Une famille de matrices sparses pour une modélisation multi-échelle par blocs

Camille Brunet*, Thomas Villman**
Vincent Vigneron*

*IBISC, Université d'Evry Val d'Essonnes
40 rue du Pelvoux - 91020 Evry Courcouronne - France
camille.brunet@ibisc.univ-evry.fr
**University of applied sciences Mittweida
Technikumplatz 17 - 09648 Mittweida - Germany

Résumé. La sériation est une technique d'analyse de données qui ordonne les observations directement à partir de leur tableau de valeurs afin de révéler une structure intrinsèque à ces données. Une telle approche présente de nombreux avantages de visualisation mais dès lors que les données sont bruitées ou que les groupes se superposent, la visualisation de toute structure devient difficile. Pour faire face à ces problèmes, nous introduisons de la parcimonie dans les données à travers une famille de matrices indicatrices de voisins communs. Celles-ci sont ordonnées selon un algorithme de type *branch and bound* et la matrice révélant la meilleure structure au sens de "diagonale par blocs" est sélectionnée au moyen d'un critère dérivé des problématiques de compression de données. Cet outil de partitionnement identifie des sous-ensembles de données relatifs aux *clusters* tout en écartant celles qui sont bruitées ou extrêmes ce qui permet de visualiser la structure globale intrinsèque aux données. Cependant, une trop grande sparsité des données amène parfois à l'éviction de données sous-représentées; nous proposons à cet effet, une approche multi-échelle combinant différents niveaux de sparsité dans une même visualisation.

1 Introduction

La notion de complexité des données peut s'appréhender de différentes manières dans la littérature. Elle peut en effet être liée à la nature et à l'hétérogénéité des données, à leur dimension plus ou moins grande, à leur qualité (données très bruitées ou non) ou encore à leur structure globale qui peut être telle que la distinction de groupes dans les données est rendue difficile. Les outils statistiques d'analyse de données ont pour objectif d'extraire de l'information. Ils cherchent à explorer des données multidimensionnelles pour les synthétiser d'une part et permettre de structurer l'information contenue dans ces données d'autre part. Dans la littérature, il existe plusieurs approches associées à cette démarche telles les méthodes factorielles (Pearson (1901); Jolliffe (1986)), les approches par classification non supervisée –

ou *clustering* – ou encore les méthodes de visualisation. Ces dernières peuvent être globales puisqu'elles se basent sur les proximités entre les différents groupes telles les cartes de Kohonen (Kohonen (1995)) ou les méthodes de graphes ; locales puisqu'elles évaluent la proximité entre les individus, comme les méthodes hiérarchiques ; ou encore elles peuvent combiner les relations locales et globales, comme dans le cas de la sériation. Cette dernière approche a la particularité de travailler directement sur le tableau de valeurs que ce soit sur des matrices symétriques pour lesquelles les éléments en lignes et en colonnes se réfèrent à la même entité, comme les tableaux de données relationnelles ou de dissimilarités ; ou que ce soit sur des matrices de type rectangulaire pour lesquelles les éléments en lignes diffèrent de ceux en colonnes ; par exemple, les individus sont représentés en lignes tandis que les attributs le sont sur les colonnes. Bien que la sériation soit un champ de la statistique très vaste du fait de ses multiples origines (cf. I.Liiv (2010)) et de ses diverses évolutions (cf. Marcotorchino (1987); Mechelen et al. (2004)), une taxonomie des méthodes a été proposée par Carroll et Arabie (1980) et associe à chaque type de tableau un mode de sériation particulier. L'objectif de la sériation est de réordonner les lignes et les colonnes d'un tableau de valeurs par des permutations successives de sorte que les lignes adjacentes (respectivement les colonnes) soient les plus semblables. Cette situation est illustrée par la figure 1 où, à partir d'un tableau de données relationnelles présenté par la figure 1 de gauche, les lignes et les colonnes ont permuté pour former une partition dans laquelle les éléments similaires ont été rassemblés entre eux, formant ainsi des groupes (figure 1 du milieu), et afin de mieux apprécier la présence de la structure diagonale par blocs, cette matrice ordonnée est pixellisée (figure 1 de droite). Une telle approche pourrait s'apparenter à une technique locale de clustering ordonné dans la mesure où une information est apportée d'une part sur les relations locales entre individus du fait d'un ordre dans les données et d'autre part sur la structure globale des données. Cependant, la sériation présente d'autres avantages mis en avant par plusieurs auteurs tels Arabie et al. (1996), comme l'absence de connaissance *a priori* sur le nombre de clusters et la visualisation directe de la structure sur le tableau de valeurs.

	A	B	C	D	E	F
A	1	0	0	1	0	0
B	0	1	0	0	0	0
C	0	0	1	0	1	1
D	1	0	0	1	0	0
E	0	0	1	0	1	1
F	0	0	1	0	1	1

	D	A	E	F	C	B
D	1	1	0	0	0	0
A	1	1	0	0	0	0
E	0	0	1	1	1	0
F	0	0	1	1	1	0
C	0	0	1	1	1	0
B	0	0	0	0	0	1

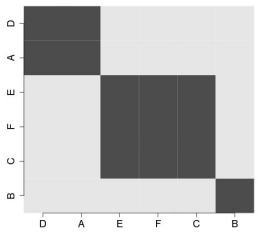


FIG. 1 – *Effets d'un algorithme de rangement sur un jeu de données : matrice de relations initiale (à gauche), matrice ordonnée (au milieu) et représentation pixellisée de la matrice ordonnée (à droite).*

Ces avantages se voient annulés dès lors que les données sont bruitées ou que les groupes de données se superposent. En effet, la présence de données bruitées empêche une visualisation claire des différents blocs et distinguer des clusters devient alors une tâche difficile. Dans ce travail nous appréhenderons la complexité des données à travers la notion

de structure : des données seront complexes dès lors que leur structuration, leur agencement en groupes sera difficilement appréhendable. Cette complexité se traduira par des données fortement bruitées ou par des recouvrements de classes rendant difficile la visualisation d'une structure. Dans ce contexte, notre travail tente d'améliorer les procédures traditionnelles de sériation dans la recherche de sous-ensembles de données caractéristiques de la structure. Pour ce faire, nous introduisons une famille de matrices parcimonieuses relatives à différents niveaux de parcimonie des données, lesquelles sont ordonnées selon un algorithme de type *branch and bound* pour faire émerger une structure compacte par blocs. Un critère dérivé de la problématique de compression des données sélectionne la matrice ordonnée la plus compacte –au sens de diagonale par blocs– afin d'obtenir la visualisation la plus informative de la structure intrinsèque des données. Cependant, dans certaines situations, une trop grande parcimonie engendre l'éviction de données sous-représentées formant de très petits clusters. Pour pallier cette non-détection, nous proposons une approche multi-échelle combinant différents niveaux de sparsité des données.

Cet article est organisé de la manière suivante : dans la Section 2, un état de l'art de la sériation sera présenté selon deux points de vue différents, l'un basé sur les fondements mathématiques du problème de la sériation et l'autre sur ses fondements algorithmiques. La Section 3 détaillera notre approche parcimonieuse ainsi que l'algorithme de rangement Parcimonious Block-Clustering multi-échelle proposé. Des expériences sur simulations et sur des données de référence seront présentées dans la Section 4.

2 La sériation

Liiv (I.Liiv (2010)) propose une définition informelle de la sériation : *[Seriation] is an exploratory data analysis technique to reorder objects into a sequence along one dimensional continuum so that it best reveals regularity and patterning among the whole series*. Autrement dit, la sériation cherche un ordre dans les données de sorte à exprimer une localité/proximité entre lignes ou colonnes adjacentes et faire ainsi émerger une structure. Cet ordre est obtenu par permutations successives des lignes et des colonnes ce qui permet d'aborder le problème de la sériation à travers deux angles différents : l'un étant de fondement mathématique puisque l'enjeu est de déterminer l'ensemble des meilleures permutations possibles, l'autre étant de fondement algorithmique du fait de la complexité de la solution (problème np-complet).

2.1 Problème d'optimisation

La problématique de la sériation se pose dans la définition et l'évaluation de la meilleure permutation possible, ce qui permet de la formuler comme un problème d'optimisation. L'approche par sériation s'applique à tout type de matrices, cependant, dans notre travail, nous nous focaliserons sur des matrices de dissimilarités. Considérons alors un ensemble de N éléments $(x_1, \ldots, x_N)$ décrits par une matrice symétrique de dissimilarité $D = (d_{ij})_{i,j \in (1,\ldots,N)}$ de taille $N \times N$ où chaque élément d_{ij} rend compte de la dissimilarité entre la paire d'observations (x_i, x_j). On définit Ψ une fonction de permutation qui ordonne, selon un certain critère $\mathcal{C}$, les éléments de la matrice D. L'objectif de la sériation est donc de trouver la fonction de permutation optimale Ψ^* qui optimise le critère $\mathcal{C}$ de rangement, telle que :

$$\Psi^* = \underset{\Psi}{\operatorname{argmax}}\, \mathcal{C}(\Psi(D)). \tag{1}$$

Ces critères prennent différentes formes dans la littérature mais une grande partie d'entre eux se base sur une mesure de similarité $s(.)$ entre les éléments successifs de la matrice D et qui vérifie

$$\max \sum_{i=1}^{n-1} s(i, i+1).$$

Cette mesure de similarité se décline de manière différente selon les auteurs comme on peut l'observer dans la table 1. McCormick et al. (1972) et Arabie et Hubert (1990) par exemple cherchent à maximiser une mesure d'efficacité (*measure of effectiveness*) (cf. critère $\mathcal{C}_6$ dans la table 1) basée sur la somme des produits scalaires en lignes et en colonnes de la matrice des données ; mesure qui a été par la suite généralisée par Climer et Zhang (2006). D'autres auteurs, tels que Hubert et al. (2001) ou Chen (2002) se sont basés sur l'optimisation d'une mesure de divergence entre la matrice de dissimilarité et une structure *anti-Robinson* cherchant à regrouper les valeurs de dissimilarités les plus petites autour de la diagonale (cf. critère $\mathcal{C}_4$). A l'inverse, certains auteurs tels Caraux et Pinloche (2005) (cf. critères $\mathcal{C}_1$ et $\mathcal{C}_2$) ou Brusco et Steinley (2006) (cf. critère $\mathcal{C}_3$) cherchent plutôt à placer les plus petites dissimilarités hors de la diagonale (structure *Robinson*). Enfin, dans le cadre du problème de compression de données, Johnson et al. (2004) ont proposé de minimiser un critère basé sur le nombre de séquences d'éléments consécutifs (en ligne) différents de 0 (cf. critère $\mathcal{C}_5$). Nombre d'auteurs ont travaillé sur la proposition de nouveaux critères de rangement comme Niermann (2005) qui cherche à comparer chaque observation à ses voisins adjacents à travers des critères relatifs au voisinage (cf. critère $\mathcal{C}_7$) ou Batagelj (1997) et Doreian et al. (2004) qui proposent des critères d'équivalence structurelle ou encore Dhillon et al. (2003) qui utilise l'information mutuelle et un critère d'entropie. Il existe une grande littérature sur ce sujet et une taxonomie de ces techniques est disponible dans Hahsler et al. (2009). De plus, cette approche par sériation peut s'inscrire dans un domaine plus étendu qu'est la classification croisée, appelée aussi *biclustering* et une vue d'ensemble de ces différentes approches est détaillée dans l'article de Mechelen et al. (2004). En particulier, des auteurs tels Long et al. (2005) se sont orientés vers une décomposition en valeurs blocs de la matrice pour traiter des données de grandes dimensions ou encore Govaert et Nadif (2008, 2010) ont abordé de manière stochastique cette problématique de permutations dans un modèle probabiliste en réduisant le rangement de la matrice à un biclustering guidé par un modèle binomial. Cependant ces approches récentes nécessitent une connaissance *a priori* du nombre de clusters formés par les individus et par les variables dont leur détermination n'est pas toujours chose facile. Enfin, notons que la sériation vue comme un problème de maximisation est comparée par certains auteurs, tels Applegate et al. (2007), au problème du voyageur de commerce (TSP). En effet, une matrice de dissimilarités peut être perçue comme un graphe pondéré que l'on parcourt. Ainsi, ordonner des lignes et colonnes de sorte que les lignes (respectivement les colonnes) adjacentes soient les plus similaires possibles revient à chercher le parcours minimal pour "visiter" chaque observation une seule fois.

TYPE	CRITÈRES À OPTIMISER SELON LA MATRICE DE DISSIMILARITÉ $D = \{d_{ij}\}_{i,j\in\{1,\dots,n\}}$
CRITÈRES STRUCTURELS	$\mathcal{C}_1 = \sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}\lvert i-j\rvert^2$
	$\mathcal{C}_2 = \sum_{i=1}^{n}\sum_{i=1}^{n}(d_{ij} - \alpha\lvert i-j\rvert^2)$
	$\mathcal{C}_3 = \sum_{i=1}^{n-2}\sum_{j=i+1}^{n-1}\sum_{k=j+1}^{n} sign(d_{ij}-d_{ik}) + sign(d_{jk}-d_{ik})$
	$\mathcal{C}_4 = \sum_{1\leqslant i<j<k\leqslant n} f(d_{ik},d_{ij}) + \sum_{1\leqslant i<j<k\leqslant n} f(d_{kj},d_{ij})$ avec $f(x,y) = \text{sign}(x-y)$ $f(x,y) = \lvert x-y\rvert\text{sign}(x-y)$ $f(x,y) = \mathbb{I}_{\{x>y\}}$ $f(x,y) = \lvert x-y\rvert\mathbb{I}_{\{x>y\}}$
	$\mathcal{C}_5 = \sum_{i=1}^{n}\sum_{j\neq i,j=1}^{n-1} \lvert d_{ij} - d_{i,j+1}\rvert$
CRITÈRES DE SIMILARITÉS	$\mathcal{C}_6 = \frac{1}{2}\sum_{i,j=1}^{n} d_{ij}(d_{i,j-1} + d_{i,j+1} + d_{i-1,j} + d_{i+1,j})$
	$\mathcal{C}_7 = \sum_{i,j=1}^{n} f_{ij}$ avec $f_{ij} = \sum_{k=\max(1,i-1)}^{\min(n,i+1)}\sum_{\ell=\max(i,j-1)}^{\min(n,j+1)}(d_{ij}-d_{k\ell})^2$ $f_{ij} = \sum_{k=\max(1,i-1)}^{\min(n,i+1)}(d_{ij}-d_{kj})^2 + \sum_{\ell=\max(i,j-1)}^{\min(n,j+1)}(d_{ij}-d_{i\ell})^2$

TAB. 1 – *Table des critères de rangement utilisés dans le cadre du one-mode clustering.*

2.2 Principe algorithmique et coût calculatoire

Trouver la permutation optimale est un problème np-complet. Il suppose le calcul de toutes les combinaisons de permutations possibles de lignes et de colonnes, soit $n!p!$ pour un tableau rectangulaire ou $n!$ dans le cas d'une matrice de dissimilarité symétrique. Le calcul de toutes ces permutations possibles peut être envisagé lorsque le jeu de données est de petite taille mais cette tâche devient difficile dès lors que $n > 30$ (plus de 2^{32} possibilités). La littérature propose un certain nombre d'algorithmes de sériation. Un des algorithmes les plus connus nommé *branch and bound* suggère de faire une recherche exhaustive des permutations non pas sur l'ensemble des données mais sur plusieurs sous-ensembles. Dans les années cinquantes, plusieurs auteurs tels Croes (1958) Eastman (1958) et Rossman et al. (1958) ont proposé de telles méthodes ; celles-ci ont été réexploitées plus récemment par Chen (2002) ou par Brusco et Stahl (2005). Un autre type de méthodes algorithmiques se base sur des recherches heuristiques ; celles-ci ont été exploitées initialement par McCormick et al. (1972), puis par Arabie et Hubert (1990) qui utilisent la notion de voisinage pour définir une liste de candidats potentiels considérés à une itération donnée. La colonne (ou la ligne) sélectionnée est celle qui permet la plus grande augmentation de la mesure d'efficacité. Dans cette même optique, Kirkpatrick et al. (1983) ont introduit la notion, anglaise, du *simulated annealing paradigm* dans la sériation qui permet à l'algorithme d'accepter, avec une certaine probabilité, un candidat pouvant être pire que la solution courante. Ainsi, au début de l'algorithme, la probabilité d'acceptation d'un candidat est élevé, mais décroit graduellement à mesure du déroulement de l'algorithme. Enfin, comme la sériation est perçue comme un TSP, bon nombre de chercheurs ont travaillé sur des approches heuristiques qui sont résumées dans Gutin et Punnen (2002). La plupart de ces méthodes sont exposées dans Applegate et al. (2007) ou dans Hahsler et al. (2009).

3 Sériation par famille

Dans cette Section, nous proposons une nouvelle méthode de sériation pour faire face aux problèmes relatifs aux données bruitées, au recouvrement de *clusters* ou à la présence de petits *clusters*. Celle-ci se base sur la recherche de sous-ensembles d'individus caractérisant la structure sous-jacente de l'ensemble des données. Pour cela, nous introduisons de la parcimonie dans les données par l'intermédiaire d'une famille de matrices binaires à degrés de voisinage commun différents, lesquelles sont ordonnées selon un algorithme nommé Parsimonious Block-Clustering. Cet algorithme ordonne la famille de matrices parcimonieuses et sélectionne le niveau de parcimonie permettant de faire émerger de la matrice une structure compacte par blocs.

3.1 Une famille de matrices binaires imbriquées

Dans notre approche, le degré de voisinage est défini comme une "valeur-seuil" du nombre de voisins communs entre paires d'observations en deçà de laquelle les paires d'observations sont éliminées. Plus le nombre de voisins communs imposé est important et plus la matrice sera parcimonieuse (remplie de zéros). Par conséquent, nous associerons au degré de voisinage

commun, le degré de parcimonie. Considérons une matrice de données Y de dimension $n \times p$. Soit $Y^d = (y_{ij}^d)_{i,j\in\{1,\ldots,n\}}$ la matrice de distances associée à Y, la nature de la fonction de distance dépendant du type de données : elle peut être une distance euclidienne entre les individus i et j (et plus généralement des normes p), une corrélation, ou toute autre fonction caractérisant la notion de proximité entre des paires d'observations. Soit $A = (a_{ij})_{i,j\in\{1,\ldots,n\}}$ la matrice binaire de similarité contruite à partir de Y^d qui vérifie :

$$a_{ij} = \begin{cases} 1 & \text{si} \quad y_{ij}^d \leq \epsilon \\ 0 & \text{si} \quad y_{ij}^d > \epsilon. \end{cases} \tag{2}$$

où ϵ est un seuil de distance caractérisant la proximité des paires d'observations. Sa valeur peut être déterminée arbitrairement ; nous proposons de la fixer au premier quartile de la distribution des distances entre paires d'observations. Par ailleurs, la matrice de similarité est symétrique d'où $a_{ij} = a_{ji}$. Soit la matrice de Gram $B = A^T A$ où chaque élément b_{ij} est le nombre de voisins aux deux points i et j. Cette matrice correspond à une matrice de voisinage commun.

Définition 1
Une matrice binaire $B_{\lambda_m} = (b_{ij}^{\lambda_m})_{i,j\in\{1,\ldots,n\}}$ parcimonieuse de degré λ_m (avec $m \in \{1 \leq m \leq M\}$) se caractérise par :

$$b_{ij}^{\lambda_m} = \begin{cases} 1 & \textit{si} \quad b_{ij} \geq \lambda_m \\ 0 & \textit{si} \quad b_{ij} < \lambda_m. \end{cases} . \tag{3}$$

où b_{ij} représente les éléments de la matrice de Gram, B, définie précédemment.
$(B_{\lambda_1}, \ldots, B_{\lambda_M})$ forme une famille de matrices binaires parcimonieuses dont le degré de parcimonie est lié au nombre de voisins communs.

Au regard de cette définition, plus le seuil de voisinage imposé λ_m est élevé, moins il y a de paires d'observations qui satisfont cette condition. La matrice associée contiendra un plus grand nombre de zéros et sera donc plus parcimonieuse. La suite des degrés de parcimonie $(\lambda_m)_{m\in\{1,\ldots,M\}}$ qui vérifie $\lambda_1 < \cdots < \lambda_M$ permet d'établir une relation d'ordre $\subset$ entre les M éléments de la famille $(B_{\lambda_m})_{m\in\{1,\ldots,M\}}$:

$$B_{\lambda_M} \quad \subset \quad B_{\lambda_{M-1}} \quad \subset \ldots \quad \subset \quad B_{\lambda_1}, \tag{4}$$

dans laquelle la matrice la plus parcimonieuse est contenue dans toutes les autres matrices de sa famille. L'un des avantages d'une telle matrice est la disparition des valeurs extrêmes et du bruit lorsque le seuil de parcimonie augmente ce qui facilite le rangement de la matrice ainsi que l'apparition d'une structure diagonale par blocs. Cependant, au regard de cette famille de matrices, se pose la question de la sélection du "meilleur" niveau de parcimonie c'est-à-dire celui qui permettra d'obtenir une visualisation claire de la structure des données. Nous introduisons pour ce faire un critère de sélection $\mathcal{C}_{\lambda_m}$ de la matrice centrale B_λ^*, tel que :

Définition 2

La matrice centrale ordonnée $(B_\lambda^*)_{ord}$ contenue dans une famille de matrices ordonnées $(B_{\lambda_m})_{ord} = ((b_{i,j}^{\lambda_m})_{ord})_{i,j\in\{1,\ldots,n\}}$, $m \in I$ et $I \in \{1, \ldots, M\}$, vérifie :

$$(B^*_\lambda)_{\text{ord}} = \arg\min_{m\in I} \quad \mathcal{C}_{\lambda_m} = \arg\min_{m\in I} \quad \sum_{i=1}^{n}\sum_{j=1}^{n-1} \frac{|(b^{\lambda_m}_{i,j})_{\text{ord}} - (b^{\lambda_m}_{i,j+1})_{\text{ord}}|}{|b^{\lambda_m}_{i,j} - b^{\lambda_m}_{i,j+1}|},$$

où $B_{\lambda_m} = (b^{\lambda_m}_{i,j})_{i,j\in\{1,\dots,n\}}$ *est la matrice binaire non ordonnée de degré* λ_m.

Matrice de voisins communs	$B = \begin{smallmatrix} 2&0&0&3&0&0\\ 0&1&0&0&0&0\\ 0&0&3&0&3&3\\ 2&0&0&3&0&0\\ 0&0&3&0&3&3\\ 0&0&3&0&3&3 \end{smallmatrix}$		
Degré de parcimonie	$\lambda \geq 1$	$\lambda \geq 2$	$\lambda \geq 3$
Matrices binaires de voisins communs	$B_1 = \begin{smallmatrix} 1&0&0&1&0&0\\ 0&1&0&0&0&0\\ 0&0&1&0&1&1\\ 1&0&0&1&0&0\\ 0&0&1&0&1&1\\ 0&0&1&0&1&1 \end{smallmatrix}$	$B_2 = \begin{smallmatrix} 1&0&0&1&0&0\\ 0&0&0&0&0&0\\ 0&0&1&0&1&1\\ 1&0&0&1&0&0\\ 0&0&1&0&1&1\\ 0&0&1&0&1&1 \end{smallmatrix}$	$B_3 = \begin{smallmatrix} 0&0&0&0&0&0\\ 0&0&0&0&0&0\\ 0&0&1&0&1&1\\ 0&0&0&0&0&0\\ 0&0&1&0&1&1\\ 0&0&1&0&1&1 \end{smallmatrix}$
$\sum_{i=1}^{6}\sum_{j=1}^{5}\left\|b^{\lambda_m}_{i,j} - b^{\lambda_m}_{i,j+1}\right\|$ *(calcul par ligne)*	17 *(3+2+3+3+3+3)*	15 *(3+0+3+3+3+3)*	9 *(0+0+3+0+3+3)*
Matrices binaires ordonnées de voisins communs	$B_1^{\text{ord}} = \begin{smallmatrix} 1&1&0&0&0&0\\ 1&1&0&0&0&0\\ 0&0&1&1&1&0\\ 0&0&1&1&1&0\\ 0&0&1&1&1&0\\ 0&0&0&0&0&1 \end{smallmatrix}$	$B_2^{\text{ord}} = \begin{smallmatrix} 1&1&0&0&0&0\\ 1&1&0&0&0&0\\ 0&0&1&1&1&0\\ 0&0&1&1&1&0\\ 0&0&1&1&1&0\\ 0&0&0&0&0&0 \end{smallmatrix}$	$B_3^{\text{ord}} = \begin{smallmatrix} 1&1&1&0&0&0\\ 1&1&1&0&0&0\\ 1&1&1&0&0&0\\ 0&0&0&0&0&0\\ 0&0&0&0&0&0\\ 0&0&0&0&0&0 \end{smallmatrix}$
$\sum_{i=1}^{6}\sum_{j=1}^{5}\left\|\left(b^{\lambda_m}_{i,j}\right)_{\text{ord}} - \left(b^{\lambda_m}_{i,j+1}\right)_{\text{ord}}\right\|$ *(calcul par ligne)*	9 *(1+1+2+2+2+1)*	8 *(1+1+2+2+2+0)*	3 *(1+1+1+0+0+0)*
critère $\mathcal{C}_{\lambda_m}$	$\mathcal{C}_{\lambda\geq 1} = \frac{9}{17} = 1.89$	$\mathcal{C}_{\lambda\geq 2} = \frac{8}{15} = 1.88$	$\mathcal{C}_{\lambda\geq 3} = \frac{3}{9} = 3.03$

TAB. 2 – *Un exemple de calcul du critère* $\mathcal{C}_{\lambda_m}$ *calculé à partir de la matrice de l'exemple introductif (cf. Figure 1 de gauche).*

Ce critère se base sur l'idée que moins il y a d'alternances entre les 0 et les 1 sur les lignes de la matrice considérée plus celle-ci présentera une structure compacte. En effet, dans la table 2, si l'on considère la quantité $\sum_{i=1}^{n}\sum_{j=1}^{n-1}|(b^{\lambda_m}_{i,j})_{\text{ord}} - (b^{\lambda_m}_{i,j+1})_{\text{ord}}|$ représentant le nombre de changements entre les 0 et les 1 d'une matrice ordonnée de degré λ_m et la quantité $\sum_{i=1}^{n}\sum_{j=1}^{n-1}|b^{\lambda_m}_{i,j} - b^{\lambda_m}_{i,j+1}|$ associée à la matrice non rangée de même degré, il est notable que le nombre de changements entre les 0 et les 1 demeure toujours plus petit dans le cas des matrices ordonnées. Cependant, à mesure que le degré de parcimonie augmente, le nombre d'alternances entre les 0 et les 1 diminue : dans l'exemple, le numérateur $\sum_{i=1}^{n}\sum_{j=1}^{n-1}|(b^{\lambda_m}_{i,j})_{\text{ord}} - (b^{\lambda_m}_{i,j+1})_{\text{ord}}|$ est égal à 9 pour un niveau $\lambda = 1$ et à 3 lorsque le degré de parcimonie est de 3. Afin que le critère de sélection ne soit pas biaisé en faveur d'une sparsité infinie, $\mathcal{C}_{\lambda_m}$ est normalisé par le nombre d'alternances entre les 0 et les 1 de la matrice binaire non ordonnée

associée au même degré de parcimonie. Ainsi, d'après l'exemple de la table 2, le niveau de parcimonie retenu est $\lambda \geq 2$. Notons que ce niveau, une structure à deux groupes est sélectionnée et une donnée isolée pouvant être considérée comme une donnée extrême est exclue. Ce critère dérive de la notion de *run* utilisé dans des problèmes de compression de données (Johnson et al. (2004); Apaydin et al. (2008)) ; ce dernier caractérisant une séquence maximale de 1 consécutifs sur une ligne d'une matrice booléenne. Le critère que nous présentons $\mathcal{C}_{\lambda_m}$ est relatif au nombre total de changements présents dans la matrice binaire non ordonnée de même degré de parcimonie afin qu'il ne soit pas biaisé en faveur d'une parcimonie infinie ou inversement, d'une parcimonie trop faible.

3.2 L'algorithme PB-Clus basé sur un critère géométrique

Il existe un grand nombre de critères pour réaliser la sériation d'une matrice (Mechelen et al. (2004)). Nous proposons d'utiliser le produit scalaire comme critère de rangement ; un critère déjà emprunté par McCormick et al. (1972) et par Arabie et Hubert (1990). Dans notre approche cependant, l'utilisation du produit scalaire normé est appliquée uniquement en ligne, du fait de la symétrie de la matrice. Soit $\frac{x_i^T x_j}{||x_i||.||x_j||}$ le produit scalaire normé entre le $i^{\text{ème}}$ et le $j^{\text{ème}}$ vecteur colonne d'une matrice X. Dans le cadre de notre étude, le produit scalaire normé est calculé sur les matrices binaires B^{λ_m} de degrés de parcimonie λ_m ($\forall m \in \{1, \ldots, M\}$) définies dans la Section 3.1. La fonction de permutation Ψ qui cherche à optimiser la somme des scalaires consécutifs peut s'écrire telle que :

$$\Psi^* = \underset{\Psi}{\operatorname{argmax}} \sum_{i=1}^{n-1} \frac{(b^{\lambda_m}_{\Psi(i)})^T b^{\lambda_m}_{\Psi(i+1)}}{||b^{\lambda_m}_{\Psi(i)}||.||b^{\lambda_m}_{\Psi(i+1)}||}. \tag{5}$$

Un tel critère possède des propriétés géométriques intéressantes pour définir la notion de groupe (ou de *cluster*) et pour interpréter sa formation.

Définition 3
Un groupe de points $\mathcal{G}$ se définit comme l'ensemble des observations dont les vecteurs de voisinage commun sont corrélés entre eux.

Ainsi, si les groupes de données sont bien séparés, les vecteurs de voisinage de chacun de ces groupes seront orthogonaux deux à deux et les observations d'un même groupe auront des vecteurs de voisinage commun corrélés. En revanche, si les groupes ne sont pas suffisamment séparés, aucune composante orthogonale ne sera détectée et les groupes ne pourront apparaître distinctement, d'où l'intérêt d'une approche parcimonieuse des données. L'algorithme que nous proposons, *Parcimonious Block Clustering* (PB-Clus), exploite donc cette définition à travers une approche de type *branch and bound* où une recherche exhaustive du meilleur candidat potentiel dans chaque sous-ensemble de vecteurs sera faite. Initialement, une observation i est choisie sur l'ensemble des données. A partir du vecteur de voisinage commun associé à i, l'algorithme recherche d'abord une liste de vecteurs indépendants entre eux, puis leurs composantes connexes. Chaque sous-ensemble de la liste est réordonné de sorte que la somme des produits scalaires adjacents soit maximale par bloc. La forme générale de l'algorithme PB-Clus est présentée ci-dessous et une implémentation plus détaillée est proposée dans la table 3 :

1. Une matrice de produits scalaires est calculée à partir de B_{λ_m}.
2. Une observation est sélectionnée et la composante connexe est recherchée i.e. l'ensemble des vecteurs corrélés. Puis un vecteur orthogonal à ces premiers éléments est sélectionné et ses vecteurs corrélés sont récupérés à leur tour. Cette étape est réitérée jusqu'à ce qu'il n'y ait plus de vecteurs à classer ou que les vecteurs restants soient nuls.
3. Dans chaque sous-groupe, les vecteurs sont rangés selon leur corrélation par rapport au dernier élément rangé.
4. L'ordre obtenu dans chaque sous-matrice est appliqué à la matrice initiale B_λ.

PB-Clus est appliqué pour différentes valeurs de λ et la "matrice centrale" est sélectionnée parmi la famille de matrices parcimonieuses ordonnées selon par le critère $\mathcal{C}_{\lambda_m}$ défini dans la section 3.1. Cependant, une trop grande sparsité peut impliquer l'évincement de petits clusters initialement présents dans la matrice des données. Comme nous disposons de plusieurs matrices ordonnées avec différents seuils de parcimonie, une combinaison de ces derniers permettrait de visualiser sur une même matrice la structure globale des données. Par ailleurs, l'algorithme de sériation PB-Clus traite les données, dans sa globalité, à travers une approche *branch and bound*. Cependant, localement, le rangement des vecteurs dans chacun des sous-ensembles suit une procédure de type *forward*, ce qui implique que la place des vecteurs déjà ordonnés reste inchangée pour une permutation courante. Aussi, l'algorithme propose une solution approchée pour le réagencement d'une matrice et ne prétend pas être optimal comparativement aux autres approches suggérées dans la littérature. Enfin, l'algorithme PB-Clus a un coût de calcul plus important que les autres méthodes de sériation dès lors que le rangement est effectué non pas sur une seule matrice mais sur M matrices relatives à des degrés de parcimonie différents. Dans le cas d'une matrice de taille $n \times n$ ayant K groupes de même taille $\frac{n}{K}$, il y a au plus, $K(\frac{n}{K}!)$ calculs. Cependant à mesure que le degré de parcimonie augmente, la matrice se remplit de colonnes (lignes) de 0 ce qui diminue le nombre d'éléments à ranger, et par conséquent le temps de calcul. Aussi, le coût calculatoire resterait bien inférieur à $M.K(\frac{n}{K}!)$.

4 Expériences sur données simulées

Nous proposons dans cette section d'évaluer notre méthode sur des simulations afin de souligner ses principales caractéristiques face à des données bruitées ou à des groupes superposés. Pour ce faire, nous allons considérer dans une première expérience le cas de groupes de données superposés et déséquilibrés afin de caractériser le comportement de la famille de matrices parcimonieuses selon le degré de superposition des clusters ; dans une seconde expérience, nous analyserons l'influence des données bruitées sur la visualisation des sous-ensembles structurels d'un jeu de données. L'ensemble des simulations évaluant le comportement de PB-Clus en face de données fortement bruitées et aléatoirement chevauchées se sont inspirées des travaux de Prelic et al. (2006). Dans chacune des expériences proposées, la famille de matrices binaires de voisinage commun est construite par seuillage de la matrice des distances calculée pour toutes les paires d'observations, le seuil étant fixé au premier quartile de la distribution de ces distances.

Algorithme PB-Clus

→Initialisation de l'algorithme PB-Clus:

seq = liste des niveaux de voisins communs désirée

$listeC = NA$: intialisation de la liste contenant les critères des matrices binaires ordonnées.

→Boucle principale de PB-Clus:

Pour i=1 à $taille\,(\text{seq})$\{

calculer $B_\lambda$la matrice binaire de voisins communs de niveau $\lambda = \text{seq}\,(i)$

calculer $S_\lambda$la matrice de produits scalaires de B_λ

$O_\lambda = ordre\,(S_\lambda)$, calcule le rangement des individus

$\mathcal{C}_\lambda = crit\,(B_\lambda, O_\lambda)$, calcule le critère de la définition 2 de la section 3.1

$listeC = merge\,(listeC, \mathcal{C}_\lambda)$\}

→Résultat de PB-Clus:

$B^*_\lambda = \min\,(listeC)$, retourne le critère minimale et le nombre de voisins communs associés.

$O_\lambda = ordre\,(S^*_\lambda)$, établit le rangement des individus pour le niveau optimal de voisins communs.

Fonction $ordre\,(S_\lambda)$ **:**

initialisation:

V = $colinear\,(\text{i.obs}, S_\lambda)$

liste = liste des individus où le nombre de voisins communs est non nul ($vc \neq 0$).

i.max = l'observation ayant vc maximum.

i.perm= 0 (liste qui donnera l'ordre des individus selon leur scalaire ordonné.)

boucle principale:

tant que ($taille\,(\text{liste}) > 1$) {

si ($taille\,(\text{V.cor}) \geq 1$) { i.perm =$merge\,(\text{i.perm,liste[V.col]})$

$Sj = Sj\,[\text{c(V.cor,V.ind),c(V.cor,V.ind)}]$

liste = liste[c(V.cor,V.ind)]

si ($taille\,(\text{V.cor}) > 1$) { V =$colinear\,(1, B_\lambda)$} }

else { i.perm =$merge\,(\text{i.perm,liste[V.col]})$

$Sj = Sj\,[\text{V.ind,V.ind}]$

liste = liste[V.ind]

si ($taille\,(\text{V.cor}) > 1$) {

i.max =observation ayant vc maximum dans S_j

$B_\lambda = B_\lambda\,[\text{V.ind,V.ind}]$

V =$colinear\,(\text{i.max}, B_\lambda)$}}

Fonction $colinear\,(i, S)$ retourne 3 listes différentes:

Soit s_i le vecteur de la ligne i de S, alors:

- **V.col** = liste des individus colinéaires à s_i.
- **V.cor** = liste des individus corrélés mais non colinéaires à s_i.
- **V.ind** = liste des individus indépendants à s_i.

TAB. 3 – *Implémentation de l'algorithme PB-Clus.*

4.1 Sériation dans le cas de groupes superposés

Sériation de groupes superposés et déséquilibrés : Dans cette expérience, les données sont simulées à partir d'un mélange de trois lois gaussiennes de dimensions 2 dans lequel deux groupes se superposent. Ces derniers sont de proportions différentes : le premier cluster est formé de 5% des données, soit 15 observations tandis que les deux autres représentent 32% et 63% des données, soit respectivement 100 et 200 observations. Cette situation est illustrée par la figure 2(a) dans laquelle les données sont représentées dans l'espace des variables. La partition centrale obtenue par notre méthode de sériation est illustrée par la Figure 2(b) et ce, avec un seuil de sparsité de 16 voisins communs. Pour ce niveau de parcimonie, plus de 6% des données ont été évincées ce qui a pour conséquence l'absence du troisième cluster, le plus petit, qui de part sa taille (15 observations) au regard du seuil de parcimonie (16 voisins communs) a été automatiquement évincé des sous-ensembles de données qui forment la matrice centrale. Afin d'intégrer la visualisation de ce cluster dans la matrice centrale, un seuil plus faible de sparsité (8 voisins communs) a été appliqué sur le sous-ensemble des données évincées dans le but d'en faire émerger une sous-structure. La combinaison de la visualisation centrale obtenue pour un niveau de parcimonie de 16 voisins communs avec celle du sous-ensemble de données évalué pour un seuil de 8 voisins communs, nous permet d'afficher sur une même matrice les 3 groupes de données. La figure 2(c) rend compte de cette approche multi-échelle de l'algorithme PB-Clus. Enfin, parmi les données sériées, i.e. celles non évincées, 98% d'entre elles ont été bien classées.

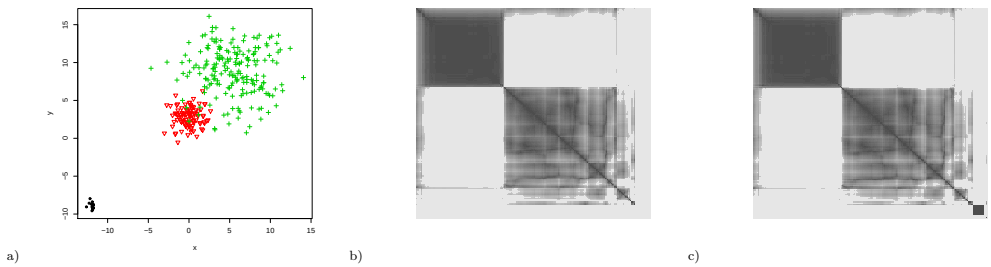


FIG. 2 – *Projection des données dans leur espace (à gauche), visualisations de la matrice centrale ordonnée par PB-Clus avec un seuil de sparsité de 16 voisins communs (milieu) et de la matrice ordonnée multi-échelle (à droite).*

Influence du niveau de superposition de clusters : Dans cette deuxième expérience, nous cherchons à évaluer l'influence du niveau de recouvrement de clusters dans la recherche d'une structuration des données. Pour ce faire, nous avons simulé 3 gaussiennes dans un espace de dimension 2 telles que leurs moyennes respectives vérifient : $m_1 = (x, y), m_2 = (x, -y), m_3 = (0, -y)$ avec $x \in [0, 0.3]$ et $y \in [0, 0.225]$. Par conséquent, la position relative des moyennes varie et cette variation détermine le niveau de superposition des groupes. Ainsi, lorsque $x = 0$ et $y = 0$, les 3 groupes sont confondus et cela correspond à un taux de recouvrement de 100%.

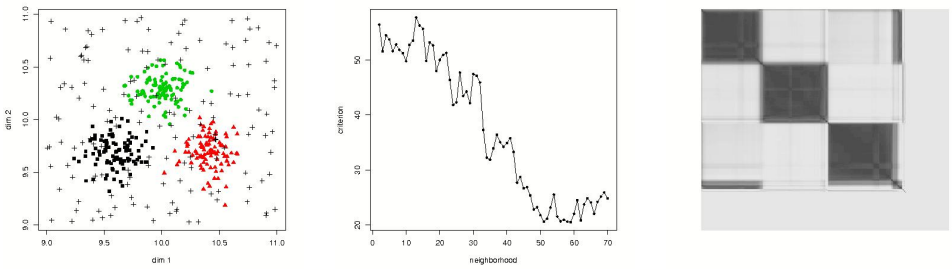
Dans le cas opposé des groupes bien séparés où le taux de recouvrement est égale à 0, les moyennes des clusters vérifient : $m_1 = (0.3, 0.225), m_2 = (0.3, -0.225), m_3 = (0, -0.225)$. La table 4 présente l'évolution du niveau des voisins communs et son taux d'éviction associé, en fonction du recouvrement des groupes. Tout d'abord, on remarque que plus les clusters se superposent, plus le critère $\mathcal{C}_\lambda$ sélectionne une représentation parcimonieuse des données. En effet, lorsque la structure visible des données devient moins prononcée, cet effet est contre-balancé par une plus importante sparsité dans les données avec l'imposition d'un voisinage commun plus élevé. De même, à mesure que la structure des données devient de plus en plus complexe, le taux de classification relatif aux sous-ensembles de données sériées diminue ainsi que la qualité de la visualisation. Dans notre exemple, au-delà d'un taux de recouvrement des données de 40%, le taux de classification devient faible (< 0.6) puisque l'algorithme PB-Clus ne détecte plus de structure dans les données et ceci, quel que soit le niveau de parcimonie imposé.

taux (en %) de recouvrement	0	6.7	13.3	20.0	26.6	33.3	40.0	46.6	53.30	73.3	100
x	0.30	0.28	0.26	0.24	0.22	0.20	0.18	0.16	0.14	0.08	0
y	0.225	0.21	0.195	0.18	0.165	0.15	0.135	0.120	0.09	0.06	0
degré de parcimonie	5	6	9	35	33	34	35	35	35	35	34
pourcentage de données évincées	0.00	0.00	0.00	0.26	0.23	0.34	0.37	0.35	0.39	0.35	0.43
valeur de $\mathcal{C}_\lambda$	1.95	2.01	2.42	2.64	2.90	2.82	3.34	3.29	3.32	3.54	3.65
taux de classification	0.99	0.99	0.99	0.99	0.95	0.90	0.86	0.60	0.49	0.44	0.39

TAB. 4 – *Influence du degré de recouvrement des clusters sur la détection de structure.*

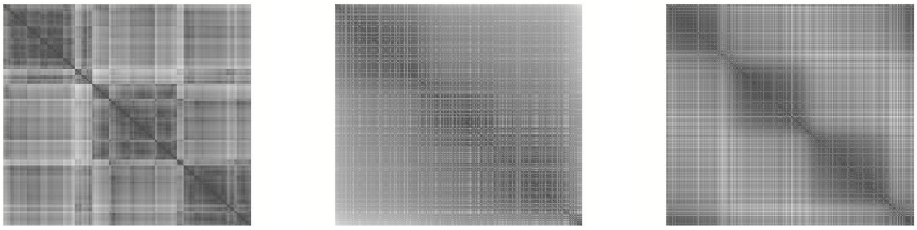
4.2 Données bruitées

Comparaison entre des méthodes de sériation : Dans cette expérience, 30% des données ont été remplacées par du bruit uniforme sur un hypercube $[-1, 1]$ de dimension 4 et l'autre moitié a été simulée à partir d'un mélange de trois gaussiennes de dimension 4 ; elles sont représentées sur leurs deux premières dimensions par la figure 3(a). La figure 3(b) présente l'évolution du critère de compacité $\mathcal{S}_\lambda$ en fonction des différents degrés de parcimonie, à savoir le nombre de voisins communs. La partition centrale (fig. 3(c)) sélectionnée est celle qui a le critère $\mathcal{C}_\lambda$ minimal, ce qui correspond à un voisinage commun de 59. Cette sparsité a pour conséquence une éviction de 16% des données et ce sont 84% des données initiales qui permettent d'obtenir une représentation diagonale par blocs ; les sous-ensembles de données exclues sont entièrement formés de données bruitées. De plus, le taux de classification correcte parmi les données sériées s'élève à 99%, ce qui implique que ces sous-ensembles de données sériées sont structurels des 3 clusters. Afin d'évaluer la performance de notre approche, trois méthodes de sériation basées sur des matrices de distances ont été appliquées : le *clustering* hiérarchique pour la sériation (fig. 4(a)), l'approche de Chen basée sur une structure anti-Robinson (Chen (2002), fig. 4(b)) ou encore une autre méthode de sériation anti-Robinson



(a) (b) (c)

FIG. 3 – *Visualisations du mélange de gaussiennes bruitées dans l'espace de données (à gauche), de l'évolution du critère $\mathcal{C}_\lambda$ selon le degré de voisins communs (milieu) et de la matrice centrale ordonnée par PB-Clus avec un seuil de sparsité de 67 voisins communs.*



(a) (b) (c)

FIG. 4 – *Visualisations de la matrice des distances sériée par une méthode de clustering hiérarchique (gauche), une méthode de Chen (milieu) et une méthode basée sur une représentation anti-Robinson (droite).*

avec une approche par *simulated annealing* (Brusco et al. (2008)) illustrée par la figure 4(c). Ces méthodes sont détaillées dans Hahsler et al. (2009) et programmées dans le package $\mathcal{R}$ *seriation*. Parmi les méthodes de sériation utilisées, nous remarquons que seule la partition centrale obtenue par l'algorithme PB-Clus apporte une visualisation claire des trois clusters. La représentation de cette structure en trois groupes distincts est possible grâce à la famille de matrices binaires parcimonieuses. En effet, plus les matrices ont un degré de parcimonie élevé, plus la quantité de données bruitées pris en compte diminue.

Influence du niveau de bruit : Cette deuxième expérience tend à montrer le comportement de l'algorithme PB-Clus en face de données fortement bruitées. A cet effet, nous avons simulé trois gaussiennes de dimension 2 de 50 observations chacune et ayant pour moyenne

niveau de bruit	0	0.2	0.4	0.6	0.8	1.0	1.2	1.4	1.6	1.8	2.0
degré de parcimonie	4	12	6	8	12	18	24	21	35	31	35
pourcentage de données évincées	0.06	0.16	0.12	0.23	0.11	0.04	0.12	0.19	0.16	0.11	0.08
Taux de classification	1	1	1	1	0.99	0.99	0.99	0.99	0.99	0.98	0.98

TAB. 5 – *Comportement de PB-Clus selon différents niveaux de bruit ajoutés aux données initiales.*

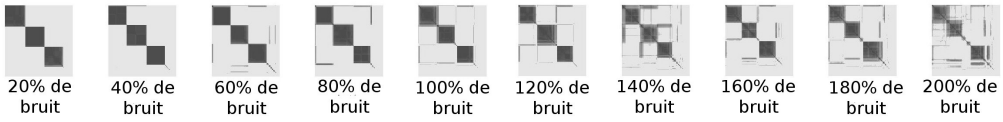


FIG. 5 – *Evolution de la visualisation de la structure des données en fonction de différents niveaux de bruit ajoutés aux données initiales.*

$m_1 = (-0.4, -0.3)$, $m_2 = (0.4, -0.3)$ et $m_3 = (0, 0.3)$ respectivement et pour matrice de variances-covariances $S = diag(0.1, 0.1)$. Ces groupes sont volontairement bien séparés afin de pouvoir évaluer la sensibilité de l'algorithme au bruit. Les données bruitées ont été générées selon une loi uniforme sur un cube $[-1, 1]$ de dimension 2. Pour évaluer l'impact du bruit sur la visualisation, nous avons fait varier le nombre de données bruitées ajoutées de 10% à 200% du nombre des données de l'échantillon initial. La figure 5 présente l'évolution de la visualisation des données à mesure de l'ajout de bruit tandis que la table 5 représente les niveaux de parcimonie et le taux de données evincées associés au critère $\mathcal{C}_\lambda$ pour chaque niveau de bruit. On remarque tout d'abord que la visualisation de groupes se dégrade peu avec l'ajout de bruit. En effet, sur la figure 5, la structure ne se dégrade qu'à partir du moment où les données bruitées représentent plus de la moitié de l'ensemble des données. La qualité de ces représentations s'explique par le seuil de parcimonie qui, augmentant avec l'ajout de bruit (cf. table 5), exclut un plus grand nombre de données. L'objectif de PB-Clus, basé sur la sélection de sous-ensembles de données structurelles est bien atteint dans cette expérience puisque la visualisation met en exergue 3 groupes et ce, quel que soit le niveau de bruit.

5 Comparaison avec des méthodes classiques

Cette dernière section est destinée à évaluer la qualité de la visualisation d'une structure dans les données ainsi que celle de la partition obtenue par l'algorithme PB-Clus. Pour cela, nous avons emprunté deux types de jeux de données appartenant à des communautés différentes : d'une part, 5 bases de données réelles et de références pour la sériation accessibles

par le logiciel $\mathcal{R}$ et d'autre part, 7 jeux de données simulées, les données FCPS, présentant des difficultés particulières pour la tâche de clustering. Les familles de matrices binaires de voisinage commun ont été obtenues de la même manière que dans la Section 4.

5.1 Données de référence pour la sériation

Dans cette section, nous comparons les performances de l'algorithme PB-Clus en termes de visualisation d'une part avec deux autres méthodes de sériation, l'une utilisant la classification hiérarchique (HC), l'autre utilisant un critère de divergence reposant sur une structure anti-Robinson (CHEN) (Chen (2002)) décrites dans Hahsler et al. (2009) et en termes de classification d'autre part, par l'intermédiaire d'une méthode de classification non supervisée basée sur les distances, le k-means. Cette comparaison a été faite sur 5 bases de données populaires dans la communauté statistique de *clustering* et de sériation qui sont disponibles dans le logiciel $\mathcal{R}$. Les 5 jeux de données que nous avons choisis sont décrits ci-dessous :

- La base de données des *iris de Fisher* a été construite par Anderson (1935) qui a collecté 3 espèces d'iris différentes dans la péninsule de Gaspé : les *setosa*, les *virginica* ainsi que les *versicolor*. Chaque espèce est représentée par 50 fleurs qui sont décrites par 4 caractéristiques morphométriques basées sur la largeur et la longueur de leurs sépales et de leurs pétales. Cette base de données est extrêmement populaire dans la communauté statistique du fait de la difficulté à distinguer les *virginica* et les *versicolor*.

- Les données *ruspini* proviennent des travaux de Ruspini (1970) sur le clustering : elles sont formées de 75 points décrits sur 2 dimensions et répartis en 4 classes homogènes et équilibrées.

- Les données *townships* sont des données binaires relatant la présence ou l'absence de 9 caractéristiques descriptives de 16 villes, telles la présence ou l'absence d'universités, de coopératives agricoles, de chemins de fer etc. Il n'y a aucune information sur le nombre de groupes structurant les données.

- Les données *faithful* sont une base de données évaluant le temps entre deux éruptions de geysers du parc national de Yellowstone de Wyoming (USA) et leur durée. Elles sont caractérisées par 272 observations (Haerdle (1991)).

- Les données *geysers* représentent une version complète des données précédentes et a été collectée par Azzalini et Bowman (1990). Ce sont quelques 299 éruptions qui ont été étudiées (mêmes types de mesures que précédemment) entre le 1er et le 15 août 1985.

Pour évaluer la qualité de la visualisation de notre approche, nous avons utilisé deux critères de Niermann (Neumann et Moore) présentés dans la Section 2 ; la partition obtenue sera quand à elle évaluée par validation croisé avec le vrai label lorsque les données en sont dotées ou dans le cas échéant, avec les labels estimés par les k-means. Comme ce dernier suppose une connaissance *a priori* du nombre de groupes du mélange, nous avons choisi d'utiliser le nombre de clusters détectés par PB-Clus afin d'obtenir des partitions comparables. La figure 6 de droite

représente les produits scalaires consécutifs des éléments i et $i+1$ ordonnés sur les 5 bases de données exposées précédemment. Ces courbes de produits scalaires consécutifs rendent compte de la proximité entre deux observations adjacentes et de points de rupture pour le passage d'un cluster à un autre, ce qui permet de sélectionner le nombre de clusters du mélange et d'obtenir une partition des données. La colonne de gauche de la figure 6 représente la visualisation centrale de la matrice parcimonieuse ordonnée avec l'algorithme PB-Clus. Dans le cas des *iris de Fisher*, l'observation de sa matrice centrale de degré de voisinage commun 8, présente une structure globale à deux *clusters* dans laquelle il préfigure localement, au niveau du premier bloc, une séparation en deux parties différentes. On retrouve ici la structure particulière des iris dans laquelle les *versicolor* et les *virginica* sont des espèces peu distinctes. Par ailleurs, cette partition en 3 groupes se voit être confirmée par les 2 points de rupture présents sur la courbes de ses produits scalaires consécutifs. L'analyse simple de ces deux graphiques rend compte de la performance de notre approche parcimonieuse pour la visualisation des données, d'autant plus que rares sont les méthodes de clustering qui sélectionnent un modèle optimal à 3 classes d'iris (cf. les modèles de mélanges Raftery et Dean (2006)). Dans le cas des données *Ruspini* et *Faithful*, les ruptures sur la courbe des produits scalaires consécutifs sont nettes et amples ce qui rend compte de la déconnexion totale des clusters entre eux ; cette même conclusion est visible sur leur matrice centrale ordonnée de degré 5 pour les données *Ruspini* et de 2 pour celles de *Faithful*. A l'inverse, les données *Geysers* et *Townships* présentent des ruptures de faibles ampleurs. On peut en effet observer sur les données *Geysers* deux petites ruptures sur la courbe de ses produits scalaires consécutifs, lesquelles s'expliquent par la proximité des clusters les uns par rapport aux autres. Ensuite, dans le cas des données *Townships*, la courbe des scalaires consécutifs nous montre que la première ville est, certes, reliée aux 7 villes suivantes mais moins fortement que ces 7 villes entre elles ce qui suppose un certain éloignement de cette donnée au reste du groupe. La visualisation centrale de la matrice parcimonieuse ordonnée de degré 2 avec l'algorithme PB-Clus apporte une meilleure compréhension des relations entre les villes. En effet, on remarque que la première donnée est fortement corrélée à deux blocs de villes distincts. Cette interprétation se voit être confirmée par les travaux de Hahsler et al. (2009) qui, par une approche de sériation sur le tableau initial rectangulaire des données, ont montré l'existence d'une structure à 3 groupes : les villes urbaines, les villes de campagnes et les villes intermédiaires avec une ville en transition. Cette première évaluation basée sur notre perception visuelle est complétée par une mesure de qualité basée sur des critères de sériation évaluant la proximité du voisinage dans la matrice ordonnée. La table 6 évalue les performances de 3 méthodes de sériation, dont PB-Clus au regard des deux mesures de Niermann (2005) (Moore et Neumann), la meilleure méthode étant celle dont le critère est minimum. On remarque que les 2 critères de Niermann sont minimum pour une approche parcimonieuse des données et ceci sur l'ensemble des bases de données ce qui rend compte de la performance de PB-Clus relativement à HC ou à la méthode de Chen. Enfin, la dernière table 7 présente les tableaux de classification croisée avec le vrai label dans le cas des *iris de Fisher* et avec les labels obtenus par k-means dans le cas des données *Ruspini, Townships, Geysers* et *Faithful*. Notons que dans le cas des données iris et *Geysers* c'est-à-dire ceux où les ruptures sur la courbe des scalaires consécutifs sont de faibles ampleurs, nous avons seuillé les scalaires afin d'obtenir une labellisation pour chaque donnée. Sur les *iris de Fisher*, notre algorithme présente un taux de classification correcte de 89.0% légèrement plus faible que est comparable à celui obtenu par les k-means (90.6%). Cette différence de taux est lié aux données situées à l'intersection

des *virginica* et des *versicolor* et à l'initialisation de notre algorithme. Pour les autres jeux de données, on observe que les partitions obtenues par PB-Clus et par les k-means concordent presque parfaitement, les taux de classification avoisinant les 98%.

Méthodes		PB-Clus		HC		Chen	
	Critères	*Moore*	*Neumann*	*Moore*	*Neumann*	*Moore*	*Neumann*
Données							
	Iris	**1 371.2**	**471.1**	31 728.8	10 893.1	19 357.8	7 304.0
	Townships	**244.5**	**91.8**	1 109.9	441.5	849.0	342.0
	Ruspini	**1 290.1**	**442.2**	8 724.9	3 036.4	6 503.7	2 277.1
	Faithful	**2 634.1**	**889.4**	34 045.5	11 503.5	23 390.0	9 894.2
	Geysers	**2 514.9**	**850.4**	68 205.3	2 302.1	12 866.8	4 501.4

TAB. 6 – *Comparaison de* 3 *méthodes de sériation, PB-Clus, HC et Chen selon les mesures de Moore et de Neumann sur les données de référence.*

IRIS DE FISHER								TOWNSHIPS				
	clusters PB-Clus				*clusters Kmeans*					*clusters PB-Clus*		
classes connues	1	2	3	*classes connues*	1	2	3	*classes connues*	1	2	3	4
Setosa	50	0	0	Setosa	50	0	0	Urbaines	8	0	0	0
Versicolor	0	50	0	Versicolor	0	49	1	Intermédiaires	0	4	0	0
Virginica	0	17	33	Virginica	0	13	37	Campagnardes	0	0	2	0
								Inclassables	0	1	0	1
Taux de classification = 0.88				*Taux de classification = 0.90*				*Taux classification = 0.94*				

RUSPINI					FAITHFUL			GEYSERS			
	clusters PB-Clus					*clusters PB-Clus*			*clusters PB-Clus*		
clusters Kmeans	1	2	3	4	*clusters Kmeans*	1	2	*clusters Kmeans*	1	2	3
groupe 1	13	0	0	0	groupe 1	168	4	groupe 1	88	2	7
groupe 2	0	35	0	0	groupe 2	0	100	groupe 2	0	105	0
groupe 3	0	0	15	0				groupe 3	0	0	97
groupe 4	0	0	0	20							
Taux de classification = 1.00					*Taux de classification = 0.98*			*Taux de classification = 0.97*			

TAB. 7 – *Tableaux de validation croisée des données benchmarks de sériation.*

5.2 Données de référence pour le clustering

Dans cette dernière expérience, nous considérons les jeux de données FCPS (Ultsch (2005)) disponibles sur le site *http ://www.informatik.uni-marburg.de/databionics* et qui sont caractérisés par des données de petite dimension ($p \leq 3$) divisées en plusieurs clusters. Chacune de ces bases présente un problème particulier pour le clustering : certains groupes d'un même mélange ne suivent pas la même distribution telles que les données *Lsun*, *Atom*, *TwoDiamonds* ou encore *WingNut* ; d'autres sont caractérisés par des variances intra différentes comme les

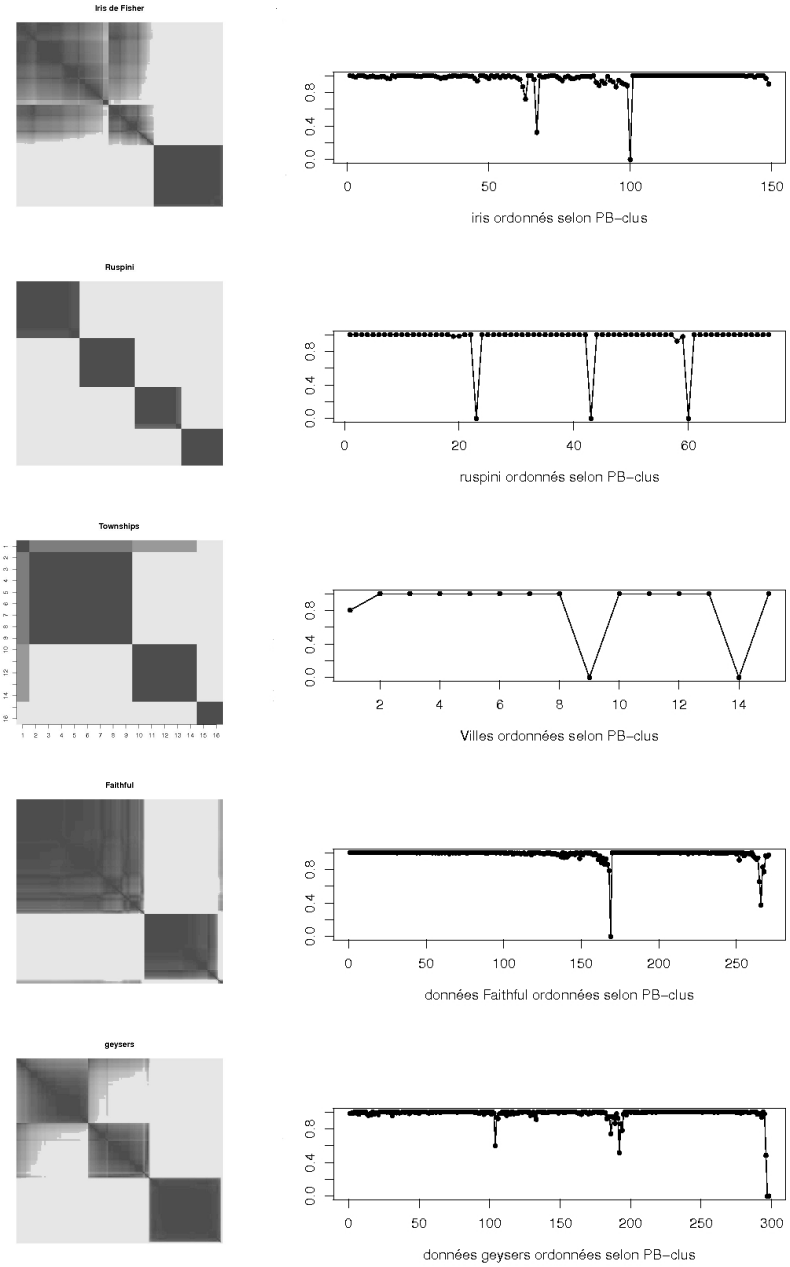


FIG. 6 – *Matrices de scalaires ordonnés (gauche) et représentation de scalaires consécutifs ordonnés sur des données de référence (droite).*

données *WingNut* ou *Lsun* et d'autres encore ne sont pas linéairement séparables comme les données *Atom*. Toutes ces spécificités sont détaillées dans la table 8. L'objectif de cette expérience est d'évaluer la capacité de notre approche à trouver des sous-ensembles structurels caractérisant chaque base de données et ce, quelles que soient la forme ou la taille des clusters. L'algorithme PB-Clus est appliqué sur l'ensemble de ces données et la première partie de la table 9 présente les degrés de parcimonie sélectionnés par le critère $\mathcal{C}_\lambda$, le taux d'éviction des données associé, le nombre de groupes détectés ainsi que la visualisation de la matrice centrale ordonnée. Notons que, comme précédemment, la partition obtenue ainsi que le nombre de groupes présents dans chaque jeu de données ont été obtenus par seuillage des scalaires consécutifs. On remarque tout d'abord, que la visualisation de la structure intrinsèque de l'ensemble des données est relativement nette. Cette qualité de la représentation des structures s'explique principalement par un niveau de parcimonie relativement élevé comme le montre les niveaux de voisinage commun sélectionnés de la table 9 mais aussi, par une analyse multi-échelle dans le cas des données *Atom* et *Target* marquées d'un astérisque. Dans le cas des données *Atom*, l'algorithme PB-Clus avait sélectionné initialement une représentation parcimonieuse excluant 49% des données (soit le cluster de grande variance) avec un voisinage commun de 58. Pour les données *Target*, 48% des données avaient été évincées pour un seuil de voisinage commun s'élevant à 34, ce qui avait pour conséquence d'omettre le cluster en anneau ainsi que les tous petits clusters. L'approche multi-échelle a donc permis d'une part, de détecter des sous-groupes dans les données évincées (le niveau de parcimonie et le taux d'éviction ont été renseignés entre des parenthèses dans la table 9) et d'autre part, de les visualiser. Dans le cas des données *Atom*, on distingue bien deux groupes : l'un compact et carré, relatif au cluster dense et central, et l'autre beaucoup plus effilé, caractérisant le second cluster de variance beaucoup plus grande. De la même manière, l'analyse multi-échelle appliquée sur les données *Target*, permettent d'introduire le cluster en forme d'anneau dans la visualisation ainsi que les 4 petits clusters. Quelle que soit la base de données considérée, la structure intrinsèque a bien été détectée par l'algorithme PB-Clus et ceci grâce à l'introduction de parcimonie dans les données. La deuxième partie de la table 9 présente les taux de classification correcte sur les données sériées (non évincées) issues de notre approche et ceux obtenus par Ultsch (2005) à partir de 3 méthodes non supervisées : les k-means et deux algorithmes de classification hiérarchiques (SingleLinkage et Ward). Certes, les taux de classification sont comparables entre PB-Clus et les méthodes proposées, cependant, ces résultats ont été obtenus en apportant une connaissance *a priori* du nombre de groupes pour chaque jeu de données ce qui facilite le partitionnement des différents jeux de données pour les algorithmes des k-means ou de classification hiérarchiques. De plus, dans notre approche, ce n'est pas tant la comparaison des taux de bonne classification entre les méthodes qui est intéressante mais plutôt le fait que d'une part, la structure intrinsèque des données est toujours bien évaluée quelle que soit leur difficulté et d'autre part, sur les 7 jeux de données proposés plus de 93% des données structurelles sont bien classées. L'ensemble de ces remarques rend compte de la performance de PB-Clus quant à la mise en exergue d'une structuration et de données structurelles et ce, quelle que soit la distribution des données, la taille des clusters ou leurs variances.

	n	K	p	Difficultés
Atom	800	2	3	différentes variances non linéairement séparables
Hepta	211	7	2	aucune difficulté
Lsun	400	3	2	différentes variances différentes distances inter
Target	770	6	2	outliers
Tetra	400	4	3	clusters très proches
TwoDiamands	800	2	2	frontière du cluster définie par une densité
WingNut	1016	2	2	densité versus distances

TAB. 8 – *Nombre d'observations (n), de clusters (K) et de dimensions (p) des bases de données FCPS.*

Données :	Atom	Hepta	Lsun	Target	Tetra	TwoDiam	WingNut
PB-Clus							
Voisins Communs	34* (2)	10	68	58* (2)	53	32	110
Taux d'éviction	0.49* (0)	0	0.17	0.48* (0)	0	0	0.087
Nombre de groupes	2	7	3	6	4	2	2
Visualisation							
Taux de classification :							
PB-Clus	0.99	1	0.93	0.96	0.98	0.99	0.94
k-means	0.71	0.97	0.76	0.63	0.99	1.00	0.96
Ward	1.00	0.97	0.99	0.99	0.26	0.50	1.00
Single Linkage	0.66	0.97	0.72	0.65	0.98	1.00	0.88

TAB. 9 – *Tableau des résultats de la sériation sur les bases de données FCPS.*

6 Conclusion

L'algorithme PB-Clus que nous proposons est une méthode de sériation qui permet de visualiser des sous-ensembles de données et leurs dépendances dans une base de données. Pour ce faire, nous avons proposé une famille de matrices parcimonieuses formées de matrices de niveaux de parcimonie différents dont le niveau est directement déterminé par le nombre de voisins communs entre paires d'observations. Cette méthode est un outil efficace pour l'analyse de données qui offre visuellement de meilleurs résultats que les méthodes traditionnelles de sériation, en particulier lorsque les données sont bruitées ou que les groupes sont superposés. De plus, cette approche parcimonieuse facilite l'interprétation des données et offre une qualité de partitionnement comparable aux k-means avec l'avantage de ne poser aucune hypothèse sur le nombre de clusters. Par ailleurs, choisir un niveau de parcimonie dans les données revient à chercher des sous-ensembles de données explicatifs d'une structure qui leur est propre. Ce nouveau point de vue peut s'apparenter à une approche par niveaux de densité communément appelé *levels set* qui a été initialement abordée par Hartigan (1987) puis repris par Nolan (1991). Une comparaison de ces deux approches et la recherche d'un lien théorique font partis de nos travaux de recherche en cours.

Remerciements

Nous tenons à remercier le Comité de Lecture du numéro spécial RNTI "Fouille de Données Complexes : données multiples" qui a contribué à l'amélioration de la qualité de l'article présenté.

Références

Anderson, E. (1935). The irises of the gaspe peninsula. *Bulletin of the American Iris Society 59*, 2–5.

Apaydin, T., A. Tosun, et H. Ferhatosmanoglu (2008). Analysis of basic data reordering techniques. *SSDBM*, 517–524.

Applegate, D., R. Bixby, V. Chvàtal, et W. Cook (2007). *The Traveling Salesman Problem*. Princeton University Press.

Arabie, P. et L. Hubert (1990). The bond energy algorithm revisited. *IEEE transactions on systems, man, and cybernetics 20*(1), 268–274.

Arabie, P., L. J. Hubert, et G. D. Soete (1996). *Clustering and Classification*, Chapter An overview of combinatorial data analysis, pp. 5–63. World Scientific.

Azzalini, A. et A. Bowman (1990). A look at some data on the old faithful geyser. *Applied Statistics 39*, 357–365.

Batagelj, V. (1997). Notes on blockmodeling. *Social Networks 7*, 143–155.

Brusco, M., H. F. Kohn, et S. Stahl (2008). Heuristic implementation of dynamic programming for matrix permutation problems in combinatorial data analysis. *Psychometrika*.

Brusco, M. et S. Stahl (2005). *Branch and Bound applications in combinatorial data analysis*. Springer.

Brusco, M. et D. Steinley (2006). Inducing a blockmodel structure of two-mode binary data using seriation procedures. *Journal of Mathematical Psychology 50*, 468–477.

Caraux, G. et S. Pinloche (2005). Permutmatrix : a graphical environment to arrange gene expression profiles in optimal linear order. *Bioinformatics 21*(7).

Carroll, D. et P. Arabie (1980). Multidimensional scaling. *Annu Rev Psychol 31*, 607–649.

Chen, C. (2002). Generalized association plots : Information visualization via iteratively generated correlation matrices. *Statistica Sinica 12*, 7–29.

Climer, S. et W. Zhang (2006). Rearrangement clustering : Pitfalls, remedies and applications. *Journal of Machine Learning Research 7*, 919–943.

Croes, G. (1958). A method for solving the traveling-salesman problems. *Operations Research 6*, 791–812.

Dhillon, I., S. Mallela, et D. Modha (2003). Information theoric co-clustering. *Ninth ACM SIGKDD, Int. Conf. Knowledge Discovery and Data Mining KDD'03*, 89–98.

Doreian, P., V. Batagelj, et A. Ferligoj (2004). Generalized blockmodeling of two-mode network data. *Social Networks 26*, 29–53.

Eastman, W. (1958). Linear programming with pattern constraints. Master's thesis, Ph.D. Dissertation, Harvard University, Cambridge, Mass.

Govaert, G. et N. Nadif (2008). Algorithms for model-based block gaussian clustering. *The 4th international Conference on datamining*, 536–272.

Govaert, G. et N. Nadif (2010). Latent block model for contingency table. *Communication in Statistics : Theory and Methods 39*, 416–425.

Gutin, G. et A. Punnen (2002). *The traveling salesman problem and its variations*, Volume 12. Springer.

Haerdle, W. (1991). *Smoothing Techniques with Implementation in S.* Springer, New York.

Hahsler, M., K. Hornik, et C. Buchta (2009). Getting things in order : an introduction to the r package seriation. Technical Report 58, R.

Hartigan, J. (1987). Estimation of a convex density contour in two dimensions. *Journal of the American Statistical Association 82*(397), 267–270.

Hubert, L., P. Arabie, et J. Meulman (2001). Combinatorial data analysis : Optimization by dynamic programming. *Society ofr industrial and Applied Mathematics*.

I.Liiv (2010). Seriation and matrix reordering methods : an historical overview. *Wiley interscience 3*(2), 70–91.

Johnson, D., S. Krishnan, et J. Chhugani (2004). Compressing large boolean matrices using reordering techniques. *Proceedings of the Thirtieth international conference on Very large data bases 30*, 13–23.

Jolliffe, I. (1986). *Principal Component Analysis*. Springer.

Kirkpatrick, S., C. Gelatt, et M. Vecchi (1983). Optimization by simulated annealing. *Science 220*, 681–690.

Kohonen, T. (1995). *Self Organizing Maps* (Heidleberg ed.). Berlin : Springer.

Long, B., Z. Zhang, et P. Yu (2005). Co-clustering by block value decomposition. *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, 635–640.

Marcotorchino, F. (1987). Block seriation problems : a unified approach. *Applied Stochastic Models and Data Analysis 3*, 73–91.

McCormick, W., P. Schweitzer, et T. White (1972). Problem decomposition and data reorganization by a clustering technique. *Operations Research 20*, 993 –1009.

Mechelen, I. V., H.-H. Bock, et P. D. Boeck (2004). Two-mode clustering methods : a structured overview. *Statistical Methods in Medical Research 13*, 363–394.

Niermann, S. (2005). Optimizing the ordering of tables with evolutionary computations. *The American Statistician 59*(1), 41–46.

Nolan, D. (1991). The excess-mass. *Journal of Multivariate Analysis 39*, 348–371.

Pearson, K. (1901). On lines and planes of closest fit to systems of points is space. *Philosophical magazine series 6 2*, 157–175.

Prelic, A., S. Bleuler, P. Zimmermann, P. B. A. Wille, W. Gruissem, L. Hennig, L. Thiele, et E. Zitzler (2006). A systematic comparison and evaluation of biclustering methods for gene expression data. *Bioinformatics 22*(9), 1122–1129.

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

Rossman, M., R. Twery, et F. Stone (1958). *A Solution to the Traveling Salesman Problem by combinatorial programming*. Peat, Marwick, Mitchell and Co., Chicago.

Ruspini, H. (1970). Numerical methods for fuzzy clustering. *Information Sciences 2*(3), 319–350.

Ultsch, A. (2005). Clustering with som : U*C. *In Proc. Workshop on Self-Organizing Maps, Paris, France*, 75–82.

Summary

Seriation is a useful statistical method to visualize clusters in a dataset but since the data are noisy, the visualization of such a structure becomes difficult. To deal with this problem, a metric based on common neighborhood is proposed to introduce several degrees of sparsity in the dataset. A family of sparse matrices are ordered by a branch and bound algorithm and the matrix which has the best block diagonal form is selected by a compressing criterion. This tool enables to identify clusters without taking into account noisy data or outliers and reveals the intrinsic structure of data. However, if the metric introduces too much sparsity in the data, then under-sampled groups of data could be ousted ; we propose then a multi-scale approach which combines different levels of sparsity in a same visualization.