Département : Informatique

Clustering multi-sujet pour la neuroimagerie

Rapport de stage de 4ème année à l'école ingénieur

Yih-Dar SHIEH

Tuteurs de stage :
Julien Lefèvre
Sylvain Takekart

Stage du 1er juin au 30 août 2017

Institut de Neurosciences de la Timone

Mots clés : machine learning, clustering, multi-sujet, neuroimagerie

Remerciements

Je tiens à remercier toutes les personnes qui ont contribué au succès de mon stage et qui m'ont aidé lors de la rédaction de ce rapport.

Tout d'abord, j'adresse mes remerciements à M. Stephane Ayache de Polytech Marseille, qui m'a aidé dans ma recherche de stage et m'a permis de trouver l'équipe d'apprentissage de marseille. Avec cela, j'ai eu la contact de mon tuteur de stage.

Je tiens à remercier vivement mes tuteurs de stage, M. Julien Lefèvre et M. Sylvain Takekart, de l'Institut de Neurosciences de la Timone de l'Université Aix-Marseille, pour leur accueil, le temps passé ensemble et le partage de leur expertise au quotidien. Grâce aussi à leur confiance, j'ai pu m'accomplir totalement dans mes missions de manière autonome. Leurs conseils m'ont aidé à trouver la bonne direction de recherche.

Enfin, je remercie également toute l'équipe MECA pour leur accueil et leur esprit d'équipe. C'est vraiment agréable de faire un stage ici.

Résumé

L'objectif de mon stage est de développer une méthode de clustering qui permette de fournir un nouveau type de parcellisation du cerveau pour un ensemble de sujet, qui respecte à la fois la différence de chaque sujet et le consensus entre les sujets (par exemple, la forme de la parcellisation).

J'ai obtenu une telle méthode de clustering conjoint, de type k-means, en ajoutant une contrainte aux positions relatives de centroïde de cluster. La méthode est étudiée très attentivement à la fois en théorie et en mise en œuvre, pour l'efficacité et la scalabilité. En théorie, la complexité est améliorée de $\tilde{O}(M^2K^4d)$ à $\tilde{O}(MK^2d)$ [1], ici M est le nombre de sujet, K est le nombre de clusters et d est la dimension. En pratique, l'implémentation rend notre méthode encore plus puissant quand d est grand.

Cette méthode est appliquée sur des données synthétique, ainsi que des données de surface de cerveau et des profils de connectivité cérébrale. La combinaison de caractéristiques et de coordonnées donne des parcellisations avec un consensus sur la forme.

1. C'est la complexité pour calculer $\mathscr{L}(\mathcal{C})$ introduit dans le chapitre 2. Sous l'hypothèse que le nombre d'étapes du gradient conjugué est borné, c'est aussi la complexité de notre méthode de clustering conjoint.

Table des matières

Chapitre 1

Introduction

1.1 Laboratoire d'accueil

Mon stage se déroule à l'institut de Neurosciences de la Timone, qui est un laboratoire du CNRS, dans l'équipe "Méthodes et anatomie computationelle". L'équipe MeCA est dirigée par Olivier Coulon (directeur de recherche au CNRS), Julien Lefèvre (maître de conférence des Universités à l'Université Aix-Marseille) et Sylvain Takekart (ingénieur Recherche au CNRS). Mes tuteurs de stage sont Julien Lefèvre et Sylvain Takekart.

1.1.1 Institut de Neurosciences de la Timone

L'objectif de l'Institut de Neurosciences de la Timone (INT) est de développer des recherches de haut niveau en neurosciences fondamentales, du cellulaires au cognitif, et de faire tomber les frontières entre les approches fondamentales et cliniques. Elle accueille 9 équipes de recherche et 1 équipe de recherche pluridisciplinaire "Méthodes et anatomie computationelle". L'organisation et la hiérarchie de INT sont présentées dans fig. 1.1.

L'INT occupe un bâtiment entier de 4500m2 sur le Campus de la Faculté de Médecine.

1.1.2 L'équipe "Méthodes et anatomie computationelle"

L'équipe **Méthodes et anatomie computationelle** (MeCA) est un groupe de recherche interdisciplinaire créé par l'Institut de Neurosciences de La Timone et le Laboratoire des Sciences de l'Information et des Systèmes, associant leurs compétences en neurosciences fondamentales et cliniques et en traitement de données afin de mieux comprendre l'organisation et le fonctionnement du cerveau normal et pathologique. Le groupe de recherche est physiquement installé sur le site de La Timone.

1.2 Sujet de stage

1.2.1 Problématique

En neurosciences, comprendre l'organisation du cerveau demande de pouvoir le subdiviser en régions homogènes. Ce processus de segmentation du cor-

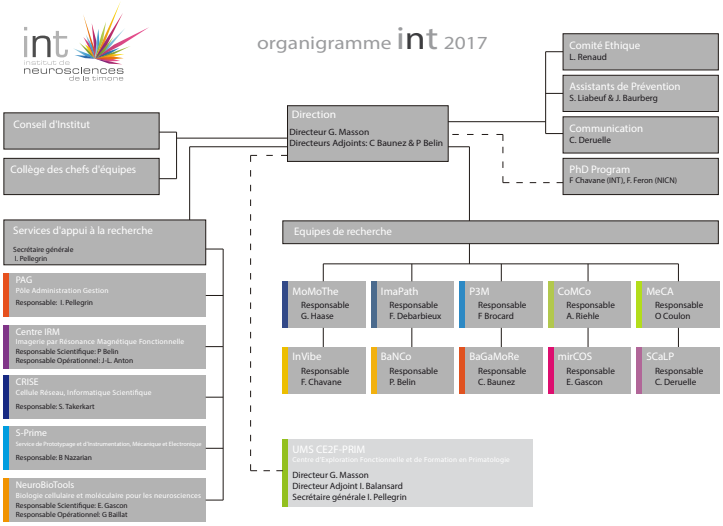


FIGURE 1.1 – Organigramme INT 2017

tex cérébral est aussi appelé parcellisation. On peut obtenir un tel ensemble de parcelles en utilisant des méthodes de clustering sur des images IRM du cerveau, visant ainsi à regrouper les pixels du cortex qui partagent des propriétés communes. De manière classique, ces parcellisations peuvent être estimées séparément pour chaque individu, ou en cherchant une parcellisation commune à un groupe d'individus. Les parcellisations individuelles sont assez souvent trop bruitées, tandis que les parcellisations de groupe ont tendance à gommer les différences inter-individuelles.

1.2.2 Sujet

L'objectif de mon stage est de **développer une méthode de clustering** qui permette de fournir un nouveau type de parcellisation, de manière intermédiaire aux deux méthodes classiques évoquées ci-dessus. La méthode de parcellisation à mettre en place devra estimer **de manière conjointe** un ensemble de parcellisations individuelles pour chaque membre d'un groupe. Ceci peut s'envisager en ajoutant des contraintes particulières dans la méthode de clustering.

1.2.3 Mission

Ma mission de stage consiste à définir et implémenter une telle méthode de clustering conjoint, à l'appliquer sur des données réelles (images IRM disponibles dans l'équipe) et à évaluer quantitativement la qualité des parcellisations obtenues. En particulier, il visera deux applications différentes en étudiant

d'une part la forme du cortex à l'aide d'IRM anatomique, et d'autre part l'organisation de la connectivité cérébrale en utilisant l'IRM de diffusion.

1.3 Planning

Mon stage de 3 mois est réparti en

1. **Étude bibliographique** (4 semaines) : méthodes de clustering, de segmentation d'images, étude antérieure sur clustering multi-sujet et multi-view.
2. **Développement** (5 semaines) : la méthode, l'implémentation, amélioration de la complexité, le bon usage des méthodes dans NumPy.
3. **Applications** (3 semaines) : sur des données synthétiques, sur des données de surface de cerveau, sur des profils de connectivité cérébrale.

1.4 Organisation du rapport

La suite de ce rapport est organisée comme suit : Le chapitre 2 décrit la contribution théorique de ce stage, **une méthode de clustering conjoint de type k-means**, notamment dans Section 2.5. Pour ce but, on introduit aussi la notion de clustering et la méthode k-means. Puis, le chapitre 3 est dédiée à mes contributions d'implémentation. Ensuite, les résultats obtenus sont montrés dans le chapitre 4, et le rapport se conclut dans le chapitre 5.

Chapitre 2

Conception

Dans ce chapitre, une nouvelle méthode de clustering de manière conjointe, proposée et étudiée par le stagiaire, est présentée. Beaucoup d'efforts sont faits pour la scalabilité [1] (scalability en anglais) de cette méthode.

2.1 Clustering

Le clustering, ou le partitionnement de données, est une méthode permettant de diviser un ensemble de données en différents groupes homogènes.

Definition 1. *Une partition $\mathcal{C}$ d'un ensemble X est un ensemble de sous-ensembles non vides de X tel que X est l'union disjointe des éléments de $\mathcal{C}$.*

Soit $\mathscr{C}_X$ l'ensemble de toutes les partitions de X. Pour le problème de clustering, on considère souvent un sous-ensemble $\mathscr{C}'$ de $\mathscr{C}_X$ des partitions d'intérêt. Une méthode de clustering définit, essentiellement, une fonction de coût $\mathscr{L} : \mathscr{C}' \to \mathbb{R}_{\geq 0}$ et donne un moyen pour trouver

$$\mathcal{C}^\star = \operatorname*{argmin}_{\mathcal{C} \in \mathscr{C}'} \mathscr{L}(\mathcal{C}). \tag{2.1}$$

Cependant, la (les) solution(s) optimale(s) est souvent difficile à trouver, de sorte que nous relâchons souvent le problème discret à un continu, et une solution approximative ou une solution optimale locale est trouvée. Enfin, un clustering optimal (ou proche) est construit.

2.2 Problématique : Clustering conjoint

Dans cette section, une formulation de la problématique de clustering multi-sujet de manière conjointe est donnée.

Soit $\mathcal{X} = \left\{X^{(m)}\right\}_{m=1}^{M}$ une famille d'ensembles et $X = \bigcup_{m=1}^{M} X^{(m)}$. Pour une partition $\mathcal{C}$ de X, soit $\mathcal{C}^{(m)} = \left\{C \cap X^{(m)}\right\}_{C \in \mathcal{C}}$. Nous ne considérons que les cas où chaque $C \cap X^{(m)}$ est non vide, donc $\mathcal{C}^{(m)}$ est une partition de $X^{(m)}$ pour tout m.

1. La scalability désigne la capacité d'une méthode à s'adapter à un changement d'ordre de taille du problème.

Comme dans Section 2.1, on dispose d'une fonction de coût $\mathscr{L}^{(m)}$ définit sur l'espace des partitions $\mathscr{C}^{(m)}$ de chaque $X^{(m)}$. Un clustering $\mathcal{C}$ de X qui minimise $\sum_{m=1}^{M} \mathscr{L}^{(m)}\left(\mathcal{C}^{(m)}\right)$ peut être obtenu en effectuant un clustering sur chaque $X^{(m)}$ indépendamment.

Pour le clustering conjoint, il faut donc ajouter un terme supplémentaire $\mathscr{L}'(\mathcal{C})$ dans le fonction de coût, qui ne peut être déterminé que par l'information global $\mathcal{C}$. Le fonction de coût que nous étudions a donc la forme

$$\mathscr{L}(\mathcal{C}) = \sum_{m=1}^{M} \mathscr{L}^{(m)}\left(\mathcal{C}^{(m)}\right) + \mathscr{L}'(\mathcal{C}) \cdot \tag{2.2}$$

2.3 Études antérieures

Dans la première partie du stage, j'ai étudié des solutions existantes et en particulier :

- Dans [3], S. RACE et C. MEYER proposent une méthode itérative pour obtenir un clustering de consensus à partir d'un ensemble de clustering. Cela s'applique plutôt à un seul sujet avec des clusterings obtenus par des moyens ou paramètres différents de clustering.
- Dans [1], A. JOULIN, F. BACH et J. PONCE proposent une méthode de segmentation des images avec le clustering discriminatif. Cette méthode combine le clustering spectral (pour le clustering indépendant de chaque image) et la classification (qui utilise les modèles discriminatifs). Cette méthode est intéressante, et peut être utilisée pour notre but. Néanmoins, elle est moins rapide que la méthode développée par nous quand le nombre de points dans chaque sujet est grand. Pour l'application de [1], l'usage de superpixel évite ce problème, mais ce n'est utile que pour la segmentation des images. Pour 30 images, leur méthode obtient une segmentation en entre 30 et 60 minutes.
- Dans [5], W. TANG, Z. LU et I. DHILLON proposent une méthode de clustering avec graphes multiples, en utilisant l'idée "Linked Matrix Factorization". L'idée est de trouver des approximations des matrices d'adjacence de graphe avec un plongement commun (qui capturent l'information de clustering commun) et des vecteurs propres de chaque graphe (qui capture l'information de clustering individuel). Mais elle ne s'applique que au cas ou tous les sujets ont le même nombre de points.

Car on n'a pas trouvé de solution satisfaite, on continue a concevoir une nouvelle solution.

2.4 k-means

La méthode k-means est une méthode de partitionnement de données basée sur les centroïdes. Car notre approche de clustering conjoint est de type k-means, nous l'introduisons brièvement dans cette section. Ici, on considère le clustering pour un seul sujet X.

Soit X un sous ensemble de l'espace vectoriel réel $\mathbb{R}^d$, et $\mathcal{C} = \{C_1, C_2, \ldots, C_K\}$ une partition de X. Soit $\{\mu_k\}_{k=1}^K$ les centroïdes des clusters C_k :

$$\mu_k = \frac{\sum_{x \in C_k} x}{|C_k|} \tag{2.3}$$

La fonction de coût dans section 2.1 pour k-means est donnée par

$$\mathscr{L}(\mathcal{C}) = \sum_{k=1}^{K} \sum_{x \in C_k} (\mu_k - x)^2 \cdot \tag{2.4}$$

Soit $\{\mathrm{Var}\,(C_k)\}_{k=1}^K$ les variances intra-cluster, on a

$$\mathscr{L}(\mathcal{C}) = \sum_{k=1}^{K} |C_k| \,\mathrm{Var}\,(C_k) \cdot \tag{2.5}$$

L'algorithme Algorithm 1 pour former un clustering est itérative et alternée entre l'étape d'affectation et l'étape de mise à jour :

Algorithm 1: k-means algorithm for clustering

clustering (K, X) :
Input : An integer K and a $N \times d$ array X. ▷ $N = |X|$
Output: A clustering $\mathcal{C}$ with K clusters for X.
Initialize a set of K centroids μ_k by an appropriate initialization method.
while the convergence condition is not satisfied **do**
  Form a clustering $\mathcal{C}$ by assigning each point $x \in X$ to the closest μ_k.
  Update the centroid μ_k of C_k for each cluster $C_k \in \mathcal{C}$ using (2.3).
end
return $\mathcal{C}$

Pour les μ_k fixés, l'étape d'affectation trouve les clusters optimaux pour minimiser la fonction de coût dans l'équation (2.4). De même, pour les C_k fixés, l'étape de mise à jour trouve les μ_k qui minimisent l'équation (2.4). Néanmoins, la solution trouvée, soit approximative ou soit exacte, n'est pas forcément la solution optimale globale.

2.5 Contribution théorique : k-means conjoint

Cette section décrit la principale contribution théorique de ce stage : Une méthode de clustering conjoint de type k-means.

Dans la suite de ce rapport, soit M le nombre de sujets, K le nombre de clusters et d la dimension de l'espace de caractéristiques [2].

Au lieu d'un seul ensemble X dans Section 2.4, maintenant soit chaque $X^{(m)}$ un sous ensemble de l'espace vectoriel réel $\mathbb{R}^d$, ici on adopte la notation

2. C'est l'espace vectoriel réel introduit avant l'équation (2.3) dans Section 2.4. Chaque coordonnée représente une certaine caractéristique d'un sujet.

dans Section 2.2. Soit $\mu_k^{(m)}$ les centroïdes des clusters $C_k^{(m)}$ de $X^{(m)}$, pour $k = 1, 2, \ldots, K$ et $m = 1, 2, \ldots, M$.

Le premier terme dans l'équation (2.2), appelé le **coût de base**, dans un clustering conjoint de type k-means, devient

$$\mathscr{L}_B(C) = \sum_{m=1}^{M} \left(\sum_{k=1}^{K} \sum_{x^{(m)} \in C_k^{(m)}} \left(\mu_k^{(m)} - x^{(m)} \right)^2 \right). \tag{2.6}$$

Soit $\{\mathrm{Var}(C_k^{(m)})\}_{k=1}^{K}$ les variances intra-cluster de chaque sujet $X^{(m)}$, pour $m = 1, 2, \ldots, M$. On a

$$\mathscr{L}(\mathcal{C}) = \sum_{m=1}^{M} \sum_{k=1}^{K} \left| C_k^{(m)} \right| \mathrm{Var}\left(C_k^{(m)} \right). \tag{2.7}$$

2.5.1 La contrainte pour le clustering conjoint

Pour le deuxième terme $\mathscr{L}'(\mathcal{C})$ dans l'équation (2.2), on impose une contrainte aux positions relatives des centroïdes dans chaque $X^{(m)}$. Cela donne

$$\begin{aligned} \mathscr{L}'(\mathcal{C}) &= \mathscr{L}'_R(\mathcal{C}) \\ &= \sum_{1 \le m < n \le M} \sum_{\substack{1 \le i < j \le K \\ 1 \le k < l \le K \\ (i,j) \le (k,l)}} \left(\left\langle \mu_i^{(m)} - \mu_j^{(m)}, \mu_k^{(m)} - \mu_l^{(m)} \right\rangle - \left\langle \mu_i^{(n)} - \mu_j^{(n)}, \mu_k^{(n)} - \mu_l^{(n)} \right\rangle \right)^2 \end{aligned} \tag{2.8}$$

Les produits scalaires $\left\langle \mu_i^{(m)} - \mu_j^{(m)}, \mu_k^{(m)} - \mu_l^{(m)} \right\rangle$ mesurent les positions relatives des centroïdes pour le clustering $\mathcal{C}^{(m)}$ du sujet $X^{(m)}$. Par exemple, $\left\langle \mu_i^{(m)} - \mu_j^{(m)}, \mu_i^{(m)} - \mu_j^{(m)} \right\rangle$ est le carré de la distance entre $\mu_i^{(m)}$ et $\mu_j^{(m)}$. La quantité $\mathscr{L}'_R(\mathcal{C})$ mesure donc la différence de positions relatives de centroïdes entre les sujets.

Quand les centroïdes $\mu_k^{(m)}$ de $X^{(m)}$ et $\mu_k^{(n)}$ de $X^{(n)}$ ne diffèrent que par une isométrie affine (une combinaison de translation, rotation et réflexion), on a $\mathscr{L}'_R(\mathcal{C}) = 0$. Pendent le processus de clustering avec les méthodes d'optimisation, plus $\mathscr{L}'_R(\mathcal{C})$ est petit, plus les positions relatives des centroïdes obtenues coïncident.

2.5.2 Complexité : Amélioration de M^2 à M

En utilisant la formule

$$\sum_{1 \le m < n \le M} (a_m - a_n)^2 = M \sum_{m=1}^{M} \left(a_m - \frac{\sum_{m=1}^{M} a_m}{M} \right)^2 = M^2 \, \mathrm{Var}\left(\{a_m\}_{m=1}^{M} \right), \tag{2.9}$$

on obtient

$$\mathscr{L}'_R(\mathcal{C}) = M^2 \sum_{\substack{1 \le i < j \le K \\ 1 \le k < l \le K \\ (i,j) \le (k,l)}} \mathrm{Var}\left(\left\{ \left\langle \mu_i^{(m)} - \mu_j^{(m)}, \mu_k^{(m)} - \mu_l^{(m)} \right\rangle \right\}_{m=1}^{M} \right). \tag{2.10}$$

L'équation (2.10) donne un moyen pour calculer $\mathscr{L}'_R(\mathcal{C})$ avec la complexité M au lieu de M^2 en M.

2.5.3 Complexité : Amélioration de K^4 à K^2

Dans un premier temps, l'équation (2.10) est simplifié par

$$\mathscr{L}'_R(\mathcal{C}) = M^2 \left(\frac{1}{8} \sum_{1\le i,j,k,l\le K} \mathrm{Var}\left(\left\{ \left\langle \mu_i^{(m)} - \mu_j^{(m)}, \mu_k^{(m)} - \mu_l^{(m)} \right\rangle \right\}_{m=1}^{M} \right) + \frac{1}{4} \sum_{1\le i,j\le K} \mathrm{Var}\left(\left\{ \left\langle \mu_i^{(m)} - \mu_j^{(m)}, \mu_i^{(m)} - \mu_j^{(m)} \right\rangle \right\}_{m=1}^{M} \right) \right) \tag{2.11}$$

Le deuxième terme dans l'équation (2.11) a K^2 termes pour chaque m, mais nous avons encore K^4 terms dans le premier terme. Soit

$$\begin{aligned} T_{i,j}^{(m)} &= \left\langle \mu_i^{(m)}, \mu_j^{(m)} \right\rangle,\ t_{i,j} = \mathrm{Var}\left(\left\{ T_{i,j}^{(m)} \right\}_{m=1}^{M} \right), \\ S_i^{(m)} &= \sum_{j=1}^{K} T_{i,j}^{(m)},\ s_i = \mathrm{Var}\left(\left\{ S_i^{(m)} \right\}_{m=1}^{M} \right), \\ U^{(m)} &= \sum_{i=1}^{K} S_i^{(m)},\ u = \mathrm{Var}\left(\left\{ U^{(m)} \right\}_{m=1}^{M} \right), \\ V_i^{(m)} &= T_{i,i}^{(m)},\ v_i = \mathrm{Var}\left(\left\{ V_i^{(m)} \right\}_{m=1}^{M} \right), \\ W^{(m)} &= \sum_{i=1}^{K} V_i^{(m)},\ w = \mathrm{Var}\left(\left\{ W^{(m)} \right\}_{m=1}^{M} \right), \end{aligned} \tag{2.12}$$

on a

$$\begin{aligned} \mathscr{L}'_R(\mathcal{C}) = \frac{1}{2} M^2 \Bigg(\Bigg(K^2 \sum_{1\le i\le j\le K} t_{i,j} - 2K \sum_{i=1}^{K} s_i + u \Bigg) \\ + \left(K \sum_{i=1}^{K} v_i + 2 \sum_{1\le i,j\le K} t_{i,j} + w \right) \Bigg) \\ - 2 \sum_{i=1}^{K} \left(M \sum_{m=1}^{M} S_i^{(m)} V_i^{(m)} - \sum_{m=1}^{M} S_i^{(m)} \sum_{m=1}^{M} V_i^{(m)} \right) \end{aligned} \tag{2.13}$$

L'équation (2.13) donne un moyen pour calculer $\mathscr{L}'_R(\mathcal{C})$ avec la complexité K^2 au lieu de K^4 en K.

2.6 La procédure

Dans Section 2.5, on développe une méthode de clustering conjoint de type k-means, en utilisant les produits scalaire des positions relatives. Dans cette

section, on décrit la procédure pour former un clustering avec une méthode de clustering conjoint de type k-means. Cette procédure générique ne dépend pas de la contrainte choisie, sous la condition que le terme $\mathscr{L}'(\mathcal{C})$ correspondant à la contrainte ne dépend que des centroïdes $\mu_k^{(m)}$.

Soit $\mathscr{L}'(\mathcal{C})$ le terme correspondant à une contrainte satisfaisant l'hypothèse au dessus. La procédure pour former un clustering avec une méthode de clustering conjoint de type k-means est similaire à laquelle de k-means pour un seul sujet :

Algorithm 2: Joint k-means for clustering multiple subjects

clustering $(K, \mathcal{X})$:
Input : An integer K and $\mathcal{X} = \left\{X^{(m)}\right\}_{m=1}^{M}$.
Output: A clustering $\mathcal{C}$ with K clusters for $\mathcal{X}$.
Initialize a set of $M \times K$ centroids $\mu_k^{(m)}$ by an appropriate initialization method.
while the convergence condition is not satisfied **do**
  for *m = 1, 2, . . . , M* **do**
    Form a clustering $\mathcal{C}^{(m)}$ by assigning each point $x^{(m)} \in X^{(m)}$ to the closest $\mu_k^{(m)}$.
  end
  Update the centroid $\mu_k^{(m)}$ of $C_k^{(m)}$ for each cluster $C_k^{(m)} \in \mathcal{C}^{(m)}$ by minimizing $\mathscr{L}(\mathcal{C})$ using an optimization method.
end
return $\mathcal{C}$

Pour l'initialisation, on peut, par exemple, fusionner les points des $X^{(m)}$ pour obtenir X, et utiliser une méthode d'initialisation de k-means sur X.

Pour notre méthode, on utilise la méthode du gradient conjugué pour la minimisation.

2.7 Le gradient

Pour la méthode du gradient conjugué, on a besoin d'un moyen pour calculer le gradient de $\mathscr{L}$. C'est facile pour le terme $\mathscr{L}_B$ dans l'équation (2.6). Pour $\mathscr{L}_R{}'$ dans l'équation (2.8), soit

$$t_{i,j,k,l}^{(m)} = \left\langle \mu_i^{(m)} - \mu_j^{(m)}, \mu_k^{(m)} - \mu_l^{(m)} \right\rangle. \tag{2.14}$$

Pour la contrainte $\mathscr{L}_R{}'$, son gradient et donné par, $\left(\nabla \mathscr{L}_R{}'\right)^{(m)} = G^{(m)} \cdot \mu^{(m)}$, ici $G^{(m)}$ est une matrice de taille $K \times K$ pour tout m, avec les entrées

$$G_{i,j}^{(m)} = 2\left(M \cdot H_{i,j}^{(m)} - \sum_{m=1}^{M} H_{i,j}^{(m)}\right), \tag{2.15}$$

ici

$$H_{i,j}^{(m)} = \sum_{1 \le k,l \le K} t_{i,k,j,l}^{(m)} + \delta_{i,j} \sum_{k=1}^{K} t_{i,k,j,k}^{(m)} - (1 - \delta_{i,j}) t_{i,j,i,j}^{(m)}, \tag{2.16}$$

et $\delta_{i,j}$ est le symbole de Kronecker. Pour calculer le gradient plus rapidement, notez que on a

$$\begin{aligned}
\sum_{1\leq k,l\leq K} t_{i,k,j,l}^{(m)} &= K^2 \cdot T_{i,j}^{(m)} - K\left(S_i^{(m)} + S_j^{(m)}\right) + U^{(m)}, \\
\delta_{i,j} \sum_{k=1}^{K} t_{i,k,j,k}^{(m)} &= \delta_{i,j}\left(K\, V_i^{(m)} - 2\, S_i^{(m)} + W^{(m)}\right), \\
t_{i,j,i,j}^{(m)} &= \left(V_i^{(m)} + V_j^{(m)} - 2\, T_{i,j}^{(m)}\right) \cdot
\end{aligned} \tag{2.17}$$

Chapitre 3

Implémentation

Dans Chapitre 2, nous présentons une méthode de clustering conjoint. Nous l'étudions très attentivement et nous obtenons un moyen de calcul très efficace du point de vue théorique.

Dans ce chapitre, nous présentons les efforts de l'implémentation qui rend notre méthode encore plus puissant.

3.1 Langage de programmation et outils

L'implémentation se fait avec la distribution **Python** de Anaconda, avec des outils

- **SciPy** : Ce projet contient un ensemble de bibliothèques Python à usage scientifique : les modules pour l'optimisation, l'algèbre linéaire, les statistiques, etc.
- **NumPy** : Une extension du langage de programmation Python, destinée à manipuler des matrices ou tableaux multidimensionnels ainsi que des fonctions mathématiques opérant sur ces tableaux.
- **scikit-learn** : Une bibliothèque libre Python dédiée à l'apprentissage automatique.

3.2 Contribution de l'implémentation

3.2.1 L'espaces de caractéristiques de grande dimension

Avec la notation définis au début de Section 2.5, on a $M \times K$ centroïdes, chacun étant un vecteur de dimension d. Dans Section 2.5.2 et Section 2.5.3, on a traité le problème de la complexité en M et en K. Pour la scalabilité de notre méthode en dimension d, c'est linéaire en d. Néanmoins, pour calculer les produits scalaires dans l'équation (2.8) (si on n'utilise pas les améliorations théoriques obtenues), ce sera très coûteux. L'équation (2.12) et l'équation (2.13) suggère d'établir une table de taille $M \times K \times K$ des produits scalaires $\left\langle \mu_i^{(m)}, \mu_j^{(m)} \right\rangle$. Ça réduit beaucoup de temps de calcul quand d est grande.

3.2.2 Calcul rapide du coût de base

Comme décrit dans Section 2.6, on utilise la méthode du gradient conjugué pour la minimisation dans l'étape de mise à jour des $\mu_k^{(m)}$ de la procédure de clustering. Dans cette étape, le calcul du coût de base $\mathscr{L}_B(\mathcal{C})$ peut être accéléré en mettant en mémoire tampon les valeurs $\sum_{x \in C_k^{(m)}} x^n$ pour $n = 0, 1, 2$. (Pour $n = 0$, c'est les cardinalités $|C_k^{(m)}|$. Pour $n = 1, 2$, les opérations sont faites élément par élément.) Ça réduit beaucoup de temps de calcul quand les nombres de points $|X^{(m)}|$ est très grand, surtout dans le cas ou le nombre d'étape de descente du gradient est grand.

3.2.3 Implémentation de la méthode du gradient conjugué

Pour l'étape de mise à jour les centroïdes dans **??**, on utilise la méthode du gradient conjugué. Le module **optimize** dans SciPy contient beaucoup de méthodes d'optimisation, y compris gradient conjugué. Néanmoins, nous implémentons la méthode du gradient conjugué nous-même avec les raisons suivantes :

- Les variables ($\mu_k^{(m)} \in \mathbb{R}^d$) que nous manipulons sont présentés plus naturellement par un tableau multidimensionnel de la forme (M, K, d), mais les méthodes de scipy.optimize prennent un tableau de dimension 1 comme les variables d'une fonction à optimiser. Cela peut être traité en utilisant la méthode numpy.reshape pour transformer la forme d'un tableau, donc ce n'est pas vraiment un problème.
- Parfois, les méthodes de scipy.optimize ne donnent pas de résultat satisfaite. Elles renvoient un avertissement "desired error not necessarily achieved due to precision loss". C'est souvent le cas pour $\mathscr{L}(\mathcal{C})$ quand M, K ou d est grand, car elle n'est pas une fonction convexe. Le gradient obtenu a une norme plus grande que la tolérance spécifiée pour la déclaration de la convergence. Notre implémentation résout ce problème.

Nous ne prétendons pas que notre implémentation est plus efficace ou complète. Mais avec moins de 100 lignes de code pour deux fonctions, on a un moyen plus flexible pour explorer ce qui s'est passé pendant l'optimisation, et un contrôle plus satisfait de convergence.

3.2.4 Utilisation de la force de NumPy

On utilise les fonctions de NumPy intensivement, car ses routines internes sont implémentées en C. Notre programme s'exécute donc très rapidement. Pour ce but, les opérations nécessaires pour calculer $\mathscr{L}(C)$ et son gradient sont faites par les fonctions de NumPy les plus appropriés possible. Par exemple, pour $T_{i,j}^{(m)}$ dans l'équation (2.12), on le calcule comme un tableau de la forme $M \times K \times K$ avec

$$\mu \cdot \mu^t, \tag{3.1}$$

ici μ est un tableau de la forme $M \times K \times d$ avec les entrées $\mu[m, i, j] = \mu_{i,j}^{(m)}$, et μ^t est la transposition de μ selon les 2e et 3e axes, qui a la forme $M \times d \times K$. Le résultat est donc de la forme $M \times K \times K$ avec les entrées $(\mu \cdot \mu^t)[m, i, j] = \left\langle \mu_i^{(m)}, \mu_i^{(m)} \right\rangle$.

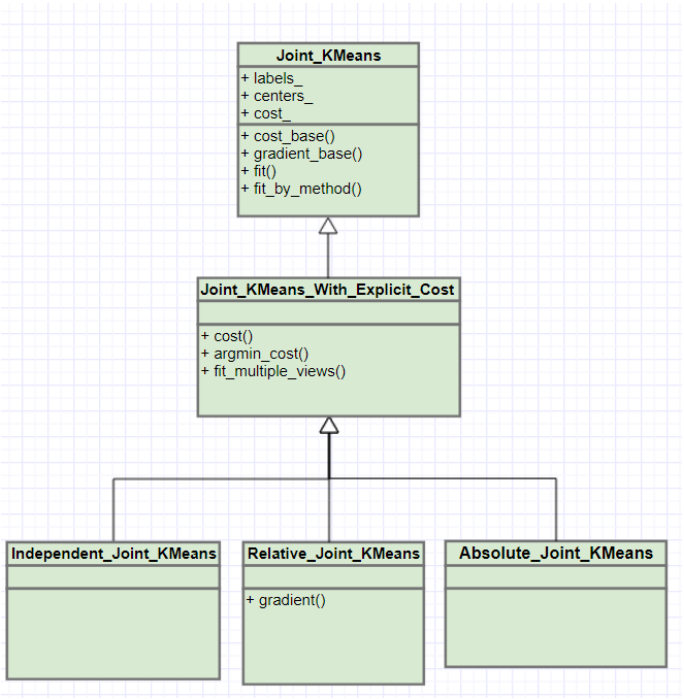


FIGURE 3.1 – Diagramme UML

3.2.5 Combinaison de différentes contraintes

Jusqu'à maintenant, on travaille avec un seul espace de caractéristiques. Pour notre application, on travaille souvent avec plusieurs espaces de caractéristique, l'un d'eux est souvent un type de coordonnées de position. C'est possible que pour un espace de caractéristiques, on veut utiliser k-means indépendamment pour chaque sujet, mais pour l'autre espace, on préfère un k-means conjoint.

Dans ce but, nous implémentons une classe générique **Joint_KMeans**, qui contient les méthodes communes pour une méthode de clustering conjointe de type k-means. Une sous classe **Joint_KMeans_With_Explicit_Cost** qui représente les clusterings conjoints de type k-means avec le coût donné explicitement. On a implémenté 3 sous classe de Joint_KMeans_With_Explicit_Cost :

- **Independent_Joint_KMeans** : La contrainte $\mathscr{L}'(\mathcal{C})$ est zéro.
- **Absolute_Joint_KMeans** La contrainte correspond aux positions absolues de centroïdes.
- **Relative_Joint_KMeans** La contrainte correspond aux positions relatives de centroïdes, décrite dans Section 2.5.

On fait des efforts pour que les méthodes soient comme les méthodes dans le module sklearn.cluster.KMeans. De plus, nous offrons une méthode pour faire le clustering conjoint avec multi-views, c'est à dire, avec plusieurs espaces de caractéristique comme discuté au-dessus. Figure 3.1 présente la structure de notre implémentation.

Chapitre 4

Résultat

Après avoir développé une méthode de clustering conjoint, nous l'avons appliquée sur des données synthétiques, sur des données de surface de cerveau et sur des profils de connectivité cérébrale.. Nous montrons les résultats dans ce chapitre.

4.1 Les Données synthétiques

Nous testons notre méthode sur 2 sujets avec 3 clusters, avec la dimension de l'espace de caractéristiques $d = 2$. Le premier sujet est fixé, et l'autre est obtenu par rotation du premier sujet. La rotation est faite du 0 degré jusqu'à 90 degré.

Dans ce test, car on génère les données, on a la vérité terrain du clustering. On utilise des mesures différentes offert par le module sklearn.metrics pour mesurer la qualité d'un clustering. La mesure est fait en utilisant l'ensemble des étiquettes de tous les sujets obtenus du clustering conjoint. Ils sont comparés collectivement avec l'ensemble des vrais étiquettes.

Figure 4.1 [1] montre les résultats. De gauche à droite, il s'agit des notes pour les méthodes Independent_Joint_KMeans, Pooling_Joint_KMeans, Absolute_Joint_KMeans et Relative_Joint_KMeans. Pour Independent_Joint_KMeans et Relative_Joint_KMeans, la qualité du clustering est encore très bons même quand la rotation est 75 degré.

Notez que Independent_Joint_KMeans et Relative_Joint_KMeans donnent des résultats très proches. Ce n'est pas étonnant. Cela montre que, dans le cas ou les données dans différents sujets sont obtenus par des rotations, Relative_Joint_KMeans est comme Independent_Joint_KMeans à l'intérieur de chaque sujet. La différence entre ces deux méthodes est que, la méthode Relative_Joint_KMeans donne la même étiquette pour les clusters correspondant dans différents sujets, mais ce n'est pas le cas pour Independent_Joint_KMeans en général. Dans ce test, l'initialisation des centroïdes est faite par fusion des données de tous les sujets, donc pour notre données synthétiques, Independent_Joint_KMeans donne aussi la même étiquette pour les clusters correspondant dans différents sujets. Si l'initialisation est faite individuellement pour

1. La mesure ARI (adjusted rand index) mesure la similarité de deux clusterings. La mesure AMI (adjusted mutual information) mesure l'accord de deux clusterings. La mesure de homogénéité mesure la homogénéité des éléments dans les mêmes clusters.

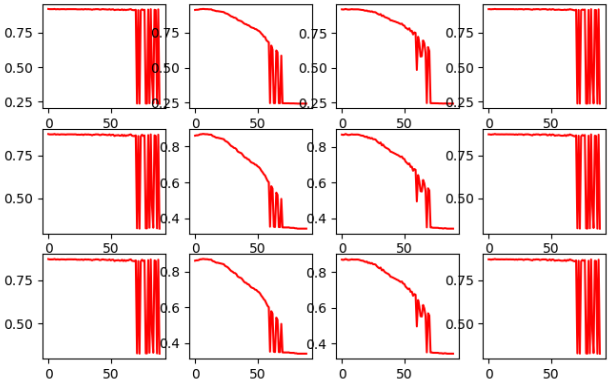


FIGURE 4.1 – La note de qualité du clustering conjoint pour les données synthétiques. L'axe x est l'angle de rotation. De haut en bas, les mesures utilisées sont adjusted rand index, adjusted mutual information et homogeneity. Un clustering avec la note 1 de ces mesures est un clustering parfait.

chaque sujet, les étiquettes sont bien associées d'un sujet à un autre avec notre méthode, mais ce n'est pas le cas pour Independent_Joint_KMeans.

4.2 Parcellisation du cerveau

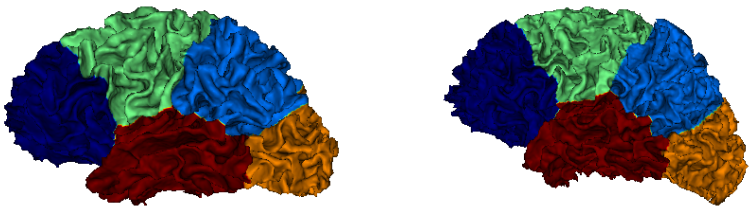
Pour la parcellisation du cerveau, on utilise comme caractéristiques les fonctions propres de l'opérateur de Laplace-Beltrami des 62 surfaces, 7 caractéristiques à chaque fois. Ces fonctions propres sont des descripteurs basse-fréquence d'objets géométriques. Voir [2] pour plus de détails. Pour cet espace de caractéristique, on utilise Independent_Joint_KMeans. Il est combiné avec l'espace de caractéristiques des coordonnées de sommet des maillages qui modélisent la surfaces de cerveau, et on utilise Relative_Joint_KMeans. Ça assure le consensus de la forme de la parcellisation.

Nous avons essayé la parcellisation pour différents nombres k de clusters. Les parcellisations de deux sujets avec k-means conjoint avec $k = 5$ sur 62 sujets sont montrées dans fig. 4.2. Comparé avec Les parcellisations avec k-means indépendant dans fig. 4.3, on obtient des parcellisations qui correspondent entre les sujets.

4.2.1 Évaluation quantitative

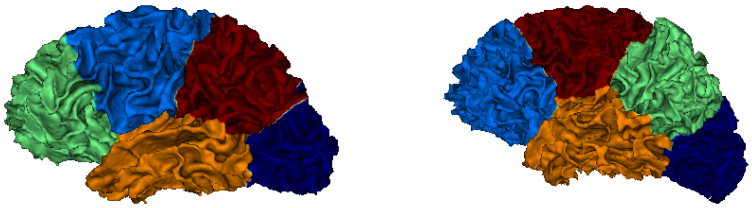
Car nous ne disposons pas de vérité terrain de parcellisation, nous ne pouvons pas l'utiliser pour évaluer la qualité de parcellisation [2]. On a déjà vu que notre méthode de k-means conjoint produit des parcellisations qui ont une similarité de forme. Sous cette hypothèse, on ne mesure pas ce niveau de consensus

2. On peut éventuellement utiliser des informations de parcellisation en lobes, mais ça reste encore très exploratoire.



(a) Sujet 1 (b) Sujet 2

FIGURE 4.2 – Parcellisation avec k-means conjoint



(a) Sujet 1 (b) Sujet 2

FIGURE 4.3 – Parcellisation avec k-means indépendant

ici, qui est un problème non trivial. Ce que nous mesurons, c'est la qualité de parcellisation dans chaque sujet. Pour chaque sujet individuel, on peut faire le k-means. Le résultat est, en général, la meilleur parcellisation que l'on peut obtenir par une méthode de type k-means. Donc on mesure la qualité en en comparant, pour chaque sujet, la parcellisation obtenue par k-means conjoint sur l'ensemble de sujet et celle de k-means sur le seul sujet.

On considère la note la plus bas et la note moyenne des notes pour l'ensemble de sujet, pour $k = 4, 5, 6, 7$. Table 4.1 et Table 4.2 donnent les résultats, qui indiquent les qualités de parcellisation obtenues sont assez bons, en général.

4.3 Profils de connectivité cérébrale

Le profil de connectivité cérébrale contient, pour chaque point de cerveau, un vecteur qui décrit les probabilités de ce point est connecté aux différentes régions de cerveau. Voir [4] pour plus de détails.

Pour l'application à l'étude des réseaux de connectivité cérébrale, Figure 4.5

	ARI	AMI
$k = 4$	0.90	0.88
$k = 5$	0.81	0.80
$k = 6$	0.61	0.67
$k = 7$	0.81	0.83

TABLE 4.1 – Les notes les plus bas.

	ARI	AMI
$k = 4$	0.96	0.94
$k = 5$	0.94	0.93
$k = 6$	0.85	0.86
$k = 7$	0.93	0.92

TABLE 4.2 – Les notes moyennes.

est obtenu par k-means sur les profils de connectivité avec k-means conjoint sur les coordonnées, et Figure 4.4 est par k-means individuel sur le profil de connectivité de chaque sujet. Ce dernier produit des parcellisations de très mauvaise forme, qui ne sont même pas connexes. Avec la contrainte sur les coordonnées, on obtient des résultats plus souhaités :

- Les régions sont cohérentes spatialement dans chaque sujet.
- Les étiquettes sont bien associées d'un sujet à un autre.
- La division obtenue sur place antérieur-postérieure correspond aux connaissances a priori.

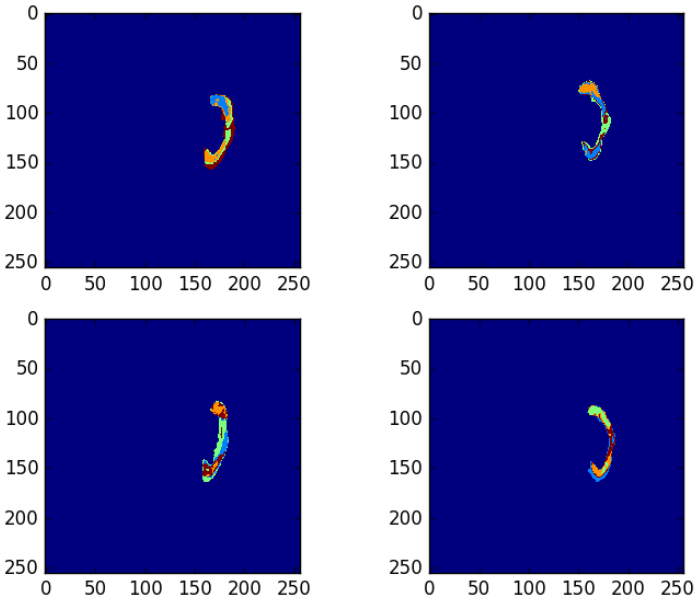


FIGURE 4.4 – Parcellisation avec k-means

4.4 Le minutage

Pour le minutage, on travaille avec 2 espaces de caractéristiques V_1 et V_2 de dimension d_1 et d_2. Le clustering est fait par la combinaison de Independent_Joint_KMeans pour V_1 et Relative_Joint_KMeans pour V_2.

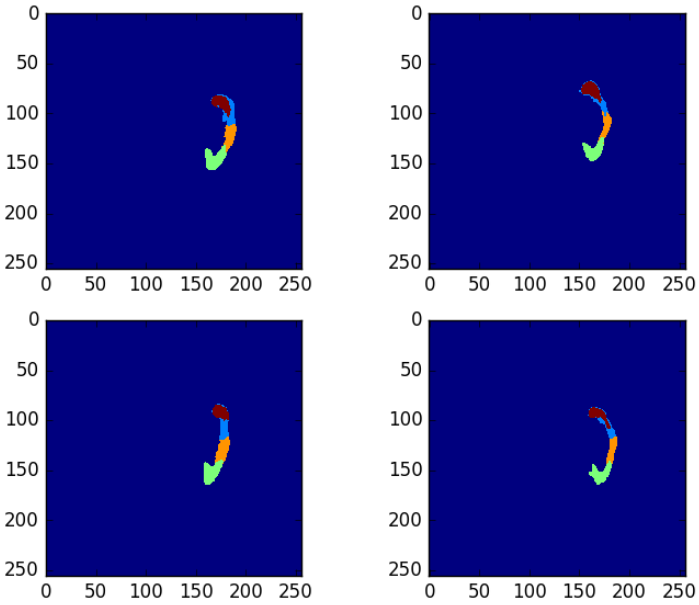


FIGURE 4.5 – Parcellisation avec k-means conjoint

Données de surface de cerveau

Pour les données de surface de cerveau, il y a 62 sujets, chacun a 40962 points. Les dimensions des espaces de caractéristiques est $d_1 = 7$ et $d_2 = 3$. Notre programme obtient le résultat en 5 minutes pour chaque initialisation.

Profils de connectivité cérébrale

Pour les profils de connectivité cérébrale, il y a 10 sujets avec le nombre total de 40962 points. Les dimensions des espaces de caractéristiques est $d_2 = 1917$ et $d_2 = 2$. Notre programme obtient le résultat en 1 minutes pour chaque initialisation.

Chapitre 5

Conclusion

5.1 Contribution

Pendant ce stage, j'ai développé une méthode de clustering conjoint de type k-means, avec la contrainte aux positions relatives des centroïdes. La méthode est étudiée très attentivement à la fois en théorie et en mise en œuvre, pour l'efficacité et la scalabilité. En théorie, la complexité est amélioré de $\tilde{O}(M^2K^4d)$ à $\tilde{O}(MK^2d)$, ici M est le nombre de sujet, K est le nombre de clusters et d est la dimension. En pratique, l'implémentation rend notre méthode encore beaucoup plus puissant quand d est grand. On a aussi implémenté une interface de k-means conjoint qui facilite la combinaison de différentes méthodes de clustering de type k-means. Cela nous permet de travailler plus facilement avec plusieurs espaces de caractéristique.

La méthode est appliquée sur des données synthétiques, ainsi des données sur la surface de cerveau et des profils de la connectivité de cerveau. La combinaison de caractéristique et coordonnée donne des parcellisations avec un consensus de la forme.

5.2 Retour

J'ai cherché un stage lié au machine learning, et ce stage est pertinent. J'ai appris :

- Utilisation avancée de NumPy pour l'efficacité des opérations sur tableaux multidimensionnels.
- Une meilleure connaissance des méthodes de clustering (k-means, spectral, etc.) et l'utilisation de sklearn pour ces méthodes.
- Certaines approches pour le problème de clustering de consensus, à partir d'un ensemble de clustering sur un seul sujet, sur différents sujet ou sur un seul sujet avec différents espace de caractéristiques.

5.3 Travaux futurs

En raison de contraintes temporelles, nous n'avons pas abordé les problèmes suivants :

- **Initialisation** : On fusionne les points des $X^{(m)}$ pour obtenir un seul sujet virtuel X, et utiliser une méthode d'initialisation de k-means sur X. Dans le cas où il y a un seul espace de caractéristiques (single view) et la correspondance des centroïdes dans différent sujets est forte, cela marche bien. Au cas contraire, surtout quand le nombre de clusters est grand, c'est difficile d'avoir une initialisation qui mène à une convergence. Pour les traitements de donnée dans Section 4.2 et Section 4.3, c'est déjà difficile pour $k = 8$.
- **Qualité du clustering conjoint** : Quand la vérité terrain du clustering est disponible, la qualité peut être mesurée facilement. Au cas contraire, pour le clustering sur un seul sujet, il y a des métriques pour mesurer la qualité des clusters obtenus. Pour le cas de multi-sujet, nous avons besoin d'une mesure qui prend en compte le niveau de consensus pour les clusters correspondants dans différents sujets.

Bibliographie

[1] Armand Joulin, Francis Bach et Jean Ponce. "Multi-class cosegmentation". In : *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE. 2012, p. 542–549.

[2] Julien Lefevre, Guillaume Auzias et David Germanaud. "Brain lobes revealed by spectral clustering". In : *Pattern Recognition (ICPR), 2014 22nd International Conference on*. IEEE. 2014, p. 562–567.

[3] S. Race et C. Meyer. "A Flexible Iterative Framework for Consensus Clustering". In : *ArXiv e-prints* (août 2014). arXiv : `1408.0972 [stat.ML]`.

[4] Sylvain Takerkart et al. "Learning from Diffusion-Weighted Magnetic Resonance Images using graph kernels". In : *International Workshop on Graph-Based Representations in Pattern Recognition*. Springer. 2017, p. 39–48.

[5] Wei Tang, Zhengdong Lu et Inderjit S Dhillon. "Clustering with multiple graphs". In : *Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on*. IEEE. 2009, p. 1016–1021.