Recherche de groupes parallèles en classification non-supervisée

Lionel Martin*, Matthieu Exbrayat*, Teddy Debroutelle**,
Aladine Chetouani**, Sylvie Treuillet**, Sébastien Jesset***

*LIFO, Batiment IIIA, rue Léonard de Vinci, B.P. 6759,
F-45067 Orléans cedex 2, prenom.nom@univ-orleans.fr,
http://www.univ-orleans.fr/lifo/
**Laboratoire PRISME, Université d'Orléans,12 rue de Blois,
F-45067 Orléans cedex 2, prenom.nom@univ-orleans.fr,
http://www.univ-orleans.fr/prisme/
***Service Archéologique Municipal d'Orléans, 13 bis rue de la Tour Neuve,
45000 Orléans, sjesset@ville-orleans.fr

Résumé. Dans cet article, nous nous intéressons à une situation de classification non supervisée dans laquelle nous souhaitons imposer une "forme" commune à tous les clusters. Dans cette approche, la "forme" commune sera caractérisée par un hyperplan qui sera le même pour tous les groupes, à une translation près. Les points sont donc supposés être distribués autour d'hyperplans parallèles. La fonction objectif utilisée peut naturellement s'exprimer comme la minimisation de la somme des distances de chaque point à son hyperplan. Comme pour le cas de k-means, la résolution est effectuée par l'alternance de phases d'affectation de chaque point à l'hyperplan le plus proche et de phases de calcul de l'hyperplan qui ajuste au mieux l'ensemble des points qui lui sont affectés. L'objectif étant d'obtenir des hyperplans parallèles, cette phase de calcul est menée simultanément pour tous les hyperplans, par une méthode de régression.

1 Introduction

Le clustering, tâche classique d'apprentissage automatique, consiste à scinder un ensemble d'objets en quelques sous-groupes cohérents. Les critères de cohérence peuvent varier, mais reposent en général sur les distances ou (dis)similarités entre les objets observés. La littérature propose plusieurs classifications des méthodes de clustering ; nous suggérons au lecteur de consulter par exemple (Aggarwal et Reddy (2013)) pour une vue d'ensemble.

Dans cet article, nous nous intéressons au cas où nous possédons une connaissance a priori sur la forme des groupes ; en particulier, les groupes sont supposés être distribués autour d'hyperplans parallèles. Divers jeux de données, notamment en reconnaissance de formes, présentent une structuration spatiale des objets en couches parallèles. La figure 1 donne un exemple de distribution spatiale en frises parallèles. Sur cette figure on observe à gauche le relevé de motifs sur des tessons de poteries carolingiennes. Ces motifs ont été imprimés lors de

la réalisation de la poterie, et présentent une structure rectiligne, dont la visualisation est possiblement altérée par la courbure du tesson. De même, plusieurs lignes - ou registres - de motifs peuvent avoir été réalisées. Notre objectif est ici d'identifier ces différentes lignes. Du fait que l'on travaille sur des tessons, la portion de motif exploitable peut être assez restreinte. Afin de simplifier le travail d'identification, le prétraitement présenté à droite a été réalisé. Il consiste à extraire des points d'intérêt, c'est à dire des points remarquables dans l'image, correspondant aux maximums locaux de critères donnés (Bay et al. (2008); Lowe (2004)). La plupart de ces points sont situés à la périphérie des motifs gravés, mais également à la périphérie du tesson, ce qui introduit des outliers. Le clustering porte sur ces points d'intérêt. On peut observer ici la structuration en frises parallèles, mais également la présence de bruit : légère courbure, registre très partiel en haut de l'image, points d'intérêt non pertinents, registres médian et inférieur qui se connectent à droite de l'image.

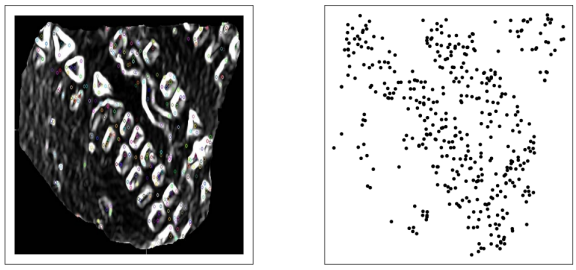

FIG. 1 – *Un exemple de distribution d'objets en frises parallèles. A gauche : reproduction d'un tesson de poterie présentant des motifs imprimés, structurés en lignes –registres– parallèles. On visualise ici les variations de profondeur de surface ; les points d'intérêt sont légèrement marqués. A droite : extraction des points d'intérêt de l'image de gauche.*

Peu d'approches permettent d'introduire une connaissance a priori sur la distribution des données. Les approches à base de mélange de lois fixent un modèle probabiliste de distribution des données –généralement des lois multinormales–. L'objectif est alors de rechercher les paramètres du modèle qui maximisent sa vraissemblance, étant données les observations. D'autres approches s'appuient sur des formes de groupes particulières. Ainsi, les techniques de clustering point-symétriques (Su et Chou (2001)) vont chercher des groupes organisés autour d'un centre, ou centroïde, tel que pour chaque objet du groupe on puisse observer un autre objet de ce même groupe qui lui soit approximativement symétrique par rapport au centre.

Dans notre situation, les approches inspirées de l'analyse factorielle typologique (AFT) (Diday (1974)) ou le *Clusterwise Linear Regression* (CLR) (Späth (1982)) semblent les mieux adaptées. L'AFT consiste à rechercher k variétés d'inertie minimum, alors que le CLR s'applique dans un contexte de régression, où l'on recherche k équations ajustant au mieux les données. Dans les deux cas, l'objectif est à la fois de répartir les données en k groupes et de construire un modèle pour chacun des groupes. Les deux méthodes alternent des phases de calcul du modèle avec des phases de modification des groupes, suivant le même principe que les k-means. Toutes deux partent d'une partition initiale et effectuent une recherche locale. Comme le montre l'exemple suivant (figure 2, table 1), l'efficacité de ce type de méthode dé-

pend de l'initialisation ; la fonction optimisée peut présenter, selon le jeu de données considéré, de nombreux optimums locaux.

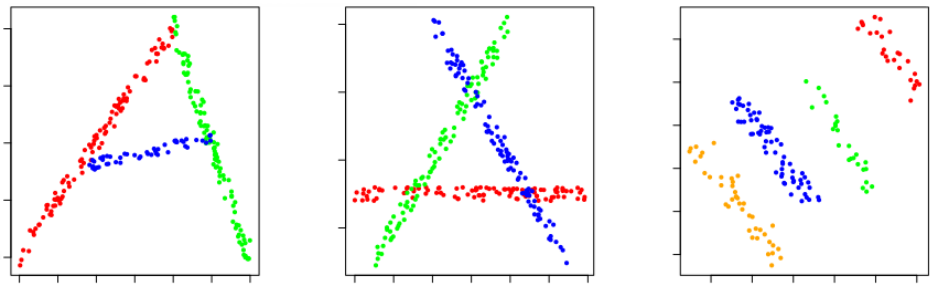

FIG. 2 – *3 jeux de données où les groupes sont alignés, avec respectivement k=3, 3 et 4*

Nous avons appliqué l'approche CLR sur ces trois jeux de données. La configuration initiale a été obtenue en tirant k centres (points) au hasard et en affectant tous les points au centre le plus proche. Dans chaque cas, nous avons répété l'algorithme 1000 fois et calculé le F-score après l'initialisation et après l'exécution. Nous avons également déterminé le taux de réussite (une exécution est considérée comme réussie si le F-score dépasse 0.90).

	taux de réussite	F-score après initialisation	F-score après exécution
Jeu n°1	83.7%	0.64 ± 0.06	0.90 ± 0.11
Jeu n°2	27.9%	0.52 ± 0.04	0.70 ± 0.18
Jeu n°3	4.7%	0.66 ± 0.10	0.61 ± 0.15

TAB. 1 – *Score de la méthode CLR sur les jeux de données de la figure 2*

Sans tirer des conclusions générales à partir de ces exemples, nous observons que ces méthodes sont très sensibles à l'initialisation, et que l'optimum global peut être difficile à atteindre. C'est en particulier le cas sur le troisième jeu de données, plus conforme à notre hypothèse de distribution, pour lequel l'exécution de l'algorithme induit une dégradation du F-score moyen (par rapport au F-score après initialisation). Pour ces raisons, nous proposons dans cet article une méthode inspirée de CLR qui a pour objectif de rechercher des hyperplans parallèles ; cette méthode est notée *P*-CLR (pour "Parallel-CLR"). Nous proposons également trois méthodes d'initialisation qui visent à trouver une partition initiale adaptée à l'hypothèse de distribution autour d'hyperplans parallèles.

Cet article est structuré comme suit : en section 2, nous présentons l'approche générale. Nous proposons ensuite diverses techniques d'initialisation de l'algorithme en section 3. Les expérimentations réalisées sont présentées et commentées en section 4. Enfin, nous concluons et évoquons les possibilités d'extension de notre méthode en section 5.

2 Approche proposée

Nous nous plaçons dans un contexte où un ensemble de points $\{x_j\}_{j=1\ldots n} \in \mathbf{R}^d$ doit être partitionné en k groupes. Dans la suite nous noterons les groupes G_i, $i = 1 \ldots k$, où G_j désigne l'ensemble des points appartenant au groupe d'indice i. Notre hypothèse de travail consiste à considérer que les k groupes forment des nuages de points qui sont distribués autour d'hyperplans parallèles ; dans $\mathbf{R}^2$, cette hypothèse implique que les points sont distribués autour de droites parallèles.

Notre objectif consiste donc à la fois à déterminer les équations de k hyperplans parallèles, et également à déterminer le groupe auquel chaque point est affecté. Cet objectif peut être formulé, comme dans le cas de l'algorithme k-means, par la minimisation d'un critère :

$$Q = \sum_{i=1}^{k} \sum_{x_j \in G_i} l(x_j, G_i) \tag{1}$$

où $l(x_j, G_i)$ désigne le coût de l'affectation de l'objet x_j au groupe G_i (dans le cas de k-means, ce coût est le carré de la distance entre x_j et le centre du groupe G_i). Dans l'approche proposée ici, chaque groupe sera caractérisé par l'équation d'un hyperplan. Ainsi, le groupe G_i sera associé à l'hyperplan d'équation :

$$h_i(x) = 0 \quad avec \quad h_i(x) = w.x + b_i \tag{2}$$

où $w.x$ désigne le produit scalaire entre w et x. Tous les hyperplans partagent le même vecteur de paramètres w dans leurs équations ; seuls les paramètres b_i sont spécifiques à chaque hyperplan, ce qui assure que les hyperplans soient parallèles. La résolution alternera des phases d'affectation (pendant lesquelles les équations des hyperplans sont inchangées) avec des phases de recalcul des équations des hyperplans (pendant lesquelles les affectations sont inchangées). En garantissant une amélioration du critère (1) lors de chacune de ces phases, la méthode est assurée de converger vers un optimum (local).

Ces deux phases de calcul sont détaillées dans la suite de cette section. Nous consacrerons la section suivante à l'étape d'initialisation, dont l'objectif est d'obtenir une première organisation en groupes qui soit assez cohérente. Plusieurs méthodes d'initialisation sont présentées ; leur influence sur la classification obtenue est évaluée dans la section d'expérimentations.

2.1 Étape de calcul des hyperplans

Dans cette phase, nous considérons que les groupes G_i, $i = 1 \ldots k$ sont fixés, donc l'affectation de chaque point à son groupe n'est pas modifiée. Si nous imposions $k = 1$, le critère (1) serait identique au critère usuel de la régression linéaire, cadre dans lequel la fonction de coût est aussi nommée "fonction de perte". Il existe un certain nombre de propositions de fonctions de perte dans la littérature (Smola et al. (1996)), les plus utilisées en apprentissage étant le critère des moindres carrés et le "ϵ-insensitive Loss Function", introduit dans le cas des machines à vecteur de support.

Dans notre cadre, puisque nous supposons $k > 1$, l'étape de calcul des hyperplans se traduit par la réalisation simultanée de k régressions. Notons que ces régressions ne sont pas indépendantes puisque le vecteur de paramètres w est commun à toutes les équations obtenues.

La réalisation de cette étape nécessite donc d'étendre les méthodes de résolution usuelles, en incluant la contrainte de parallèléité des hyperplans (w commun). Dans la suite de cet article, nous nous concentrons sur le critère des moindres carrés.

Dans la méthode de régression linéaire par moindres carrés (Cornillon et Matzner-Løber (2007)), l'une des variables, notée y, est considérée comme la variable cible, les autres variables sont dites explicatives (notons z le vecteur des variables explicatives). Les entrées du problèmes sont donc des couples $\{(z_i, y_i)\}_{i=1\ldots n}$. L'objectif consiste à chercher une expression linéaire $f(z)$, minimisant la somme des écarts quadratiques :

$$\sum_{j=1}^{n}(y_j - f(z_j))^2 \tag{3}$$

Ce modèle est particulièrement adapté à un contexte où la valeur de la variable y doit être prédite à partir des valeurs des variables explicatives z. En ce sens, la fonction de perte mesure l'erreur (quadratique) de la prédiction.

Dans notre cadre, il n'y a pas de distinction entre variable cible et variable explicative. Nous pouvons arbitrairement choisir une variable comme variable cible, les autres seront considérées comme explicatives. À partir de nos données $\{x_j\}_{j=1\ldots n} \in \mathbf{R}^d$, notons $\{y_j\}_{j=1\ldots n} \in \mathbf{R}$ l'une des variables et $\{z_j\}_{j=1\ldots n} \in \mathbf{R}^{d-1}$ les autres variables. Tout hyperplan correspondant à l'équation (2) peut être écrit sous la forme $y = f(z)$ à condition que le coefficient associé à y dans l'équation (2) soit non nul. La nullité de ce coefficient pourrait survenir si tous les groupes étaient orientés exactement suivant l'axe y, ce qui est extrêmement peu probable. En pratique nous avons choisi la dernière variable pour y.

Finalement nous cherchons k équations de la forme :

$$y = f_i(z) \quad avec \quad f_i(z) = w'.z + b_i \qquad i = 1 \ldots k \tag{4}$$

La somme quadratique des écarts (3) peut alors s'écrire sous forme matricielle :

$$\sum_{i=1}^{k}\sum_{x_j \in G_i}(y_j - f_i(z_j))^2 = ||Au - y||^2 \tag{5}$$

où $A \in \mathbf{R}^{n\times(k+d-1)}$ est la matrice $A = (BZ)$ avec :

— $B = (b_{ij}) \in \mathbf{R}^{n\times k}$ est définie par $b_{ij} = 1$ si $x_i \in G_j$ et $b_{ij} = 0$ si $x_i \notin G_j$,

— $Z \in \mathbf{R}^{n\times(d-1)}$ est la matrice composée des vecteurs z_i (en ligne),

— $y \in \mathbf{R}^n$ est le vecteur $(y_1, \ldots y_n)$,

— $u \in \mathbf{R}^{k+d-1}$ est le vecteur $(b_1, \ldots, b_k, w'_1, \ldots, w'_{d-1})$ qui représente l'ensemble des inconnues du problème.

Comme pour le cas de la régression linéaire classique, les conditions d'annulation des dérivées partielles impliquent que la solution satisfait :

$$A'Au = A'y \tag{6}$$

La réalisation de l'étape de calcul des hyperplans, impliquant la minimisation du critère (5) nécessitera donc la résolution de ce système de $(k + d - 1)$ équations linéaires.

2.2 Étape d'affectation et Critère d'arrêt

Pour cette étape, nous considérons l'ensemble des k équations des hyperplans $h_i(x) = 0$ (2) fixé. L'objectif étant d'affecter chaque point à l'hyperplan le plus proche, nous avons une mesure naturelle de l'écart entre un point $x_j = (y_j, z_j)$ et l'hyperplan $y = f_i(z)$ donné par $|y_j - f_i(z_j)|$. Chaque point $x_j, j = 1 \dots n$ sera affecté au groupe d'indice $g(x_j)$, défini par :

$$g(x_j) = Min_{i=1\dots k}|y_j - f_i(z_j)| \tag{7}$$

L'amélioration systématique du critère (5) à chacune des étapes décrites précédemment, assure la convergence de la méthode (le critère (5) étant positif). L'arrêt des itérations intervient lorsque deux solutions consécutives sont identiques, i.e. dès que la partition ne change plus.

3 Initialisation

La section précédente présente les deux étapes de calcul mises en œuvre. Nous proposons ici plusieurs méthodes d'initialisation, dont l'objectif est d'obtenir une première partition qui soit assez proche de la solution recherchée. L'idée commune aux méthodes d'initialisation proposées consiste à rechercher un sous-espace de dimension 1, orthogonal aux hyperplans recherchés. Une fois ce sous-espace trouvé, la partition initiale sera obtenue en exécutant l'algorithme k-means dans ce sous-espace.

Pour obtenir ce sous-espace, nous allons en quelque sorte rechercher des contraintes must-link et cannot-link (Wagstaff et al. (2001)) à l'aide de l'arbre de recouvrement minimal (ARM) (Joseph B. Kruskal (1956)). Étant donné un graphe connexe (S, A) où S est un ensemble de sommets et A un ensemble d'arêtes pondérées, l'ARM est le sous-graphe de (S, A) qui connecte tous les sommets et dont la somme des poids des arêtes est minimale.

Considérons le graphe dont les sommets sont les points $\{x_i\}_{i=1\dots n}$ et ayant une arête entre chaque couple de points x_i et x_j, pondérée par la distance entre ces 2 points [1]. L'ARM obtenu à partir de ce graphe présente une structure très simple et s'il existe une arête entre deux points dans ce graphe :

— plus la distance entre ces 2 points est faible (relativement aux poids des autres arêtes de l'ARM), plus il est envisageable que ces deux points soient dans le même groupe,

— plus la distance entre ces 2 points est grande, moins il est envisageable que ces deux points soient dans le même groupe. En effet, l'arête qui les relie peut être considérée comme un pont entre deux groupes relativement lointains.

L'idée utilisée ici consiste à éliminer les plus grandes arêtes de ce graphe, ce qui aura pour effet d'isoler les outliers et de fragmenter en sous-graphes plus petits. Nous faisons l'hypothèse que les points d'un même sous-graphe appartiennent majoritairement à un même groupe.

3.1 Minimisation des poids minimaux (minDmin)

Le principe de cette méthode consiste à rechercher une partition initiale dans laquelle les points reliés par une arête de faible poids appartiennent si possible au même groupe. Nous plaçons en quelque sorte des contraintes must-link sur ces couples de points.

1. notons que le clustering hiérarchique par "lien simple" peut être obtenu directement à partir de cet ARM

Pour cela, nous choisissons un taux noté δ qui définit le nombre d'arêtes retenues : nous conservons les $n \times \delta$ arêtes de poids minimum dans l'ARM. Nous cherchons alors le sous-espace de projection qui minimise la somme des carrés des distances entre ces points. Ce sous-espace est obtenu par une forme restreinte de l'analyse en composantes principales (ACP) en retenant le vecteur propre associé à la plus petite valeur propre. L'ACP a pour objectif d'optimiser la somme des distances entre tous les couples de points, nous l'adaptons ici pour optimiser la somme partielle des distances correspondant aux $n \times \delta$ arêtes retenues.

La figure 3 illustre cette initialisation. Le choix de la valeur de δ sera discuté en section 4.

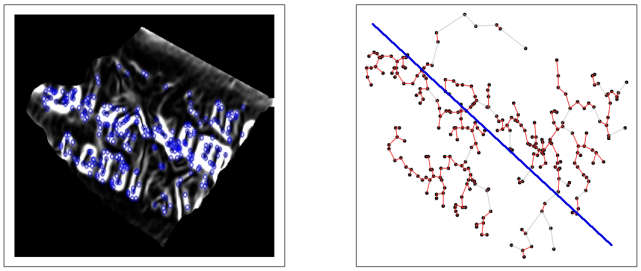

FIG. 3 – *À gauche une image avec 3 fragments de motifs alignés et les points d'intérêt (ronds). À droite l'ARM, les 90% d'arêtes les plus petites (épaisses) et la droite orthogonale au sous-espace de projection, orientation présumée des motifs.*

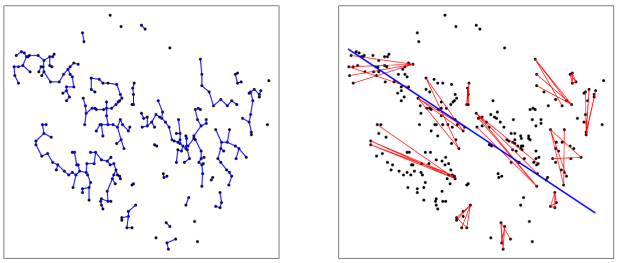

FIG. 4 – *À gauche : les 90% d'arêtes les plus petites. À droite : les 5 couples de points les plus éloignés (reliés par des arêtes) de chaque composante connexe du graphe de gauche et la droite orthogonale au sous-espace de projection.*

3.2 Minimisation des poids maximaux (minDmax)

Lorsque nous éliminons les $n \times \delta$ plus grandes arêtes de l'ARM, le graphe obtenu n'est pas connexe (figure 4, à gauche). Il est dès lors possible de retenir les N couples de points les plus éloignés dans chacune des composantes connexes, et d'ajouter en quelque sorte des contraintes must-link sur ces couples. Le principe de calcul du sous-espace de projection consiste, comme

dans le cas précédent, à minimiser la somme quadratique des distances entre ces couples de points, par une ACP restreinte.

3.3 Écart maximal entre distance euclidienne et ISOMAP (maxIsomap)

La dernière méthode d'initialisation proposée est fondée sur la distance Isomap induite à partir de l'ARM : la distance Isomap entre 2 points est obtenue en calculant le plus court chemin dans l'ARM entre ces 2 points. Nous exploitons ici les distorsions entre cette distance Isomap et la distance euclidienne (dans l'espace d'origine). Si la distance Isomap entre 2 points est grande alors que la distance euclidienne est petite, nous imposons que ces 2 points ne soient pas dans le même groupe (contrainte cannot-link). Le sous-espace de projection est obtenu en maximisant la somme quadratique des distances (euclidiennes) pour les couples de points retenus, à nouveau par une ACP restreinte (figure 5). Le paramètre à fixer est ici le nombre de couples retenus. Dans nos expérimentations nous en avons retenu 200.

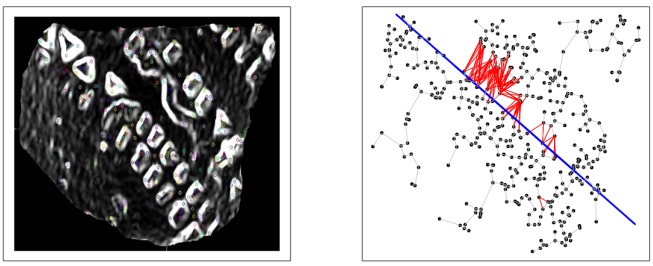

FIG. 5 – *À gauche une image avec 3 fragments de motifs alignés et les points d'intérêt. À droite : les arêtes avec la plus grande distorsion et la droite orthogonale au sous-espace de projection.*

3.4 Complexité des méthodes d'initialisation

Rappelons que n désigne le nombre d'objets, d la dimension de l'espace de représentation. Chaque méthode d'initialisation nécessite la calcul de l'ARM dont la complexité est en $\mathcal{O}(n^2 log(n))$, par l'algorithme de Kruskal (Joseph B. Kruskal (1956)). Pour rechercher les plus petites arêtes ou les plus grandes, il suffit de trier les n arêtes de l'ARM (complexité en $\mathcal{O}(nlog(n))$). Pour les méthodes minDmin et minDmax, il faut alors construire la matrice à diagonaliser (au maximum n produits scalaires dans un espace de dimension d, complexité en $\mathcal{O}(n \times d)$) et enfin diagonaliser la matrice $d \times d$ obtenue (complexité en $\mathcal{O}(d^3)$).

Pour la méthode d'initialisation maxIsomap, il faut en plus calculer la matrice des distances Isomap à partir de l'ARM (graphe ayant n sommets et $n-1$ arêtes). Cette matrice peut être obtenue par l'algorithme de Floyd-Warshall (complexité en $\mathcal{O}(n^3)$) ou en itérant l'algorithme de Dijkstra pour chaque noeud : cet algorithme peut être implémenté en $\mathcal{O}(nlog(n))$ pour chacun des noeuds, soit une complexité globale en $\mathcal{O}(n^2 log(n))$. La complexité de cette méthode d'initialisation est donc du même ordre que celle des 2 méthodes précédentes.

4 Expérimentations

Nous présentons des résultats expérimentaux sur des données réelles et sur des données synthétiques. Dans les deux cas, il s'agit de points en 2 dimensions :

— les données réelles sont les points d'intérêt obtenus à partir de 32 images de fragments de tessons avec des motifs. Nous avons manuellement identifié les groupes ainsi que l'orientation optimale pour la projection initiale,

— nous avons développé un générateur qui produit des points avec un nombre variable de groupes, une orientation variable des nuages, un écart (distance) aléatoire entre les nuages de points (les groupes sont plus ou moins proches) et une quantité de bruit (uniforme) paramétrable. Pour ces données de synthèse, nous avons également les groupes cibles ainsi que l'orientation optimale.

Dans chaque cas, le nombre de groupes recherché correspond au nombre de groupes réel. Dans ces expérimentations, nous avons choisi $\delta = 0.9$ et $N = 10$ pour l'initialisation **minDmax**.

4.1 Influence de l'initialisation

Nous proposons dans un premier temps, de mesurer l'influence de la méthode d'initialisation sur les résultats obtenus. La première expérience porte sur le paramètre δ qui spécifie le taux de sélection des arêtes dans l'ARM, pour l'initialisation **minDmin**. Nous avons généré des données avec 0%, 10% et 20% de bruit. Nous pouvons étudier l'erreur entre l'orientation obtenue par notre méthode et l'orientation optimale (angle mesuré en degrés), en faisant varier le taux entre 70% et 100% (moyenne sur 1000 jeux de données générés). Le but est de déterminer un paramétrage satisfaisant.

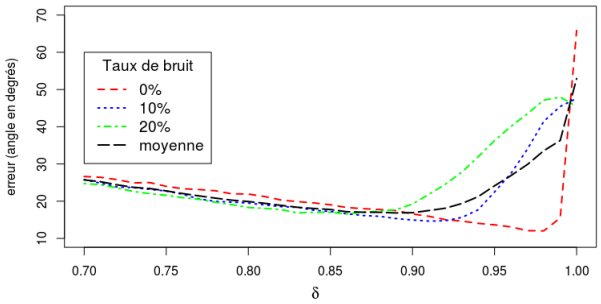

FIG. 6 – *Erreur lors de l'initialisation en fonction de δ et du taux de bruit.*

Sans surprise, la figure 6 montre que le bruit dégrade la qualité de l'initialisation. Par ailleurs, plus les données sont bruitées, plus il est nécessaire de réduire δ. L'erreur minimale pour la courbe moyenne est obtenue pour $\delta = 0.89$, nous utiliserons cette valeur dans nos tests, cette méthode d'initialisation sera notée **minDmin0.89**.

Une autre possibilité de choix repose sur la valeur relative de la plus petite valeur propre dans l'ACP restreinte : plus cette valeur (relative) est petite, plus la projection réduit les dis-

tances entre les points liés par une contrainte must-link. Nous appelons **minDmin*** la variante de **minDmin** reposant sur le paramètre δ associé à la plus petite valeur propre (relative).

Le tableau (2) présente l'erreur moyenne sur les données réelles, entre l'angle obtenu après initialisation et l'angle optimal (pour comparaison, la colonne "alea" présente l'erreur avec un angle aléatoire). On peut noter la mauvaise performance de *maxIsomap* sur les données réelles, très précise sur certains jeux de données, très mauvaise sur d'autres.

	alea	minDmin0.89	minDmin*	minDmax	maxIsomap
Erreur	45.22 ($\pm$24.04)	15.01 ($\pm$19.11)	11.09 ($\pm$14.54)	**7.57 ($\pm$7.85)**	42.59 ($\pm$35.18)

TAB. 2 – *Erreur de l'initialisation sur l'angle optimal (en degrés)*

Le tableau suivant (3) montre l'impact de l'initialisation sur la F-mesure (200 répétitions pour les données synthétiques, bruit entre 0% et 25%). Pour cette comparaison, nous avons ajouté deux méthodes d'initialisation plus simples : **base** où des centres sont tirés aléatoirement et les points sont affectés aux centres les plus proches ; **projAlea** où k-means est exécuté après projection suivant un vecteur tiré aléatoirement.

	Données synth.			Données réelles		
	init.	final	+ rot.	init.	final	+ rot.
P-CLR+base	0.550	0.530	0.611	0.526	0.663	0.667
P-CLR+projAlea	0.514	0.546	0.536	0.574	0.675	0.715
P-CLR+minDmin0.89	0.814	0.764	0.827	0.715	0.737	0.758
P-CLR+minDmin*	**0.819**	0.776	**0.841**	0.738	0.752	0.771
P-CLR+minDmax	0.743	0.719	0.776	**0.767**	0.741	**0.792**
P-CLR+maxIsomap	0.798	0.768	0.830	0.608	0.720	0.734

TAB. 3 – *Influence de l'initialisation sur la F-mesure avec la méthode* P-*CLR*

Ces résultats mettent en évidence l'importance de l'étape d'initialisation : le F-score obtenu après l'initialisation est présenté colonne "init.". Ils montrent également que l'exécution de l'algorithme proposé peut dégrader le score obtenu après initialisation (colonnes "final"). Ce problème est essentiellement dû à la méthode de régression choisie. En effet, la méthode des moindres carrés ne minimise pas la distance entre l'hyperplan et les points mais plutôt une erreur de prédiction de la variable cible.

Dans un espace à 2 dimensions, plus le nuage de points est incliné, plus grande est la valeur du critère (5). Ainsi, il est possible d'obtenir une droite qui ajuste au mieux un nuage de points (au sens des moindres carrés (3)) mais qui n'est pas optimale en terme de distances entre les points et la droite. Pour corriger cette anomalie, nous avons intégré dans notre approche une étape de rotation après le calcul des hyperplans : la rotation est déterminée pour que les hyperplans deviennent alignés avec l'axe horizontal. Le critère des moindres carrés correspond dans ce cas à la somme des distances entre les points et la droite. Les résultats obtenus avec cette version sont présentés dans la colonne "+rot". Nous utilisons la version avec rotation dans les tests suivants.

4.2 Comparaison des approches

Comparons maintenant *P*-CLR avec des approches existantes : k-means, clustering hiérarchique avec "lien simple" (hclust), mélange de gaussiennes (EM-GMM), clustering "points symétriques" (SBKM, (Su et Chou (2001))), dbscan [2] et bien entendu CLR. Pour CLR et *P*-CLR nous utilisons les différentes initialisations proposées. Compte tenu de la présence de bruit, à la fois dans les données synthétiques et les données réelles, le calcul de la F-mesure ne reflète que partiellement la qualité de la partition produite (colonnes "avec bruit"). Pour avoir une idée plus précise, nous avons recalculé la F-mesure en éliminant les points appartenant à la classe "bruit" dans les données source (colonnes "sans bruit"). Notons que ces points ont été supprimés uniquement pour le calcul de la F-mesure, ils ont été conservés pour la phase de classification.

	Données synthétiques		Données réelles	
	avec bruit	sans bruit	avec bruit	sans bruit
k-means	0.526	0.584	0.529	0.571
hclust	0.444	0.484	0.604	0.659
EM-GMM	0.761	0.852	0.627	0.655
SBKM	0.488	0.540	0.575	0.617
dbscan	0.721	0.748	0.633	0.680
CLR+base	0.550	0.631	0.666	0.713
CLR+minDmin0.89	0.735	0.860	0.733	0.783
CLR+minDmin*	0.736	0.861	0.738	0.787
CLR+minDmax	0.648	0.752	0.741	0.791
CLR+maxIsomap	0.698	0.812	0.705	0.754
P-CLR+minDmin0.89	0.835	0.948	0.763	0.813
P-CLR+minDmin*	**0.836**	**0.950**	0.771	0.821
P-CLR+minDmax	0.766	0.842	**0.796**	**0.847**
P-CLR+maxIsomap	0.791	0.897	0.737	0.794

TAB. 4 – *Comparaison de la F-mesure selon la méthode utilisée*

Nous pouvons remarquer que quelle que soit la méthode d'initialisation, *P*-CLR obtient les meilleurs résultats alors que, comme nous l'avons évoqué en introduction, la méthode CLR n'induit pas d'amélioration du F-score, voire une dégradation sur les données synthétiques (par rapport au F-score obtenu après initialisation présenté dans le tableau 3).

5 Conclusion

Dans cet article, nous avons présenté une méthode capable de partitionner un ensemble de points en groupes, en intégrant une contrainte sur la forme géométrique partagée par tous les groupes. Cette contrainte impose que les points soient distribués autour d'hyperplans parallèles. Les résultats expérimentaux montrent à la fois le bon comportement de la méthode

2. Pour dbscan, nous avons itéré l'algorithme afin d'obtenir le nombre de groupes souhaités, en fixant le paramètre de taille du voisinage à 3, et en faisant varier le paramètre de distance.

présentée et l'importance de l'étape d'initialisation. Une amélioration possible pour cette étape serait de pouvoir prévoir quelle méthode utiliser. Nous avons en effet noté des performances diverses pour un même jeu de données réelles, selon la méthode d'initialisation utilisée.

Dans nos données réelles, les motifs sont distribués autour de lignes plus ou moins incurvées. Une extension naturelle de l'approche présentée ici consisterait à utiliser une méthode de résolution permettant d'intégrer un noyau, par exemple basée sur l'analyse factorielle. Une autre extension possible consisterait à rechercher des groupes distribués non pas autour d'hyperplans parallèles, mais autour de sous-espaces parallèles de dimension inférieure à $d - 1$.

Références

Aggarwal, C. C. et C. K. Reddy (2013). *Data Clustering : Algorithms and Applications* (1st ed.). Chapman & Hall/CRC.

Bay, H., A. Ess, T. Tuytelaars, et L. Van Gool (2008). Speeded-up robust features (surf). *Comput. Vis. Image Underst. 110*(3), 346–359.

Cornillon, P.-A. et É. Matzner-Løber (2007). *Régression : Théorie et applications*. Statistique et probabilités appliquées. Springer.

Diday, E. (1974). Introduction à l'analyse factorielle typologique. *Revue de Statistique Appliquée 22*(4), 29–38.

Joseph B. Kruskal, J. (1956). On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society 7*(1), 48–50.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision 60*(2), 91–110.

Smola, A., C. Burges, H. Drucker, S. Golowich, L. V. Hemmen, K.-R. Müller, B. Schölkopf, et V. Vapnik (1996). Regression estimation with support vector learning machines.

Späth, H. (1982). A fast algorithm for clusterwise linear regression. *Computing 29*(2), 175–181.

Su, M. et C. Chou (2001). A modified version of the k-means algorithm with a distance based on cluster symmetry. *IEEE Trans. Pattern Anal. Mach. Intell. 23*(6), 674–680.

Wagstaff, K., C. Cardie, S. Rogers, et S. Schroedl (2001). Constrained k-means clustering with background knowledge. In *In ICML*, pp. 577–584. Morgan Kaufmann.

Summary

In this paper we focus on an unsupervised classification case, where clusters share a common "shape". We consider that this shape consists of a given hyperplane, common to all clusters up to a given a translation. Points are thus considered as distributed around a set of parallel hyperplanes. The underlying objective function can be seen as minimizing the sum of distances of each point to its hyperplane. Similarly to k-means, this goal is achieved by alternating affectation- (of each point to an hyperplane) and computation- (of the hyperplane equation) phases. Seeking for parallel hyperplanes, this computation phase is conducted simultaneously for all hyperplanes.