et la Table 2.3 indiquent que les méthodes sont équivalentes pour ce scénario en termes d'ARI et de sélection de variables importantes. Par contre seules les méthodes VarSelLCM, SKM_chgpt et SelvarMIX ne sélectionnent pas de variables de bruit ce qui leur donnent un clair avantage en terme d'interprétabilité. De plus la méthode SKM_chgpt est en moyenne trois fois plus rapide que Sparcl, du fait de l'utilisation du Gap Statistic, 5 fois plus rapide que VarSelLCM, 8 fois plus rapide que SFEM, 13 fois plus rapide que SelvarMix et 33 fois plus rapide que Clustvarsel.

### 2.9.2 Résultats scénario 2 : $K = 2; n = 100; p_K = 10; d = 100; m = 0.85$

Pour résumer, la Figure 2.6 et la Table 2.4 indiquent, comme pour le scénario précédent, que les méthodes sont équivalentes pour ce scénario en termes d'ARI et de sélection de variables importantes ce qui est aussi la conclusion de tous les articles comparant (en partie) les différentes méthodes. Par contre, seules les méthodes VarSelLCM et SKM_chgpt ne sélectionnent pas de variables de bruit. Le WT-$K$-means combiné au Gap Statistic (Sparcl) sélectionne beaucoup de variables de bruit ce qui coïncide avec les résultats des simulations des autres articles. De plus la méthode Vscc est la plus rapide, la méthode SKM_chgpt est en moyenne deux fois et demi plus lente que Vscc et toutes les autres méthodes sont dix à cent fois plus lentes que Vscc.

### 2.9.3 Conclusion sur les simulations

Nous retrouvons globalement les conclusions faites par les auteurs des différents articles étudiés. Les méthodes ont des performances équivalentes en terme d'ARI et de sélection de variables importantes mais certaines

**Table 2.3 :** Le tableau représente les moyennes et écarts-types par méthode pour le scénario $K = 2, n = 100, p_K = 2, d = 20, m = 1.5$ sur 20 simulations de l'ARI, le ratio de variables importantes (Ratio V.Imp) et de bruit (Ratio V.Bruit) sélectionnées et le temps de calcul (Temps).

| | ARI | | Ratio V.Imp | | Ratio V.Bruit | | Temps | |
|---|---|---|---|---|---|---|---|---|
| noms | moyenne | sd | moyenne | sd | moyenne | sd | moyenne | sd |
| VarSelLCM | **0.95** | 0.06 | **1.00** | 0.00 | **0.01** | 0.02 | 5.87 | 1.24 |
| SFEM | **0.93** | 0.06 | **1.00** | 0.00 | 0.85 | 0.21 | 9.31 | 3.13 |
| Sparcl | **0.90** | 0.22 | **0.97** | 0.11 | 0.82 | 0.23 | 3.52 | 0.94 |
| SWKM_chgpt | **0.89** | 0.22 | **0.93** | 0.24 | **0.00** | 0.02 | **1.20** | 0.31 |
| Clustvarsel | **0.89** | 0.22 | **0.95** | 0.22 | 0.37 | 0.11 | 40.61 | 58.74 |
| SelVarMix | **0.61** | 0.46 | **0.97** | 0.11 | **0.05** | 0.21 | 14.71 | 3.91 |
| Vscc | 0.00 | 0.00 | **1.00** | 0.00 | 1.00 | 0.00 | **1.18** | 0.21 |

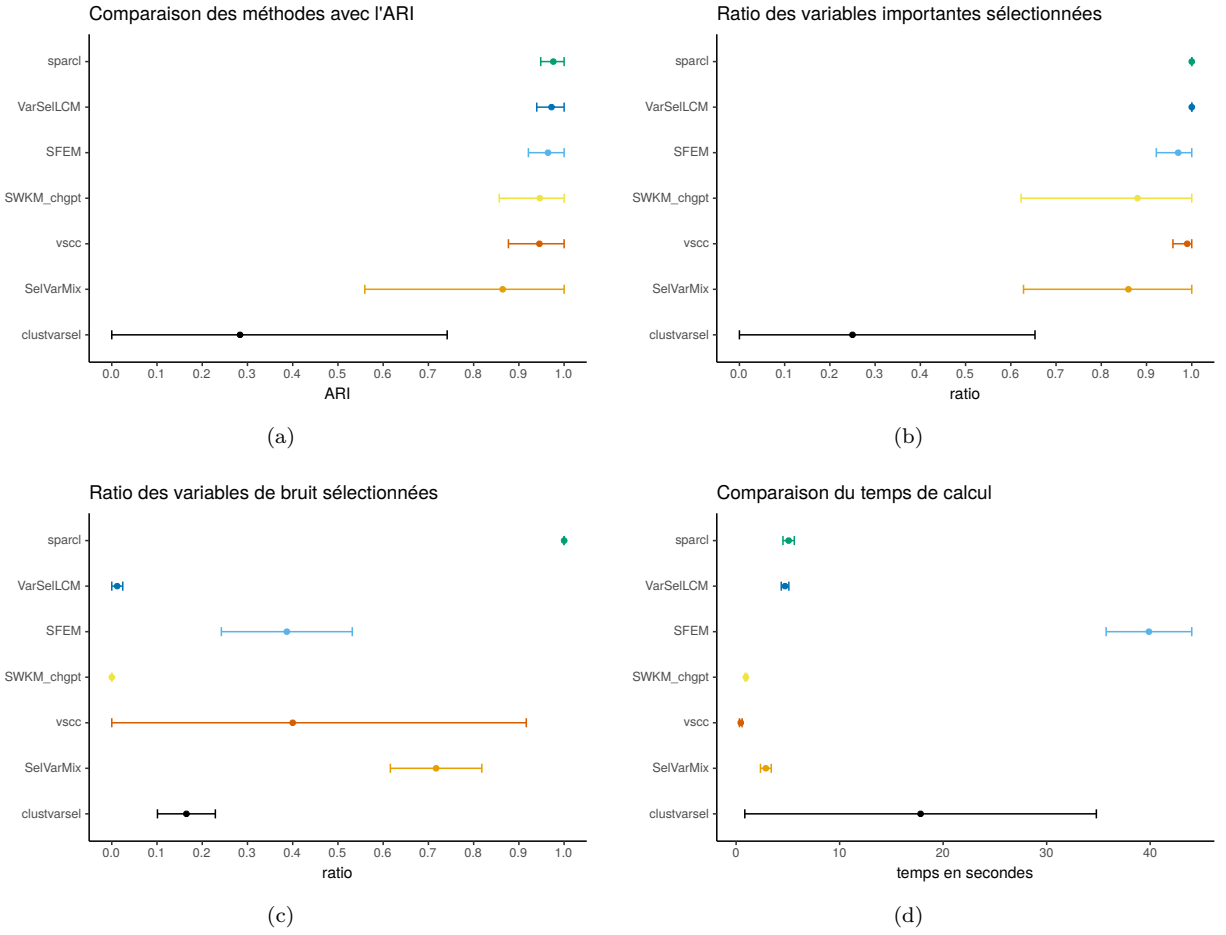



**Figure 2.6 :** Les quatre graphiques représentent les moyennes et écarts-types par méthode pour le scénario $K = 2, n = 100, p_K = 10, d = 100, m = 0.85$ sur 20 simulations de l'ARI (a), le ratio de variables importantes (b) et de bruit (c) sélectionnées et le temps de calcul (d).

**Table 2.4 :** Le tableau représente les moyennes et écarts-types par méthode pour le scénario $K = 2, n = 100, p_K = 10, d = 100, m = 0.85$ sur 20 simulations de l'ARI, le ratio de variables importantes (Ratio V.Imp) et de bruit (Ratio V.Bruit) sélectionnées et le temps de calcul (Temps).

| | ARI | | Ratio V.Imp | | Ratio V.Bruit | | Temps | |
|---|---|---|---|---|---|---|---|---|
| noms | moyenne | sd | moyenne | sd | moyenne | sd | moyenne | sd |
| Sparcl | **0.98** | 0.03 | **1.00** | 0.00 | 1.00 | 0.00 | 5.07 | 0.55 |
| VarSelLCM | **0.97** | 0.03 | **1.00** | 0.00 | **0.01** | 0.01 | 4.73 | 0.37 |
| SFEM | **0.96** | 0.04 | **0.97** | 0.05 | 0.39 | 0.14 | 39.88 | 4.14 |
| SWKM_chgpt | **0.95** | 0.09 | 0.88 | 0.26 | **0.00** | 0.00 | 0.94 | 0.09 |
| Vscc | **0.95** | 0.07 | **0.99** | 0.03 | 0.40 | 0.52 | **0.45** | 0.13 |
| SelVarMix | **0.86** | 0.30 | 0.86 | 0.23 | 0.72 | 0.10 | 2.87 | 0.52 |
| Clustvarsel | 0.28 | 0.46 | 0.25 | 0.40 | 0.17 | 0.06 | 17.82 | 16.98 |

méthodes échouent à supprimer les variables de bruit notamment le WT-$K$-means avec le Gap statistic (Sparcl). Pour cette dernière, c'est bien entendu le choix du paramètre $\lambda$ qui reste difficile. Étant donné que l'ARI est satisfaisant, on en conclut que les poids des variables de bruit sont proches de zéro mais que le choix du paramètre $\lambda$ fait qu'elles n'ont pas été supprimées. VarSelLCM et SKM_chgpt sont les méthodes les plus compétitives notamment en terme d'interprétabilité, d'importance et de sélection de variables. Parmi celles-ci, SKM_chgpt est en moyenne cinq fois plus rapide que les autres, ce qui justifierait son utilisation sur des ensembles de données de grande taille et de grande dimension.

## 2.10 Conclusion

Dans ce chapitre,

# 3

# Clustering sparse de groupe de variables et application à des données mixtes



## 3.1 Introduction

### 3.1.1 Contexte

Comme on l'a vu dans le chapitre précédent, les méthodes sparses tendent à choisir les variables les plus informatives et permettent de résoudre des tâches de clustering en diminuant le nombre de paramètres utiles, ce qui améliore les performances. Désormais, on se pose la question d'améliorer encore ces méthodes si les variables décrivant les données ont une structure de groupe a priori. On constate que dans l'état de l'art réalisé au chapitre précédent, seule une méthode prend en compte la structure de groupe des variables (Huo and Tseng, 2017). Pourtant, il semble intuitif qu'en intégrant cette information dans la sélection des variables, on pourra obtenir des clusters plus précis et plus proches de la structure sous-jacente.

### 3.1.2 Motivations

De nombreux types de données présentent une structuration des variables en groupes, comme par exemple des ensembles de données d'expression génétique, où les gênes ne fonctionnent pas individuellement, mais par

paquets. On peut voir à ce sujet les études de Simon et al. (2013) et Burczynski et al. (2006) qui montrent que les gênes sont regroupés en *ensemble de gênes* à l'aide des données de position cytogénétique. On peut aussi trouver dans Higuera et al. (2015), un jeu de données qui décrit les expressions des protéines chez la souris, que nous avons pris comme exemple d'application dans un document de recherche*.

Un autre exemple de données ayant une structure de groupe naturelle sont les données mixtes, c'est-à-dire mélangeant des variables numériques et catégorielles, et aussi les données seulement catégorielles. En effet, une caractéristique des données mixtes est qu'elles conduisent souvent à définir des groupes de variables, car on représente la plupart du temps une variable catégorielle $\mathbf{x}^j$ à $c_j$ modalités par un groupe de $c_j$ variables indicatrices numériques. Tous ces exemples montrent l'intérêt d'étudier les méthodes de clustering qui prennent en compte l'information de l'existence de groupes de variables.

Dans le chapitre précédent, on s'est intéressé à la méthode de Huo and Tseng (2017) avec la pénalité overlapping group qui est adaptée à ce type de données. Mais si la procédure d'optimisation employée est convaincante en régression (ADMM) les difficultés posées par le clustering nécessitent, selon nous, des solutions simplifiées. En dehors de cette méthode, il existe une première méthode de clustering sparse basée sur les GMM et applicable à des données mixtes a été présentée : la méthode du package `VarSelLCM` (Marbac and Sedki, 2017 ; Marbac et al., 2020). À notre connaissance c'est la seule et il n'y a pas de méthode de clustering sparse basée sur le $K$-means qui soit adaptée aux données mixtes.

### 3.1.3 Objectifs et contributions

Les objectifs principaux, qui sont aussi ceux de l'ensemble de ce manuscrit, sont d'améliorer l'interprétabilité et les performances du clustering en proposant un algorithme de sélection et d'importance de variables applicable à des données numériques ayant une structure de groupe. La méthode proposée sera ainsi parfaitement adaptée au cas des données mixtes.

Notre contribution, décrite dans (Chavent et al., 2020), consiste à proposer conjointement une extension de l'algorithme de WT-$K$-means (Witten and Tibshirani, 2010) à des données numériques sous forme de groupes de variables en introduisant une pénalité adaptée et une méthode de transformation des données mixtes tout en utilisant la méthode de détection de rupture présentée au chapitre précédent pour choisir le paramètre $\lambda$. Cette méthode est implémentée sous forme de package `R` nommé `vimpclust`† et publié sur le `CRAN`.

Par souci de clarté, les remarques et les réflexions préliminaires sur la normalisation des données, l'initialisation des algorithmes, leur convergence, la sélection de modèles et l'analyse de l'état de l'art (méthodes, packages, simulations, résultats) ont été présentées au chapitre précédent. Ainsi, ce chapitre est consacré à la définition de l'algorithme proposé.

### 3.1.4 Pourquoi avoir fait ce choix de modèle ?

Nous avons choisi d'étendre le WT-$K$-means pour plusieurs raisons. Premièrement, la formulation est élégante et bien définie. Le problème d'optimisation est clair : il s'agit optimiser un critère d'inertie pour lequel les poids des variables s'interprètent comme leur contribution à l'inertie. Deuxièmement, l'algorithme des $K$-means est connu pour être peu coûteux en temps de calcul et les méthodes sparses basées sur les GMM qui ont le même avantage font souvent des hypothèses similaires à celles qui sont sous-jacentes à l'algorithme des $K$-means. Troisièmement, le WT-$K$-means est souvent choisi comme algorithme de référence, utilisé pour comparer les résultats d'autres méthodes et aucune autre n'a encore montré clairement sa supériorité. Quatrièmement, la pénalité lasso a une extension directe au cas des variables structurées en groupes, le *group lasso*, que nous pourrons utiliser. Enfin, dans le contexte supervisé, on sait que les méthodes pénalisées sont plus stables et ne souffrent pas d'autant de variabilité que les méthodes de type stepwise par exemple (Friedman et al., 2001). Nous aimerions hériter du succès que ces méthodes ont obtenu en apprentissage supervisé, en les adaptant au cadre non supervisé du clustering.

## 3.2 Détails du modèle WT-$K$-means et extension avec la pénalité *group-lasso*

Dans cette section, toutes les subtilités de l'algorithme WT-$K$-means seront détaillées et ensuite notre extension pour les variables groupées sera présentée.

*https ://cran.r-project.org/web/packages/vimpclust/vignettes/groupsparsewkm.html
†https ://cran.r-project.org/web/packages/vimpclust/index.html

### 3.2.1 Retour sur le modèle WT-$K$-means

FORMULATION Rappelons la formulation du WT-$K$-means qui est un problème peut alors se réécrire sous la forme d'une double optimisation :

$$\underset{C_1,\dots,C_K,\mathbf{w}}{\text{maximiser}}\ \mathbf{w}^\intercal\mathbf{b} - \lambda\|\mathbf{w}\|_1 \ \text{ s.c. } \ \|\mathbf{w}\|_2^2 \leqslant 1, w_j \geqslant 0\ \forall j. \tag{3.1}$$

ALGORITHME ITÉRATIF Le problème (3.1) est un double problème d'optimisation non convexe. Ainsi, la solution envisagée pour optimiser cette équation est un algorithme itératif où les clusters sont trouvés à poids fixés et les poids sont optimisés à clusters fixés. L'algorithme itératif est résumé sous forme de diagramme Figure 3.1 à $K$ et $\lambda$ fixés.

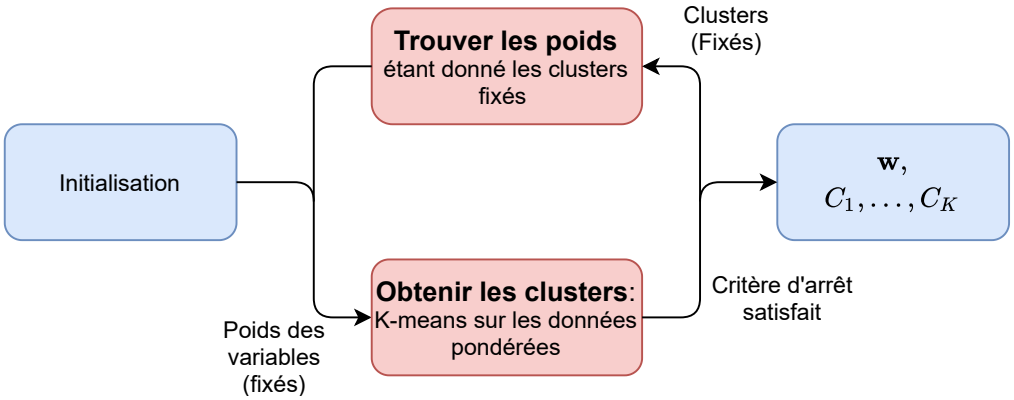



**Figure 3.1 :** Le graphique illustre l'algorithme itératif du WT-$K$-means. Les clusters sont trouvés à poids fixés et les poids sont optimisés à clusters fixés.

Pour aboutir à une solution, le double problème d'optimisation est scindé en deux comme détaillé ci-dessous.

- Initialisation : On initialise les poids tous égaux à $w_j = 1/\sqrt{p}, \forall j = 1, ..., p$ ce qui permet de satisfaire les deux contraintes et on répète l'enchaînement suivant.
  1. Obtenir les clusters : $\underset{C_1,\dots,C_K}{\text{maximiser}}\ \mathbf{w}^\intercal\mathbf{b}$. La pénalité n'apparait plus car les poids sont fixés.
  2. Obtenir les poids : $\underset{\mathbf{w}}{\text{maximiser}}\ \mathbf{w}^\intercal\mathbf{b} - \lambda\|\mathbf{w}\|_1$, s.c. $\|\mathbf{w}\|_2^2 \leqslant 1, w_j \geqslant 0\ \forall j$. Ici les clusters sont fixés donc $\mathbf{b}$ est fixé.
- Arrêt : La procédure itère les étapes 1. et 2. jusqu'à ce que le critère d'arrêt suivant soit satisfait

$$\frac{\left\|\mathbf{w}^r - \mathbf{w}^{r-1}\right\|_1}{\left\|\mathbf{w}^{r-1}\right\|_1} < 10^{-4}, \tag{3.2}$$

où $\mathbf{w}^r = (w_1^r, \dots, w_p^r)^\intercal$ est le vecteur des poids à l'itération $r$.

SOLUTIONS Le fonctionnement de l'algorithme itératif est désormais clair, mais reste à expliciter le double problème d'optimisation. Il faut comprendre pourquoi d'une part résoudre le problème à poids fixés revient à faire un $K$-means sur les données pondérées et d'autre part quelle est la solution obtenue pour les poids lorsque les clusters sont fixés.

1. Trouver les clusters : pour trouver les clusters dans l'étape 1, il faut

$$\underset{C_1,\dots,C_K}{\text{maximiser}}\ \mathbf{w}^\intercal\mathbf{b}.$$

On sait que les $K$-means maximisent l'inertie inter-classes, or les poids peuvent se factoriser dans l'inertie inter-classes pondérées

$$\mathbf{w}^\intercal\mathbf{b} = \sum_{j=1}^{p}\sum_{k=1}^{K}\frac{n_k}{n}(\sqrt{w_j}\times\bar{x}_k^j - \sqrt{w_j}\times\bar{x}^j)^2.$$

L'équation ci-dessus nous apprend que, optimiser l'inertie inter-classes pondérée revient à optimiser l'inertie inter-classes de l'ensemble des données où chaque variable $\mathbf{x}^j$ aurait été multipliée au préalable par $\sqrt{w_j}$. Cela revient donc à effectuer un algorithme $K$-means sur des variables multipliées par un facteur $\mathbf{w}^{\frac{1}{2}}$.

2. Trouver les poids : pour trouver les poids dans l'étape 2, il faut

$$\underset{\mathbf{w}}{\text{maximiser}}\ \mathbf{w}^\intercal \mathbf{b} - \lambda \left\|\mathbf{w}\right\|_1 ,\ \text{s.c.}\ \ \left\|\mathbf{w}\right\|_2^2 \leqslant 1, w_j \geqslant 0\ \forall j.$$

Les conditions d'optimalité s'écrivent sous la forme d'un système Karush-Kuhn Tucker (KKT). En utilisant les conditions KKT comme expliqué dans Boyd and Vandenberghe (2004) on obtient :

$$\mathbf{w}^\star = \begin{cases} \mathbf{0} \text{ si } \lambda \geqslant b_j\ \forall j = 1, \ldots, p, \\ \dfrac{S_\lambda(\mathbf{b})}{\left\|S_\lambda(\mathbf{b})\right\|_2} \text{ sinon,} \end{cases}$$

avec l'opérateur de seuillage doux $S_\lambda(\mathbf{b}) = (\mathbf{b} - \boldsymbol{\lambda})_+$ et $(\mathbf{x})_+ = (\max(0, x_1), \ldots, \max(0, x_p))^\intercal$ que l'on peut aussi écrire comme $S_\lambda(b_j) = (b_j - \lambda)_+$. Cela signifie qu'une variable $j$ est supprimée du modèle si $\lambda > b_j$ et sinon que son poids est proportionnel à $b_j - \lambda$. Les détails de cette solution sont donnés dans l'annexe, Section 3.7.1.

### 3.2.2 Clustering sparse de groupes de variables

INTRODUCTION DE LA PÉNALISATION PAR GROUPE DE VARIABLES Nous généralisons le WT-$K$-means en introduisant la régularisation par groupe. Cela va permettre de prime abord de sélectionner des groupes de variables numériques, mais nous verrons que dans le cadre présenté cela permet aussi de faire de la sélection de variables sur des données mixtes.

Supposons que les $p$ variables sont divisées en $L$ groupes connus à l'avance, tels que $\mathbf{X} = \left[\mathbf{X}^1 | \ldots | \mathbf{X}^L\right]$, avec $\mathbf{X}^l \in \mathbf{R}^{n \times p_l}$, $p_l$ étant la taille du groupe $l$, et $p_1 + ... + p_L = p$. Le vecteur de variance entre classes $\mathbf{b}$ et le vecteur de poids $\mathbf{w}$ peuvent également être décomposés en $\mathbf{b}^\intercal = (\mathbf{b}_1, \ldots, \mathbf{b}_L)$ et $\mathbf{w}^\intercal = (\mathbf{w}_1, \ldots, \mathbf{w}_L)$.

Pour les données de groupe, on définit une pénalité de groupe $\ell_1$ spécifique, qui a déjà été utilisée dans le cadre de la régression Yuan and Lin (2006),

$$h(\mathbf{w}) = \left\|\mathbf{w}\right\|_{1,group} = \sum_{l=1}^{L} \upsilon_l \left\|\mathbf{w}_l\right\|_2 , \tag{3.3}$$

où $\boldsymbol{\upsilon}^\intercal = (\upsilon_1, ..., \upsilon_L)$ est un vecteur de poids appliqué aux groupes de variables. Dans la littérature sur la régression par group sparse, deux choix courants semblent émerger, soit $\upsilon_l = 1$, $\forall l = 1, ..., L$, soit $\upsilon_l = \sqrt{p_l}$ $\forall l = 1, ..., L$. Cette dernière méthode, que nous utiliserons par la suite, consiste à pénaliser chaque groupe par sa taille.

NOUVEAU PROBLÈME D'OPTIMISATION Avec les notations précédentes, le nouveau problème d'optimisation s'écrit comme suit :

$$\underset{C_1,\ldots,C_K,\mathbf{w}}{\text{maximiser}}\ \mathbf{w}^T \mathbf{b} - \lambda \sum_{l=1}^{L} \sqrt{p_l} \left\|\mathbf{w}_l\right\|_2 \text{ s.c.} \left\|\mathbf{w}\right\|_2 \leq 1\ ,\ \mathbf{w} \geq \mathbf{0}. \tag{3.4}$$

L'Équation (3.4) représente bien la version *group-sparse* du problème WT-$K$-means (2.4). Néanmoins, ce problème se présente naturellement sous des formes plus générales (Simon et al., 2013) où les normes $\ell_1$ et la norme $\ell_2$ peuvent être combinées pour former une pénalité connue sous le nom *sparse-group lasso* en régression, qui est une extension de la pénalité *elastic-net* (Zou and Hastie, 2005) en appliquant la pénalité $\ell_2$ à des groupes de variables devenant ainsi une pénalité de groupes Yuan and Lin (2006).

Faisons un point sur la sémantique des termes utilisés. Littéralement, en anglais lasso signifie *least absolute shrinkage and selection operator* ce qui implique déjà l'idée d'une sélection de variable et donc de sparsité. Ainsi, group lasso signifie que l'on applique de la sparsité par groupe de variables et sparse-group lasso correspond à une sparsité entre les groupes de variables (group) et à l'intérieur des groupes (sparse). Dans le cas du clustering, le terme lasso n'intervient pas. Nous faisons alors le choix de nommer la sparsité entre les groupes de variables *group-sparse* et cela donne le modèle Group-Sparse-$K$-means défini par l'Équation (3.4). Étendre le modèle (3.4) à la pénalité combinant les normes $\ell_1$ et $\ell_2$ par groupe, pénalité que l'on nomme *sparse-group-sparse*, donnera naissance au modèle Sparse-Group-Sparse-$K$-means ou SGS-$K$-means pour faire court. Ce modèle permet donc une sparsité entre les groupes et à l'intérieur des groupes de variables. Formellement, il s'écrit :

$$\underset{C_1,\ldots,C_K,\mathbf{w}}{\text{maximiser}}\ \mathbf{w}^\intercal \mathbf{b} - \lambda \Big[(1-\alpha) \sum_{l=1}^{L} \upsilon_l \left\|\mathbf{w}_l\right\|_2 + \alpha \left\|\mathbf{w}\right\|_1\Big] \text{ s.c.} \left\|\mathbf{w}\right\|_1 \leq 1\ ,\ \mathbf{w} \geq \mathbf{0}, \tag{3.5}$$

avec $\lambda \geqslant 0$ le paramètre déterminant le niveau de pénalisation, $\alpha \in [0,1]$ le paramètre qui arbitre entre une pénalisation inter-groupes ($\alpha \to 1$) ou intra-groupes ($\alpha \to 0$). En effet, dans les cas extrêmes où $\alpha = 0$ et $\alpha = 1$, le modèle (3.6) correspond respectivement au modèle WT-$K$-means et Group-Sparse-$K$-means.

RÉSOLUTION DU PROBLÈME La résolution du problème décrit par l'Équation (3.6) est similaire à celle présentée Section 2.4.1. En résumé, les poids sont initialisés tous égaux à $1/\sqrt{p}$ pour satisfaire les contraintes. Ensuite on itère les deux étapes suivantes jusqu'à ce que le critère de convergence défini Section 2.4.1 soit satisfait. La première étape consiste à trouver les clusters à poids fixés en résolvant (3.6) par rapport à $C_1, ..., C_K$, ce qui revient à exécuter les $K$-means sur les données pondérées par $\mathbf{w}$. La deuxième étape consiste à trouver les poids à clusters fixés en résolvant (3.6) par rapport à $\mathbf{w}$, et la solution de ce problème est :

$$\mathbf{w}^\star = \begin{cases} \mathbf{0} \text{ si } \|\tilde{S}\big((\mathbf{b}_j - \lambda\alpha)_+, \lambda\sqrt{\mathbf{p}}(1-\alpha)\big)\|_2 = 0, \\ \dfrac{\tilde{S}\big((\mathbf{b}_j - \lambda\alpha)_+, \lambda\sqrt{\mathbf{p}}(1-\alpha)\big)}{\|\tilde{S}\big((\mathbf{b}_j - \lambda\alpha)_+, \lambda\sqrt{\mathbf{p}}(1-\alpha)\big)\|_2} \text{ sinon,} \end{cases}$$

et donc naturellement la solution du problème Group-Sparse-$K$-means, c'est à dire lorsque $\alpha = 0$ peut s'écrire comme

$$\mathbf{w}^\star = \begin{cases} 0 \text{ si } \|\tilde{S}(\mathbf{b}, \sqrt{\mathbf{p}}\lambda)\|_2 = 0, \\ \dfrac{\tilde{S}(\mathbf{b}, \sqrt{\mathbf{p}}\lambda)}{\|\tilde{S}(\mathbf{b}, \sqrt{\mathbf{p}}\lambda)\|_2} \text{ si } \|\tilde{S}(\mathbf{b}, \sqrt{\mathbf{p}}\lambda)\|_2 \neq 0, \end{cases}$$

où $\sqrt{\mathbf{p}}^\intercal = (\sqrt{p_1}, ..., \sqrt{p_L})$,

$$\tilde{S}(\mathbf{b}, \sqrt{\mathbf{p}}\lambda)^\intercal = \Big(S_g(\mathbf{b_1}, \sqrt{p_1}\lambda)^\intercal, ..., S_g(\mathbf{b_L}, \sqrt{p_L}\lambda)^\intercal\Big) \in \mathbf{R}^p,$$

$$S_g(\mathbf{b_l}, \sqrt{p_l}\lambda) = \frac{\mathbf{b_l}}{\|\mathbf{b_l}\|_2}\left(\|\mathbf{b_l}\|_2 - \sqrt{p_l}\lambda\right)_+ \in \mathbf{R}^{p_l} \ \forall l = 1, ..., L,$$

et $S_g$ est l'opérateur de seuillage doux par groupe. Cela signifie qu'une variable $x^j$ peut être supprimée uniquement si la norme de son groupe $\|\mathbf{b}_\ell\|_2 < \sqrt{p_\ell}\lambda$. Les variables d'un même groupe sont donc sélectionnées toutes ensemble. La résolution des problèmes (2.4), (3.4) et (3.6) est donnée dans l'annexe, Section 3.7.1.

DISCUSSION SUR LE PARAMÈTRE $\lambda$ L'opérateur de seuillage doux par groupe supprime du modèle tous les groupes dont la norme de la variance inter-classes correspondante $\mathbf{b}_\ell$ est inférieure au seuil fixe $\lambda$ normalisé par la taille du groupe, et réduit d'autant les normes des groupes de variables restants. Dans le Chapitre 2, nous avions déjà eu une discussion similaire, où infinité de valeurs de $\lambda$ peuvent être définies mais il est d'usage de fixer une grille de $\lambda$ entre 0 et une valeur maximum $\lambda_{max}$ à déterminer. Witten and Tibshirani (2010) cherchent le $\lambda_{max}$ à l'aide d'une fonction de recherche dichotomique[‡], où *binary search* en anglais. Mais on peut noter que pour la version originale de l'algorithme, $\lambda_{max} = \max(\mathbf{b})$ et que pour notre version group-sparse $\lambda_{max} = \max((\frac{\|\mathbf{b}_\ell\|_2}{v_\ell})_\ell)$ car au-dessus de ces valeurs tous les coefficients sont mis à zéro.

Une remarque tout aussi importante à noter est que $\frac{\tilde{S}\left((\mathbf{b}_j - \lambda\alpha)_+, \lambda(1-\alpha)\right)}{\|\tilde{S}\left((\mathbf{b}_j - \lambda\alpha)_+, \lambda(1-\alpha)\right)\|_2} = \frac{\mathbf{b}}{\|\mathbf{b}\|_2}$ si $\lambda = 0$ et donc l'algorithme ne se réduit pas aux $K$-means standard mais à des $K$-means pondérées car l'optimisation des poids et le fait que les variances inter-classes des variables sont le plus souvent différentes implique des résultats différents du $K$-means standard. Néanmoins à $\lambda = 0$ l'algorithme n'effectue pas de sélection de variable.

DISCUSSION SUR LE PARAMÈTRE $\alpha$ La formulation SGS-$K$-means a été introduite par souci de complétude, d'exhaustivité et de clarté. En effet, cette méthode permet de faire le lien entre plusieurs pénalités : sparse, group-sparse, sparse-group-sparse, elastic-net. Sa résolution donne une solution commune à toutes les méthodes déterminées par ces pénalités. Néanmoins, nous nous restreignons dans la suite à l'utilisation du Group-Sparse-$K$-means ($\alpha = 0$, pénalité group-sparse)

## 3.3 Clustering sparse de données mixtes et groupe de variables

Cette section est donc consacrée à la définition d'une méthode explicite de clustering sparse, dans le cadre de données mixtes, en utilisant un critère de pénalisation.

[‡]https ://github.com/cran/sparcl/blob/master/R/SparseClustering.R

### 3.3.1 Discussion sur le clustering de données mixtes

Le clustering de données mixtes est complexe car il porte sur deux types de variables différents. La première difficulté est de définir un type de normalisation qui permette aux variables d'avoir des contributions équitables. En effet, pour le clustering de variables numériques, si les variables ne sont pas normalisées pour être en variance, l'algorithme de clustering met en évidence la structure en variance des données plutôt que la structure en clusters. Cela est aussi le cas pour les algorithmes de clustering sparses basés sur les K-means, qui de plus nécessitent d'employer une procédure itérative qui prend en compte la contribution des variables à la variance inter-classes.

On cherche donc une normalisation appropriée pour les données mixtes. Mais cela n'est pas facile à définir comme nous pouvons le voir dans ce premier exemple : soit un ensemble de données décrites par une variable numérique qui a une structure en clusters, disons un mélange de deux Gaussiennes, et une variable binaire. Si la variable binaire est très dépendante de la variable numérique, alors elle contient elle aussi l'information sur les clusters et a une structure en clusters. En revanche, si la variable binaire est indépendante de la variable numérique, alors rien ne peut être dit sur la structure de cette variable (elle pourrait être liée à une variable non observée, qui pourrait ou non, avoir une structure en clusters). Le raisonnement inverse peut être fait si la variable numérique n'a pas de structure en clusters.

Des méthodes d'analyse factorielles mixtes (Chavent et al., 2012) existent et exploitent les liens entre les variables de différents types. Malheureusement ce type de méthodes ne peut pas être utilisé ici. En effet, ces méthodes impliquent la création de nouvelles variables qui sont des combinaisons linéaires des variables de départ, et qui expliquent la structure en variance des données. C'est contradictoire avec le but des algorithmes sparses qui cherchent au contraire à diminuer ce nombre. Dans le cas d'analyse factorielles sparses (Witten et al., 2009 ; Chavent and Chavent, 2017), les combinaisons linéaires sparses obtenues obtenues pourraient ne pas correspondre à un choix optimal de sous-espace pour le clustering, sans oublier que ces méthodes requièrent le choix d'au moins deux hyperparamètres, le paramètre contrôlant la sparsité et le second fixant la dimension.

### 3.3.2 Méthodes existantes pour traiter les données mixtes

Il existe peu de méthodes de clustering de données mixtes, notamment avec les $K$-means. Les méthodes développées consistent soit à transformer les données en données numériques et utiliser une distance à priori, soit à utiliser un algorithme de pondération des variables catégorielles.

Une première stratégie (recodage) consiste à d'abord transformer les variables catégorielles, puis à appliquer une méthode conçue pour les données numériques. Pour une variable catégorielle $\mathbf{x}^j$ ayant $c_j$ modalités on crée $c_j$ variables indicatrices distinctes, une pour chacune des modalités, prenant les valeurs 0 ou $a \in \mathbb{R}$ et le problème réside dans la sélection de la constante $a$ (Foss et al., 2016). L'approche usuelle qui consiste à fixer $a = 1$ est arbitraire et ne prend pas en compte la structure des données. Une autre approche consiste à normaliser toutes les variables à variance unitaire ce qui, pour les variables catégorielles, correspond à une normalisation par $1/\sqrt{\frac{n_{j,l}}{n} \times (1 - \frac{n_{j,l}}{n})}$ où $n_{j,l}$ est le nombre de données prenant la $l$-ième valeur pour la $j$-ième variable et donc $\frac{n_{j,l}}{n}$ représente la fréquence de la $l$-ième modalité pour cette variable.

D'autres méthodes de clustering de données mixtes consistent à utiliser une métrique compatible avec les données mixtes, telle que la distance de Gower (Gower, 1971), puis à utiliser une méthode de clustering qui se base sur une matrice de distances telles que la classification ascendante hiérarchique. Cependant, chaque variable dans la distance de Gower doit se voir attribuer un poids spécifié par l'utilisateur qui détermine sa contribution relative à la distance, ce qui présente essentiellement le même dilemme que précédemment.

Plusieurs autres méthodologies pour le clustering de données mixtes ont ce problème d'arbitrage car elles exigent qu'on fixe la contribution relative des variables numériques par rapport aux variables catégorielles. C'est notamment le cas pour la méthode des $K$-prototypes de Huang (1998) qui utilise comme métrique la distance euclidienne au carré pour les variables continues et catégorielles, où les variables catégorielles sont recodées en indicatrices unitaires et pondérées suivant une constante à optimiser.

Enfin, un dernier algorithme de clustering de données mixtes est présenté. Il est décrit dans Modha and Spangler (2003) et est similaire à celui des $K$-prototypes. Il utilise une combinaison linéaire pondérée de la distance euclidienne au carré pour les variables numériques et de la distance cosinus pour les variables catégorielles. La pondération entre les variables numériques et catégorielles est identifiée par une recherche dite *brute force*. Cela requiert donc de multiples exécutions de l'algorithme et la sélection d'un hyperparamètre supplémentaire.

En résumé, il ne semble pas y avoir de solution idéale dans l'absolu. Il faut faire un choix entre fixer des hypothèses sur la structure des données ou ne pas le faire aux dépens de la simplicité et de la rapidité. Sachant que le Group-Sparse-$K$-means est pourvu de deux paramètres $(K, \lambda)$ qui, nous comme nous l'avons vu pour

le WT-$K$-means, sont extrêmement compliqués à optimiser nous faisons le choix d'un recodage à priori des données.

### 3.3.3 Recodage des données

L'idée dans cette section est de présenté un recodage des données. Ce recodage transforme les variables catégorielles pour pouvoir utiliser des algorithmes de clustering utilisant et la distance euclidienne directement sur les données transformées.

Soit $p_1$ le nombre de variables numériques, $p_2$ le nombre de variables catégorielles et $p = p_1 + p_2$ le nombre total de variables. On désigne donc par $\mathbf{X}_1$ la matrice $n \times p_1$ de variables numériques et $\mathbf{X}_2$ la matrice $n \times p_2$ de variables catégorielles où chaque variable $\mathbf{x}^j$ de $\mathbf{X}_2$ possède $c_j$ catégories telles que $\sum_{j=1}^{p_2} c_j = c$ catégories.

Soit $\mathbf{G}_j \in \{0,1\}^{n \times c_j}$ la matrice indicatrice de la $j$-ième variable catégorielle avec $c_j$ catégories et soit $\mathbf{D}_j$ la matrice diagonale des fréquences des catégories de cette variable.

Soit $\mathbf{J} = \mathbf{I}_n - \frac{\mathbf{1}_n \mathbf{1}_n'}{n}$ l'opérateur de centrage où $\mathbf{I}_n$ désigne la matrice d'identité $n \times n$ et $\mathbf{1}_n$ un vecteur $n$ à entrées unitaires.

Soit $\mathbf{G} = (\mathbf{G}_1|, \ldots, |\mathbf{G}_{p_2})$ la matrice $n \times m$ des variables indicatrices des $m$ catégories des $p_2$ variables catégorielles et soit $\mathbf{D} = \text{diag}(\mathbf{D}_1, \ldots, \mathbf{D}_{p_2})$ la matrice diagonale $n \times m$ des fréquences des $m$ catégories.

Nous supposons ici que $\mathbf{X}_1$ centrée et de variance unitaire. Pour normaliser la matrice catégorielle, nous procédons comme dans Chavent et al. (2007), en remplaçant $\tilde{\mathbf{X}}_2$ par la matrice $n \times m$ obtenue en recodant $\mathbf{G}$ de telle sorte que

$$\tilde{\mathbf{X}}_2 = \frac{1}{\sqrt{n}} \mathbf{J}\mathbf{G}\mathbf{D}^{-\frac{1}{2}},$$

puis nous fusionnons les deux matrices pour obtenir la matrice de données $[\mathbf{X}_1, \tilde{\mathbf{X}}_2]$.

En d'autres termes, $\tilde{\mathbf{X}}_2$ doit être centré et normalisé par $1/\sqrt{\frac{n_{j,l}}{n}}$, où $n_{j,l}$ est le nombre de données d'entrée prenant la $l$-ième valeur de la $j$-ième variable. La mise à l'échelle appliquée aux variables indicatrices conduit en fait à utiliser une distance de type $\chi^2$ sur les variables catégorielles, tandis que pour les variables numériques cela revient à utiliser la distance euclidienne.

Le diagramme résumant les étapes nécessaire à la création de la matrice $[\mathbf{X}_1, \tilde{\mathbf{X}}_2]$ sont résumées ci-dessous dans la Figure 3.2. On peut alors remarquer que cette transformation est équivalente à utiliser la distance

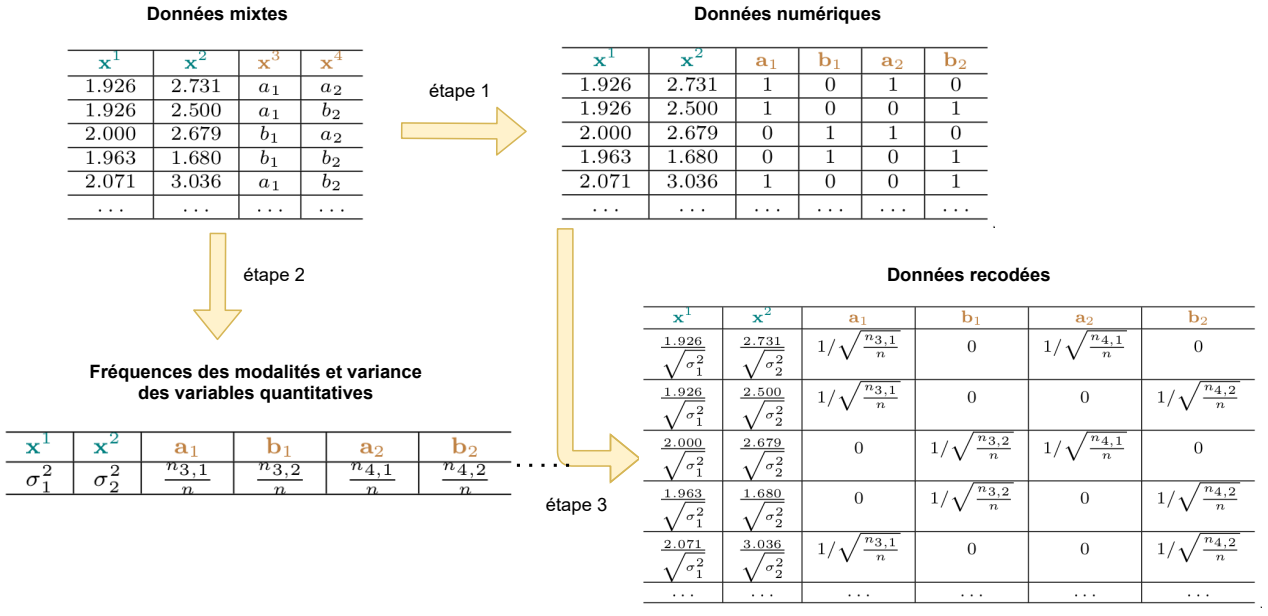



**Figure 3.2 :** Diagramme illustrant les étapes de recodage des données sur une matrice avec 4 variables dont deux variables numériques et deux variables catégorielles. La première étape consiste à transformer les modalités des variables catégorielles en indicatrices. La deuxième étape consiste à calculer la fréquence des modalités et la variance des variables numériques. Enfin, la troisième étape consiste à normaliser les données numériques par la racine carrée de la fréquence des modalités ou de la variance selon le type de variable.

euclidienne avec la métrique $\mathbf{M}^{-1}$ où $\mathbf{M} = \text{diag}(\sigma_1, \ldots, \sigma_{p_1}, \frac{n_{1,c_1}}{n}, \ldots, \frac{n_{p,c_p}}{n})$ sur le tableau de données $[\mathbf{X}_1, \mathbf{G}]$.

Comme on peut le constater dans cette section ou sur le diagramme 3.2, le recodage des données mixtes définit naturellement des groupes de variables. Chaque variable catégorielle donne lieu à un groupe de variables

indicatrices représentant la présence ou l'absence d'une modalité. Alors l'utilisation d'algorithmes classiques de sélection de variables tel que le WT-$K$-means ne va pas nécessairement sélectionner toutes les variables indicatrices associées à une variable et n'aboutira pas dans ce cas à une véritable sélection de variables. De plus, il n'est pas directement possible de comprendre l'importance d'une variable catégorielle si seulement une partie de ses modalités est sélectionnée. C'est pourquoi le Group-Sparse-$K$-means est tout à fait adapté à ce type d'application.

## 3.4 Illustration du package `vimpclust` sur des données réelles

Dans cette section, nous présentons le package **vimpclust** qui implémente, entre autres, la méthode Group-Sparse-$K$-means que nous proposons dans ce manuscrit et qui a été décrite dans Chavent et al. (2020). Ce package dispose de deux fonctions principales, **sparsewkm** qui permet de faire du clustering sparse de données numériques et/ou catégorielles et **groupsparsewkm** qui permet de faire du clustering sparse de groupes de variables numériques. Ces méthodes généralisent l'algorithme de Witten and Tibshirani (2010). Notamment si les variables utilisées pour le clustering sont uniquement numériques sans structure de groupe a priori, l'algorithme de clustering se réduit à celui de Daniella Witten et Robert Tibshirani. Si certaines ou toutes les variables sont catégorielles, **sparsewkm** sparsewkm transforme les données en utilisant une étape d'analyse factorielle décrite dans la Section 3.3.3 ci-avant et c'est cette fonction que nous illustrons ici.

Commençons par décrire la fonction et ses arguments. Plusieurs arguments peuvent être donnés en entrée de à **sparsewkm**, mais seuls les deux premiers sont requis,

- **X** : est la matrice des données. Les données doivent sous le format **data.frame** et les variables catégorielles en format **factor**.
- **centers** : est le nombre de clusters $K$ à calculer.

Les autres arguments sont liés aux choix du paramètre de régularisation, ou du nombre d'itérations et d'initialisations aléatoires de l'algorithme. Les valeurs par défaut sont fixées pour ces paramètres et plus d'informations sont disponibles en utilisant l'aide **help(sparsewkm)**.

Pour illustrer le fonctionnement du package, nous utilisons les données sur les maladies cardiaques HDdata* qui décrivent 270 patients décrits par six variables numériques et huit variables catégorielles.

La fonction **sparsewkm** est appliquée aux données HDdata sur toutes les variables sauf la dernière, **hd**, qui code la présence ou l'absence d'une maladie cardiaque. On fixe le nombre de clusters à deux. Ci-dessous est présenté le code permettant d'exécuter la fonction et d'afficher deux graphiques, le chemin de régularisation qui montre le poids des variables en fonction des valeurs du paramètre $\lambda$ et le chemin de variance expliquée des données non pondérées en fonction des valeurs du paramètre $\lambda$. C'est ce graphe qui est utilisé par l'algorithme de détection de rupture pour le choix du paramètre $\lambda$.

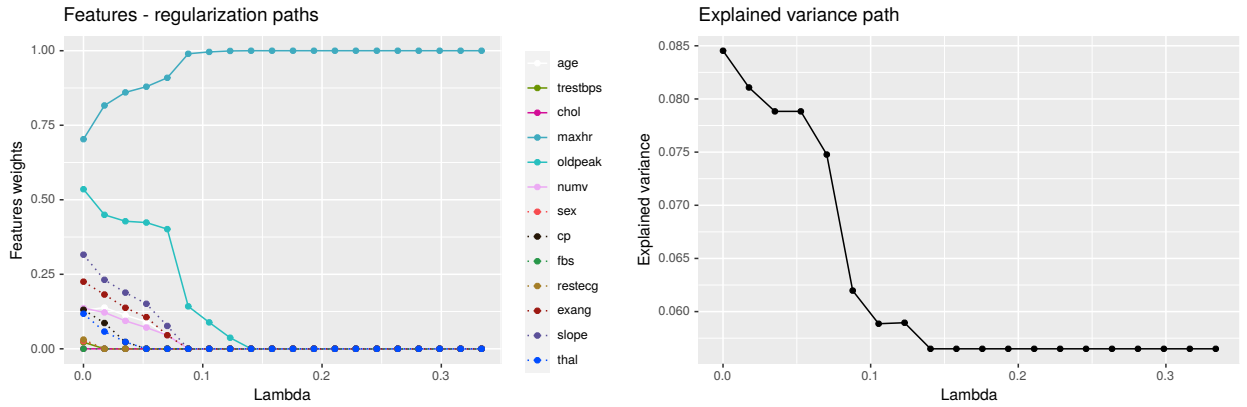
```
res <- sparsewkm(X = HDdata, centers = 2)
plot(res, what="weights.features")
plot(res, what="expl.var")
```

Les graphiques sont présentés sur la Figure 3.3 : le graphique (a) indique le chemin de régularisation c'est-à-dire le poids des variables en fonction des valeurs de $\lambda$, le graphique (b) représente le chemin de variance expliqué en fonction des valeurs de $\lambda$. On peut voir que, lorsque $\lambda$ augmente, les poids de certaines variables atteignent la valeur zéro. Par défaut, les chemins associés aux variables numériques sont affichés avec des lignes continues, tandis que ceux associés aux variables catégorielles sont affichés avec des lignes pointillées.

D'après les résultats, les variables numériques **maxhr** et **oldpeak**, et les variables catégorielles **slope** et **exang** apparaissent comme les plus discriminantes pour de petites valeurs de $\lambda$. Lorsque $\lambda$ augmente, seul **maxhr** est sélectionné par l'algorithme. L'algorithme de détection de rupture choisit la huitième valeur de $\lambda$ en partant de la gauche. D'autres graphiques sont disponibles, comme le chemin de régularisation des modalités qui fournit une image plus détaillée de la façon dont chaque modalité d'une variable catégorielle contribue au clustering et le comme nombre de variables sélectionnées en fonction des valeurs de $\lambda$.

LA NORMALISATION DES DONNÉES Comme on l'a vu dans la Section 2.6, la variance des variables affecte énormément le clustering, et encore d'avantage le clustering sparse. Spécifiquement, pour la méthodologie destinée aux données mixtes, étant donné le recodage que l'on effectue en entrée de l'algorithme (voir section suivante), il n'y a plus besoin de normaliser les données par la suite. Mais lorsque l'algorithme est utilisée sur des groupes de variables numériques (non recodées), à priori les données ne sont pas transformées en amont et

*https ://archive.ics.uci.edu/ml/datasets/statlog+(heart)




(a) (b)

**Figure 3.3 :** Le graphique (a) représente le chemin de régularisation c'est à dire le poids des variables en fonction des valeurs de $\lambda$, le graphique (b) représente le chemin de variance expliqué en fonction des valeurs de $\lambda$.

il faut donc normaliser les variables à moyenne nulle et variance unitaire. Tous ces détails sont pris en charge automatiquement dans le package `vimpclust` qui implémente nos solutions.

INIALISATION DE L'ALGORITHME Dans la suite de ce qui a été discuté dans la Section 2.6, nous avons implémenté dans notre package `R vimpclust`, la possibilité à de renseigner le nombre de fois que le $K$-means standard doit être relancé (comme expliqué dans l'introduction 2.2.2) et nous avons développé dans une version bêta une initialisation au moyen de l'algorithme $K$-means++. Ces fonctionnalités ne sont pas disponibles dans le package de Daniela Witten `sparcl`. Cette proposition est utile car on constate que l'initialisation a un très grand impact sur les résultats de l'algorithme, et de manière générale des algorithmes sparses. La version avec l'initialisation $K$-means++ n'a pas été utilisé dans ce manuscrit et donc les différences observées entre les fonctions des deux packages codant le WT-$K$-means sont bien dû au choix du paramètre $\lambda$.

## 3.5 Simulations : comparaison des méthodes de clustering sparse sur des données mixtes

Dans cette section nous allons comparer deux méthodes de clustering sparse qui traitent des données mixtes : la méthode du package `VarSelLCM` (Marbac and Sedki, 2017) décrite dans la Section 2.5.4 et `vimpclust` (Chavent et al., 2020) qui est le Group-Sparse-$K$-means avec détection de rupture pour le choix du paramètre $\lambda$. On reprend le même schéma de simulation que celui qui a été décrit par l'Équation 2.11, et on discrétise en variables binaires (deux catégories) la moitié des variables importantes et la moitié des variables de bruit. L'opération est simple : puisque par définition toutes les variables sont centrées et de médiane zéro on discrétise les variables de la façon suivantes :

$$\tilde{x}_i^j = \begin{cases} 1 & \text{si} \quad x_i^j > 0, \\ 0 & \text{sinon.} \end{cases}$$

Nous testons les deux scénarios, avec $K = 2, p_K = 2, d = 20, m = 1.5$ où donc une des deux variables importantes et dix variables de bruit sont discrétisées et $K = 2 p_K = 10; d = 100$ où cinq variables importantes et dix variables de bruit sont discrétisées. La valeur $m$ est indiquée dans le tableau.

La Table 3.1 résume les résultats obtenus suivant les différents scénarios. Les résultats sont clairs, les méthodes sont équivalentes en terme de performances mais pas en termes de sélection de variables. En effet, Vimpclust a tendance à ne pas sélectionner les variables catégorielles. On peut expliquer ce fait par deux arguments : le premièr c'est qu'en général les variables catégorielles se voient attribuer un poids plus faible que les variables numériques. Le second est une conséquence du premier, car si les variables catégorielles ont un poids plus faible, elles affectent moins le chemin de variance inter-classes et le choix du paramètre $\lambda$ se fait en grande partie à l'aide des variables numériques, ce que l'on vérifie dans les simulations.

## 3.6 Conclusion

Dans ce chapitre,

**Table 3.1 :** Le tableau représente les moyennes et écart type par méthode pour le scénario $K = 2, p_K = 10, d = 100, m = 1.7$ sur 20 simulations de l'ARI, le ratio de variables importantes (Ratio V.Imp) et de bruit (Ratio V.Bruit) sélectionnées et le temps de calcul (Temps).

| | scénario | | | ARI | | Ratio V.Imp | | Ratio V.Bruit | | Temps | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| noms | $p_k$ | $d$ | $m$ | moyenne | sd | moyenne | sd | moyenne | sd | moyenne | sd |
| VarSelLCM | 2 | 20 | 1.5 | 0.92 | 0.06 | 1.00 | 0.00 | 0.02 | 0.03 | 1.06 | 0.13 |
| Vimpclust | 2 | 20 | 1.5 | 0.84 | 0.09 | 0.50 | 0.00 | 0.03 | 0.02 | 0.42 | 0.06 |
| VarSelLCM | 10 | 100 | 0.6 | 0.74 | 0.12 | 0.97 | 0.07 | 0.03 | 0.02 | 4.75 | 0.31 |
| Vimpclust | 10 | 100 | 0.6 | 0.42 | 0.10 | 0.30 | 0.15 | 0.01 | 0.02 | 2.17 | 0.48 |
| VarSelLCM | 10 | 100 | 0.7 | 0.77 | 0.28 | 0.99 | 0.03 | 0.02 | 0.02 | 3.92 | 0.35 |
| Vimpclust | 10 | 100 | 0.7 | 0.57 | 0.15 | 0.35 | 0.14 | 0.01 | 0.02 | 1.95 | 0.50 |
| VarSelLCM | 10 | 100 | 0.8 | 0.93 | 0.04 | 1.00 | 0.00 | 0.02 | 0.02 | 3.54 | 0.18 |
| Vimpclust | 10 | 100 | 0.8 | 0.62 | 0.25 | 0.28 | 0.18 | 0.00 | 0.01 | 1.93 | 0.32 |
| VarSelLCM | 10 | 100 | 0.9 | 0.96 | 0.05 | 1.00 | 0.00 | 0.03 | 0.02 | 3.26 | 0.18 |
| Vimpclust | 10 | 100 | 0.9 | 0.75 | 0.19 | 0.30 | 0.16 | 0.00 | 0.00 | 1.65 | 0.34 |
| VarSelLCM | 10 | 100 | 1 | 0.98 | 0.04 | 1.00 | 0.00 | 0.03 | 0.01 | 3.16 | 0.15 |
| Vimpclust | 10 | 100 | 1 | 0.88 | 0.17 | 0.41 | 0.15 | 0.01 | 0.02 | 1.51 | 0.29 |

## 3.7 Annexe

### 3.7.1 Solution du Group-Sparse-$K$-means

Nous allons dans cette section donner la solution du problème Group-Sparse-$K$-means explicité dans ce chapitre. Dans un soucis de généralité et pour rendre cette section plus compacte, nous allons écrire le problème sous une forme plus générale que l'on appellera Sparse-Group-Sparse-$K$-means qui mélange deux pénalités deux normes $\ell_1$ et $\ell_2$ ce qui revient à utiliser la même forme de pénalisation que le sparse group lasso en régression Simon et al. (2013). Cette pénalisation permet de pénaliser par groupe mais aussi à l'intérieur des groupes. Malheureusement elle introduit un paramètre supplémentaire qui semble trop compliqué à optimiser. Néanmoins, ce cadre offre la possibilité d'obtenir les solutions aux problèmes WT-$K$-means, Group-Sparse-$K$-means et Sparse-Group-Sparse-$K$-means avec une seule formulation.

Supposons que les $p$ variables soient divisées en $L$ groupes connus à l'avance, tels que $\mathbf{X} = \left[\mathbf{X}^1 | \dots | \mathbf{X}^L\right]$, avec $\mathbf{X}^\ell \in \mathbf{R}^{n \times p_\ell}$, $p_\ell$ étant la taille du groupe $\ell$, et $p_1 + \dots + p_L = p$. Le vecteur de variance entre classes $\mathbf{b}$ et le vecteur de poids $\mathbf{w}$ peuvent également être décomposés en $\mathbf{b}^\intercal = (\mathbf{b}_1, \dots, \mathbf{b}_L)$ et $\mathbf{w}^\intercal = (\mathbf{w}_1, \dots, \mathbf{w}_L)$.

On définit l'algorithme Sparse-Group-Sparse-$K$-means comme solution du problème suivant :

$$\underset{C_1,\dots,C_K,\mathbf{w}}{\text{maximiser}} \ \mathbf{w}^\intercal\mathbf{b} - \lambda\Big[(1-\alpha)\sum_{\ell=1}^{L} \upsilon_\ell \|\mathbf{w}_\ell\|_2 + \alpha\|\mathbf{w}\|_2\Big] \ \text{s.c.} \|\mathbf{w}\|_1 \leq 1 \ , \ \mathbf{w} \geq \mathbf{0}, \tag{3.6}$$

où ici nous supposerons $\upsilon_\ell = 1$ par soucis de clarté. Rappelons que la procédure itérative est en deux étapes, où dans une étape les poids sont fixés.

SOLUTION Présentons à présent la solution au problème. Tout d'abord, Nous pourrions réécrire le problème dans sa formulation complète avec contraintes et comme un problème de minimisation :

$$\underset{C_1,\dots,C_K,\mathbf{w}}{\text{minimiser}} \ -\mathbf{w}^\intercal\mathbf{b} + \lambda\Big[(1-\alpha)\sum_{\ell=1}^{L} \|\mathbf{w}_\ell\|_2 + \alpha\|\mathbf{w}\|_1\Big] \ \text{s.c.} \ \|\mathbf{w}\|_2^2 \leqslant 1, \mathbf{w} > \mathbf{0}.$$

Soit $\mathcal{L}(\mathbf{w}, \lambda, \gamma) = f(\mathbf{w}) + \gamma g(\mathbf{w})$, où $f(\mathbf{w}) = -\mathbf{w}^\intercal\mathbf{b} + \lambda\Big[(1-\alpha)\sum_{\ell=1}^{L} \|\mathbf{w}_\ell\|_2 + \alpha\|\mathbf{w}\|_1\Big]$, $g(\mathbf{w}) = \|\mathbf{w}\|_2^2 - 1$. La solution doit satisfaire aux conditions de Karush-Kuhn-Tucker (KKT) suivantes :

$$\begin{aligned} &\frac{\partial\mathcal{L}(\mathbf{w}, \lambda, \gamma)}{\partial\mathbf{w}} = 0, \\ &\gamma \geqslant 0, \\ &g(\mathbf{w}) \leqslant 0, \\ &\gamma g(\mathbf{w}) = 0, \end{aligned}$$

où $\gamma$ est le multiplicateur de Lagrange. Cependant, la norme $\ell_1$ n'a pas de dérivée à 0. Ainsi, la première condition KKT s'écrit comme suit :

$$-\mathbf{b}_\ell + 2\gamma\mathbf{w}_\ell + \lambda\Big[(1-\alpha)\Gamma^2(\mathbf{w}_\ell) + \alpha\Gamma^1(\mathbf{w}_\ell)\Big] = 0, \tag{3.7}$$

où $\Gamma^1(\mathbf{w}_\ell)$ est la sous-différentiel du groupe $\ell$ évaluée en $\mathbf{w}_\ell$et $\Gamma^1(\mathbf{w}_\ell)$ est définie comme :

$$\Gamma^1(\mathbf{w}_\ell) = \big(\Gamma^1(w_{\ell_1}), ..., \Gamma^1(w_{\ell_{\mathbf{p}_\ell}})\big)^\intercal \in \mathbb{R}^{p_\ell} \text{ avec } \Gamma^1(w_{\ell_j}) = \begin{cases} 1 \text{ si } w_{\ell_j} > 0, \\ c \in [-1; 1] \text{ si } w_{\ell_j} = 0, \\ -1 \text{ si } w_{\ell_j} < 0, \end{cases}$$

où le troisième cas n'est pas possible ici avec la variable $j$ appartenant forcément au groupe $\ell$, $\mathbf{x}^j \in \mathbf{X}^\ell$, et $\Gamma^2(\mathbf{w}_\ell)$ est la sous-différentiel de la norme $\ell_2$ et $\Gamma^2(\mathbf{w}_\ell)$ est définie comme :

$$\Gamma^2(\mathbf{w}_\ell) = \begin{cases} \dfrac{\mathbf{w}_\ell}{\|\mathbf{w}_\ell\|_2} \text{ si } \|\mathbf{w}_\ell\|_2 \neq 0, \\ \{\mathbf{u}, \|\mathbf{u}\|_2 \leqslant 1\} \text{ si } \|\mathbf{w}_\ell\|_2 = 0. \end{cases}$$

Nous pouvons réécrire l'Équation 3.7 comme

$$2\gamma\mathbf{w}_\ell = \mathbf{b}_\ell - \lambda\Big[(1-\alpha)\Gamma^2(\mathbf{w}_\ell) + \alpha\Gamma^1(\mathbf{w}_\ell))\Big]. \tag{3.8}$$

Tout d'abord considérons deux cas qui dépendent de la pénalité $\lambda$ :

1. si $\lambda = 0$, le problème se résout directement avec $w_j = \dfrac{b_j}{\|\mathbf{b}\|_2}$. En effet,

$$\sum_{j=1}^{p} w_j^2 = 1 \iff \sum_{j=1}^{p} b_j^2 \times (2\gamma)^{-2} = 1 \iff 2\gamma = \sqrt{\sum_{j=1}^{p} b_j^2}.$$

2. si $\lambda > 0$ alors il faut encore simplifiée l'Équation 3.8 sous une forme d'opérateur de seuillage. Les sous-différentiels ne sont pas évaluables en 0 et donc nous devons distinguer les deux cas suivants :
   (a) $\|\mathbf{w}_\ell\|_2 = 0$ si $\|(\mathbf{b}_\ell - \lambda\alpha)_+\|_2 \leqslant \lambda(1-\alpha)$ avec un petit travail algébrique et on obtient comme solution de l'Équation 3.8 avec $\Gamma^2(\mathbf{w}_\ell) = \frac{(\mathbf{b}_\ell - \lambda\alpha)_+}{\lambda(1-\alpha)}$ et $\Gamma^1(\mathbf{w}_\ell) = \frac{\mathbf{b}_\ell - (\mathbf{b}_\ell - \lambda\alpha)_+}{\lambda\alpha}$.
   (b) $\|\mathbf{w}_\ell\|_2 \neq 0$ si $\|\mathbf{w}_\ell\|_2 = 0$ si $\|(\mathbf{b}_\ell - \lambda\alpha)_+\|_2 \geqslant \lambda(1-\alpha)$ et l'Équation 3.8 s'écrit comme,

$$\mathbf{w}_\ell\Big(2\gamma + \frac{\lambda(1-\alpha)}{\|\mathbf{w}_\ell\|_2}\Big) = \mathbf{b}_\ell - \lambda\alpha\Gamma^1(\mathbf{w}_\ell). \tag{3.9}$$

   De cette équation nous pouvons décrire deux nouveaux cas si nous obervons particulièrement une variable la variable $j$ appartenant au groupe $\ell$, $\mathbf{x}^j \in \mathbf{X}^\ell$ :
   i. $w_j = 0$ si $b_j \leqslant \lambda\alpha$ pour tout $j$ dans le groupe $\ell$ et $\Gamma^1(w_{\ell_j}) = \frac{b_j}{\lambda\alpha}$.
   ii. $w_j > 0$ si $b_j > \lambda\alpha$ pour tout $j$ dans le groupe $\ell$ et $w_j\Big(2\gamma + \frac{\lambda(1-\alpha)}{\|\mathbf{w}_\ell\|_2}\Big) = b_j - \lambda\alpha$.
   En joignant les deux cas précédents on obtient :

$$w_j\Big(2\gamma + \frac{\lambda(1-\alpha)}{\|\mathbf{w}_\ell\|_2}\Big) = (b_j - \lambda\alpha)_+ \tag{3.10}$$

   Désormais nous avons résolu le problème de la non différentiabilité et nous pouvons décrire le problème dans le cas $\lambda > 0, \|\mathbf{w}_\ell\|_2 \neq 0, w_j > 0$. Nous pouvons ainsi repartir de l'Équation 3.10 et nous distinguons encore deux cas :
   A. si $\gamma > 0$ alors on peut réécrire l'Équation 3.10 sous forme vectorielle :

$$\mathbf{w}_\ell\Big(2\gamma + \frac{\lambda(1-\alpha)}{\|\mathbf{w}_\ell\|_2}\Big) = (\mathbf{b}_\ell - \lambda\alpha)_+ \iff 2\gamma\|\mathbf{w}_\ell\|_2 = \|(\mathbf{b}_j - \lambda\alpha)_+\|_2 - \lambda(1-\alpha) \tag{3.11}$$

   Ainsi nous pouvons injecter l'Équation 3.11 dans 3.10 et nous obtenons :

$$\mathbf{w}_\ell = \frac{1}{2\gamma}\frac{(\mathbf{b}_j - \lambda\alpha)_+}{\|(\mathbf{b}_j - \lambda\alpha)_+\|_2}\Big(\|(\mathbf{b}_j - \lambda\alpha)_+\|_2 - \lambda(1-\alpha)\Big) \tag{3.12}$$

$$= \frac{1}{2\gamma}\tilde{S}\big((\mathbf{b}_j - \lambda\alpha)_+, \lambda(1-\alpha)\big) \tag{3.13}$$

   or $\gamma > 0$ implique $\|\mathbf{w}\|_2 = 1$ donc

$$\mathbf{w}_\ell = \frac{\tilde{S}\big((\mathbf{b}_j - \lambda\alpha)_+, \lambda(1-\alpha)\big)}{\|\tilde{S}\big((\mathbf{b}_j - \lambda\alpha)_+, \lambda(1-\alpha)\big))\|_2} \tag{3.14}$$

B. si $\gamma = 0$ on obtient directement par l'Équation 3.11 que $\mathbf{w} = 0$.

Ainsi si on résume, on a que

$$\mathbf{w}^\star = \begin{cases} \mathbf{0} \text{ si } \|\tilde{S}\big((\mathbf{b}_j - \lambda\alpha)_+, \lambda(1-\alpha)\big)\|_2 = 0, \\ \dfrac{\tilde{S}\big((\mathbf{b}_j - \lambda\alpha)_+, \lambda(1-\alpha)\big)}{\|\tilde{S}\big((\mathbf{b}_j - \lambda\alpha)_+, \lambda(1-\alpha)\big)\|_2} \text{ sinon,} \end{cases}$$

avec $\frac{\tilde{S}\left((\mathbf{b}_j - \lambda\alpha)_+, \lambda(1-\alpha)\right)}{\|\tilde{S}\left((\mathbf{b}_j - \lambda\alpha)_+, \lambda(1-\alpha)\right)\|_2} = \frac{\mathbf{b}}{\|\mathbf{b}\|_2}$ si $\lambda = 0$ et $\gamma$ toujours choisi supérieur à 0 pour une solution non-triviale.

Désormais, nous pouvons donner la solution du problème WT-$K$-means, c'est à dire lorsque $\alpha = 1$, qui peut s'écrire comme

$$\mathbf{w}^\star = \begin{cases} \mathbf{0} \text{ si } \lambda \geqslant b_j \ \forall j = 1, \ldots, p, \\ \dfrac{(\mathbf{b}_j - \lambda)_+}{\|(\mathbf{b}_j - \lambda)_+\|_2} = \dfrac{S(\mathbf{b}_j, \lambda)}{\|S(\mathbf{b}_j, \lambda)\|_2} \text{ sinon,} \end{cases}$$

$S$ étant l'opérateur de seuillage doux et nous pouvons donner la solution du problème Group-Sparse-$K$-means, c'est à dire lorsque $\alpha = 0$, qui peut s'écrire comme

$$\mathbf{w}^\star = \begin{cases} \mathbf{0} \text{ si } \|\tilde{S}\big(\mathbf{b}_j, \lambda\big)\|_2 = 0, \\ \dfrac{\tilde{S}\big(\mathbf{b}_j, \lambda\big)}{\|\tilde{S}\big(\mathbf{b}_j, \lambda\big)\|_2} \text{ sinon.} \end{cases}$$

# 4

# Clustering sparses et données corrélées



## 4.1 Introduction

Les méthodes de clustering sont essentielles pour réaliser une analyse complète de données multivariées. Le clustering multivarié n'est utile que si les variables sont dépendantes. En effet, si les variables sont indépendantes, les données ne sont pas vraiment en grande dimension ou multivariées, mais doivent être considérées

plutôt comme une collection de variables univariées non liées. En revanche, s'il existe des corrélations, une analyse de clustering sur l'ensemble des variables peut s'avérer intéressante. Nous abordons le problème à l'aide d'un exemple introductif.

### 4.1.1 Un exemple introductif

Nous simulons un exemple qui illustre le problème des données corrélées. Soit $\mathbf{X}$ une matrice avec $n = 500$ observations et $p = 3$, simulée selon le mélange de deux Gaussiennes suivant :

$$\sum_{k=1}^{2} 0.5 \times \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \text{ avec } \boldsymbol{\mu}_k = \begin{pmatrix} 7 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \text{ et } \boldsymbol{\Sigma}_k = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0.5 \\ 0 & 0.5 & 1 \end{pmatrix} \text{ pour } k = 1, 2.$$

où $\boldsymbol{\mu}_k$ et $\boldsymbol{\Sigma}_k$ sont respectivement les moyennes et les matrices de covariance des deux distributions. Ainsi les centres sont séparés uniquement sur $\mathbf{x}^1$ et les variables $\mathbf{x}^2, \mathbf{x}^3$ sont moyennement corrélées.

On normalise maintenant la matrice $\mathbf{X}$ pour que les variables soient de moyenne nulle et de variance unitaire. Certains affirment qu'il ne faut pas normaliser des données simulées car par définition la structure voulue peut être simulée au préalable et tout peut être indiqué dans le modèle. Ici, nous décidons de normaliser car c'est généralement une opération qui est faite sur les données réelles et cela va permettre d'illustrer en détail notre propos.

La Figure 4.1 montre les résultats du meilleur clustering 2-means, c'est-à-dire celui qui maximise la fonction objectif sur le jeu de données (en répétant sur plusieurs milliers d'initialisations aléatoires) et les 2-means ne parviennent pas à retrouver les clusters sous-jacents. On observe que lorsque les données ne sont pas normalisées l'algorithme des $K$-means retrouvent les clusters car la variance de $\mathbf{x}^1$ est grande par rapport aux autres variables. Néanmoins, nous aurions pu simuler des composantes plus rapprochées avec une variance par composante sur la première dimension beaucoup plus faible pour obtenir le même résultat qu'après normalisation.

Il est clair que si on ne retient que les variables $\mathbf{x}^1, \mathbf{x}^2$, on obtient les résultats attendus au départ. Et, il est surprenant que l'ajout d'une seule variable change complètement les résultats.

Malgré la popularité de l'algorithme des $K$-means, on sait très peu de choses sur son comportement lorsque les données sont corrélées. On pourrait penser qu'il s'agit d'un problème inhérent à l'algorithme des K-means. Cependant, d'autres algorithmes de clustering bien connus se comportent de manière similaire et reproduisent les mêmes résultats.

### 4.1.2 Le constat et les motivations

LES CORRÉLATIONS AFFECTENT LE CLUSTERING La motivation principale de ce chapitre peut se résumer par l'énoncé suivant : les corrélations affectent les résultats du clustering et la présence de corrélations ne doit pas être négligée. Des travaux antérieurs ont déjà prouvé que l'algorithme des $K$-means est fortement lié à l'ACP (Ding and He, 2004) et ont conclu qu'un *bon clustering* est celui dont les $K$ centres appartiennent au sous-espace des $K - 1$ premières composantes principales : "a "good clustering" is one whose representation lies close to the principal subspace". L'intuition derrière cette assertion est que les premières composantes principales sont celles qui ont les plus grandes variances et donc la variance inter-classes sera plus grande dans ce sous espace ""well separated" refers not only to the distances between the clusters, but to the volume (of the polyhedron) spanned by them, which should be as large as possible" (Meilă, 2006).

LES MÉTHODES COMBINANT ACP ET $K$-MEANS POUR AMÉLIORER LE CLUSTERING Sur la base de ces résultats, d'innombrables études ont été réalisées pour améliorer les $K$-means à l'aide de l'ACP. Certains articles proposent d'initialiser les $K$-means à l'aide de l'ACP. Une des premières études à ce sujet est celle de Su and Dy (2004) qui développent une initialisation déterministe basée sur l'ACP. La procédure est simple, il suffit de couper en zéro la première composante principale pour former deux clusters, puis choisir le cluster dont première la composante principale à la plus grande variance (pour chaque cluster on réalise une ACP et on compare les premières valeurs propres de chacune) et enfin de couper à nouveau en son centre. La procédure continue jusqu'à ce qu'une partition en $K$ clusters soit obtenue ce qui permet de calculer les centres initiaux dans l'espace de départ. D'autres auteurs proposent de simplement d'exécuter un $K$-means sur les $K$ principales composantes pour déterminer les clusters qui initialiseront les centres dans l'espace de départ (Xu et al., 2015). Leur méthodologie est appuyée par une analyse théorique démontrant "the optimal solution lies in the $K$-dimensional PCA subspace (within the accuracy of spectral relaxation)". D'autres ont aussi combiné les algorithmes ACP et $K$-means dans des procédures itératives (Honda et al., 2008, 2009).

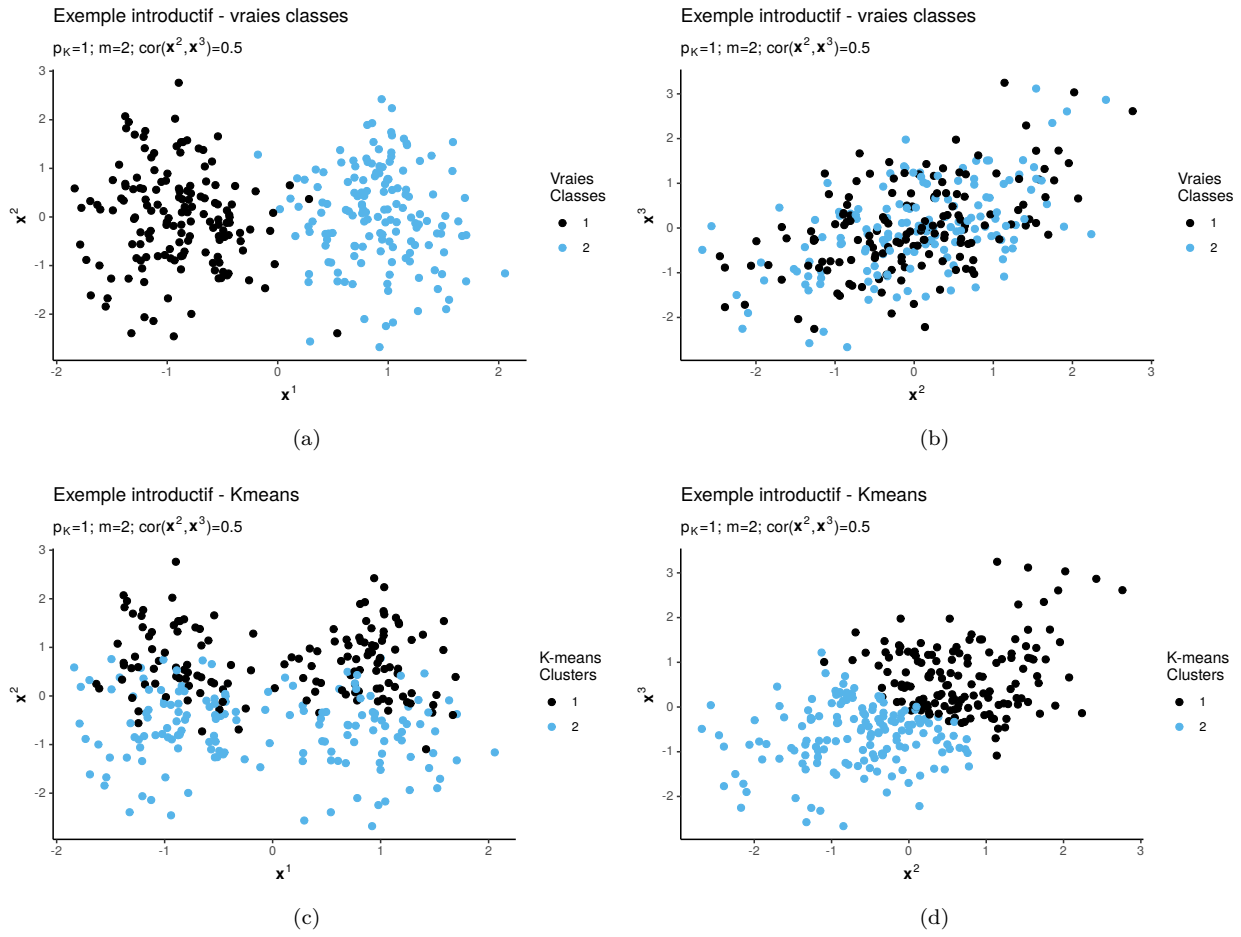



**Figure 4.1 :** Données simulées où $\mathbf{x}^1$ est sépare bien les deux clusters et où $\mathbf{x}^2, \mathbf{x}^3$ sont légèrement corrélés ($\text{cor}(\mathbf{x}^1, \mathbf{x}^2) = 0.5$). (a) représente les vraies classes représentées sur le plan $\mathbf{x}^1, \mathbf{x}^2$ ; (b) représente les vraies classes représentées sur le plan $\mathbf{x}^2, \mathbf{x}^3$ ; (c) représente les clusters trouvés par les 2-means représentées sur le plan $\mathbf{x}^1, \mathbf{x}^2$ ; (d) représente les clusters trouvés par les 2-means représentées sur le plan $\mathbf{x}^2, \mathbf{x}^3$. Les clusters sont le résultat des meilleurs 2-means (sur un millier d'initialisations aléatoires). L'optimum des $K$-means est atteint en divisant la Gaussienne corrélée suivant la direction donnée par $\mathbf{x}^2, \mathbf{x}^3$ et les clusters sous-jacents n'ont pas été trouvés.

Le lien entre ACP et $K$-means met en lumière le problème des données corrélées Le lien entre l'ACP et les $K$-means met en évidence certains problèmes. L'algorithme des $K$-means peut être utilisé pour trouver des clusters au sens décrit dans l'introduction de ce manuscrit, c'est-à-dire des zones de forte densité séparées par des zones de faible densité. Or la littérature décrivant les liens entre $K$-means et ACP nous révèle que ce qui compte le plus c'est la variance du sous espace dans lequel résident les centres. Alors, on peut se demander quels sont les avantages de l'utilisation de la méthode des $K$-means par rapport à celle de l'ACP, si la première n'est que dans une certaine mesure une version discrète de la seconde qui de surcroît n'a pour objectif que de comprendre la structure en variance des données. Manifestement, si l'objectif principal est de découvrir la structure en clusters, c'est-à-dire des régions de densités différentes, il serait nécessaire de prendre en compte les corrélations entre les variables. Étant donné que l'ACP maximise l'information restituée en représentant les données dans un sous-espace d'inertie maximale il faut adopter un point de vue différent, arguant que les corrélations peuvent être considérées comme des redondances dans les données et peuvent nuire aux résultats.

### 4.1.3 Introduction au problème de corrélation en clustering

Sur la base des observations faites dans la section précédente il est légitime de se poser la question suivante : quel est concrètement le problème causé par les corrélations ? Imaginez que l'ensemble de données soit décrits par trois variables, dont deux sont identiques et que l'on souhaite construire des clusters avec un algorithme qui utilise la distance euclidienne, comme les $K$-means. L'information apportée par les variables identiques compte double dans le calcul de la distance euclidienne entre deux observations. Par conséquent, il est probable que le clustering dépende exclusivement de ces deux variables identiques, quelle que soit la structure de l'ensemble de données.

Pour autant que nous le sachions, aucune recherche spécifique n'a été effectuée pour gérer les corrélations en clustering. Il existe néanmoins des solutions pour traiter les variables corrélées dans les données. L'une d'elles, appelée blanchiment, consiste à transformer les données de telle sorte que la matrice de covariance des données transformées soit la matrice identité et que les corrélations soient ainsi supprimées. Il existe deux transformations de blanchiment très connues, le blanchiment par ACP et le blanchiment par Zero-phase Component Analysis (ZCA) également appelé blanchiment de Mahalanobis (Kessy et al., 2018). Cependant, les méthodes de blanchiment présentent plusieurs inconvénients que nous présentons dans les sections suivantes.

D'autre part, on pourrait penser que la sélection des variables résoudrait le problème causé par les corrélations des variables de bruit, en supprimant du modèle ces variables. Nous observerons que ce n'est généralement pas le cas.

### 4.1.4 Les contributions de ce chapitre

La gestion des corrélations dans le clustering est primordiale, comme l'est celle de la variance des variables. En effet, comme décrit dans le dernier paragraphe de la Section 2.8.1, les variables importantes de grandes variances biaisent les résultats en faveur des algorithmes basés sur la distance euclidienne et il en va de même si les variables importantes sont corrélées. Évidemment si l'ensemble de données contient uniquement des variables importantes et qui plus est corrélées, l'intérêt d'une telle normalisation est moindre. En revanche, si dans des ensembles de données réels il existe une structure en clusters il est raisonnable de supposer, premièrement la présence de variables de bruit et deuxièmement l'existence de corrélation entre les variables notamment entre les variables de bruit. De plus, le biais induit par les corrélations sera d'autant plus fort lorsque le nombre de variables de bruit est grand notamment par rapport au nombre de variables importantes.

Ainsi le problème des corrélations entre variables devra être géré dans un cas général, tant pour l'utilisation d'algorithmes de clustering classiques que pour celle d'algorithmes sparses. C'est pourquoi, pour résoudre le problème causé par les corrélations dans le clustering, nous proposons une nouvelle transformation. Cette transformation pondère les variables par l'inverse de la somme des carrés des corrélations avec les autres variables. Le but est de réduire la redondance de l'information dans les données. Ainsi la transformation proposée se définit comme une simple normalisation applicable en entrée de n'importe quel algorithme.

De plus, dans l'ACP, les valeurs propres fournissent des informations sur les corrélations entre les variables. Plus les variables sont corrélées entre elles, plus les premières valeurs propres seront grandes. Par conséquent, les corrélations entraînent une redondance qui est quantifiée par les valeurs propres. La solution proposée peut être interprétée comme le calcul de la moyenne pondérée des variances des composantes principales (valeurs propres) où les pondérations sont données par les corrélations des variables avec les composantes principales.

Dans ce chapitre, nous allons illustrer avec des exemples simples les problèmes que causés par les corrélations. Ensuite, nous allons montrer que les transformations de blanchiment peuvent altérer la structure des données, notamment la structure en clustering, rendant plus difficile le clustering après transformation. Nous allons également mettre en évidence que les algorithmes de clustering sparses amplifient généralement

les problèmes causés par l'existence de corrélations. Enfin, l'efficacité de notre méthode est illustrée sur des simulations basées sur le schéma de simulations présenté dans la Section 2.8.

## 4.2 Une vue d'ensemble du problème, et des questions de blanchiment et de clustering sparse.

Les méthodes permettant de résoudre le problème de la corrélation sont essentiellement de deux types : soit des méthodes de type pré-traitement, qui vont être effectuées indépendamment de l'algorithme de clustering, soit des méthodes intégrées qui modifient l'algorithme.

### 4.2.1 Méthodes de pré-traitement

Nous avons classé les méthodes de pré-traitements en deux catégories, les méthodes de blanchiment et les autres méthodes qui se basent sur des algorithmes plus complexes.

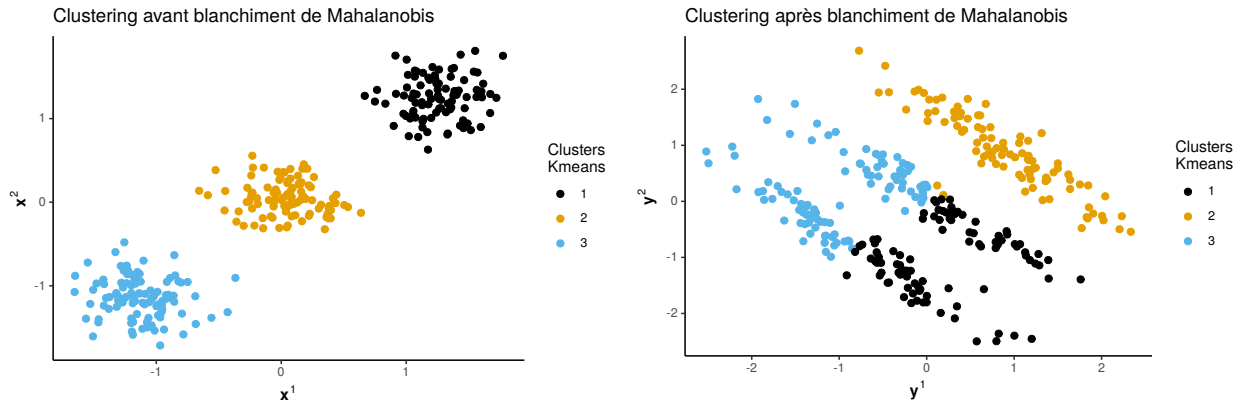
#### 4.2.1.a Méthodes de blanchiment

DÉFINITION ET EXEMPLE DE BLANCHIMENT Le blanchiment, *whitening* ou *sphering* en anglais, est une transformation linéaire qui convertit une matrice $\mathbf{X} \in \mathbb{R}^{n \times p}$ ayant matrice de covariance estimée définie positive $\hat{\mathbf{\Sigma}}_{\mathbf{X}}$ en une nouvelle matrice de données

$$\mathbf{Y} = \mathbf{X}\mathbf{W} \text{ où } \mathbf{W}^\intercal \mathbf{W} = \hat{\mathbf{\Sigma}}_{\mathbf{X}}^{-1}, \tag{4.1}$$

de sorte que la matrice de covariance de $\mathbf{Y}$ est $\hat{\mathbf{\Sigma}}_{\mathbf{Y}} = \mathbf{I}$. Le blanchiment est alors une généralisation de la normalisation qui prend en compte non seulement les variables séparément mais avec leurs dépendances. La contrainte donnée dans l'Équation (4.1) ne détermine pas de façon unique la matrice de blanchiment $\mathbf{W}$ : il existe une infinité de matrices possibles $\mathbf{W}$ et chaque $\mathbf{W}$ conduit à une transformation des données en des données sphériques. Il existe donc différentes formes de blanchiment qui se différencient par le choix de la matrice $\mathbf{W}$. Avant de présenter les inconvénients des méthodes de blanchiment de manière générale, nous allons illustrer les problèmes rencontrés avec deux méthodes de blanchiment très connues.

BLANCHIMENT DE MAHALANOBIS Le blanchiment de Mahalanobis ou aussi appelé blanchiment par ZCA définit par $\mathbf{W} = \hat{\mathbf{\Sigma}}_{\mathbf{X}}^{-1/2}$. Ce type de blanchiment transforme les données pour les rendre sphériques et de ce fait les distances entre les individus sont modifiés. Dans certains cas, la structure en clusters des données peut être détériorée. Reprenons le schéma de simulation décrit dans la Section 2.8 qui est le schéma défini par Daniella Witten et Robert Tibshirani avec $K = 3, p_K = 2, \mu = 5$ et sans variable de bruit $d = 0$. On a donc trois clusters qui sont alignés sur deux variables. Les données sont comme d'habitude centrées réduites. La Figure




(a) (b)

**Figure 4.2 :** Le graphique (a) représente les clusters trouvés par le $K$-means sur les données simulées. Le graphique (b) représente les clusters trouvés par le $K$-means sur les données transformées par le blanchiment de Mahalanobis. La transformation a changé la structure en clusters.

4.2 représente les résultats du meilleur $K$-means (au sens de sa fonction objectif pour des milliers d'exécutions) sur les données avant et après un blanchiment de Mahalanobis. Nous pouvons constater que le blanchiment détruit la structure des données. Le fait de rendre sphérique les données écrase les clusters.

BLANCHIMENT PAR L'ACP Un autre blanchiment bien connu est basée sur l'analyse en composantes principales normalisée et utilise $\mathbf{W} = \mathbf{\Lambda}^{-\frac{1}{2}}\mathbf{V}^\intercal$, où $\mathbf{\Lambda} = \text{diag}(\lambda_1, ..., \lambda_p) \in \mathbb{R}^{p\times p}$ est la matrice diagonale des valeurs propres et $\mathbf{V} \in \mathbb{R}^{p\times p}$ est la matrice de changement de bases associées à la base des vecteur propres, telle que la matrice de covariance $\hat{\mathbf{\Sigma}}_\mathbf{X} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\intercal$. On peut interpréter la multiplication avec la matrice $\mathbf{V}^\intercal$ soit comme une rotation des données, soit comme une rotation du système de coordonnées. Par conséquent, cette transformation fait d'abord pivoter les variables en utilisant les vecteurs propres de la matrice de covariance, comme cela est fait dans l'ACP standard. On obtient ainsi des composantes orthogonales, mais avec des variances généralement différentes. Pour obtenir des données blanchies, les variables qui ont subi une rotation sont ensuite normalisées par la racine carrée des valeurs propres $\mathbf{\Lambda}^{-\frac{1}{2}}$.

Le problème du blanchiment par ACP est tout à fait différent de celui présenté précédemment. Par définition, l'ACP n'affecte pas la structure des données. En particulier si on conserve toutes les composantes principales, les données sont exactement reconstruites et les résultats de clustering produits sur ces données sont équivalents à ceux construits à partir des données originales. Cependant, le blanchiment ACP implique également une normalisation des composantes principales par la racine carrée des valeurs propres $\mathbf{\Lambda}^{-\frac{1}{2}}$. Par conséquent, on peut facilement construire un exemple avec $p$ variables importantes partageant la même clusters comme sur la Figure (a) 4.2. Alors, quasiment toute la variance des données est représentée sur laquelle les vrais centres des clusters sont alignés. Dans ce cas, les autres directions ne sont pas pertinentes pour le clustering et pourtant, la normalisation implique que toutes les directions sont de même variance. Ainsi, le blanchiment de l'ACP a crée artificiellement $p-1$ variables de bruit. Dit autrement, le blanchiment par ACP peut augmenter la variance dans des directions non pertinentes, ici non redondantes et cela peut fortement affecter les résultats du clustering.

LES INCONVÉNIENTS DU BLANCHIMENT Nous pouvons maintenant nous intéresser aux problèmes associés au blanchiment de manière globale. Nous en listons notamment trois.

1. L'opération de blanchiment est impossible lorsque $p \gg n$ : certains blanchiments, dont celui de Mahalanobis, ne sont pas possibles lorsque $p \gg n$, car ils nécessitent l'inversion de matrices qui sont singulières dans ce cas. De manière générale le problème d'inversion de matrice est complexe surtout en grande dimension et les méthodes pour pallier ce problème, telles que la régularisation de Tykhonov Tikhonov (1943) ou la pseudo-inverse de Moore Penrose Penrose (1955) sont plus coûteuses en temps de calcul et peuvent être numériquement instables.

2. Le blanchiment ne permet pas de faire du clustering sparse : le blanchiment par l'ACP ou de Mahalanobis ne permettent de faire du clustering sparse car les variables transformées sont des combinaisons linéaires de toutes les variables de départ.

3. Cela transforme la structure des données : comme on l'a vu précédemment, les opérations de blanchiment modifient la structure des données de telle sorte que les clusters ne peuvent plus être trouvés et que les distances dans certaines directions sont modifiées, augmentées ou réduites. On peut souhaiter réduire la redondance mais quel est l'intérêt d'augmenter la variance dans les directions des dernières composantes principales que l'on souhaiterait considérer comme du bruit.

#### 4.2.1.b Autres méthodes algorithmiques

Il existe d'autres méthodes qui pourraient offrir des solutions au problème engendré par les corrélations.

LE CLUSTERING DE VARIABLES CLUSTOFVAR : Une solution possible serait d'effectuer un clustering de variables, où le clustering a pour but de regrouper les variables en se basant sur la corrélation. Une telle méthodologie, permettant notamment le clustering de variables numériques et/ou catégorielles, existe sous le nom de ClustOfVar (Chavent et al., 2011). Le clustering de variables est une alternative à des méthodes type ACP puisqu'il permet d'organiser les variables en groupes homogènes afin d'obtenir des structures informatives des variables. D'un point de vue général, le clustering de variables regroupe des variables qui sont fortement corrélées les unes aux autres et qui apportent donc la même information. Une fois les variables regroupées de manière homogène le but est de résumer chaque groupe par une variable. On aura donc réduit la dimension et réduit la corrélation dans les données.

Pour résumer un cluster de variables, l'utilisateur peut faire le choix de sélectionner une variable du groupe comme la représentante de celui-ci. On peut aussi vouloir construire une variable synthétique. Par exemple, dans le cas de variables quantitatives, une solution consiste à réaliser une ACP dans chaque groupe et à retenir la première composante principale comme variable synthétique du groupe.

L'avantage du clustering de variables est qu'il réduit le nombre de variables en entrée de l'algorithme, réduit évidemment la corrélation et améliore l'interprétabilité car les variables de départ sont maintenant regrouper en clusters ce qui donne un avantage explicatif pour des analyses exploratoires. En revanche, pour notre application, cette méthode présente des défauts notamment celui d'avoir un paramètre supplémentaire à optimiser, le nombre de clusters de variables, il faut aussi choisir l'algorithme de clustering de variables. Par exemple, pour l'algorithme ClustOfVar, par défaut la CAH est utilisé pour faire le clustering de variables et celle-ci est très coûteuse en temps de calcul notamment lorsque l'on dépasse les quelques milliers de variables, ce qui pose empêcherait l'utilisation de cette méthode en entrée d'algorithmes sparses pour des données de très grande dimension. Plus de détails sont donnés sur la méthode ClustOfVars dans les chapitres suivants.

### 4.2.2 Méthodes intégrées

Les méthodes intégrées sont des méthodes qui modifient des algorithmes existants dans le but de prendre en compte les corrélations entre les variables. Nous faisons deux catégories, sans faire une liste exhaustive de toutes les méthodes. La première catégorie liste des méthodes avec transformation de l'espace de départ et la seconde regroupe des méthodes de sélections de variables, par exemple avec une pénalisation prenant en compte la corrélation.

#### 4.2.2.a Transformation de l'espace

Comme nous l'avons déjà vu dans les chapitres précédents, certaines méthodes impliquent une transformation de l'espace. Nous rappelons que ces méthodes ont vu le jour avec l'ouvrage pionnier de Ding and Li (2007) et malheureusement la fonction objectif de cette algorithme est mal définie (Wang et al., 2019). Néanmoins, l'étape dite discriminative de l'algorithme implique une transformation de l'espace à l'aide d'une LDA ce qui se rapproche en un sens d'un blanchiment de Mahalanobis, ou en tout cas cette transformation par LDA a exactement les mêmes inconvénients et problèmes que le blanchiment de Mahalanobis puisque les deux impliquent l'inversion d'une matrice de covariance (pour la LDA c'est la "pooled within-groups covariance matrix"). Ainsi, toute cette famille d'algorithme va garder les mêmes inconvénients. De plus, il est très difficile de sparsifier ce type d'algorithme. Déjà en apprentissage supervisé il est difficile de sparsifier la LDA. Daniella Witten et Robert Tibshirani ont proposé une version sparse (Witten and Tibshirani, 2011), mais ils ont dû assumer que les matrices de covariance par classe était diagonale, ce qui est une grosse hypothèse et l'intérêt de la transformation par la LDA est quelque peu perdu. Ils ont ensuite avec d'autres auteurs proposés une autre version basée sur du *optimal scoring* (Clemmensen et al., 2011). Il n'en reste pas moins que ce sont des méthodes algorithmiquement complexes et difficile à intégrer à une méthodologie de clustering.

#### 4.2.2.b Sélection de variables

Il serait possible de développer un algorithme de sélection de variables qui sélectionnerait qu'une seule des variables corrélées. Malheureusement ce type de méthode ne fonctionnerait pas comme nous l'avons vu dans les simulations de la Section 2.8. En effet, dans ce schéma de simulation, les clusters sont alignés sur les variables importantes et donc cela crée implicitement de la corrélation entre les variables. Nous avions aussi remarqué que plus le nombre de variables importantes était grand plus les clusters étaient séparés et de surcroît les algorithmes avaient plus de facilité à retrouver les classes. Sélectionner qu'une seule variable d'un groupe de variables corrélées importantes engendré une perte d'information précieuse en clustering.

Une autre possibilité serait de pénaliser les algorithmes de clustering par la corrélation entre les variables. Par exemple, l'algorithme Sparse $K$-means peut être reformulé avec une pénalité dépendant des corrélations entre les variables. En revanche, cela ajouterait un hyperparamètre supplémentaire à optimiser, en plus de celui sur les poids pour obtenir de la sélection de variables, ce qui n'est pas souhaitable.

### 4.2.3 Pourquoi les algorithmes de clustering sparses sont-ils affectés par les corrélations ?

Les variables corrélées impliquent des axes de grande de variance. Or, pour la majorité des algorithmes sparses basés sur la distance euclidienne, qui pour la plupart font partie de la famille des $K$-means sparses, les variables les plus importantes sont celles qui contribuent le plus à l'inertie inter-classes et donc dans une certaine mesure à la variance. Cependant, les variables qui contribuent le plus à la variance des données, selon la théorie de l'ACP, sont les variables corrélées.

De plus, les axes de grande variance ont une grande inertie inter-classes mais aussi une grande inertie intra-classes. En effet, les variables corrélées impliquent des axes de grande variance, où les centres des clusters sont bien séparés. En revanche sur ces axes, les clusters sont également moins homogènes (grande somme des distances euclidienne au carré intra clusters). Certains algorithmes supervisés tentent de résoudre des

problèmes similaires et c'est notamment le cas de la LDA qui cherche le sous-espace ou l'inertie intra-classes est maximum et ou l'inertie inter-classes est minimum. Cela est très bien illustré dans le livre Bishop (2006) (pages 187-188) "The simplest measure of the separation of the classes is the separation of the projected class means". Par contre l'auteur ajoute "There is still a problem with this approach, if for example two classes that are well separated in the original space but that have considerable overlap when projected onto the line joining their means. This difficulty can arise from the strongly nondiagonal covariances of the class distributions. The idea proposed by Fisher is to maximize a function that will give a large separation between the projected class means while also giving a small variance within each class, thereby minimizing the class overlap". Ainsi il faut que les classes soient séparées et homogènes ce qui n'est pas forcément le cas sur les premières composantes principales. C'est aussi pourquoi certains auteurs ont voulu associer les $K$-means et l'algorithme LDA pour faire du clustering mais malheureusement ces méthodes ont de nombreux inconvénients qui ont déjà été discutés ci-dessus.

Finalement, au vu des problèmes posés, il est naturel de se demander si la fonction objectif des $K$-means munis de la distance euclidienne est bien définie (Ben-David, 2018). En effet, nous avons observé que le meilleur $K$-means au regard de sa fonction objectif n'est souvent pas le meilleur modèle à sélectionner. Malgré tout les $K$-means restent très largement utilisés et il est nécessaire de proposer une solution.

## 4.3 La solution proposée : normaliser les variables en fonction des corrélations

### 4.3.1 Présentation de la solution

LES AVANTAGES ATTENDUS DE LA SOLUTION Nous avons déjà vu que les transformations de blanchiment impliquent des rotations qui peuvent provoquer des transformations indésirables. C'est pourquoi nous souhaitons une transformation par une matrice diagonale afin que les variables ne changent qu'en variance. La normalisation proposée doit tenir compte de la corrélation entre les variables et elle doit être suffisamment simple pour être utilisée avant tout algorithme de clustering, dans toutes les situations, même par exemple en grande dimension lorsque $p \gg n$.

DÉFINITION DE NOTRE SOLUTION DE NORMALISATION Nous proposons de normaliser les variables en fonction de la racine carrée de la somme de leur corrélation au carré avec les autres variables. Formellement, $\forall l, j = 1, \ldots, p$ $\text{cor}(\mathbf{x}^l, \mathbf{x}^j)$ est la corrélation entre les variables $l$ et $j$ de $\mathbf{X}$, où $\mathbf{X}$ est supposée normalisée à moyenne 0 et variance unitaire, alors la matrice de normalisation $\mathbf{N}$ est définie comme suit

$$\mathbf{N} = \text{diag}(\nu_1^2, \ldots, \nu_p^2), \tag{4.2}$$

avec $\forall j = 1, \ldots, p, \nu_j^2 = \sum_{l=1}^{p} \text{cor}(\mathbf{x}^l, \mathbf{x}^j)^2$ et diag désigne la matrice diagonale avec $\nu^2 = (\nu_1^2, \ldots, \nu_p^2)^\intercal$ ses éléments diagonaux. Par conséquent, notre nouvelle matrice de données $\mathbf{Y}$ peut être calculée comme suit

$$\mathbf{Y} = \mathbf{X}\mathbf{N}^{-\frac{1}{2}}, \tag{4.3}$$

ce qui revient à calculer $\forall j = 1, \ldots, p, \mathbf{y}_j = \frac{\mathbf{x}_j}{\sqrt{\nu_j^2}}$ en supposant que les variables de $\mathbf{X}$ sont centrées et de variance unitaire. Dans le cas où les variances sont différentes, remplacer la corrélation par la covariance ne donne pas les mêmes résultats. Il faut impérativement que les variables soient normalisées à moyenne 0 et variance unitaire au préalable. C'est donc une normalisation par l'inverse des corrélations, ou Inverse Correlation Scaling en anglais (ICS) et par la suite nous la nommerons normalisation par ICS.

PROPRIÉTÉS DE LA SOLUTION La normalisation par ICS détient plusieurs bonnes propriétés qui peuvent être mises en avant. Premièrement, les corrélations entre les variables ne sont pas affectées. Cela signifie que la structure en corrélation des données n'est pas modifiée, seule la structure en variance change dans le but de découvrir plus facilement la structure en clusters. En effet, les variances des variables sont maintenant différentes car chaque variable à sa variance divisée par $\nu_j^2$. Une variable très corrélée aux autres, c'est-à-dire une variable ayant une valeur de $\nu_j^2$ grande, aura une variance plus petite après transformation et prendra donc moins d'importance dans le calcul des distances euclidiennes. La distance euclidienne entre deux observations

$\mathbf{y}_i$ et $\mathbf{y}_t$ est la distance euclidienne pondérée par les inverses des coefficients $\nu_j^2$ entre les observations $\mathbf{x}_i$ et $\mathbf{x}_t$ :

$$\begin{aligned} d^2(\mathbf{y}_i, \mathbf{y}_t) &= \sum_{j=1}^{p} (y_i^j - y_t^j)^2, \\ &= \sum_{j=1}^{p} (\frac{x_i^j}{\sqrt{\nu_j^2}} - \frac{x_i^j}{\sqrt{\nu_j^2}})^2, \\ &= \sum_{j=1}^{p} \frac{1}{\nu_j^2} (x_i^j - x_t^j)^2. \end{aligned}$$

Cela peut aussi s'interpréter comme un poids qui serait attribué à chaque variable en fonction de sa contribution à la variance des données. Dans la prochaine sous-section, nous allons analyser plus précisément les propriétés théoriques de cette normalisation.

### 4.3.2 Analyse théorique : lien avec l'ACP

Nous souhaitons analyser théoriquement notre solution pour en comprendre tous les aspects. Pour ce faire, nous allons mettre en lumière le lien entre l'ACP et la normalisation proposée. Rappelons que les colonnes de $\mathbf{X}$ sont centrées en 0 et de variance 1 et que nous calculons le coefficient suivant :

$$\nu_j^2 = \sum_{l=1}^{p} \text{cor}(\mathbf{x}^l, \mathbf{x}^j)^2, \forall j = 1, \ldots, p.$$

Ainsi, nous obtenons :

$$\nu_j^2 = (\hat{\mathbf{\Sigma}}_{\mathbf{X}}^{\intercal} \hat{\mathbf{\Sigma}}_{\mathbf{X}})_{j,j}$$

Or, par la décomposition en valeur propre de la matrice de covariance nous obtenons :

$$\hat{\mathbf{\Sigma}}_{\mathbf{X}} = \mathbf{V} \mathbf{\Lambda} \mathbf{V}^{\intercal},$$

avec $\mathbf{\Lambda} \in \mathbb{R}^{p \times p}$ la matrice diagonale des valeurs propres et $\mathbf{V} \in \mathbb{R}^{p \times p}$ la matrice des vecteurs propres de $\frac{1}{n} \mathbf{X}^{\intercal} \mathbf{X}$. Finalement, nous obtenons l'équation suivante :

$$\nu_j^2 = \sum_{\alpha=1}^{p} (\mathbf{v}_j^{\alpha})^2 \lambda_{\alpha}^2. \tag{4.4}$$

Une explication plus précise de l'obtention de l'Équation (4.4) est disponible dans la Section 4.6.1. La solution proposée s'interprète donc comme la somme des variances des composantes principales pondérées par les corrélations des variables aux axes. Or les valeurs propres donnent l'information des corrélations entre les variables. Plus les variables sont corrélées entre elles, plus les premières valeurs propres seront grandes. Par conséquent, en clustering, la corrélation amène une redondance qui est quantifiée par les valeurs propres.

De surcroît, plus il y a de la redondance dans les données, plus les premières valeurs propres seront grandes. Comme nous l'avons vu précédemment, cette redondance affecte négativement les algorithmes de clustering et notamment les $K$-means. Ainsi, il s'avère que cela soit une bonne pratique que de normaliser le poids des variables par leur contribution à la redondance dans les données.

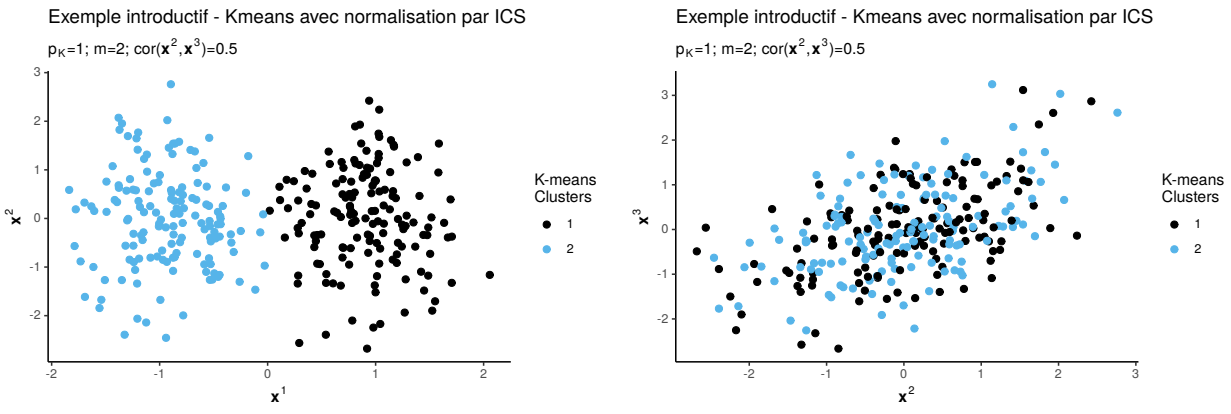
De plus on a la relation suivante : $\mathbf{F} = \mathbf{U} \mathbf{\Lambda}^{\frac{1}{2}}$ avec $\mathbf{U} \in \mathbb{R}^{n \times p}$ la matrice des vecteurs propres de $\frac{1}{n} \mathbf{X} \mathbf{X}^{\intercal}$ et $\mathbf{F}$ les composantes principales, ce qui correspond à :

$$\mathbf{f}^{\alpha} = \mathbf{u}^{\alpha} \sqrt{\lambda_{\alpha}}.$$

Or en pratique, on peut exactement reconstruire les données en prenant toutes les composantes principales. Cela signifie que les $K$-means exécutés toutes les composantes principales donneront le même clustering que sur les données de départ. La distance euclidienne au carrée entre deux observations $i, t$ est définie comme :

$$d(\mathbf{x}_i, \mathbf{x}_t) = \sum_{j=1}^{p} (x_i^j - x_t^j)^2 = \sum_{\alpha=1}^{p} (f_i^{\alpha} - f_t^{\alpha})^2 = \sum_{\alpha=1}^{p} (u_i^{\alpha} - u_t^{\alpha})^2 \lambda_{\alpha}.$$

Ainsi cette redondance $\lambda_{\alpha}$ doit être supprimée lorsque $\lambda_{\alpha} > 1$ et ne pas être augmentée lorsque $\lambda_{\alpha} < 1$.




(a) (b)

**Figure 4.3 :** Données simulées et normalisées par ICS où $\mathbf{x}^1$ est composé de clusters bien séparés et $\mathbf{x}^2, \mathbf{x}^3$ sont légèrement corrélés ($\text{cor}(\mathbf{x}^1, \mathbf{x}^2) = 0.5$). Le graphique (a) représente les clusters trouvés par les 2-means affichés sur $\mathbf{x}^1, \mathbf{x}^2$ ; le graphique (b) représente les clusters trouvés par les 2-means affichés sur $\mathbf{x}^2, \mathbf{x}^3$. Les clusters sont le résultats des meilleurs 2-means par rapport à leur fonction objectif (sur un millier d'initialisation aléatoire). Les 2-means retrouvent bien les clusters sous-jacents.

### 4.3.3 Résultats des $K$-means sur l'exemple introductif normalisé par ICS

La nouvelle transformation est utilisée sur l'exemple d'introductif. La Figure 4.3 donne la partition en deux classes trouvée par les des 2-means sur les données normalisées par ICS. Comme prévu, la normalisation par ICS aide les $K$-means à retrouver les vrais clusters sous-jacents. Ce n'est pas distinguable sur les échelles des figures, mais la variance des variables $\mathbf{x}^2, \mathbf{x}^3$ a été réduite à 0.786 sur cette simulation alors que évidemment celle de $\mathbf{x}^1$ est restée inchangée (autour de 1). Finalement, l'information apportée par $\mathbf{x}^2, \mathbf{x}^3$ étant la même, leur contribution à la variance s'est vue réduite.

Cet effet qui est provoqué par les variables corrélées est, comme nous l'avons déjà vu, en partie dû à l'utilisation de la distance euclidienne. Par ailleurs, les modèles de mélanges ont le pouvoir de modéliser différentes formes de matrice de covariances par classe, donc sur cet exemple ce type de modèle peut obtenir de bons résultats. Malheureusement cela n'est pas toujours vrai et cela va dépendre notamment de l'initialisation choisie. En effet, de nombreuses implémentions des GMM comme Scrucca et al. (2016) ; Lebret et al. (2015) préconisent une initialisation à l'aide des $K$-means ou d'une CAH. Comme ces deux algorithmes sont impactés par les corrélations, l'initialisation sera mauvaise et il peut être difficile de se sortir du minimum local trouvé par ceux-ci. Néanmoins sur cet exemple, les GMM vont fonctionner même avec une mauvaise initialisation. Nous illustrons cette assertion à l'aide de la Table 4.1 qui compare le $K$-means sur les données de départ, le $K$-means sur les données normalisées par ICS, les GMM avec le package `Rmixmod` initialisé avec les $K$-means. la Table 4.1 montre clairement que les $K$-means ne fonctionnent pas sur des données corrélées même pour

**Table 4.1 :** Le tableau représente les moyennes et écart type de l'ARI par méthode pour l'exemple introductif avec cette fois-ci $\text{cor}(\mathbf{x}^2, \mathbf{x}^3) = 0.9$ sur 100 simulations.

| | $K$-means | $K$-means ICS | GMM (mixmod) |
|---|---|---|---|
| moyenne | 0.01 | 0.90 | 0.88 |
| sd | 0.02 | 0.05 | 0.15 |

un cas aussi simple. La normalisation par ICS montre ici son utilité pour l'algorithme $K$-means et obtient des résultats équivalents à ceux des GMM. En revanche il est possible que sur des cas plus compliqués les GMM restent bloqués dans les minimums locaux donnés par les $K$-means et donc une initialisation avec les $K$-means ou une CAH sur des données normalisées par ICS peut être souhaitable.

### 4.3.4 Avantages et inconvénients de la normalisation par ICS

La normalisation par ICS présentent de nombreux avantages que nous avons listé ci-dessous.

1. Premièrement la solution est simple et rapide. En effet, elle est très facilement implémentable en une ou deux lignes de code et elle requiert un temps de calcul très faible ce qui donnera un temps d'exécution quasi instantanée pour la majorité des applications.

2. La normalisation par ICS peut être utilisée en entrée de n'importe quel algorithme. Cela est surtout utile pour les algorithmes basés sur la distance euclidienne car comme on l'a vu avec cette distance la normalisation par ICS a de bonnes propriétés. En revanche, son intérêt semble plus limité pour les GMM sauf si on prend en compte leur initialisation. L'initialisation des GMM se base le plus souvent sur les $K$-means ou une CAH qui sont donc sensibles aux corrélations et il faut espérer que les GMM puissent se sortir du minimum local trouvé par ces deux algorithmes.

3. L'algorithme peut être utilisé en grande dimension c'est-à-dire lorsque $p >> n$, ce qui n'est pas toujours le cas de toutes les autres solutions comme le blanchiment de Mahalanobis par exemple car celui-ci nécessite l'inversion de matrice singulière.

4. La normalisation par ICS permet de faire du clustering sparse. En effet, encore une fois, les solutions présentées dans l'état de l'art de ce chapitre Section 4.2 telles que le blanchiment ou les transformations de l'espace utilisent des matrices de rotation de sorte que les variables résultantes sont des combinaisons linéaires des variables originale, ce qui ne permet pas de faire sélection des variables de départ.

5. La solution s'étend naturellement au cas des données mixtes, qui est souvent négligé mais est très courant dans la pratique. La méthode nécessite la définition d'une mesure de similarité entre deux variables de tout type, numérique et catégorielle. De plus, une variable catégorielle peut être transformée en un groupe de variable binaire (Section 3.3.3). Ainsi, l'analyse canonique des corrélations peut être utilisée pour calculer une similarité entre deux groupes de variables Chavent et al. (2012) et notre solution pourrait s'appliquer à la matrice des similarités obtenues. Cette extension ne sera pas discutée plus précisément dans ce manuscrit.

6. Dans nos simulations, la normalisation proposée a un impact négatif très limité sur les résultats des algorithmes clustering.

Toutefois, en toute honnêteté, la normalisation par ICS présente certaines limites que nous listons ci-dessous.

1. La méthode est uniquement bien définie pour les algorithmes de clustering basés sur la distance euclidienne mais nous pensons qu'elle peut être facilement étendue à d'autres mesures de similarité si nécessaire.

2. Dans le cas où il y a uniquement des variables importantes qui sont corrélées, la normalisation par ICS va forcément réduire la variance et le poids de ces variables dans le clustering ce qui est un cas défavorable pour notre solution. En revanche, il en est de même pour la normalisation à variance unitaire dans le cas où les variables importantes ont des variances plus grandes que les variables de bruit, or celle-ci reste très majoritairement utilisée. De plus, la présence de corrélation uniquement sur les variables importantes, qui ont une structure en clusters, semble être un cas rare.

3. La normalisation par ICS considère les corrélations entre les variables, or les corrélations sont des dépendances linéaires. Il n'est pas évident d'imaginer comment des dépendances non linéaires peuvent affecter la distance euclidienne. Honnêtement, il est possible que des dépendances non linéaires affectent la distance euclidienne, mais à priori cela sera dans une moindre mesure car deux variables identiques ont aussi une corrélation de 1 et comptent double dans la distance euclidienne et ce cas ne peut pas être décrit avec des relations non linéaires. De plus, il est très facile d'imaginer des groupes de variables linéairement corrélées qui jouent sur la variance. En revanche, cela l'est beaucoup moins pour des groupes de variables non linéairement dépendantes car elles doivent justement être non linéairement dépendante une à une, ce qui nécessiterait autant de relations non linéaires différentes que de variables.

4. La solution proposée considère des corrélations globales entre les variables. Or, il peut exister des corrélations uniquement localement, par exemple par cluster. En fait, en ce sens les $K$-means avec notre normalisation sont robustes aux corrélations entre les variables globalement mais pas localement comme le serait des GMM qui pourraient modéliser des covariances par cluster différentes. Cela serait toutefois une extension intéressante.

## 4.4 Simulations

Dans cette section, nous comparons et analysons les méthodes décrites dans le Chapitre 2 dans un contexte de données très corrélées. Nous allons reprendre les mêmes schémas de simulations et les mêmes mesures d'évaluations.

LE MODÈLE GLOBAL Les simulations ont pour but ici d'évaluer et de comparer des algorithmes de clustering avec sélection de variables dans le cas où il y a des groupes de variables importantes et de variables de bruit corrélées. Ainsi, nous allons reprendre le schéma de simulation vu à la Section 2.8 du Chapitre 2 mais nous allons ajouter un groupe de variables et il y aura donc trois groupes de variables :

- le groupe des variables importantes qui a une structure en clustering formée par un mélange de Gaussiennes. Notons $p_K$ le nombre de variables importantes. On sait désormais qu'en prenant le schéma 1 décrit par l'Équation 2.11, c'est à dire lorsque les clusters sont alignés sur un axe, les variables importantes ont une corrélation implicite dû à l'alignement des clusters.
- le groupe des variables de bruit, où les variables du groupe sont indépendantes du clustering, indépendantes des variables importantes et indépendantes entre elles. Elles suivront ici une distribution Gaussienne sphérique. Notons $d$ le nombre de variables de bruit indépendantes.
- le groupe des variables de bruit corrélées, où les variables du groupe sont indépendantes du clustering, indépendantes des variables importantes et indépendantes du premier groupe de variable de bruit mais dépendantes (corrélées) entre elles. Elles suiveront ici une distribution Gaussienne. Notons $q$ le nombre de variables de bruit indépendantes.

Par conséquent $p = p_K + d + q$. Le modèle de mélange sous-jacent global s'écrit désormais :

$$\sum_{k=1}^{K} \frac{1}{K} \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_p) \text{ avec } \boldsymbol{\mu}_k = (\mathbf{m}_k, 0, ..., 0)^\intercal \in \mathbb{R}^{p_K+d+q} \text{ et } \mathbf{m}_k \in \mathbb{R}^{p_K}. \tag{4.5}$$

avec $\boldsymbol{\Sigma}_k = \begin{pmatrix} \mathbf{I}_{p_K} & 0 & 0 \\ 0 & \mathbf{I}_d & 0 \\ 0 & 0 & \mathbf{S}_q \end{pmatrix}$ qui est bien sûr une matrice par bloc où les 0 représentent ici des matrices de 0 avec les bonnes tailles et $\mathbf{S}_q$ a 1 sur toute sa diagonale et $r$ ailleurs. Donc $r$ correspond au coefficient de corrélation global de cette matrice.

Ce mélange Gaussien décrivant les variables ne sera que d'un seul type, un mélange de 2 Gaussiennes sphériques équiprobables, ainsi $K = 2$, $\sum_{k=1}^{K} \frac{1}{K} \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ avec $\boldsymbol{\mu}_1 = -\boldsymbol{\mu}_2 = (\mathbf{m}_1, 0, ..., 0)_p^\intercal$ et donc $\mathbf{m}_1 = -\mathbf{m}_2 = (m, ..., m)_{p_K}^\intercal$.

### 4.4.1 Simulations pour des algorithmes de clustering non sparses

DESCRIPTION DES SIMULATIONS Nous reprenons le schéma de simulation décrit par l'Équation 4.5 et nous allons comparer trois types d'algorithmes non sparses : les $K$-means, une CAH et les GMM. Pour les $K$-means et la CAH nous comparerons respectivement deux modèles par famille d'algorithme : les $K$-means sur données centrées et normalisées à variance unitaire ($K$M dans le tableau) et les $K$-means et les données normalisées par ICS ($K$M (ICS) dans le tableau) et nous ferons de même pour la CAH (respectivement CAH et CAH (ICS) dans le tableau). Pour les GMM, nous utiliserons le package `R Rmixmod` et nous utiliserons deux modèles : un GMM où la forme de la vraie matrice de covariance sera donnée au modèle (GMM dans le tableau) et un autre GMM ou toutes les formes contraintes et non contraintes de la matrice de covariance seront testées et la meilleure sera déterminée grâce au BIC (GMM (All) dans le tableau). Les GMM seront initialisés à l'aide de l'algorithme des $K$-means. Un dernier modèle sera testé, (GMM2 (All) dans le tableau) qui est quasiment le même modèle que GMM (All) mais la fonction provient du package `mclust`. La seule différence entre les deux et l'initialisation, où dans le package `mclust` il est imposé de faire du clustering hiérarchique agglomératif basé sur des critères de vraisemblance maximale pour les modèles de mélange Gaussien paramétrés par décomposition des valeurs propres Scrucca et al. (2016).

RÉSULTATS DES SIMULATIONS Pour tous les modèles, le nombre de clusters $K$ sera donné, avec on le rappelle $K = 2$ pour toutes les simulations et par ailleurs nous fixons $n = 100, r = 0.9$ et $m = 2$ donc les clusters sont très séparés. Nous résumons les simulations suivant les différents scénarios à l'aide d'un simple tableau donné par la Table 4.2. Dans les scénarios nous avons mis peu de variables et notamment peu de variables de bruit car les méthodes ne permettent pas explicitement de gérer ce type de variables. La Table 4.2 montre que la normalisation par ICS contribue largement à améliorer les résultats des algorithmes de clustering en présence de variables corrélées et elle a aussi un impact négatif limité lorsque les variables corrélées sont uniquement des variables importantes. De surcroît, nous observons que les GMM avec la vraie forme de matrice de covariance ont des résultats similaires aux $K$-means avec normalisation ICS. En revanche, lorsque la forme de la matrice de variance covariance n'est pas contrainte l'ARI des GMM est grandement affecté.

**Table 4.2 :** Le tableau représente les moyennes et écart type de l'ARI par méthode pour le schéma de simulation décrit par l'Équation 4.5 avec $n = 100, m = 2, r = 0.9$ et $p_K = 2$ sur 100 simulations. Les valeurs de $d$ et $q$ suivant les scénarios sont indiquées dans le tableau.

| $d$ | $q$ | $K$M | $K$M(ICS) | CAH | CAH(ICS) | GMM | GMM(All) | GMM2(All) |
|---|---|---|---|---|---|---|---|---|
| 5 | 0 | **0.98**(0.03) | **0.94**(0.09) | **0.90**(0.09) | 0.52(0.25) | **0.98**(0.10) | **0.95**(0.20) | **0.97**(0.23) |
| 0 | 5 | 0.01(0.03) | **0.90**(0.15) | 0.08(0.10) | 0.46(0.26) | **0.83**(0.34) | 0.21(0.42) | 0.42(0.32) |
| 5 | 5 | 0.02(0.04) | **0.85**(0.24) | 0.09(0.12) | 0.42(0.26) | **0.72**(0.40) | 0.15(0.39) | 0.26(0.29) |

UN POINT SUR LES MODÈLES DE MÉLANGE Souvent avec le package `Rmixmod`, les GMM non contraints obtiennent de mauvais résultats car ils sélectionnent un modèle avec une seule classe. L'implémentation et surtout la convergence de ces algorithmes font que même en fixant $K = 2$, il est tout à fait possible d'obtenir un clustering avec tous les individus dans le même cluster et c'est souvent le résultat que l'on obtient dans ces simulations notamment lorsque $q > 5$. En effet, le vrai modèle qui est un modèle dont les clusters ont le même volume, proportion et orientation et ont une forme ellipsoïdale est testé et ce modèle retrouve bien les clusters. Malheureusement, le BIC n'est pas optimal sur ce modèle et il n'est donc pas choisi. Nous avons contacté les auteurs mais il n'y pas eu de retour sur ce point précis, laissant penser que ce n'est pas un problème algorithmique. Nous observerons les mêmes résultats dans la section suivante, avec les algorithmes sparses se basant sur `Rmixmod`, telle que la méthode `SelVarMix`.

Un autre problème fait surface avec le package `mclust`. Les partitions en utilisant ce package sont similaires à celles obtenues en utilisant les $K$-means sur les données centrées réduites, c'est à dire que les clusters vont dépendre des variables corrélées.

À ce stade, les raisons profondes de l'échec des GMM ne sont pas claires. Les résultats restent identiques si l'on change l'initialisation du modèle GMM(All), notamment en l'initialisant avec la partition obtenue par le modèle $K$M(ICS). Sachant que les utilisateurs ne peuvent peuvent pas initialiser eux mêmes les GMM avec le package `mclust` (ce qui est d'ailleurs le cas pour la majorité des packages pour les GMM en clustering) et que les méthodes d'initialisations sont très restreintes nous pouvons pas conclure sur ce sujet. D'autre part, même en donnant la bonne forme de la matrice de covariance au GMM avec le package `mclust`, nous n'obtenons pas de bons résultats

En résumé, les deux packages donnent des résultats différents, ils échouent à trouver et/ou simplement sélectionner le bon modèle. Les GMM étant normalement en capacité de modéliser ce type de données, nous faisons l'hypothèse que c'est un problème d'initialisation (`mclust`) doublé d'un problème d'implémentation (`Rmixmod`) sans pouvoir la vérifier.

CONCLUSION Ces simulations sont représentatives des résultats que l'on peut attendre en appliquant la normalisation par ICS sur les données dans le but de faire du clustering. Néanmoins, en présence de variables de bruit cela peut être plus approprié d'utiliser des algorithmes de clustering sparses et cela serait intéressant d'observer le comportement de ces algorithmes sur des normalisées par ICS.

### 4.4.2 Simulations pour les algorithmes de clustering sparses

Dans cette sous-section, nous allons comparer les algorithmes de clustering sparses utilisés dans les simulations présentées dans la Section 2.9 avec les mêmes paramètres pour les fonction `R` et nous allons ajouter deux algorithmes à comparer : le Sparse K-means avec détection de ruptures sur les données normalisées par ICS (SKM_chgpt ICS) et le Sparse K-means avec le Gap Statistic sur les données normalisées par ICS (Sparcl ICS). Nous reprenons le schéma de simulation décrit par l'Équation (4.5) avec $K = 2, n = 100$ et nous fixons $m = 0.85$ comme pour les simulations la Section 2.9 et $r = 0.7$ ce qui correspond à des données sensiblement corrélées sans que cela soit exagéré ou excessif ce qui peut donner un bon aperçu du comportement de ces algorithmes sur des données réelles. Nous allons aussi reprendre le même nombre de variables de bruit que dans la Section 2.9 mais cette fois nous allons former deux groupes avec des variables indépendantes et des variables corrélées. Ainsi, nous considérons trois scénarios : $p_K = 10, d = 50, q = 0$ qui est le scénario le plus défavorable pour notre solution mais aussi le plus facile pour les autres algorithmes ; $p_K = 10, d = 0, q = 50$ qui permettra d'observer l'impact de la présence des variables de bruit corrélées par rapport aux variables de bruit indépendantes sur les algorithmes ; $p_K = 10, d = 50, q = 50$ qui est le scénario le plus difficile mais aussi le plus réaliste. Enfin, nous ajouterons un dernier scénario avec peu de variables qui montrera que le problème existe même en petite dimension.

#### 4.4.2.a Résultats scénario 1 : $p_K = 10; d = 50, q = 0$

Les résultats du scénario 1 sont disponibles dans la Table 4.3. Toutes les méthodes sont équivalentes sauf vscc et clustvarsel qui ne parviennent pas à retrouver les clusters. Comme précédemment, nous observons que la méthode sparcl (Sparse $K$-means avec le Gap Statistic) retient toutes les variables de bruit contrairement à notre solution SWKM_chgpt. De plus la normalisation par ICS n'a aucune incidence sur les résultats car les mêmes algorithmes avec et sans normalisation par ICS ont des ARI, des ratios de variables importantes et de bruits et des temps de calculs qui sont équivalents, ce qui était l'effet escompté pour ce scénario.

**Table 4.3 :** Le tableau représente les moyennes et écart type de l'ARI par méthode pour le schéma de simulation décrit par l'Équation 4.5 avec $n = 100, m = 0.85$ sur 20 simulations. Les nombres de variables par groupe sont $p_K = 10; d = 50; q = 0$.

| | ARI | | Ratio V.Imp | | Ratio V.Bruit | | Temps | |
|---|---|---|---|---|---|---|---|---|
| algorithmes | moyenne | sd | moyenne | sd | moyenne | sd | moyenne | sd |
| Sparcl | **0.98** | 0.03 | **1.00** | 0.00 | 1.00 | 0.00 | 7.24 | 0.25 |
| SFEM_init_km | **0.98** | 0.04 | **1.00** | 0.00 | 0.93 | 0.03 | 39.46 | 3.49 |
| VarSelLCM | **0.98** | 0.04 | **1.00** | 0.00 | 0.02 | 0.01 | 5.99 | 0.12 |
| Sparcl_ICS | **0.97** | 0.05 | **1.00** | 0.00 | 1.00 | 0.00 | 6.58 | 0.19 |
| SWKM_chgpt_ICS | **0.93** | 0.09 | **0.78** | 0.24 | **0.00** | 0.00 | **1.30** | 0.05 |
| SWKM_chgpt | **0.90** | 0.11 | 0.71 | 0.25 | **0.00** | 0.00 | **1.12** | 0.08 |
| clustvarsel | 0.30 | 0.44 | 0.36 | 0.45 | 0.17 | 0.14 | 27.64 | 6.63 |
| SelVarMix | 0.20 | 0.42 | 0.36 | 0.34 | 0.49 | 0.52 | 3.37 | 0.17 |
| vscc | 0.00 | 0.00 | 1.00 | 0.00 | 1.00 | 0.00 | **1.43** | 0.28 |

#### 4.4.2.b Résultats scénario 2 : $p_K = 10; d = 0; q = 50$

Les résultats du scénario 2 sont disponibles dans la Table 4.4. Cette fois-ci, seules les méthodes où les données d'entrée ont été normalisées par ICS et SelVarMix fonctionnent. Cela est très étonnant que les GMM sparses ne fonctionnent pas alors que le scénario 2 n'est pas très différent du scénario 1. Pourtant ces méthodes ne font pas d'hypothèses à priori sur la forme de la matrice de covariance $\mathbf{S}_q$ mis à part VarSelLCM qui suppose l'indépendance des variables importantes par cluster ce qui peut-être pose problème lors de la recherche de la solution à revoir. Sparcl sélectionne toutes les variables de bruit ce qui n'est pas un bon résultat du point de vue de la sélection de variables.

**Table 4.4 :** Le tableau représente les moyennes et écart type de l'ARI par méthode pour le schéma de simulation décrit par l'Équation 4.5 avec $n = 100, m = 0.85$ et $r = 0.7$ sur 20 simulations. Les nombres de variables par groupe sont $p_K = 10; d = 0; q = 50$.

| | ARI | | Ratio V.Imp | | Ratio V.Bruit | | Temps | |
|---|---|---|---|---|---|---|---|---|
| algorithmes | moyenne | sd | moyenne | sd | moyenne | sd | moyenne | sd |
| sparcl_ICS | **0.97** | 0.03 | **1.00** | 0.00 | 1.00 | 0.00 | 6.93 | 0.38 |
| SWKM_chgpt_ICS | **0.92** | 0.12 | 0.71 | 0.23 | **0.00** | 0.00 | **1.69** | 0.36 |
| SelVarMix | **0.83** | 0.35 | 0.80 | 0.35 | 0.86 | 0.07 | 4.24 | 0.20 |
| clustvarsel | 0.10 | 0.32 | 0.11 | 0.31 | 0.08 | 0.07 | 24.37 | 10.37 |
| SFEM_init_km | 0.00 | 0.02 | 0.80 | 0.12 | 0.71 | 0.07 | 67.64 | 2.92 |
| sparcl_norm | 0.00 | 0.01 | 1.00 | 0.00 | 1.00 | 0.00 | 8.10 | 0.83 |
| SWKM_chgpt_norm | 0.00 | 0.01 | 0.00 | 0.00 | 0.12 | 0.09 | **1.40** | 0.19 |
| VarSelLCM | 0.00 | 0.01 | 0.01 | 0.03 | 0.51 | 0.01 | 6.32 | 0.45 |
| vscc | 0.00 | 0.01 | 0.30 | 0.48 | 0.60 | 0.29 | **1.75** | 1.70 |

#### 4.4.2.c Résultats scénario 3 : $p_K = 10; d = 50; q = 50$

Les résultats du scénario 3 sont disponibles dans la Table 4.5. Cette fois, seules les méthodes où les données d'entrée ont été normalisées par ICS fonctionnent et finalement les résultats sont très similaires de ceux présentés dans la Table 4.4. Nous pouvons noter que nous obtenons de meilleurs résultats que dans la Section 2.9 pour le scénario $p_K = 10; d = 100$ décrit dans la Table 2.4. Le nombre de variables de bruit est le même mais la quantité de bruit est moindre car les $q = 50$ variables de bruit corrélées impliquent de l'information redondante qui est plus facile à gérer par l'algorithme que 50 variables de bruit indépendantes.

#### 4.4.2.d Résultats scénario supplémentaire

Nous ajoutons un dernier scénario pour illustrer

**Table 4.5 :** Le tableau représente les moyennes et écart type de l'ARI par méthode pour le schéma de simulation décrit par l'Équation 4.5 avec $n = 100, m = 0.85$ et $r = 0.7$ sur 20 simulations. Les nombres de variables par groupe sont $p_K = 10; d = 50; q = 50$.

| | ARI | | Ratio V.Imp | | Ratio V.Bruit | | Temps | |
|---|---|---|---|---|---|---|---|---|
| algorithmes | moyenne | sd | moyenne | sd | moyenne | sd | moyenne | sd |
| sparcl_ICS | **0.97** | 0.03 | **1.00** | 0.00 | 1.00 | 0.00 | 6.58 | 0.28 |
| SWKM_chgpt_ICS | **0.92** | 0.12 | 0.71 | 0.23 | **0.00** | 0.00 | **1.62** | 0.34 |
| SelVarMix_model=all | **0.53** | 0.50 | 0.54 | 0.45 | 0.86 | 0.10 | 4.16 | 0.19 |
| clustvarsel | 0.10 | 0.32 | 0.11 | 0.31 | 0.08 | 0.07 | 24.28 | 10.43 |
| SFEM | 0.00 | 0.02 | 0.80 | 0.12 | 0.71 | 0.07 | 65.94 | 4.35 |
| sparcl_norm | 0.00 | 0.01 | 1.00 | 0.00 | 1.00 | 0.00 | 7.10 | 0.62 |
| SWKM_chgpt_norm | 0.00 | 0.01 | 0.00 | 0.00 | 0.12 | 0.09 | **1.32** | 0.18 |
| VarSelLCM | 0.00 | 0.01 | 0.01 | 0.03 | 0.51 | 0.01 | 6.20 | 0.46 |
| vscc | 0.00 | 0.01 | 0.30 | 0.48 | 0.60 | 0.29 | **1.72** | 1.69 |

### 4.4.3 Résumé des résultats

Les résultats des scénarios mettent en lumière un point très important qui est qu'aucune des méthodes testées ne semble fonctionner sur des données contenant des variables bruit corrélées, mise à part les méthodes avec normalisation par ICS. Les scénarios présentés sont pourtant extrêmement simples et réalistes. Ils représentent assez bien des données auxquelles nous pourrions être confrontées dans la vie réelle.

Par ailleurs les résultats sont encore plus tranchés que dans la Section 2.9. En effet, l'algorithme Sparcl sélectionne toutes les variables importantes et les variables de bruit. En fait, c'est simplement le modèle $\lambda = 0$ qui est à chaque fois choisi. Les variables de bruit ont forcément des poids très faibles, proches de 0 car l'ARI n'est pas affecté. En un sens c'est donc un bon modèle du point de vue de l'ARI et le classement des variables doit aussi être bon. Néanmoins ce scénario reste très simple. Sur les données réelles, le groupe de variables importantes peut contenir des variables qui portent plus ou moins d'informations sur le clustering, ce qui donnera des poids à $\lambda = 0$ beaucoup plus bruités. Dans la Section 2.9, sur un schéma tout aussi simple mais avec plus de variables de bruit indépendantes, l'algorithme Sparcl (c'est-à-dire le choix du $\lambda$ avec le Gap statistic) avait déjà de grandes difficultés pour sélectionner les variables.

Comme expliqué à la Section 4.4.1, SelVarMix a des résultats très aléatoires. Après vérification et comme pour le package `Rmixmod`, il peut arriver que le modèle donne un clustering en une seule classe alors même que $K = 2$ était fixé dans les paramètres et que $K = 2$ est bien affiché dans le paramètre de sortie `nbcluster` : "The selected number of clusters". C'est donc le même problème que celui rencontré précédemment et cela peut s'expliquer par le fait que le package `SelVarMix` dépende du package `Rmixmod` à vérifier. Nous avons tenté de contacter les auteurs des deux packages en leur envoyant le code et les simulations qui posaient problème. Nous avons bien reçu un retour accusant la réception de notre demande mais cela n'a pas donné suite. Nous pouvons tout de même conclure que, en l'état actuelle des choses, le package `SelVarMix` doit être utilisé avec précaution par les utilisateurs.

Finalement l'algorithme SWKM_chgpt_ICS, c'est-à-dire le Sparse $K$-means avec détection de rupture dont les données d'entrée sont normalisées par ICS, offre un bon compromis en terme de performance, de sélection de variables et de coût algorithmique. Il peut être utilisé comme un outil exploratoire en grande dimension et sur des données très corrélées.

### 4.4.4 Des explications sur les résultats

LES MODÈLES GMM SPARSE HÉRITENT DES PROBLÈMES DES GMM Dans la Section 4.4.1, que la méthode implémentée dans le package *mclust* ne parvenait pas à retrouver les clusters dans des simples de variables de bruit corrélées. Or Clustvarsel se base dans son implémentation sur `mclust`. Clustvarsel permet de faire de la sélection de variables à l'aide d'une procédure stepwise forward-backward (Section 2.5.4) et considère tour à tour chaque variable qui ne fait pas partie des variables déjà sélectionnées et évalue s'il y est pertinent d'ajouter cette variable. En revanche, si au départ `mclust` ne sélectionne pas le bon modèle, alors il semble logique que la procédure tout entière soit compromise.

De même, le modèle Vscc se base aussi dans son implémentation sur le package `mclust` et donc cette méthode hérite des mêmes problèmes. Néanmoins, nous pouvons donner une explication encore plus précise. Dans la Section 2.5.4, nous avons vu que la méthode procédera par étapes, où la première consiste à calculer les variances intra-classes, dont les classes sont calculées à l'aide d'un GMM. La première variable sélectionnée est celle qui a la plus petite variance inter-classes $\frac{v^j}{n}$. Ainsi, si le clustering est basé sur les variables de bruit corrélées, alors la première variable sera forcément une de ses variables de bruit. Ensuite, la méthode tente de ne pas sélectionner des variables corrélées au variables déjà sélectionnées en fixant un seuil sur les corrélations mais visiblement cela n'est pas une bonne solution au regard du schéma de simulations que nous proposons.

Au final, cela est proche de la discussion tenue dans la Section 2.6.2 sur l'initialisation des algorithmes.

Simplement dans ces méthodes non convexes il faut être vigilant sur les solutions obtenues aux premières étapes de l'algorithme.

LES HYPOTHÈSES DE CERTAINS MODÈLES GMM SPARSES Dans ce paragraphe, les résultats des méthodes GMM sparses sont analysées. Nous constatons que les résultats des méthodes GMM sparse sont très inférieurs à ceux des méthodes normalisées par ICS et cela peut être attribué aux hypothèses faites par ses modèles. En effet, VarSelLCM et SFEM supposent que le bruit est isotrope, c'est-à-dire que la matrice de covariance des variables de bruit est diagonale et que les éléments diagonaux sont tous égaux (cite). Pour le modèle VarSelLCM, cela a été introduit dans la Section 2.5.4 et plus précisément dans l'article les auteurs indiquent "A variable is said to be *irrelevant* to the clustering if its one-dimensional marginal distributions are equal between classes" et de surcroît ils indiquent que cela correspond à $\mu_{k,l} = \mu_{k,j}$ et $\sigma_{k,l} = \sigma_{k,j}$, $\forall k = 1, ..., K$ (Marbac and Sedki, 2017). Pour le modèle SFEM, les auteurs indiquent que "the discriminative and the non-discriminative subspaces are orthogonal, which suggests in practice that all the relevant clustering information remains in the latent subspace" et que le bloc correspondant à la covariance du bruit s'écrit $\text{diag}(\beta_k, ..., \beta_k) \in \mathbb{R}^{p-d}$ dans le cas le plus général, avec $p$ le nombre de variables de départ et $d$ la dimension de l'espace latent et $\beta_k$ détermine la variance du bruit par cluster (Bouveyron and Brunet-Saumard, 2014a,b).

LES AUTRES MODÈLES Sparcl_norm et SWKM_norm ont les problèmes expliqués précédemment. SelVarMix hérite du problème du package *Rmixmod*, où un modèle avec une partition en une classe peut être modélisé et sélectionné (malgré le fait que le nombre de classes soit fixé à 2). Néanmoins, dans sa formulation SelVarMix

## 4.5 Conclusion

Dans ce chapitre,

## 4.6 Annexe

### 4.6.1 Explication supplémentaire sur le lien de la normalisation par ICS avec l'ACP

Dans cette section, nous allons tenter d'expliquer précisément et différemment l'obtention de l'Équation 4.4. Rappelons que nous calculons le coefficient suivant $\nu_l^2 = \sum_{j=1}^{p} \text{cor}(\mathbf{x}^l, \mathbf{x}^j)^2$, or étant donné que les colonnes de $\mathbf{X}$ sont centrées en 0 et de variance 1 on obtient :

$$\sum_{j=1}^{p} \text{cor}(\mathbf{x}^l, \mathbf{x}^j)^2 = \sum_{j=1}^{p} (\mathbf{x}^{l\intercal}\mathbf{x}^j)^2 \text{ car } \mathbf{X} \text{ est normalisée} \tag{4.6}$$

$$= \sum_{j=1}^{p} \mathbf{x}^{l\intercal}\left(\mathbf{x}^j\mathbf{x}^{j\intercal}\right)\mathbf{x}^l \tag{4.7}$$

$$= \mathbf{x}^{l\intercal}\left(\sum_{j=1}^{p} \mathbf{x}^j\mathbf{x}^{j\intercal}\right)\mathbf{x}^l \tag{4.8}$$

$$= \mathbf{x}^{l\intercal}\left(\mathbf{X}\mathbf{X}^\intercal\right)\mathbf{x}^l. \tag{4.9}$$

Or on a que $\mathbf{X} = \mathbf{F}\mathbf{V}^\intercal$ avec $\mathbf{F} = [\mathbf{f}^1, \ldots, \mathbf{f}^p] \in \mathbb{R}^{n\times p}$ les composantes principales en supposant $p > n$ et $\mathbf{V} = [\mathbf{v}^1, \ldots, \mathbf{v}^p] \in \mathbb{R}^{p\times p}$ la matrice de coefficient qui permet de ramener les points du nouveau système de coordonnées $\mathbf{F}$ à l'ancien $\mathbf{X}$, sans qu'aucune dimension ou information ne soient perdues car on peut totalement reconstruire l'espace en prenant toutes les composantes principales. Alors, nous obtenons :

$$(4.9) = \mathbf{x}^{l\intercal}\left(\mathbf{F}\mathbf{V}^\intercal\mathbf{V}\mathbf{F}^\intercal\right)\mathbf{x}^l. \tag{4.10}$$

De plus, $\mathbf{V}^\intercal\mathbf{V} = \mathbf{I}$, cette dernière est évidemment une matrice $p \times p$ qui effectue une transformation du nouveau système de coordonnées (à basse dimension) à l'ancien système de coordonnées (à haute dimension), et vice-versa, ce qui n'a aucun effet. Ainsi, nous obtenons :

$$(4.10) = \left(\mathbf{x}^{l\intercal}\mathbf{F}\right)\left(\mathbf{x}^{l\intercal}\mathbf{F}\right)^\intercal. \tag{4.11}$$

Par ailleurs, $(\mathbf{x}^l)^\intercal \mathbf{F} = [(\mathbf{x}^l)^\intercal \mathbf{f}^1, \ldots, (\mathbf{x}^l)^\intercal \mathbf{f}^p]$ donc

$$\sum_{j=1}^{p} \text{cor}(\mathbf{x}^l, \mathbf{x}^j)^2 = \sum_{j=1}^{p} \text{cov}(\mathbf{x}^l, \mathbf{x}^j)^2 \text{ car } \mathbf{X} \text{ est normalisée} \tag{4.12}$$

$$= \sum_{\alpha=1}^{p} \text{cov}(\mathbf{x}^l, \mathbf{f}^\alpha)^2 \tag{4.13}$$

$$= \sum_{\alpha=1}^{p} \text{cor}(\mathbf{x}^l, \mathbf{f}^\alpha)^2 \times \lambda_\alpha \text{ car } \mathbf{f}^\alpha \text{ est de variance } \lambda_\alpha, \tag{4.14}$$

or $\mathbf{x}^j = \sum_{\alpha'=1}^{p} \mathbf{f}^{\alpha'} \mathbf{v}_j^{\alpha'}$ donc

$$\frac{< \mathbf{x}^l, \mathbf{f}^\alpha >}{\lambda_\alpha} = \frac{< \sum_{\alpha'=1}^{p} \mathbf{f}^{\alpha'} \mathbf{v}_j^{\alpha'}, \mathbf{f}^\alpha >}{\sqrt{\lambda_\alpha}} \tag{4.15}$$

$$= \frac{< \mathbf{f}^\alpha \mathbf{v}_j^\alpha, \mathbf{f}^\alpha >}{\sqrt{\lambda_\alpha}}, \tag{4.16}$$

et $\forall l, j = 1, \ldots, p,\ l \neq j$ on a $< \mathbf{f}^l, \mathbf{f}^j >= 0$ car les composantes principales sont indépendantes. De plus, $< \mathbf{f}^\alpha, \mathbf{f}^\alpha >= \text{var}(\mathbf{f}^\alpha) = \lambda_\alpha$, donc on obtient :

$$(4.16) = \frac{\lambda_\alpha \mathbf{v}_i^{\alpha'}}{\sqrt{\lambda_\alpha}} = \sqrt{\lambda_\alpha} \mathbf{v}_j^{\alpha'}. \tag{4.17}$$

Ainsi, nous obtenons la formulation suivante :

$$\sum_{j=1}^{p} \text{cor}(\mathbf{x}^l, \mathbf{x}^j)^2 = \sum_{\alpha=1}^{p} \text{cor}(\mathbf{x}^l, \mathbf{f}^\alpha)^2 \times \lambda_\alpha = \sum_{\alpha=1}^{p} (\mathbf{v}_j^{\alpha'})^2 \lambda_\alpha^2, \tag{4.18}$$

ce qui explique plus précisément et différemment l'obtention de l'Équation 4.4.

# Articles publiés et en cours de publication

# Bibliographie


M. Al Hasan, V. Chaoji, S. Salem, and M. J. Zaki. Robust partitional clustering by outlier and density insensitive seeding. *Pattern Recognition Letters*, 30(11) :994–1002, 2009.

J. L. Andrews and P. D. McNicholas. Variable selection for clustering and classification. *Journal of Classification*, 31(2) :136–153, 2014.

E. Arias-Castro and X. Pu. A simple approach to sparse clustering. *Computational Statistics & Data Analysis*, 105 :217–228, 2017.

D. Arthur and S. Vassilvitskii. k-means++ : The Advantages of Careful Seeding. In *ACM-SIAM Symposium on Discrete Algorithms*, pages 1027–1035, 2007.

J. D. Banfield and A. E. Raftery. Model-based gaussian and non-gaussian clustering. *Biometrics*, pages 803–821, 1993.

S. Ben-David. Clustering-what both theoreticians and practitioners are doing wrong. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

P. J. Bickel and E. Levina. Some theory for fisher's linear discriminant function,naive bayes', and some alternatives when there are many more variables than observations. *Bernoulli*, 10(6) :989–1010, 2004.

C. Biernacki, G. Celeux, and G. Govaert. Exact and monte carlo calculations of integrated likelihoods for the latent class model. *Journal of Statistical Planning and Inference*, 140(11) :2991–3002, 2010.

C. M. Bishop. Pattern recognition. *Machine learning*, 128(9), 2006.

C. Bouveyron and C. Brunet. Simultaneous model-based clustering and visualization in the fisher discriminative subspace. *Statistics and Computing*, 22(1) :301–324, 2012.

C. Bouveyron and C. Brunet-Saumard. Discriminative variable selection for clustering with the sparse fisher-em algorithm. *Computational Statistics*, 29(3) :489–513, 2014a.

C. Bouveyron and C. Brunet-Saumard. Model-based clustering of high-dimensional data : A review. *Computational Statistics & Data Analysis*, 71 :52–78, 2014b.

C. Bouveyron, G. Celeux, T. B. Murphy, and A. E. Raftery. *Model-based clustering and classification for data science : with applications in R*, volume 50. Cambridge University Press, 2019.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. doi : 10.1017/CBO9780511804441.

S. Boyd, N. Parikh, and E. Chu. *Distributed optimization and statistical learning via the alternating direction method of multipliers*. Now Publishers Inc, 2011.

M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. Lof : identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 93–104, 2000.

Š. Brodinová, P. Filzmoser, T. Ortner, C. Breiteneder, and M. Rohm. Robust and sparse k-means clustering for high-dimensional data. *Advances in Data Analysis and Classification*, 13(4) :905–932, 2019.

P. Bühlmann, P. Rütimann, S. van de Geer, and C.-H. Zhang. Correlated variables in regression : clustering and sparse estimation. *Journal of Statistical Planning and Inference*, 143(11) :1835–1858, 2013.

M. E. Burczynski, R. L. Peterson, N. C. Twine, K. A. Zuberek, B. J. Brodeur, L. Casciotti, V. Maganti, P. S. Reddy, A. Strahs, F. Immermann, et al. Molecular classification of crohn's disease and ulcerative colitis patients using transcriptional profiles in peripheral blood mononuclear cells. *The journal of molecular diagnostics*, 8(1) :51–61, 2006.

N. Campbell and R. Mahon. A multivariate study of variation in two species of rock crab of the genus leptograpsus. *Australian Journal of Zoology*, 22(3) :417–425, 1974.

G. Celeux and G. Govaert. Gaussian parsimonious clustering models. *Pattern recognition*, 28(5) :781–793, 1995.

G. Celeux, M.-L. Martin-Magniette, C. Maugis-Rabusseau, and A. E. Raftery. Comparing model selection and regularization approaches to variable selection in model-based clustering. *Journal de la Societe francaise de statistique*, 155(2) :57–71, 2014.

G. Celeux, C. Maugis-Rabusseau, and M. Sedki. Variable selection in model-based clustering and discriminant analysis with a regularization approach. *Advances in Data Analysis and Classification*, 13(1) :259–278, 2019.

S. Chakraborty and S. Das. A strongly consistent sparse $k$-means clustering with direct $l\_1$ penalization on variable weights. *arXiv preprint arXiv :1903.10039*, 2019.

S. Chakraborty, D. Paul, S. Das, and J. Xu. Entropy weighted power k-means clustering. In *International Conference on Artificial Intelligence and Statistics*, pages 691–701. PMLR, 2020.

M. Chavent and G. Chavent. Group-sparse block pca and explained variance. *arXiv preprint arXiv :1705.00461*, 2017.

M. Chavent, Y. Lechevallier, and O. Briant. Divclus-t : A monothetic divisive hierarchical clustering method. *Computational Statistics & Data Analysis*, 52(2) :687–701, 2007.

M. Chavent, V. Kuentz, B. Liquet, and L. Saracco. Clustofvar : An r package for the clustering of variables. *arXiv preprint arXiv :1112.0295*, 2011.

M. Chavent, V. Kuentz-Simonet, and J. Saracco. Orthogonal rotation in pcamix. *Advances in Data Analysis and Classification*, 6(2) :131–146, 2012.

M. Chavent, J. Lacaille, A. Mourer, and M. Olteanu. Sparse k-means for mixed data via group-sparse clustering. ESANN, 2020.

X. Chen, Y. Ye, X. Xu, and J. Z. Huang. A feature group weighting method for subspace clustering of high-dimensional data. *Pattern Recognition*, 45(1) :434–446, 2012.

L. Clemmensen, T. Hastie, D. Witten, and B. Ersbøll. Sparse discriminant analysis. *Technometrics*, 53(4) : 406–413, 2011.

J. A. Cuesta-Albertos, A. Gordaliza, and C. Matrán. Trimmed $k$-means : An attempt to robustify quantizers. *The Annals of Statistics*, 25(2) :553–576, 1997.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society : Series B (Methodological)*, 39(1) :1–22, 1977.

J. Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7 :1–30, 2006.

A. W. Diallo, N. Niang, and M. Ouattara. Sparse subspace k-means. 2021.

C. Ding and X. He. K-means clustering via principal component analysis. In *Proceedings of the twenty-first international conference on Machine learning*, page 29, 2004.

C. Ding and T. Li. Adaptive dimension reduction using discriminant analysis and k-means clustering. In *Proceedings of the 24th international conference on Machine learning*, pages 521–528, 2007.

C. Domeniconi, D. Gunopulos, S. Ma, B. Yan, M. Al-Razgan, and D. Papadopoulos. Locally adaptive metrics for clustering high dimensional data. *Data Mining and Knowledge Discovery*, 14(1) :63–97, 2007.

B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *The Annals of statistics*, 32(2) : 407–499, 2004.

M. Fop and T. B. Murphy. Variable selection methods for model-based clustering. *Statistics Surveys*, 12 : 18–65, 2018.

M. Fop, K. M. Smart, and T. B. Murphy. Variable selection for latent class analysis with application to low back pain diagnosis. *The Annals of Applied Statistics*, pages 2080–2110, 2017.

A. Foss, M. Markatou, B. Ray, and A. Heching. A semiparametric method for clustering mixed data. *Machine Learning*, 105(3) :419–458, 2016.

C. Fraley and A. E. Raftery. Model-based clustering, discriminant analysis, and density estimation. *Journal of the American statistical Association*, 97(458) :611–631, 2002.

J. Friedman, T. Hastie, R. Tibshirani, et al. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.

J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3) :432–441, 2008.

J. Friedman, T. Hastie, and R. Tibshirani. A note on the group lasso and a sparse group lasso. *arXiv preprint arXiv :1001.0736*, 2010a.

J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1) :1, 2010b.

J. H. Friedman and J. J. Meulman. Clustering objects on subsets of attributes (with discussion). *Journal of the Royal Statistical Society : Series B (Statistical Methodology)*, 66(4) :815–849, 2004.

R. Gnanadesikan, J. R. Kettenring, and S. L. Tsao. Weighting and selection of variables for cluster analysis. *Journal of classification*, 12(1) :113–136, 1995.

W. Gong, R. Zhao, and S. Grünewald. Structured sparse k-means clustering via laplacian smoothing. *Pattern Recognition Letters*, 112 :63–69, 2018.

J. C. Gower. A general coefficient of similarity and some of its properties. *Biometrics*, pages 857–871, 1971.

J. A. Hartigan and M. A. Wong. Algorithm as 136 : A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)*, 28(1) :100–108, 1979.

T. Hastie, R. Tibshirani, and M. Wainwright. *Statistical learning with sparsity : the lasso and generalizations*. Chapman and Hall/CRC, 2019.

C. Higuera, K. J. Gardiner, and K. J. Cios. Self-organizing feature maps identify proteins critical to learning in a mouse model of down syndrome. *PloS one*, 10(6) :e0129126, 2015.

D. P. Hofmeyr. Degrees of freedom and model selection for k-means clustering. *arXiv preprint arXiv :1806.02034*, 2018.

K. Honda, H. Araki, T. Matsui, and H. Ichihashi. A new approach to robust k-means clustering based on fuzzy principal component analysis. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pages 208–213. IEEE, 2008.

K. Honda, A. Notsu, and H. Ichihashi. Pca-guided k-means with variable weighting and its application to document clustering. In *International Conference on Modeling Decisions for Artificial Intelligence*, pages 282–292. Springer, 2009.

J. Z. Huang, M. K. Ng, H. Rong, and Z. Li. Automated variable weighting in k-means type clustering. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (5) :657–668, 2005.

Z. Huang. Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data mining and knowledge discovery*, 2(3) :283–304, 1998.

L. Hubert and P. Arabie. Comparing partitions. *Journal of classification*, 2(1) :193–218, 1985.

Z. Huo and G. Tseng. Integrative sparse $k$-means with overlapping group lasso in genomic applications for disease subtype discovery. *Ann. Appl. Stat.*, 11(2) :1011–1039, 06 2017. doi : 10.1214/17-AOAS1033. URL https://doi.org/10.1214/17-AOAS1033.

L. Jacob, G. Obozinski, and J.-P. Vert. Group lasso with overlap and graph lasso. In *Proceedings of the 26th annual international conference on machine learning*, pages 433–440, 2009.

L. Jing, M. K. Ng, and J. Z. Huang. An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data. *IEEE Transactions on Knowledge & Data Engineering*, (8) :1026–1041, 2007.

C. M. Judd, G. H. McClelland, and C. S. Ryan. *Data analysis : A model comparison approach*. Routledge, 2011.

A. Kessy, A. Lewin, and K. Strimmer. Optimal whitening and decorrelation. *The American Statistician*, 72 (4) :309–314, 2018.

Y. Kondo, M. Salibian-Barrera, and R. Zamar. A robust and sparse k-means clustering algorithm. *arXiv preprint arXiv :1201.6082*, 2012.

Y. Kondo, M. Salibian-Barrera, and R. Zamar. Rskc : an r package for a robust and sparse k-means clustering algorithm. *Journal of Statistical Software*, 72(1) :1–26, 2016.

K. Lange. *MM optimization algorithms*. SIAM, 2016.

M. H. Law, M. A. Figueiredo, and A. K. Jain. Simultaneous feature selection and clustering using mixture models. *IEEE transactions on pattern analysis and machine intelligence*, 26(9) :1154–1166, 2004.

É. Lebarbier and T. Mary-Huard. Une introduction au critère bic : fondements théoriques et interprétation. *Journal de la Société française de statistique*, 147(1) :39–57, 2006.

R. Lebret, S. Iovleff, F. Langrognet, C. Biernacki, G. Celeux, and G. Govaert. Rmixmod : the r package of the model-based unsupervised, supervised and semi-supervised classification mixmod library. *Journal of Statistical Software*, 67(6) :241–270, 2015.

S. P. Lloyd. Least Squares Quantization in PCM. *IEEE Transactions on Information Theory*, 28(2) :129–137, 1982. ISSN 15579654. doi : 10.1109/TIT.1982.1056489.

J. MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.

M. Marbac and M. Sedki. Variable selection for model-based clustering using the integrated complete-data likelihood. *Statistics and Computing*, 27(4) :1049–1063, 2017.

M. Marbac, M. Sedki, and T. Patin. Variable selection for mixed data clustering : application in human population genomics. *Journal of Classification*, 37(1) :124–142, 2020.

B. W. Matthews. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 405(2) :442–451, 1975.

C. Maugis, G. Celeux, and M.-L. Martin-Magniette. Variable selection for clustering with gaussian mixture models. *Biometrics*, 65(3) :701–709, 2009a.

C. Maugis, G. Celeux, and M.-L. Martin-Magniette. Variable selection in model-based clustering : A general variable role modeling. *Computational Statistics & Data Analysis*, 53(11) :3872–3882, 2009b.

M. Meilă. The uniqueness of a good optimum for k-means. In *Proceedings of the 23rd international conference on Machine learning*, pages 625–632, 2006.

D. S. Modha and W. S. Spangler. Feature weighting in k-means clustering. *Machine learning*, 52(3) :217–237, 2003.

W. Pan and X. Shen. Penalized model-based clustering with application to variable selection. *Journal of Machine Learning Research*, 8(May) :1145–1164, 2007.

R. Penrose. A generalized inverse for matrices. In *Mathematical proceedings of the Cambridge philosophical society*, volume 51, pages 406–413. Cambridge University Press, 1955.

A. E. Raftery and N. Dean. Variable selection for model-based clustering. *Journal of the American Statistical Association*, 101(473) :168–178, 2006.

S. Romano, N. X. Vinh, J. Bailey, and K. Verspoor. Adjusting for chance clustering comparison measures. *arXiv preprint arXiv :1512.01286*, 2015.

G. Schwarz. Estimating the dimension of a model. *The annals of statistics*, pages 461–464, 1978.

L. Scrucca and A. E. Raftery. clustvarsel : a package implementing variable selection for gaussian model-based clustering in r. *Journal of Statistical Software*, 84, 2018.

L. Scrucca, M. Fop, T. B. Murphy, and A. E. Raftery. mclust 5 : clustering, classification and density estimation using gaussian finite mixture models. *The R journal*, 8(1) :289, 2016.

M. Sedki, G. Celeux, and C. Maugis-Rabusseau. Selvarmix : Ar package for variable selection in model-based clustering and discriminant analysis with a regularization approach. *INRIA Techical report*, 2014.

N. Simon, J. Friedman, T. Hastie, and R. Tibshirani. A sparse-group lasso. *Journal of Computational and Graphical Statistics*, 22(2) :231–245, 2013.

T. Su and J. Dy. A deterministic method for initializing k-means clustering. In *16th IEEE international conference on tools with artificial intelligence*, pages 784–786. IEEE, 2004.

W. Sun, J. Wang, Y. Fang, et al. Regularized k-means clustering of high-dimensional data and its asymptotic consistency. *Electronic Journal of Statistics*, 6 :148–167, 2012.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society : Series B (Methodological)*, 58(1) :267–288, 1996.

R. Tibshirani, G. Walther, and T. Hastie. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society : Series B (Statistical Methodology)*, 63(2) :411–423, 2001.

A. N. Tikhonov. On the stability of inverse problems. In *Dokl. Akad. Nauk SSSR*, volume 39, pages 195–198, 1943.

N. T. Trendafilov and I. T. Jolliffe. Dalass : Variable selection in discriminant analysis via the lasso. *Computational Statistics & Data Analysis*, 51(8) :3718–3736, 2007.

L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

F. Wang, Q. Wang, F. Nie, Z. Li, W. Yu, and R. Wang. Unsupervised linear discriminant analysis for jointly clustering and subspace learning. *IEEE Transactions on Knowledge and Data Engineering*, 2019.

H. Wang and C. Leng. A note on adaptive group lasso. *Computational statistics & data analysis*, 52(12) : 5277–5286, 2008.

S. Wang and J. Zhu. Variable selection for model-based high-dimensional clustering and its application to microarray data. *Biometrics*, 64(2) :440–448, 2008.

J. H. Ward. Hierarchical Grouping to Optimize an Objective Function. *Journal of the American Statistical Association*, 58(301) :236–244, 1963. https://www.jstor.org/stable/2282967.

D. M. Witten and R. Tibshirani. A framework for feature selection in clustering. *Journal of the American Statistical Association*, 105(490) :713–726, 2010. doi : 10.1198/jasa.2010.tm09415. URL https://doi.org/10.1198/jasa.2010.tm09415. PMID : 20811510.

D. M. Witten and R. Tibshirani. Penalized classification using fisher's linear discriminant. *Journal of the Royal Statistical Society : Series B (Statistical Methodology)*, 73(5) :753–772, 2011.

D. M. Witten, R. Tibshirani, and T. Hastie. A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. *Biostatistics*, 10(3) :515–534, 2009.

B. Xie, W. Pan, and X. Shen. Penalized model-based clustering with cluster-specific diagonal covariance matrices and grouped variables. *Electronic journal of statistics*, 2 :168, 2008.

J. Xu and K. Lange. Power k-means clustering. In *International Conference on Machine Learning*, pages 6921–6931. PMLR, 2019.

Q. Xu, C. Ding, J. Liu, and B. Luo. Pca-guided search for k-means. *Pattern Recognition Letters*, 54 :50–55, 2015.

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society : Series B (Statistical Methodology)*, 68(1) :49–67, 2006.

H. Zhou, W. Pan, and X. Shen. Penalized model-based clustering with unconstrained covariance matrices. *Electronic journal of statistics*, 3 :1473, 2009.

H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the royal statistical society : series B (statistical methodology)*, 67(2) :301–320, 2005.