Modélisation du flux de patients aux urgences liés aux maladies respiratoires par analyse géométrique de séries temporelles

Clément Pealat

N°d'ordre NNT : 2022LYSEI017

THESE de DOCTORAT DE L'UNIVERSITE DE LYON
opérée au sein de :
INSA LYON

Ecole Doctorale N° ED512
InfoMaths

Spécialité/ discipline de doctorat : Mathématiques et applications

Soutenue publiquement le 22/03/2022, par :
Clément PEALAT

Modélisation du flux de patients aux urgences liés aux maladies respiratoires par analyse géométrique de séries temporelles

---

Devant le jury composé de :

Absil, Pierre-Antoine, Professeur, UC Louvain, Rapporteur
Le Bihan, Nicolas, Directeur de Recherche, GIPSA Lab Grenoble INP, Rapporteur
Berthoumieu, Yannick, Professeur des universités, IMS
Poisson Caillault, Émilie, Maitre de Conférence HDR, Université du littoral Côte d'Opale
Cheutet, Vincent, Professeur des universités, DISP INSA-LYON, Directeur de thèse
Bouleux, Guillaume, Maître de Conférence, Université Jean Monnet, Co-directeur de thèse

Département FEDORA – INSA Lyon - Ecoles Doctorales

SIGLE	ECOLE DOCTORALE	NOM ET COORDONNEES DU RESPONSABLE
CHIMIE	**CHIMIE DE LYON** **https://www.edchimie-lyon.fr** Sec. : Renée EL MELHEM Bât. Blaise PASCAL, 3e étage secretariat@edchimie-lyon.fr	**M. Stéphane DANIELE** C2P2-CPE LYON-UMR 5265 Bâtiment F308, BP 2077 43 Boulevard du 11 novembre 1918 69616 Villeurbanne **directeur@edchimie-lyon.fr**
E.E.A.	**ÉLECTRONIQUE, ÉLECTROTECHNIQUE, AUTOMATIQUE** **https://edeea.universite-lyon.fr** Sec. : Stéphanie CAUVIN Bâtiment Direction INSA Lyon Tél : 04.72.43.71.70 secretariat.edeea@insa-lyon.fr	**M. Philippe DELACHARTRE** INSA LYON Laboratoire CREATIS Bâtiment Blaise Pascal, 7 avenue Jean Capelle 69621 Villeurbanne CEDEX Tél : 04.72.43.88.63 **philippe.delachartre@insa-lyon.fr**
E2M2	**ÉVOLUTION, ÉCOSYSTÈME, MICROBIOLOGIE, MODÉLISATION** **http://e2m2.universite-lyon.fr** Sec. : Sylvie ROBERJOT Bât. Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 secretariat.e2m2@univ-lyon1.fr	**M. Philippe NORMAND** Université Claude Bernard Lyon 1 UMR 5557 Lab. d'Ecologie Microbienne Bâtiment Mendel 43, boulevard du 11 Novembre 1918 69 622 Villeurbanne CEDEX **philippe.normand@univ-lyon1.fr**
EDISS	**INTERDISCIPLINAIRE SCIENCES-SANTÉ** **http://ediss.universite-lyon.fr** Sec. : Sylvie ROBERJOT Bât. Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 secretariat.ediss@univ-lyon1.fr	**Mme Sylvie RICARD-BLUM** Institut de Chimie et Biochimie Moléculaires et Supramoléculaires (ICBMS) - UMR 5246 CNRS - Université Lyon 1 Bâtiment Raulin - 2ème étage Nord 43 Boulevard du 11 novembre 1918 69622 Villeurbanne Cedex Tél : +33(0)4 72 44 82 32 **sylvie.ricard-blum@univ-lyon1.fr**
INFOMATHS	**INFORMATIQUE ET MATHÉMATIQUES** **http://edinfomaths.universite-lyon.fr** Sec. : Renée EL MELHEM Bât. Blaise PASCAL, 3e étage Tél : 04.72.43.80.46 infomaths@univ-lyon1.fr	**M. Hamamache KHEDDOUCI** Université Claude Bernard Lyon 1 Bât. Nautibus 43, Boulevard du 11 novembre 1918 69 622 Villeurbanne Cedex France Tél : 04.72.44.83.69 **hamamache.kheddouci@univ-lyon1.fr**
Matériaux	**MATÉRIAUX DE LYON** **http://ed34.universite-lyon.fr** Sec. : Yann DE ORDENANA Tél : 04.72.18.62.44 yann.de-ordenana@ec-lyon.fr	**M. Stéphane BENAYOUN** Ecole Centrale de Lyon Laboratoire LTDS 36 avenue Guy de Collongue 69134 Ecully CEDEX Tél : 04.72.18.64.37 **stephane.benayoun@ec-lyon.fr**
MEGA	**MÉCANIQUE, ÉNERGÉTIQUE, GÉNIE CIVIL, ACOUSTIQUE** **http://edmega.universite-lyon.fr** Sec. : Stéphanie CAUVIN Tél : 04.72.43.71.70 Bâtiment Direction INSA Lyon mega@insa-lyon.fr	**M. Jocelyn BONJOUR** INSA Lyon Laboratoire CETHIL Bâtiment Sadi-Carnot 9, rue de la Physique 69621 Villeurbanne CEDEX **jocelyn.bonjour@insa-lyon.fr**
ScSo	**ScSo*** **https://edsciencessociales.universite-lyon.fr** Sec. : Mélina FAVETON INSA : J.Y. TOUSSAINT Tél : 04.78.69.77.79 melina.faveton@univ-lyon2.fr	**M. Christian MONTES** Université Lumière Lyon 2 86 Rue Pasteur 69365 Lyon CEDEX 07 **christian.montes@univ-lyon2.fr**

*ScSo : Histoire, Géographie, Aménagement, Urbanisme, Archéologie, Science politique, Sociologie, Anthropologie

Table des matières

Liste des notations

— $y = (y_(1), y_(2), ..., y_(l))$ désigne une série temporelle de longueur l.

— $\mathbb{R}^l$ désigne l'espace Euclidien des vecteurs de dimension l.

— $\mathbb{R}^{n \times p}$ désigne l'espace Euclidien des matrices rectangulaires de n lignes et p colonnes.

— Pour une matrice A, A^T désigne sa transposée, I_n désigne la matrice identité de $\mathbb{R}^{n \times n}$.

— M désigne une variété.

— $T_X M$ désigne l'espace tangent à la variété M en X.

— $V_{n,p} = \{A \in \mathbb{R}^{n \times p} : U^T U = I_p\}$ désigne la variété de Stiefel.

— $S(n) = \{X \in \mathbb{R}^n :< X, X >= X^T X = 1\}$ désigne la Sphère unité dans $\mathbb{R}^n$

— $Gr_{n,p}$ désigne la variété de Grassman.

— $P_{n,p}$ désigne la variété des matrices de $\mathbb{R}^{n \times p}$ symétriques définies positives de rang fixe p.

— $SPD(n)$ désigne la variété des matrices de $\mathbb{R}^{n \times p}$ symétriques strictement définies positives.

— log désigne la fonction logarithme classique définie sur les scalaires et les matrices carrés.

— exp désigne la fonction exponentielle classique définie sur les scalaires et les matrices carrés.

— Log désigne la fonction logarithme de Riemann définie sur une variété.

— Exp désigne la fonction exponentielle de Riemann définie sur une variété.

Liste des tableaux

Table des figures

Chapitre 1

Introduction

1.1 Services des urgences et virus hivernal

1.1.1 Contexte

Les hôpitaux sont des services publics essentiels qui doivent toujours être en capacité d'assurer les soins des patients entrants. Leurs services sont nombreux : chirurgie, radiologie, médecine générale, etc... Une partie importante du fonctionnement des hôpitaux est le service d'accueil des urgences (SAU). Ce service est chargé d'accueillir et de prendre en charge les malades. Une fois qu'un patient a été ausculté et enregistré administrativement, il est assigné au service approprié. C'est un dispositif multi-aspects, à la fois technique, matériel et social, complexe qui est donc facilement mis en danger par d'éventuelles perturbations. Il est d'autant plus sensible aux perturbations puisque son activité globale a fortement augmenté ses dernières années. En Fig. 1.1, on observe qu'en 20 ans, le nombre de patients annuels a quasiment doublé. On l'explique par un vieillissement global de la population : en 2000, 16% de la population avait plus de 65 ans contre 20,1% en 2020 (chiffres INSEE). C'est aussi couplé à des moyens limités dans les services hospitaliers. On obtient un système qui fonctionne toujours à la limite de ses capacités et qui est donc impacté par chaque augmentation inhabituelle d'activités.

Quand une perturbation atteint ce système, il met un certain temps à revenir à son état normal. Le temps de récupération est directement lié à l'importance de la perturbation et à la capacité du système à réagir (voir Fig. 1.2). Or, le SAU est la porte d'entrée des hôpitaux. Son bon fonctionnement est primordial pour assurer la qualité des soins des patients. Ainsi, un dérèglement de ce

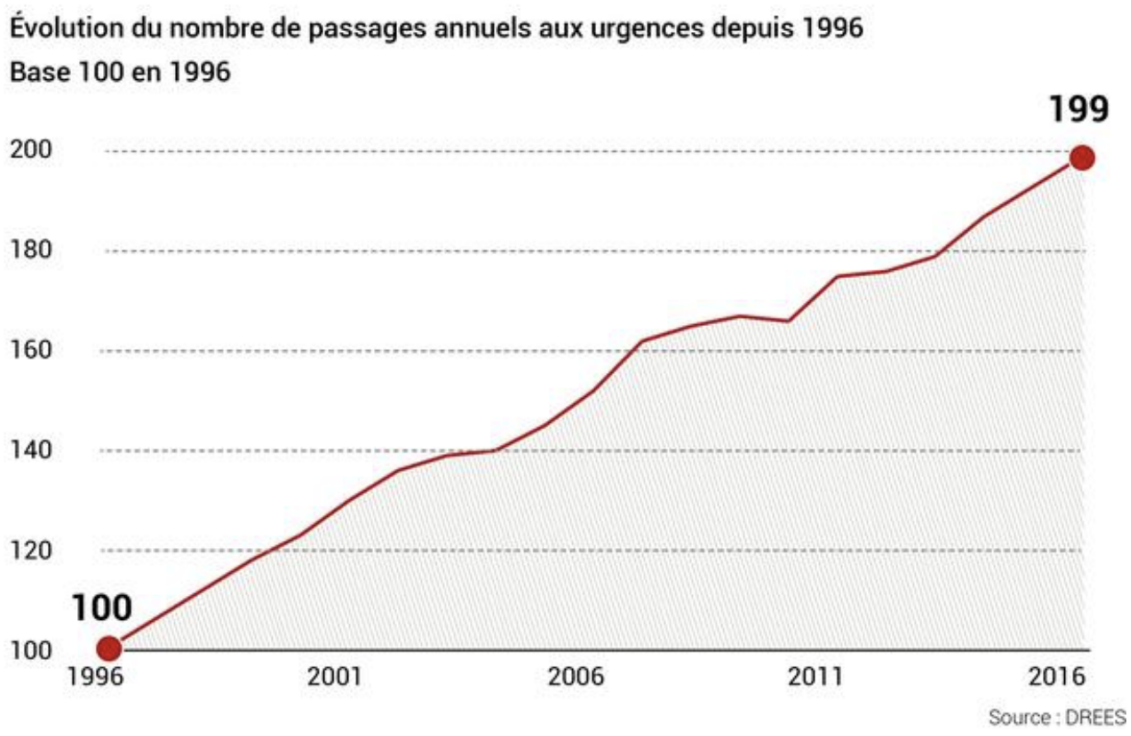


FIGURE 1.1 – Augmentation constante de l'activité des urgences. Chiffres issus de l'étude "Drees, Panorama des établissements de santé 2015".

système se propage très vite à tout l'hôpital [1]. Il est donc nécessaire de maîtriser au mieux ces perturbations.

Les perturbations que subit le SAU peuvent être classées en deux catégories :

— **Les pics d'activités intrinsèques** : dus à une augmentation des temps de séjour des patients déjà admis.

— **Les pics d'activités extrinsèques** : dus à une augmentation du flux entrant de patients au SAU.

Ces pics d'activités, selon leur amplitude, peuvent créer un engorgement dans les urgences de l'hôpital. Il en résulte une qualité de soin qui se dégrade. Dans [4], les auteurs soulignent l'augmentation du temps d'attente et de l'insatisfaction du patient (pas de lits disponibles, personnels moins présents, ...). [5] rejoint ces constatations et met aussi en évidence la tension entraînée par cet engorgement. En effet, la charge de travail trop importante et l'énervement des patients créent des frictions allant jusqu'à des situations de violences. L'efficacité du personnel médical s'en retrouve affectée : les erreurs médicales deviennent plus fréquentes et la formation du nouveau personnel

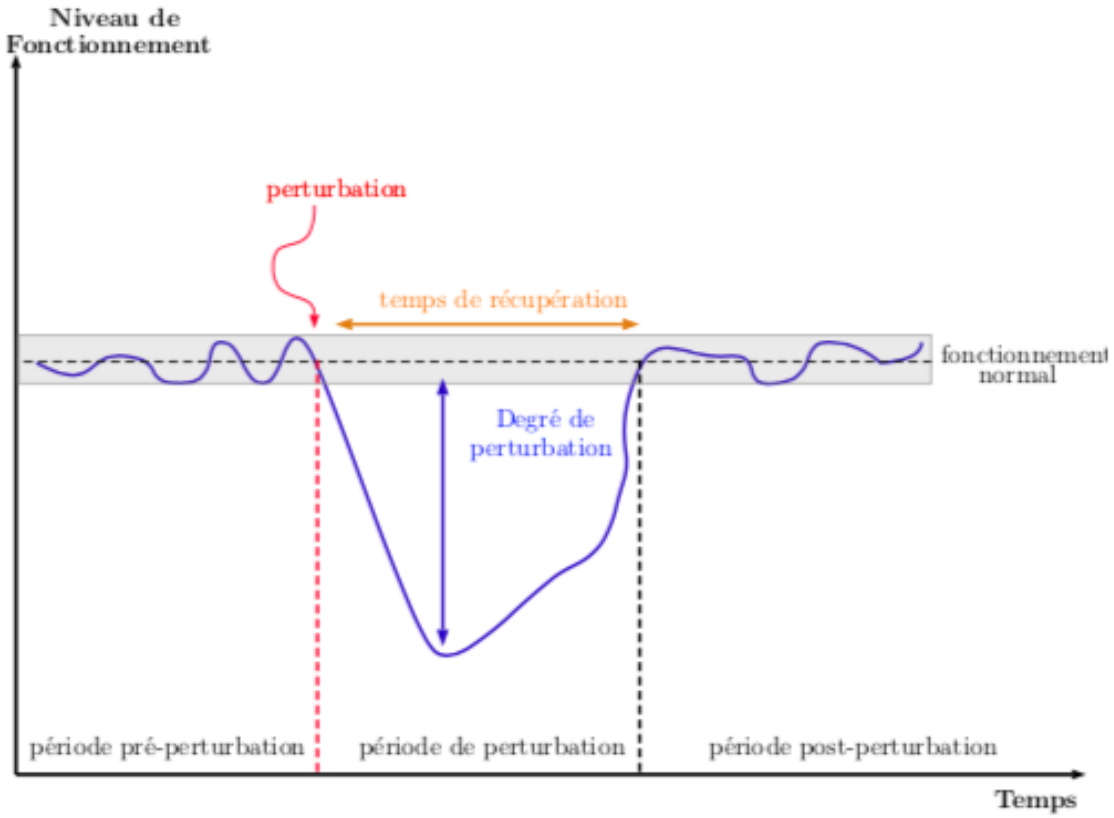


FIGURE 1.2 – Illustration d'une perturbation appliquée à un système sociotechnique d'après [2, 3]. L'arrivée d'une perturbation (comme un afflux de patient) entraîne un degré de perturbation plus ou moins important. Une bonne capacité d'adaptation du système permet de limiter le temps de récupération.

médical (mission essentielle des hôpitaux universitaires) se complique. Ces pics d'activités peuvent être aléatoires dûs à la nature irrégulière de l'activité des hôpitaux. Par exemple, un accident routier aléatoire important peut entraîner un afflux simultané de patients qui engorge le service d'urgence. Ce genre de perturbation est imprévisible, il est donc très compliqué de prendre des mesures contre celles-ci. Les virus saisonniers peuvent aussi créer des pics d'activités. En particulier, chaque hiver, un afflux de patients entraîne des énormes difficultés pour les hôpitaux [5]. Ces pics d'épidémies virales imposent une tension extrêmement forte sur le SAU de façon brutale et imprévisible. Jusqu'à la crise due à la pandémie de Covid-19, **les principaux virus responsables du dysfonctionnement des urgences étaient le virus respiratoire syncytial (VRS) et le virus de la grippe [6, 7] :**

— **Le VRS** touche principalement les enfants même si des études plus récentes [8, 9] montrent

aussi son impact sur les adultes. On estime qu'en France la majorité des enfants de moins de 2 ans vont contracter au moins une fois ce virus. Il est particulièrement grave chez les nourrissons et infecte les poumons et les voies respiratoires. Les symptômes peuvent être différents selon l'âge du patient. En effet, les enfants ont tendance à développer des symptômes cliniques clairs, ceux de la bronchiolite (rhume, toux), tandis que les adultes ont plutôt des symptômes pulmonaires. Pour les urgences pédiatriques, il est facile d'observer la présence du virus : un patient présentant les symptômes de la bronchiolite est porteur du VRS.

— **La grippe** est courante et touche toute la population. Elle dure en général de 3 à 7 jours. Cependant, sur des personnes à la santé fragile, comme les personnes âgées ou les enfants de moins de deux ans, la grippe peut devenir dramatique et impliquer un passage à l'hôpital. En général, ce déplacement à l'hôpital est dû à des complications cardio-respiratoires et non à des symptômes directs de la grippe. Il n'y pas de symptômes évidents pour ce virus car les complications possibles sont nombreuses. Il est donc compliqué pour l'hôpital de détecter précisément la présence du virus : un patient porteur de la grippe ne vient pas forcément avec des symptômes indiquant clairement la présence du virus. Pour limiter l'impact de ce virus, il est possible d'utiliser des vaccins. Cependant, plusieurs souches différentes existent pour cette maladie et il est compliqué d'anticiper quelle souche va apparaître. Ainsi, les vaccins ne sont pas toujours efficaces car ils ne correspondent pas forcément à la bonne souche.

Ces pathologies souvent anodines peuvent être dramatiques, en particulier pour des personnes ayant une santé fragile, ce qui peut amener un passage aux urgences. Chez les plus jeunes, on estime que 30% des nouveaux nés de moins d'1 an sont impactés par ces virus, ce qui entraîne environ 30 000 hospitalisations d'après les bulletins de veille sanitaire.

Pour les urgences pédiatriques, une étude sur l'impact de ces virus saisonniers sur le flux de patients entrants a été réalisée dans la région Rhône-Alpes. Il en ressort que l'activité globale du service de pédiatrie est plus importante durant la période hivernale (de novembre à avril) qu'en période estivale. Plus particulièrement, ce sont les temps de séjour qui augmentent durant cette période. C'est donc une activité extrinsèque changeante (afflux de patients porteurs des virus) qui entraîne un pic d'activité intrinsèque avec des temps de séjour qui explosent.

Contrairement aux pics aléatoires et imprévisibles d'activités, les épidémies saisonnières sont

récurrentes. Ce problème se produisant chaque année, un hôpital a tout intérêt à pouvoir l'anticiper le plus précisément possible. Il est donc important de l'étudier pour en limiter au maximum l'impact.

1.1.2 Bases de données disponibles

Pour réaliser un suivi de ces pics d'épidémies virales, un hôpital a à sa disposition deux bases de données.

Base clinique : Cette base est directement issue du service des urgences. Elle contient les activités journalières du SAU. Quand un patient arrive aux urgences, il est ajouté à cette base de données. La date d'arrivée, le(s) code(s) diagnostique(s), l'âge, etc. sont enregistrés dans celle-ci. Les codes diagnostiques servent à classifier les symptômes présentés par le patient. Un code diagnostique est déterminé par la Classification Internationale des Maladies (CIM). L'ensemble des codes et leur signification est disponible à `https://icd.who.int/browse10/2008/fr`. Un code est composé d'une lettre et de 4 chiffres nous permettant d'entrer plus ou moins dans le détail du symptôme. Par exemple, le code J18.0 peut être décortiqué ainsi : J correspond à un symptôme respiratoire, J18 est pour la pneumonie et J18.0 pour la bronchopneumonie. De plus, un patient peut être enregistré avec plusieurs codes diagnostiques (en général entre 1 et 3).

Base virologique : Un hôpital peut disposer aussi d'une base virologique. Durant le séjour à l'hôpital, un test virologique (PCR) peut être réalisé sur le patient. Ce test détermine le virus dont le patient est atteint. Le résultat est alors ajouté à la base de données virologique. Par exemple, si un patient est testé positif le 10/01/17 à la grippe, une occurrence est alors ajoutée ce jour-là dans la catégorie grippe. C'est donc une base de données non exhaustive, car seulement une partie des patients est testée. C'est aussi une base de données en "différé", le temps que les prélèvements soient analysés puis que les résultats soient ajoutés à cette base.

Dans la Table 1.1, les principales caractéristiques des deux types de bases de données sont récapitulées.

TABLE 1.1 – Principales caractéristiques des deux bases de données utilisées.

Base de données des urgences	**Base de données laboratoire**
Contient les codes CIM des patients des urgences Ajout immédiat du patient Mise à jour tous les jours	Contient les tests positives aux virus hivernaux Ajout plusieurs jours après le test (temps d'analyse) Mise à jour toutes les semaines

1.2 Projet PREDAFLU : Limiter l'impact des virus hivernaux

Pour limiter l'impact des pics d'activités, les systèmes hospitaliers doivent avoir une organisation flexible et dynamique pour pouvoir s'adapter rapidement aux perturbations extérieures. Pour le cas particulier du système des urgences, créer une telle organisation est un enjeu important mais qui reste difficile à réaliser. On retrouve dans la littérature de nombreux travaux [2, 10] à ce sujet. Face à ces augmentations temporaires d'activités, les hôpitaux peuvent agir sur plusieurs aspects :

— Aspect humains : augmentation du personnel médical, du nombre de gardes et de leurs durées

— Aspect matériel : augmentation du nombre de lits pour éviter des nuits sur brancards aux patients

Cette flexibilité du système des urgences ne peut être efficace que si ces mesures sont prises au bon moment. Il est donc nécessaire pour les hôpitaux de se munir d'outils d'aide à la décision efficaces leur permettant d'anticiper au mieux les augmentations d'activités. De nombreux travaux ont déjà été menés dans ce cadre-là. On peut les classer en deux catégories.

Tout d'abord, on retrouve les travaux qui s'occupent du côté "intrinsèque". Le but devient d'optimiser au mieux l'organisation du service d'urgence pour limiter le temps de séjour du patient. Par exemple, [11] réalise une simulation à évènements discrets des services d'urgence pour anticiper les situations d'engorgements et [12] crée une modélisation de type files d'attentes couplée avec un modèle à agents pour limiter les temps d'attente.

On a ensuite les travaux qui s'intéressent au côté "extrinsèque" et donc au flux de patients. Il est possible de directement s'intéresser au nombre de nouvelles admissions par jour. On peut citer [9] qui met en place une méthode de détection de changement de régime basée sur les statistiques

à l'ordre 4 de la série temporelle des excès d'admissions. [13] utilise un réseau de neurones avec en entrée les données climatiques et environnementales et en sortie la série temporelle des admissions quotidiennes, le but étant de prédire les pics d'admissions dus à des phénomènes extérieurs. **L'étude des épidémies hivernales ([14, 8],...) s'inscrit dans cette deuxième catégorie** : on s'intéresse aux flux de patients dus aux virus hivernaux.

En France, la gestion administrative des hôpitaux est réalisée par la région. En effet, chaque région dispose d'une agence administrant les soins (l'agence régionale de la santé (ARS)). Chacune de ces agences décide du déclenchement d'un plan hivernal (augmentation temporaire des moyens humains et matériels) pour lutter contre les épidémies virales. Par exemple, pour le service de pédiatrie du CHU de Saint-Étienne, **l'agence régionale de Rhône-Alpes peut décider d'activer le plan "GO Plan Hivernal" qui prévoit l'ouverture de 7 lits supplémentaires, de renforts infirmiers et, selon la gravité de la situation, de l'ajout d'un médecin de garde aux urgences jusqu'à minuit.** Évidemment, cette agence pourrait mettre en place cette aide très tôt et la finir très tard durant la saison hivernale mais le coût économique en serait très important et donc au final impossible à mettre en place. Il est donc nécessaire de mettre en place des outils pour aider à la décision de la mise en place de ce plan. Actuellement, cette décision est prise en se reposant sur des statistiques nationales et des critères globaux. Or, les différentes études montrent que la présence de virus saisonniers est très localisée. Rien qu'entre les hôpitaux de Grenoble et de Saint-Étienne, il est possible d'avoir plusieurs semaines de décalage. **Il faut donc faire un travail beaucoup plus fin et local pour déterminer la mise en place du plan hivernal. C'est un des objectifs du projet lancé par le ministère de la Santé, le projet PREDAFLU.** Ce projet concerne deux hôpitaux : le CHU de Saint-Étienne et le CHU de Grenoble. Il s'intéresse aux patients fragiles (enfants, personnes âgés,...) atteint de pathologies respiratoires. En alliant virologie, suivi d'activités et traitement du signal, le but est de développer des outils permettant de déterminer la date de déclenchement du plan hivernal. Ce projet est résumé à la Fig. 1.3.

Cependant, plusieurs problèmes empêchent de directement mettre en place cette alerte. Pour l'illustrer, on peut s'appuyer sur la Fig. 1.4. Elle a été obtenue grâce aux données d'admission du service des urgences (base clinique) et de tests virologiques (tests PCR, base virologique) réalisés sur une partie des patients des urgences qui permettent de mettre en lumière a posteriori (le temps

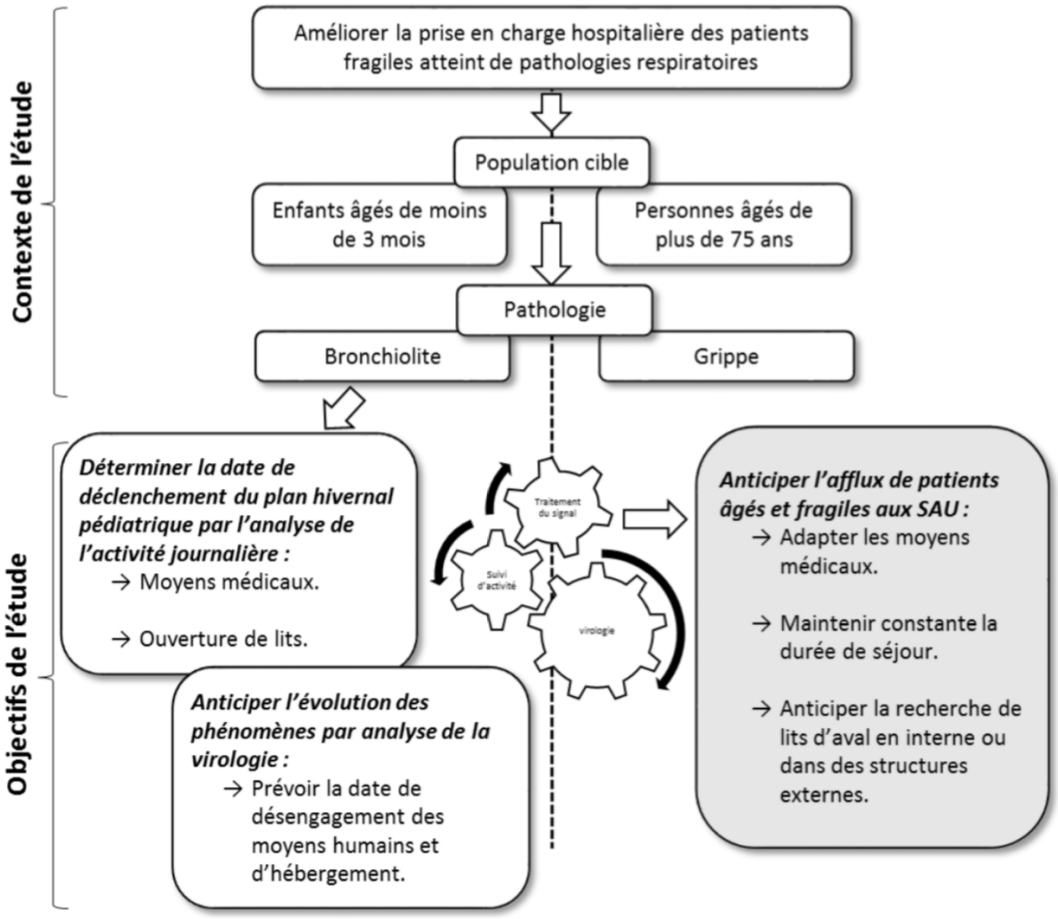


FIGURE 1.3 – Vision globale du projet PREDAFLU d'après [14]. Ce qui nous intéresse principalement ici est l'anticipation de l'afflux de patient et de l'aide à la décision pour la date de déclenchement du plan hivernal.

d'analyse) la présence ou non du virus.

Sur cette figure, on retrouve en rouge le nombre d'arrivée globale aux urgences par jour, ainsi que le nombre de tests positifs par jour pour chacun des deux virus sur la même période de temps. Tout d'abord, il apparaît clairement au vu de la courbe qu'aucune périodicité ou pics d'activité ne peuvent être directement détectés à partir du nombre d'admissions par jour. On ne retrouve aucun des pics d'activités des deux virus VRS et Grippe. Ceci s'explique par le fait que les épidémies hivernales amènent un flux de patient limité en nombre mais présentant des temps de séjour longs. En effet, les complications créées par ces virus sont souvent très lourdes et nécessitent un temps de rétablissement plus long. C'est pour cette raison que les hôpitaux ont des difficultés à gérer l'arrivée de ces virus. On est donc dans le cas d'une activité extrinsèque changeante (apparition de

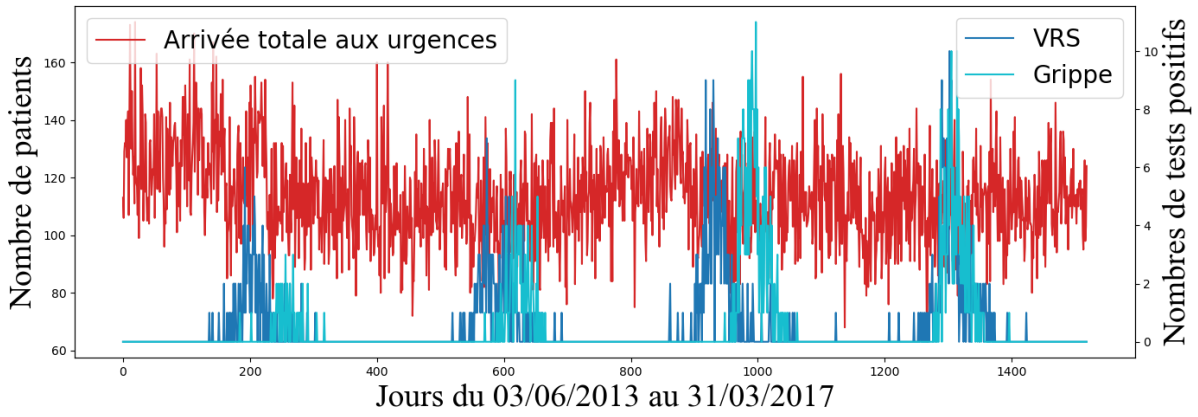


FIGURE 1.4 – Courbes du nombre d'entrée par jour aux urgences du CHU de Saint-Étienne ainsi que du nombre de tests positifs pour chaque virus. On remarque que les pics de présence du virus n'entraînent aucun pic d'activités extrinsèques pour les urgences.

maladies hivernales) qui n'entraîne pas de pics extrinsèques mais bien un pic intrinsèque dû à une explosion des temps de séjour. On remarque ensuite que les pics d'activités des virus ne sont pas parfaitement périodiques ni synchronisés. Il y a une saisonnalité (les virus reviennent chaque hiver) mais le temps entre les pics n'est pas constant, il varie entre environ 300 et 400 jours.

L'objectif de notre thèse est d'affiner notre compréhension du flux de patients dû aux virus hivernaux pour ensuite aider à la mise en place d'outils d'aide à la décision.

1.3 Notre solution scientifique : Clustering de séries temporelles

Pour résumer, nous avons des épidémies virales qui ont une dynamique très changeante variant d'un hôpital à l'autre et d'une année à l'autre. De plus, le nombre d'arrivée par jour n'est pas un bon indicateur pour anticiper une augmentation forte de l'activité, car ces virus impactent principalement la durée de séjour et non le nombre de patients. Il est donc nécessaire de traiter plus en détails le flux de patients arrivant aux urgences.

Pour aider à la compréhension du flux de patients, un hôpital peut disposer de la base de données clinique et de la base de données virologique. Nous proposons de voir chaque élément de ces bases de données comme une série temporelle. Pour la base clinique, une série temporelle par code CIM est construite. Cette série temporelle est caractérisée par le nombre de patient arrivant aux urgences avec ce code chaque jour. Pour ne pas avoir une multitude de séries temporelles avec

très peu d'occurrences, nous avons décidé de garder un niveau de détail d'une lettre et 2 chiffres au niveau du code CIM. Dans la Fig. 1.5, nous illustrons comment ces séries temporelles sont construites.

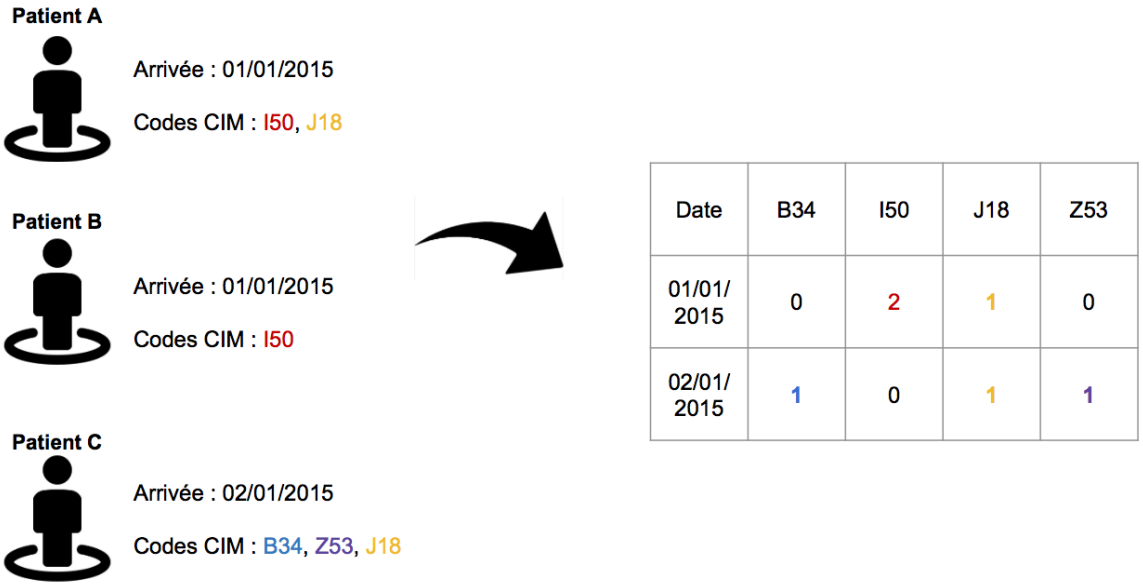


FIGURE 1.5 – Illustration de la construction des séries temporelles à partir de la base de données des urgences. Trois patients sont enregistrés ainsi que leurs codes CIM lors de leurs arrivées aux urgences. Nous en tirons ensuite une série temporelle pour chaque code CIM correspondant aux nombres d'arrivées par jour pour ce code.

Du côté de la base virologique, on obtient une série temporelle par virus. Par exemple, la série temporelle du VRS est déterminée par le nombre de patients testés à une date donnée dont le test est positif au VRS.

Nous voulons mettre en lumière les symptômes marqueurs de la grippe ou du VRS à partir de ces deux bases de données. Le VRS chez les enfants a des symptômes très clairs (bronchiolite), mais chez les adultes, en particulier pour la grippe, les symptômes sont très variables.

On cherche donc à extraire des informations sur un ensemble de séries temporelles sur lesquelles on n'a pas d'information préalable. On est donc dans le cas de machine learning non supervisé. De plus, nous voulons créer 3 groupes dans la base de données : un avec les codes diagnostiques liés au VRS, un avec les codes diagnostiques liés à la grippe, puis un avec le reste des codes diagnostiques. **Notre solution est donc de réaliser du clustering de séries temporelles.** Ce clustering va réunir dans les mêmes groupes les séries temporelles présentant des similitudes.

Dans cette optique, toutes les séries temporelles sont réunies dans une même base de données

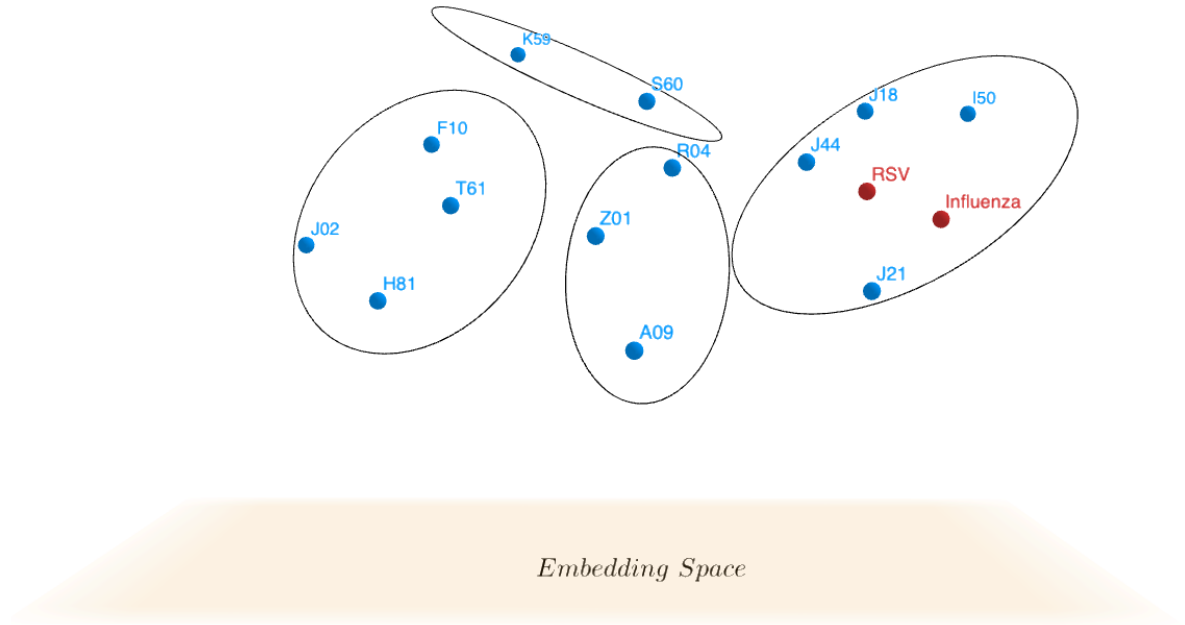


FIGURE 1.6 – Le clustering est appliqué à une base de donnée commune contenant toutes les séries temporelles (tests virologiques et codes CIM). Il permet de souligner les diagnostiques qu'il faut surveiller. Ainsi, et comme simple exemple théorique, les séries temporelles associées aux symptômes J18, I50, J44 et J21 appartiennent au même groupe que les séries temporelles associés aux tests virologiques du VRS et de la grippe. Surveiller ces codes diagnostiques est donc de grande importance.

sur une même période de temps. Enfin, via du clustering (de série temporelle uni-dimensionnelle de même longueur), les séries temporelles de codes CIM présents dans le même groupe que les séries temporelles des virus sont donc considérées comme ayant un comportement similaire. Les codes CIM associés à ces séries temporelles sont donc des codes CIM dus à l'épidémie hivernale. On propose un exemple illustratif dans la Fig. 1.6. Les codes CIM synonymes d'épidémies hivernales dans ce cas sont ceux dont la série temporelle est dans le même groupe que les séries temporelles des virus.

L'avantage de cette méthode est qu'elle permet à chaque hôpital de déterminer ses propres codes à risque. En effet, les codes CIM, bien que basés sur une nomenclature internationale, sont quand même ouverts à l'interprétation ce qui mène à des légères différences de classifications des patients entre les hôpitaux. Avec cette méthode, chaque hôpital peut déterminer ses propres codes à "risques" et s'assurer qu'ils sont bien adaptés à sa situation. De plus, il est possible de reconstruire une série temporelle proche de celle des virus directement depuis les codes CIM. En effet, en prenant la moyenne des séries temporelles des codes CIM présents dans le groupe des virus, nous pouvons

obtenir une série temporelle qui donne en temps réel, depuis la base de données des urgences seulement, la présence du virus. Il est ensuite possible à partir de cette nouvelle série temporelle de développer des méthodes pour détecter des changements de régime symptomatiques d'une arrivée massive de patients due aux virus hivernaux. On peut par exemple penser à des méthode de change point detection ([15], etc.). Ces méthodes peuvent permettre aux hôpitaux de décider plus précisément quand mettre en place le plan hivernal. En particulier, on peut citer [14], travaux précédemment réalisés dans le cadre du projet régional Go plan hivernal. Durant cette thèse, un outil de détection sur les urgences pédiatriques a montré de très bons résultats pour déterminer quand l'épidémie allait exploser. Un travail similaire peut maintenant être réalisé sur les urgences adultes maintenant que les diagnostiques à risques ont été détectés.

Il existe d'autres avantages avec cette approche. Tout d'abord, les patients qui ne sont pas nécessairement porteurs du virus mais qui viendraient sur une même saisonnalité et donc contribueraient aussi à la congestion des urgences seraient aussi surveillés. De plus, déterminer une population susceptible d'avoir un des virus hivernaux a plusieurs bénéfices. Il peut permettre de limiter la propagation du virus au sein même de l'hôpital [16, 17]. En effet, il est possible d'isoler ces patients pour éviter une propagation vers les autres patients mais aussi vers le personnel médical. De plus, les patients à risques et le personnel médical travaillant avec eux peuvent porter des masques. On peut aussi se focaliser l'utilisation de tests PCR sur cette population à risques. Cela permet à l'hôpital de mieux utiliser ses ressources car le taux de tests positifs augmente si la population testée est mieux définie. Enfin, cela peut aider à mettre en place un traitement adapté à la pathologie plus rapidement.

1.4 Synthèse

Cette thèse se déroule dans le cadre d'un projet national PREDAFLU. Le but de ce projet est d'aider les hôpitaux à limiter l'impact des épidémies hivernales dû à deux virus : la grippe et le VRS. Dans ce cadre, il est possible de mettre en place un plan hivernal augmentant temporairement les moyens matériels et humains. Cependant, il est nécessaire de savoir quand mettre en place ce plan. Hors, ces épidémies sont difficilement lisibles avec des symptômes divers chez les adultes et une périodicité non parfaite. Pour avoir une meilleure compréhension en temps réel de ce flux, on

propose de relier la base de données des urgences (symptômes des patients) et la base virologique (tests PCR sur la grippe et le VRS). Ce lien se fait via du clustering de séries temporelles. Les symptômes se retrouvant dans le même groupe qu'un virus sont donc des symptômes à surveiller pour avoir une image de la présence en temps réel du virus dans le système des urgences.

Dans le prochain chapitre, on s'intéresse donc aux différentes méthodes existantes pour réaliser du clustering de séries temporelles uni-dimensionnelles. Ces méthodes nous donnent des premiers résultats sur les données de l'hôpital qui ne sont pas satisfaisants. Nous proposons alors dans le chapitre 3 une représentation des séries temporelles par des matrices de trajectoire pour mi eux prendre en compte la dynamique. Ces matrices de trajectoires peuvent être analysées via différentes géométries non nécessairement planes. Cela nous amène au chapitre 4 qui présente la géométrie Riemannienne qui permet d'étendre la notion de distance sur des géométries non planes. Les différentes pistes abordées sont alors comparées dans le chapitre 5 sur de multiples bases de données dont le résultat de clustering attendu est connu. Dans le chapitre 6, la méthode la plus pertinente au vu des résultats du chapitre 5 est appliquée au cas hospitalier.

Chapitre 2

1-D time series clustering : Premières pistes

2.1 Introduction

Une des méthodes non supervisées les plus connue de machine learning est le clustering [18]. Ce type de méthode permet de donner des indications sur la structure des données. En effet, le but est de déterminer depuis un ensemble d'éléments non labélisés (ces éléments sont tous mélangés sans indications) des groupes d'éléments ayant des similarités. Pour cela, un algorithme de clustering va chercher à déterminer des groupes d'éléments pour lesquels la similarité intra-groupe est maximisée et la similarité inter-groupe est minimale. Il y a plusieurs possibilités pour définir cette similarité. Par exemple, on peut considérer la similarité intra-groupe simplement comme la moyenne de la distance entre chaque élément 2 à 2. Le clustering est une méthode primordiale pour toute base de données non labélisée, quelque soit la nature de cette base : binaire, spatiale, image, etc... En général, ces données sont de nature statique dans le sens où les caractéristiques des observations ne dépendent pas du temps. Sur ce type de base de données, la littérature est vaste et de multiples méthodes ont été développées. Cependant, ce type de méthode n'est pas forcément approprié pour les données qui nous intéressent. Dans le cadre de notre clustering sur l'hôpital, nous travaillons avec des séries temporelles uni-dimensionnelles (1-D). Ce type de structure est courant et se retrouve dans de nombreux domaines comme la météorologie [19], l'industrie [20], les soins de santé [9, 21], etc... Les données de séries temporelles sont souvent larges et présentent un ordonnancement temporel. Ce type de données se démarque donc du reste par leur dynamique temporelle. Cette

dynamique lie les caractéristiques de la série temporelle entre elles. Une série temporelle présente donc souvent des corrélations intéressantes entre ses caractéristiques.

Évaluation d'un résultat de clustering : Une fois que les résultats de clustering ont été déterminés, il est nécessaire d'avoir des indicateurs pour juger de leur qualité. On peut les regrouper en 2 catégories : internes et externes.

— Un indicateur interne va juger de la qualité du clustering vis à vis d'un objectif d'optimisation sans information extérieure. L'un des plus utilisés d'entre eux est le Silhouette Score [22]. Pour chaque k groupe dans le résultat de clustering noté C, on calcule la moyenne de la distance 2 à 2 entre tous les éléments de ce groupe $(a_i, i = 1..k)$ puis la moyenne de la distance entre chaque élément de ce groupe et son plus proche voisin n'appartenant pas au même groupe $(b_i, i = 1..k)$. Le Silhouette Score est ensuite défini par :

$$Sil(C) = \frac{1}{k} \sum_{i=1}^{k} \frac{b_i - a_i}{max(a_i, b_i)} \tag{2.1}$$

Avec cette équation, on remarque que le Silhouette Score est borné entre -1 pour un mauvais résultat de clustering et 1 pour un bon. En effet, pour que le Silhouette Score soit proche de 1 (resp. -1), il faut que la distance moyenne intra-groupe (resp. inter-groupe) soit négligeable face à la distance moyenne inter-groupe (resp. intra-groupe).

— Un indicateur externe peut être utilisé quand les labels réels sont connus. Le "meilleur" résultat de clustering possible est alors connu. Ces ensembles d'éléments où le résultat de clustering attendu est connu sont très utiles pour évaluer l'efficacité d'une méthode de clustering. Un indicateur externe prend donc en entrée le résultat de clustering obtenu (C_1) et le résultat de clustering attendu (C_2). Il calcult ensuite un score permettant d'évaluer la proximité entre ces deux éléments. Le v-measure score [23] par exemple est un indicateur externe basé sur l'entropie. Ce score réalise une moyenne géométrique entre la Complétude et l'Homogénéité entre $C1$ et $C2$. L'Homogénéité est bornée entre 0 et 1, 1 étant obtenu quand, pour chaque groupe de C_1, tous ses éléments appartiennent au même groupe dans C_2. De même, par symétrie, la Complétude est aussi bornée entre 0 et 1 et atteint son maximum quand, pour chaque groupe de C_2, tous ses éléments appartiennent au même groupe dans C_1. L'Homogénéité ou la Complétude seule peut être trompeur. Par exemple, si dans C_1, tous

les éléments appartiennent à un groupe unique, alors la Complétude est de 1. C'est pourquoi le v-measure score est plus pertinent en utilisant une moyenne géométrique entre ces deux scores. Une autre possibilité pour estimer la qualité du clustering est l'indicateur Adjusted Rad Index Score (ARI) [24]. C'est une extension du Rand Index Score (RI). ARI prend des valeur entre -1 et 1 (avec correspondant à un 1 un clustering parfait) et assure qu'un résultat aléatoire a une valeur de 0. Pour deux résultats de clustering C_1, C_2, le score ARI est défini par :

$$ARI(C_1, C_2) = \frac{RI - \mathbb{E}[RI]}{\max(RI) - \mathbb{E}[RI]} \tag{2.2}$$

Un résultat de clustering est déterminé en deux étapes :

— **Choix d'une distance :** Il est possible de considérer un ensemble d'éléments de différentes manières, chacune entraînant des distances différentes qui mettent en lumière des éléments spécifiques. Entre deux éléments, une distance importance (resp. faible) est synonyme de faible (resp. forte) similarité.

— **Application d'un algorithme de clustering :** Ils existent différentes méthodes pour déterminer les groupes dans l'ensemble des éléments.

Des algorithmes et des distances couramment utilisés dans la littérature sont présentés dans la suite de cette partie.

2.2 Mesure de similarité/distance classique

Plusieurs mesures de similarité sont applicables sur les séries temporelles. Elles prennent plus ou moins compte de la particularité dynamique des séries temporelles. Ce choix de distance a un impact important sur la performance du clustering.

2.2.1 Distance Euclidienne

L'une des distances les plus classiques, utilisée dans un large éventail d'applications, est la distance Euclidienne [25]. Soit deux séries temporelles de longueur l, $y_1 = (y_1(1), y_1(2), ..., y_1(l))$ et $y_2 = (y_2(1), y_2(2), ..., y_2(l))$, la distance Euclidienne va simplement les considérer comme des

vecteurs de $\mathbb{R}^l$ et est définie par :

$$d(y_1, y_2) = \sqrt{\sum_{i=1}^{l} (y_1(i) - y_2(i))^2} \tag{2.3}$$

C'est la distance classique sur $\mathbb{R}^l$. Cette distance a plusieurs avantages. Tout d'abord, elle est rapide à calculer. En effet, on a l répétitions d'opérations, nous avons donc une complexité en $\mathcal{O}(l)$. De plus, elle ne nécessite aucun travail particulier sur les séries temporelles avant d'être appliquée. Cependant, on peut remarquer qu'avec cette distance, on ne fait aucune différence entre des éléments de $\mathbb{R}^l$ issus de données statiques ou des éléments issus de données dynamiques. On ne prend donc pas du tout en compte la temporalité de la série temporelle.

2.2.2 Dynamic Time Warping (DTW)

Dynamic Time Warping (DTW) est une distance dérivée de la distance Euclidienne qui prend mieux en compte la dynamique de la série temporelle. Cette distance est actuellement une des distances les plus utilisées dans la littérature dans le cas des séries temporelles [26, 25, 27]. Avec la distance Euclidienne, on mesure directement la différence entre les points 2 à 2. DTW propose de réaliser tout d'abord un alignement entre les deux séries temporelles ayant pour but est de minimiser la distance Euclidienne. Pour deux séries temporelles y_1 et y_2 de longueur l_1, l_2, on commence par aligner les derniers éléments de chaque séries temporelles $D(y_1(l), y_1(l)) = |y_1(l_1) - y_2(l_2)|$. On remonte ensuite récursivement le long des séries temporelles avec :

$$D(y_1(i), y_1(j)) = |y_1(i) - y_2(j)| + min \begin{cases} d(y_1(i-1), y_2(j-1)), \\ d(y_1(i), y_2(j-1)), \\ d(y_1(i-1), y_2(j)) \end{cases} \tag{2.4}$$

On obtient ainsi l'alignement optimal et la distance DTW est donnée par $d_{DTW} = D(y_1(0), y_1(0))$. Une illustration de la différence entre l'alignement classique Euclidien et celui obtenu en utilisant DTW est proposée en Fig. 2.1.

La complexité de calcul est plus élevée avec cette distance que avec la distance Euclidienne à cause de la recherche de l'alignement optimal. Pour deux séries temporelles de longueurs l_1, l_2,

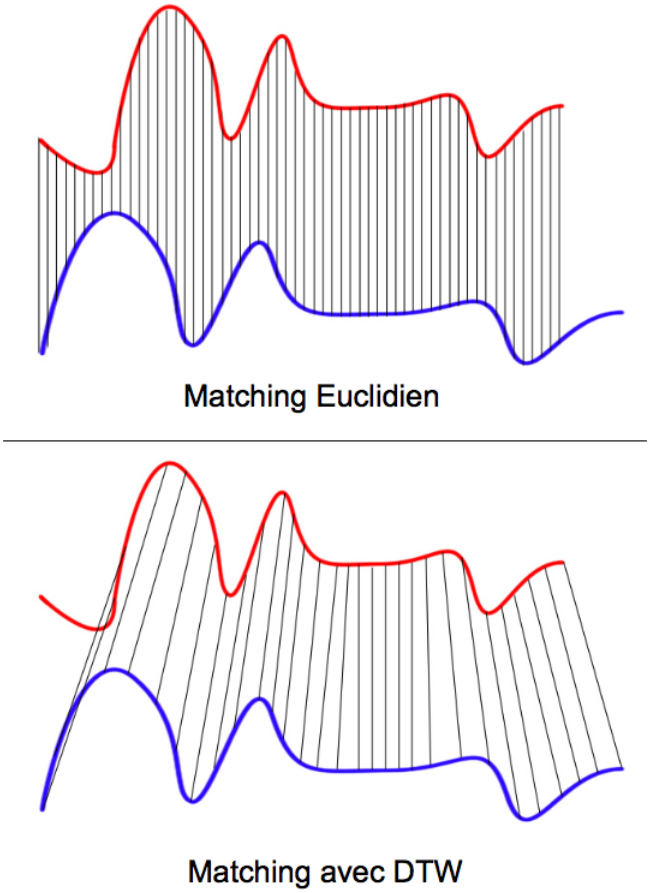


FIGURE 2.1 – Différence entre la distance Euclidienne et DTW. Avec la distance Euclidienne, les caractéristiques des deux séries temporelles sont directement associées 2 à 2. Avec DTW, les caractéristiques peuvent être associées à une caractéristique plus proche (dans le sens Euclidien). Figure tirée de `https://paperswithcode.com/method/dtw`.

la complexité est de $\mathcal{O}(l_1 \times l_2)$. Cette distance a aussi l'avantage de pouvoir mesurer des séries temporelles de longueurs différentes. De plus, elle est particulièrement bien adaptée pour comparer des séries temporelles de "vitesses" différentes. Par exemple, pour de la reconnaissance vocale, la distance DTW est bien adaptée [28] : deux séries temporelles, issues de la même phrase mais dite à des vitesses différentes, seront considérées comme similaires par la distance DTW grâce à l'alignement, contrairement à la distance Euclidienne.

2.2.3 Autres distances

Les deux distances précédentes sont les plus couramment utilisées pour des séries temporelles. D'autres distances existent dans la littérature. On peut citer :

— La distance de Mikowski : Cette distance est une généralisation de la distance Euclidienne définie par :

$$d(y_1, y_2) = \sqrt[q]{\sum_{i=1}^{l}(y_1(i) - y_2(i))^q} \tag{2.5}$$

Différentes valeurs de q sont possibles. On retrouve ainsi la distance Euclidienne pour $q = 2$, la distance de Manhattan pour $q = 1$ et la norme infini pour $q = +\infty$. La distance de Manhattan est particulièrement adaptée pour des données en quadrillage (type déplacement dans une ville) et d'après [29] aussi pour des données de grande dimension.

— Pour les séries temporelles courtes, [30] propose la "Short Time Series" distance (STS distance) définie comme la somme des différences au carrés des pentes des séries temporelles x,y de longueur l :

$$d_{STS} = \sqrt{\sum_{k=1}^{l-1}(\frac{y_{k+1} - y_k}{t_{k+1} - t_k} - \frac{x_{k+1} - x_k}{t_{k+1} - t_k})^2}$$

Dans [30], les auteurs montrent que dans le cas de séries temporelles courtes et où le pas de temps peut varier selon la série temporelle, cette distance est plus efficace que la distance Euclidienne. C'est, par exemple, souvent le cas en biologie.

— Certaines distances sont définies à partir de la corrélation entre les deux séries temporelles, proposée par exemple dans [31]. Elle est définie par $d_c = 2(1 - c)$ avec c, le facteur de corrélation de Pearson, défini pour deux séries temporelles x,y de longueur l par :

$$c = \frac{\sum_{k=1}^{l}(x_k - \mu_x)(y_k - \mu_y)}{S_x S_y}$$

avec μ la valeur moyenne d'une série temporelle et $S_x = \sqrt{\sum_{k=1}^{l}(x_i - \mu_x)^2}$. Ce facteur de Pearson prend des valeurs entre -1 et 1 avec 0 indiquant une relation nulle entre x et y, une valeur positive indiquant que les deux séries temporelles varient globalement ensemble dans le même sens et une valeur négative qu'elles varient dans des sens opposés. Ainsi, avec cette distance, plus la corrélation au sens de Pearson est grande, plus les séries temporelles seront considérées comme proches.

2.3 Algorithmes de clustering

Une fois la distance choisie, il est possible d'appliquer un algorithme de clustering. Les différents algorithmes de clustering se caractérisent par des stratégies différentes pour maximiser la similarité intra-groupe et pour minimiser la similarité inter-groupe. Il existe trois grandes familles d'algorithmes de clustering : algorithmes par partition, algorithmes hiérarchiques et algorithmes par densité.

2.3.1 K-means

Le K-means est un des algorithmes de clustering les plus connu. Il fait partie de la famille des **algorithmes par partition**. Pour une famille de n séries temporelles $\{y_i, i = 1, ..., n\}$, l'algorithme initialise au hasard k "centres" parmi les n séries temporelles. Ensuite, chaque série temporelle est associée à son plus proche centre. L'algorithme crée ainsi k groupes. Puis, chaque centre est recalculé comme la moyenne du groupe qu'il a défini. Les itérations s'enchaînent jusqu'à stabilisation [32]. Une illustration du fonctionnement de cet algorithme est proposée dans la Fig. 2.2. On peut remarquer qu'ici, chaque série temporelle appartient à un unique groupe. K-means est donc un algorithme de type "hard" clustering. De plus, cet algorithme impose de définir une moyenne en plus d'une distance, ce qui peut être compliqué et long selon la nature de l'ensemble des données sur lequel on applique K-means. Hors calcul de la moyenne, l'algorithme a une complexité en $\mathcal{O}(n^2)$ avec n le nombre d'éléments à clusteriser.

Dans la même famille des algorithmes par partition, on retrouve des dérivés de cet algorithme comme le K-medoids [33] (au lieu de prendre la moyenne d'un groupe, on prend l'élément de l'ensemble des données le plus proche de la moyenne d'un groupe) ou le Fuzzy C-means [34] (une série temporelle peut appartenir à plusieurs groupes, un "soft" algorithme). Cette famille d'algorithmes est particulièrement efficace pour trouver des groupes de forme sphérique dans la base de données. Ce type d'algorithme nécessite de définir à l'avance k le nombre de groupes que l'on souhaite. Pour cela, on peut utiliser le Silhouette Score [22] que l'on a défini précédemment. L'algorithme de clustering par partition est lancé pour plusieurs valeurs de k. On garde ensuite la valeur de k qui maximise le Silhouette Score, signe d'un clustering de qualité.

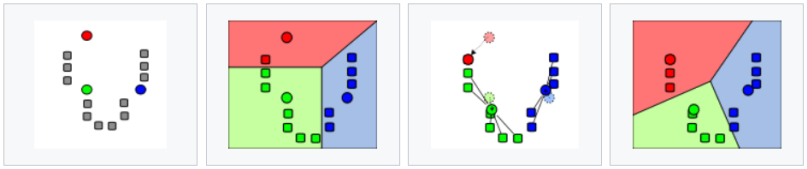


FIGURE 2.2 – Illustration de l'algorithme K-means pour $k = 3$ groupes. On commence avec 3 points choisis au hasard comme "centres" (a). On obtient ainsi 3 premiers groupes (b). Puis le centre de chacun de ces groupes est re-calculé (c) suivi par la composition des groupes (d). Cette itération continue jusqu'à convergence.

2.3.2 Clustering Hiérarchique

Un algorithme de clustering Hiérarchique [32] réunit les éléments de la base de données dans un dendrogramme de groupe. Cet algorithme peut être de deux types : Agglomerative (ou "bottom-up") ou Divisive (ou "top-down").

La méthode Agglomerative commence par considérer chaque élément de la base de données comme étant un groupe à part entière. Ensuite, les groupes les plus proches fusionnent, créant des groupes de plus en plus larges jusqu'à avoir un unique groupe. Le résultat final est un dendrogramme où chaque fusion correspond à un noeud. Plusieurs méthodes existent pour définir la distance entre les groupes et ainsi définir le plus proche voisin. La méthode de la moyenne définit la distance entre deux groupes comme la moyenne des distances entre chaque paire d'éléments (avec un élément appartenant au premier groupe et un élément appartenant au deuxième groupe). La méthode de Ward [35] prend en compte la variance entre deux groupes. On peut aussi citer la méthode "single" (resp. "complete") où la distance entre deux groupes est la distance minimale (resp. maximale) entre une paire d'éléments venant des deux groupes.

La méthode Divisive fonctionne à l'inverse : tous les éléments appartiennent à un même groupe puis les groupes sont divisés en plus petits groupes jusqu'à atteindre un groupe par élément. Sur la Fig. 2.3, on illustre le fonctionnement des deux types d'algorithmes hiérarchiques.

Dans la littérature, la méthode Agglomerative est la plus populaire des deux [36].

Cette fois encore, il est nécessaire de connaître à l'avance le nombre de groupes voulu pour le clustering. De la même manière que pour K-means, il est possible d'utiliser le Silhouette Score. Cet algorithme est aussi plutôt lent avec une complexité en $\mathcal{O}(n^3)$, avec n le nombre d'éléments à clusteriser.

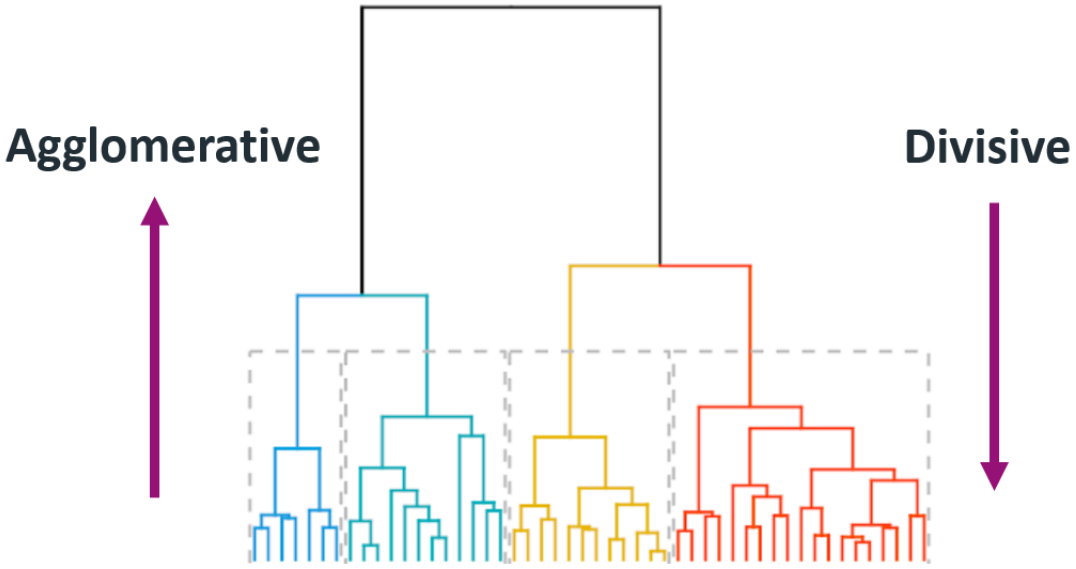


FIGURE 2.3 – Exemple de dendrogramme pouvant être obtenu avec l'algorithme de clustering Hiérarchique. Si l'algorithme est de type Agglomerative, on commence avec chaque élément qui est dans son propre groupe (partie basse du dendrogramme), puis on réunit les groupes les plus proches jusqu'à obtenir un seul groupe (partie haute du dendrogramme). Si l'algorithme est de type Divisive, c'est l'inverse.

2.3.3 Algorithmes de clustering basés sur la densité

L'idée générale des algorithmes de clustering basés sur la densité est d'avoir un groupe d'éléments qui continue à s'étendre tant que la densité d'éléments dans le voisinage est suffisamment importante. Un groupe d'éléments obtenu avec ce type d'algorithme correspond donc à une région de haute densité entourée d'une région à faible densité [37].

Un des algorithmes de densité les plus connus est le "Density-Based Spatial Clustering of Applications with Noise" (DBSCAN) [38, 37]. Il prend en entrée deux paramètres : un seuil de distance ϵ et un nombre de voisins m. À partir de ces deux paramètres, on définit la notion de "atteignable par densité". On dit qu'un point A de la base de données est atteignable par densité depuis un autre point B s'il existe une suite de points dans la base de données telle que l'on puisse définir un chemin de A vers B et telle que la distance entre deux éléments successifs de cette suite soit inférieure à ϵ. Ensuite, l'algorithme DBSCAN classe en trois catégories les éléments de la base de données :

— Point Central : Si un point de la base de données a, dans son ϵ-voisinage, au moins m autres

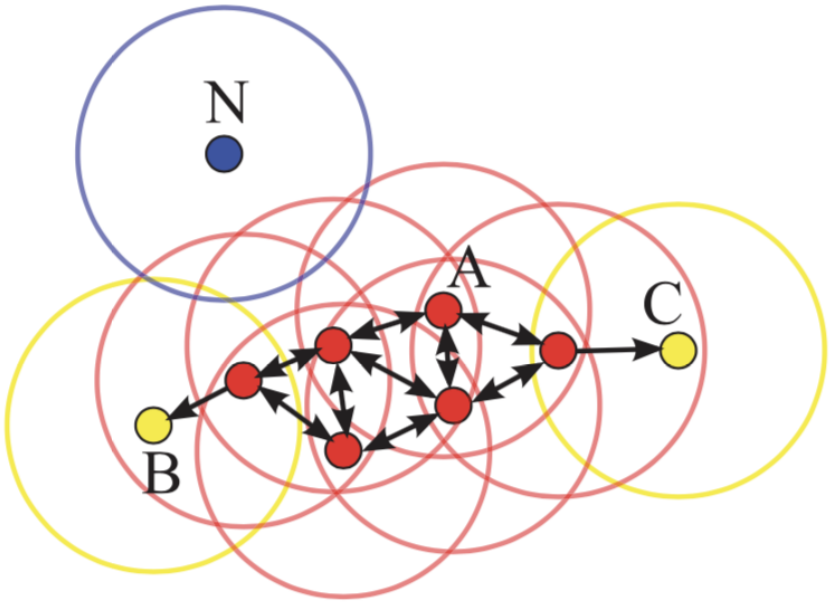


FIGURE 2.4 – Illustration du clustering par DBSCAN. Les points sont des éléments à clusteriser. Autour de chacun de ses points, des cercles de diamètre ϵ ont été tracés. Le point en bleu est classé en bruit car il n'a pas suffisamment de voisins dans son voisinage ϵ et n'est pas atteignable par densité par un point central. De même, rouge correspond aux points centraux et jaune aux points à la frontière. Ainsi, dans cet exemple, A, B et C sont dans le même groupe.

points de la base de données, c'est un point central ;

— Point à la frontière : Un point est un point à la frontière si ce n'est pas un point central et si il est atteignable par densité par un point central ;

— Point bruit : Un point est considéré comme étant du bruit si il n'est ni un point central ni un point à la frontière.

Un groupe est alors défini ainsi : il a au moins un point central et n'importe quel point est atteignable par densité par n'importe quel autre. Une illustration tirée de [38] de la construction des groupes par cet algorithme est proposée en Fig. 2.4.

Le choix du seuil ϵ a donc un impact important sur le résultat de clustering. Choisir la bonne valeur est une question compliquée. De plus, avoir un seuil constant à travers toute la base de données peut être problématique. En effet, si la base de données est composée de deux régions de densités différentes, avec un ϵ constant, une des régions risque d'être considérée comme du bruit. Pour ces deux raisons, une prolongation de cet algorithme a été créée : "Density-Based Clustering Based on Hierarchical Density Estimates" (HDBSCAN) [39]. Cet algorithme calcule

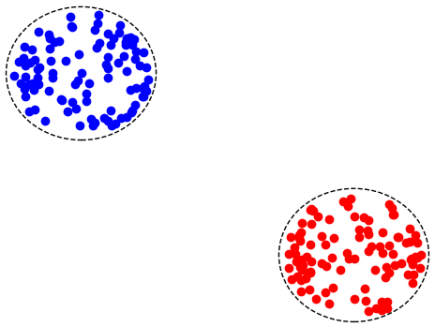
les résultats obtenus avec DBSCAN pour des valeurs d'ϵ entre $]0; +\infty[$. Pour une valeur suffisamment large d'ϵ, tous les éléments seront dans un même groupe. Ensuite, au fur et à mesure qu'ϵ diminue, la taille du groupe va diminuer jusqu'à se diviser en deux groupes. On continue jusqu'à ce que tous les éléments soient considérés comme du bruit. On a donc un moment de vie et un moment de mort associé à chaque groupe en fonction d'ϵ. Le résultat de clustering correspond alors aux groupes qui ont la durée de vie la plus longue. Cet algorithme permet donc de déterminer des groupes même si ils existent des différences de densité et il n'y a pas besoin de choisir de valeur pour ϵ. En terme de complexité, une étude est proposée sur leur site web : `https://hdbscan.readthedocs.io/en/latest/performance_and_scalability.html`. La complexité est au moins bornée par $\mathcal{O}(n^2)$, avec n le nombre d'éléments à clusteriser.

Contrairement aux algorithmes de clustering par partition (comme K-means) qui se focalisent sur des formes circulaires dans la base de données, les algorithmes de densité sont plus versatiles. On propose de le mettre en évidence dans la Fig. 2.5. On a créé deux bases de données de 200 éléments de $\mathbb{R}^2$ répartis en 2 groupes. Dans la première base de données, les deux groupes sont sphériques, et dans la deuxième base, ils sont issus de deux droites. HDBSCAN et K-means (avec $k = 2$) sont appliqués aux deux bases de données. On représente les résultats en 2D avec les points coloriés en bleu ou rouge selon le résultat de clustering. De plus, pour K-means, on rajoute en noir les centres de chacun des groupes déterminés par l'algorithme. HDBSCAN récupère parfaitement les deux groupes dans les deux cas contrairement à K-means. Dans le cas des deux droites, cette figure met en évidence que quelque soit les centres des groupes, les éléments ne pourront pas être correctement séparés.

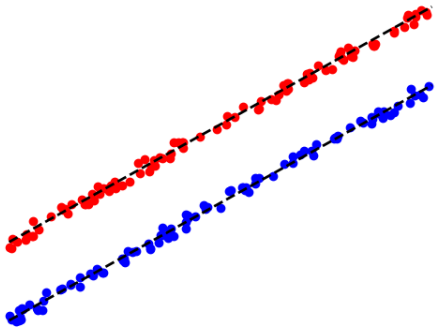
En Table 2.1, les caractéristiques des différents algorithmes de clustering sont récapitulées.

2.4 UMAP : Uniform Manifold Approximation and Projection

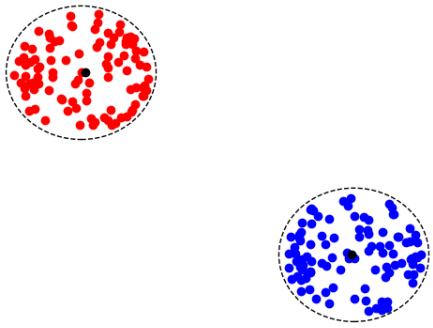
Pour améliorer des algorithmes de clustering, il est possible d'utiliser avant un algorithme de réduction de dimension (et après avoir défini la mesure de similarité). En trop grande dimension, un algorithme va souvent manquer d'éléments pour être précis. C'est ce qu'on appelle le "fléau de la dimension" [40]. En réduisant la dimension, on augmente la densité des données.



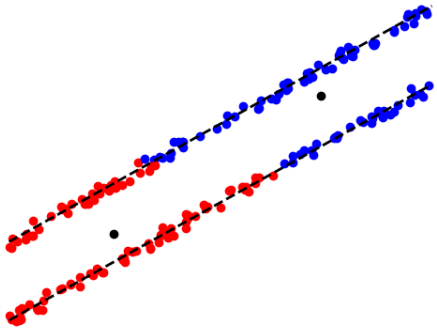
(a) Résultat de clustering par HDBSCAN dans le cas de groupes sphériques.



(b) Résultat de clustering par HDBSCAN dans le cas de groupes issus de droite.



(c) Résultat de clustering par K-means dans le cas de groupes sphériques.



(d) Résultat de clustering par K-means dans le cas de groupes issus de droite.

FIGURE 2.5 – Résultats de clustering pour deux algorithmes (HDBSCAN et K-means) et deux formes de groupes différents (2 cercles ou 2 droites). On remarque que K-means n'est pas du tout approprié au cas des droites comme le met en évidence le résultat de clustering et le centre de chacun des groupes déterminé par K-means.

TABLE 2.1 – Présentation rapide des différents algorithmes de clustering. Les trois grandes familles d'algorithmes de clustering sont rappelées. Pour chacune d'elle, on donne quelques exemples d'algorithmes et on précise sur quel type de données les utiliser et la complexité algorithmique.

Type d'algorithmes	Algorithmes par partition	Algorithmes hiérarchiques	Algorithmes par densité
Exemples	K-means, K-medoids, Fuzzy C-means	Hiér. Agglomerative, Hiér. Divise	DBSCAN, HDBSCAN
Type de données	Groupes sphériques	Nombre d'éléments limités	N'importe quelle forme
Complexité	$\mathcal{O}(n^2)$	$\mathcal{O}(n^3)$	$< \mathcal{O}(n^2)$

En Fig. 2.6, une hiérarchie rapide des différents algorithmes de réduction de dimension est présentée. Ces algorithmes peuvent être séparés en deux types. Tout d'abord, des algorithmes de réduction de dimension proposent de détecter les caractéristiques qui présentent de la redondance par rapport aux autres. Ces algorithmes gardent uniquement les caractéristiques les plus pertinentes. Il n'y a donc pas de transformation des données. On peut citer Forêt aléatoire, Forward Selection et Backward Elimination dans ce domaine. Une autre possibilité est de transformer les données en déterminant un nouveau jeu de caractéristiques. Le but est donc de déterminer un jeu de caractéristiques réduit qui garde au mieux l'information originale. On peut le réaliser de manière linéaire, avec par exemple Principal Component Analysis (PCA) [41] ou Singular Value Decomposition (SVD) [42]. Un autre exemple de réduction de dimension linéaire plus adaptée aux séries temporelles est l'algorithme Dynamic Mode Decomposition (DMD) [43]. Cet algorithme garde les principaux modes de la série temporelle. Il est possible aussi de transformer les données de manière non linéaire. On peut citer kernel PCA [44] et MultiDimensional Scaling (MDS) [45]. Kernel PCA projette tout d'abord les données dans un espace de plus grande dimension où l'on suppose qu'elle devient alors linéaire. Il est ensuite possible de réduire la dimension avec une simple PCA. MDS de son côté essaye de préserver au mieux les distances entre les éléments en grande dimension et en dimension réduite.

Durant nos travaux, l'algorithme de réduction de dimension "Uniform Manifold Approximation and Projection" (UMAP) [46] s'est montré particulièrement pertinent. Cet algorithme réalise une

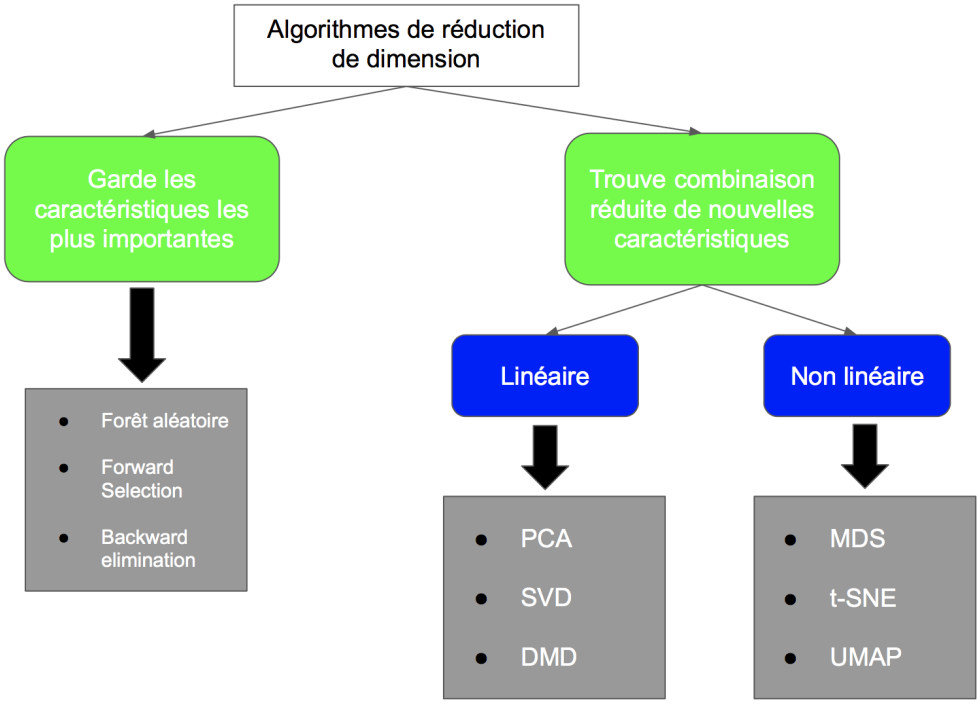


FIGURE 2.6 – Présentation des différents algorithmes de réduction de dimension. Ils sont séparables en deux familles : les algorithmes sans transformation des données qui gardent seulement les caractéristiques les plus importantes et les algorithmes qui transforme les données en créant une combinaison réduite de nouvelles caractéristiques. En particulier, dans cette deuxième famille, les transformations peuvent être linéaires ou non.

transformation non linéaire pour obtenir un nouveau jeu de caractéristiques réduit pertinent. Le fonctionnement de cet algorithme est présenté dans cette partie.

UMAP réalise une réduction de dimension vers $\mathbb{R}^m$ en utilisant la distance entre chaque élément de la base de données. Cet algorithme commence par créer un graphe représentant la topologie de la base de données. Plus particulièrement, le but est d'obtenir une représentation par un "Local Fuzzy Simplicial Set", un graphe pondéré dont les poids représentent une probabilité d'appartenance. Pour obtenir une représentation qui respecte la topologie, [47] montre qu'il est suffisant d'avoir des éléments uniformément distribués. Ce n'est évidemment pas vrai pour toute base de données mais UMAP redéfinit une mesure de similarité entre les éléments de la base de données pour que soit le cas.

Soit une base de données $X = \{X_1, ..., X_n\} \in M$, associée à une distance $d_M : X \times X \to \mathbb{R}^{*+}$. Un graphe est créé dont les sommets sont les éléments de la base de données. Pour chaque sommet X_i, on le relie à ses k (variable d'entrée à choisir) plus proches voisins (au sens de d_M) $(X_{ij})_{1\leq j\leq k}$ (ordonnés par distance croissante) par des arêtes. On note ρ_i la distance au plus proche voisins de X_i ($\rho_i = \min_{1\leq j\leq n, j\neq i}(d_M(X_i, X_j))$). Une première modification de la métrique est alors proposée avec $d = (d_i)_{1\leq i\leq n}$ une famille de métriques définie par :

$$d_i(X_j, X_k) = \begin{cases} d_M(X_j, X_k) - \rho_i \text{ si j=i ou k=i} \\ \infty \text{ sinon} \end{cases} \tag{2.6}$$

On associe aussi une constante σ_i à X_i définie par :

$$\sum_{j=1}^{k} exp\left(\frac{d_i(X_i, X_ij)}{\sigma_i}\right) = log_2(k)$$

Les poids $(w_j)_{1\leq j\leq k}$ des arêtes partant de X_i dans le graphe G sont alors définis par :

$$w_j = exp\left(-\frac{d_i(y_i, y_{ij})}{\sigma_i}\right) \tag{2.7}$$

Avec ce choix de ρ_i, chaque élément est alors relié à au moins un élément avec un poids de 1. On est assuré d'avoir une connexion locale sur l'ensemble du graphe. Le choix de σ_i lui assure une distribution uniforme : pour chaque point X_i la densité du cercle de centre X_i et de rayon $d(X_i, X_{ik)}$

est sensiblement la même. Ce choix pour la définition du poids (de type noyau de chaleur, Éq. 2.7) est justifié dans [47]. Les éléments sont donc connectés par des poids compris entre 0 et 1, interprétables comme une probabilité d'appartenance.

On obtient ainsi un graphe G dont on note la matrice d'adjacence A. Pour uniformiser la famille de métriques d sur ce graphe, on définit un nouveau graphe $\bar{G}$ pondéré non orienté dont la matrice d'adjacence B est donnée par :

$$B = A + A^T - A \circ A^T$$

(avec $\circ$ le produit d'Hadamard défini par $(A \circ B)_{i,j} = A_{i,j}B_{i,j}$). Le but est maintenant de déterminer un graphe G' d'éléments $(y_i)_{1 \leq i \leq n}$ de $\mathbb{R}^m$ dont la cross-entropy avec $\bar{G}$ est minimale. Pour cela, une première initialisation pour le placement des Y_i est obtenue en utilisant le spectre du Laplacien associé à $\bar{G}$. Les coordonnées du i-ème élément de la base de données dans $\mathbb{R}^m$ sont alors $(f_1(i), ..., f_m(i))$ avec $f_0, .., f_m$ les vecteurs propres du Laplacien (triés par valeurs propres : $\lambda_0 < \lambda_1 < ... < \lambda_m$). À partir de ces éléments de $\mathbb{R}^m$, le graphe g' est construit de manière similaire à $\bar{G}$. Cependant, la formule des poids est différente. En effet, la précédente définition (Éq. 2.7) ne permet pas de dériver la cross-entropy. Une fonction dérivable proche est alors déterminée. Deux paramètres a et b sont choisis tel que la fonction ψ réalise une approximation dérivable de Ψ avec ψ et Ψ définies par :

$$\psi \ : \begin{cases} \mathbb{R}^m \times \mathbb{R}^m \to [0,1] \\ Y_i, Y_j \to \frac{1}{1+a(\|Y_i - Y_j\|_2^2)^b} \end{cases}$$

$$\Psi \ : \begin{cases} \mathbb{R}^m \times \mathbb{R}^m \to [0,1] \\ Y_i, Y_j \to exp(-\|Y_i - Y_j\|_2 - \rho_i) \end{cases}$$

La fonction Ψ correspond à la définition des poids vu en Éq. 2.7. Le poids entre $Y_i, Y_j \in \mathbb{R}^m$ est alors défini par :

$$w(Y_i, Y_j) = \frac{1}{1 + a(\|Y_i - Y_j\|_2^2)^b}$$

Cette nouvelle expression du poids permet d'obtenir une expression littérale pour la dérivé de la cross-entropy entre $\bar{G}$ et G'. Avec cette dérivée, on détermine une force d'attraction et de répulsion et on applique un algorithme de "forced directed graph layout" pour minimiser cette cross-entropy.

Ces forces sont définies par :

$$F_{attraction}(Y_i, Y_j) = \frac{-2ab\|Y_i - Y_j\|_2^{2(b-1)}}{1 + \|Y_i - Y_j\|_2^2} w(X_i, X_j)\overrightarrow{Y_j Y_i} \tag{2.8}$$

$$F_{répulsion}(Y_i, Y_j) = \frac{2b}{(\|Y_i - Y_j\|_2^2)(1 + a\|Y_i - Y_j\|_2^{2b})}(1 - w(X_i, X_j))\overrightarrow{Y_j Y_i} \tag{2.9}$$

Pour résumer :

- **UMAP crée un graphe G en tenant en compte la distance sur la base de données et le k-voisinage de chaque élément. C'est donc un algorithme de réduction de dimension basé sur la densité. Le but est maintenant de déterminer des éléments de $\mathbb{R}^m$ définissant de la même manière un graphe G' proche de G au sens de la cross-entropy.**
- **Une initialisation de ce graphe G' sur $\mathbb{R}^m$ est obtenue par une réduction de dimension utilisant le spectre du Laplacien du graphe G.**
- **Le graphe G' est ensuite modifié via un algorithme de "forced directed graph layout" permettant de minimiser la cross-entropy entre G et G'.**

Le but premier d'UMAP est de visualiser des bases de données de hautes dimensions comme dans [48]. Pour le clustering, une utilisation de cet algorithme est moins évidente. En effet, UMAP ne peut complètement garder la densité réelle de la base de données quand la dimension est réduite. En particulier, [49] a montré sur t-SNE (une réduction de dimension similaire à UMAP) que ce type d'algorithme peut créer de pseudo-groupes qui pourraient empêcher des algorithmes de clustering de bien fonctionner. Cependant, dans les chapitres suivants, on montre à travers une analyse comparative exhaustive que son utilisation dans les cas réels améliore clairement les résultats.

En terme de complexité, une analyse est disponible sur leur site web : `https://umap-learn.readthedocs.io/en/latest/benchmarking.html`.

TABLE 2.2 – Bases de données disponibles à Saint-Étienne et Grenoble.

Base de données des urgences	**Base de données laboratoire**
Saint-Étienne :	***Saint-Étienne :***
- Du 03/06/13 au 31/03/17 - 150 codes CIM - 41 456 patients en moyenne par an	- Du 03/06/13 au 31/03/17 - 2 virus - 380 tests positifs en moyenne par an
Grenoble :	***Grenoble :***
- Du 01/09/15 au 19/04/19 - 220 codes CIM - 56 619 patients en moyenne par an	- Du 01/09/15 au 19/04/19 - 2 virus - 774 tests positifs en moyenne par an

2.5 Premiers résultats sur l'hôpital

Suite à ce rapide état des lieux de la littérature sur le clustering de séries temporelles unidimensionnelles, on peut obtenir des premiers résultats sur notre cas d'étude. Nous disposons des bases cliniques et virologiques de deux hôpitaux : l'hôpital de Grenoble et l'hôpital de Saint-Étienne. Ces deux bases nous permettent d'extraire des séries temporelles. Certaines d'entre elles présentaient très peu d'occurrences. On a donc décidé de garder les séries temporelles qui ont plus d'arrivée sur la longueur de la série temporelle qu'un certain seuil. Ce premier tri nous laisse avec 220 séries temporelles (pour 220 codes CIM) à l'hôpital de Grenoble et 150 pour l'hôpital de Saint-Étienne. Ces bases de données englobent la période du 01/09/2015 au 19/04/2019 pour Grenoble et du 03/06/13 au 31/03/17 pour Saint-Étienne. Les caractéristiques des bases de ces deux hôpitaux sont données en Table 2.2.

Le clustering étant une méthode non supervisée de machine learning, il est compliqué d'avoir un retour sur la qualité et le sens des résultats obtenus. Ici, on a l'avantage de pouvoir s'appuyer sur des connaissances médicales pour valider les résultats. En effet, la majorité des complications dues aux virus saisonniers sont de type cardiaque (code I) ou de type respiratoire (code J). On souhaite donc obtenir des résultats de clustering avec une présence importante de ces codes-là. Une absence ou une présence très partielle de codes I et J dans les groupes des virus est signe d'un

clustering inefficace. De plus, on peut s'appuyer aussi sur la reconstruction de la série temporelle moyenne des codes à risque. En effet, cette reconstruction doit être assez similaire aux courbes des virus présentées en Fig. 1.4.

Dans ce chapitre, nous avons présenté les méthodes les plus classiques à appliquer pour l'analyse des virus saisonniers. En terme de distance, DTW n'est pas forcément approprié car des séries temporelles avec des saisonnalités différentes seraient considérées comme similaires. Par exemple, un code diagnostique fréquent en été et absent en hiver serait proche (au sens de DTW) des virus hivernaux. Nous avons donc choisi d'appliquer en première approche la distance Euclidienne. Pour avoir un premier aperçu de ce que donne cette distance, nous avons crée une base de données commune contenant les deux séries temporelles des virus (VRS et grippe) ainsi que l'ensemble des séries temporelles associées à un code CIM. Nous avons ensuite déterminé les 6 éléments les plus proches (au sens Euclidien) des deux virus. Les résultats sont proposés en Table 2.3. On remarque une faible présence de codes J (symptômes respiratoires). De plus, à part pour la grippe à Grenoble où le VRS apparaît proche, l'autre virus n'apparaît pas dans les éléments les plus proches d'un virus donné. Par exemple, le VRS à Grenoble et Saint-Étienne n'a pas la grippe dans ses 6 éléments les plus proches. Or, au sein d'un même hôpital, les deux virus ont des dynamiques très semblables avec juste un léger décalage. On peut en déduire deux conclusions : on devrait retrouver des codes en commun et pour un virus donné, l'autre virus devrait faire partie des éléments les plus proches. C'est très peu le cas ici. Cette distance ne semble donc pas adaptée.

TABLE 2.3 – Table des éléments les plus proches au sens Euclidien pour les deux hôpitaux et les deux virus.

Virus	**Hôpital**	**Éléments les plus proches**
Grippe	Grenoble	J11 VRS B34 J44 R04 S22
	Saint-Étienne	J20 E11 B34 G51 D62 R00
VRS	Grenoble	R05 J42 D62 C34 I20 R54
	Saint-Étienne	J20 I61 N45 B34 I47 G51

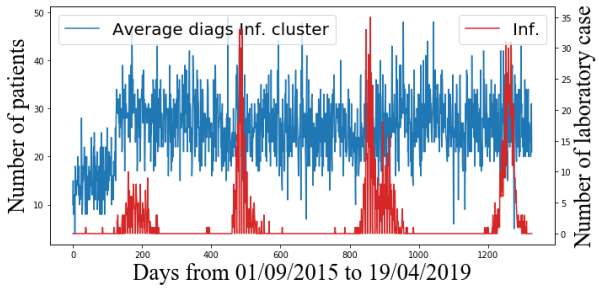
Pour le confirmer, nous avons appliqué K-means (étant un des algorithmes de clustering les plus utilisés) avec cette distance. Pour chaque hôpital et pour chaque virus, nous avons donc réuni la série temporelle du virus avec celles des codes diagnostiques (issues des urgences de cet hôpital), puis nous avons appliqué la distance Euclidienne avec K-means. On obtient quatre résultats de clustering (un par hôpital et par virus).

Pour l'hôpital de Grenoble, on obtient 42 éléments dans le groupe de la grippe tout comme dans le groupe du VRS. Cependant, il y a encore peu d'éléments en commun malgré le comportement similaire des deux virus. De plus on retrouve peu de code J (6 pour la grippe et 1 pour le VRS). Du côté de Saint-Étienne, les groupes sont encore plus imposants (55 pour le RSV et 84 pour la grippe). De même, on retrouve peu de codes J et peu de codes en commun. On a donc a priori des résultats de clustering non pertinents. Pour le vérifier, on propose de reconstruire une série temporelle, sensée être la signature des virus, à partir de ces résultats. On rappelle que cette reconstruction d'un virus est simplement la moyenne des séries temporelles présentes dans le groupe de ce virus. Les reconstructions obtenues sont tracées en même temps que les virus en Fig. 2.7. Le comportement particulier des virus n'est pas du tout visible et les admissions dans le groupe des virus sont à peu près constantes. Il est donc impossible d'utiliser ces résultats pour, par exemple, détecter un flux inhabituel dû aux virus.

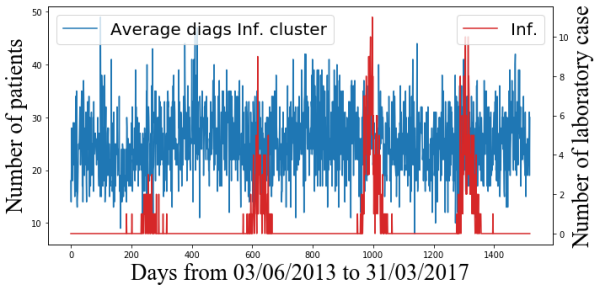
Il est donc nécessaire de trouver une meilleure mesure de similarité et un meilleur algorithme qui seraient plus adaptés à notre cas d'étude. Une cause potentielle des mauvais résultats obtenus est que la distance Euclidienne se focalise sur des différences locales sans prendre en considération la dynamique globale de la série temporelle.

2.6 Synthèse

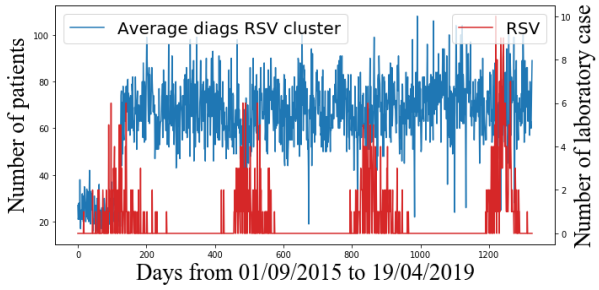
Pour aider à comprendre le flux de patient dans les hôpitaux, on a besoin de réaliser un clustering de séries temporelles uni-dimensionnelles. Il est donc nécessaire de définir une distance. Il en existe une multitude plus ou moins adaptée aux séries temporelles. Les distances les plus classiques sont présentées dans ce chapitre. On propose aussi une revue de différents algorithmes de clustering et de réduction de dimensions. Nous pouvons ainsi appliquer une première méthode de clustering que l'on peut qualifier de naïve sur les urgences hospitalières. Cette méthode est déterminée par une distance



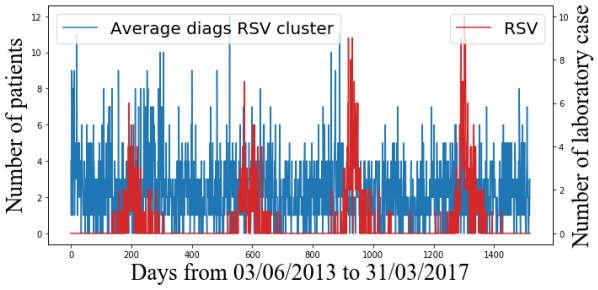
(a) Reconstruction de la série virologique Grippe obtenue avec les résultats de clustering de la distance Euclidienne associée à K-means à l'hôpital de Grenoble.



(b) Reconstruction de la série virologique Grippe obtenue avec les résultats de clustering de la distance Euclidienne associée à K-means à l'hôpital de Saint-Étienne.



(c) Reconstruction de la série virologique VRS obtenue avec les résultats de clustering de la distance Euclidienne associée à K-means à l'hôpital de Grenoble.



(d) Reconstruction de la série virologique VRS obtenue avec les résultats de clustering de la distance Euclidienne associée à K-means à l'hôpital de Saint-Étienne.

FIGURE 2.7 – Reconstructions obtenues pour chaque virus et chaque hôpital avec la méthode la plus classique. Les résultats ne sont pas satisfaisants et n'apportent pas d'informations intéressantes pour les hôpitaux.

Euclidienne combinée avec l'algorithme K-means, ce qui se fait de plus classique. Cependant, cette première approche nous donne des résultats très mauvais, mis en évidence par une reconstruction très éloignée de la série temporelle réelle du virus. Pour améliorer ce résultat de clustering, on peut travailler sur deux aspects : la distance et l'algorithme de clustering. On va tout d'abord se focaliser sur la distance. On propose ainsi d'appliquer le Delay Coordinate Embedding, présenté dans le prochain chapitre, pour avoir une distance qui prend mieux en compte la dynamique de la série temporelle.

Chapitre 3

Delay Coordinate Embedding et matrices de trajectoire

3.1 Introduction

Les résultats précédents obtenus avec la distance Euclidienne mettent en évidence que ce choix de distance passe à côté de la dynamique qui nous intéresse dans les séries temporelles des virus. Si on se penche sur les séries temporelles multivariées (plusieurs caractéristiques suivant une évolution temporelle), on découvre d'autres méthodes pour traiter les séries temporelles. Par exemple, le mouvement d'un corps dans le temps [50] peut être caractérisé par un vecteur de $\mathbb{R}^3$ suivant une évolution temporelle. Dans ce type de cas, il a été démontré dans la littérature [51, 52] que l'apprentissage sur variété permet une meilleure représentation des données comparée à la distance Euclidienne.

Apprentissage sur variété : L'idée ici est que les données sont à première vue dans un espace ambiant A mais qu'en réalité elles se situent sur une variété de dimension réduite [53]. On appelle variété un espace localement similaire à un espace Euclidien, comme par exemple une Sphère dans $\mathbb{R}^3$ qui ressemble localement à un plan de $\mathbb{R}^2$. La notion de variété et leurs géométries sont définies proprement dans le prochain chapitre. On peut prendre l'exemple de l'espace des images. Une image est définie par un certain nombre de pixels n eux-mêmes définis par 3 paramètres (quantité de rouge, quantité de bleu, quantité de vert) pour déterminer sa couleur. L'ensemble des images se situent donc dans un espace de grande dimension n^3. Or, en général, cet espace en entier ne nous intéresse pas car un élément pris au hasard dans cet espace serait presque sûrement semblable à

du bruit pour l'humain. Si on s'intéresse par exemple à l'ensemble des images de paysages, il est plus pertinent de déterminer un espace de dimension réduit contenant cet ensemble plutôt que de traiter cet ensemble directement sur $\mathbb{R}^{\kappa^{\#}}$. C'est ce qu'on appelle l'apprentissage sur variété.

On se propose d'utiliser ce même type de méthode. Si la littérature est vaste concernant l'apprentissage sur variété dans le cas de séries temporelles multivariées, on n'en trouve pas à notre connaissance pour les séries temporelles uni-dimensionnelles. Ainsi, [54], qui balayent plusieurs méthodes de clustering pour des séries temporelles uni-dimensionnelles, ne citent pas du tout ce type d'approche. Pour pallier ce manque, on propose d'appliquer le Delay Coordinate Embedding qui transforme une série temporelle en une matrice de trajectoire. Une matrice de trajectoire permet de mieux prendre en compte la dynamique d'une série temporelle et dispose d'une structure similaire à une série temporelle multivariée. Cette méthode du Delay Coordinate Embedding a donc un double intérêt :

— Une meilleur image de la dynamique des virus,

— L'application de méthodes qui ont fait leur preuve dans le cas de séries temporelles multivariées.

3.2 Delay coordinate embedding

Pour un système mécanique, l'espace de phase est un espace où tous les états possibles du système sont représentés. Cet espace permet de mettre en lumière la dynamique du système mécanique. On peut prendre l'exemple d'un pendule simple. L'angle θ entre la verticale et la direction du fil du pendule vérifie, pour des petits angles, l'équation différentielle $\ddot{\theta} + \theta = 0$, ce qui nous donne $\theta(t) = cos(t)$. Cela nous permet de construire l'espace de phase (θ, $\dot{\theta}$) qui décrit ce système (voir Fig. 3.1).

Le problème dans les cas réels est qu'on n'a pas accès à l'état global du système mais plutôt à une mesure discrète de celui-ci. Soit un système mécanique défini à chaque instant t par $u(t) \in S$ avec S l'espace de phase. On mesure un signal de celui-ci défini par :

$$x_n = X(u(t_n))$$

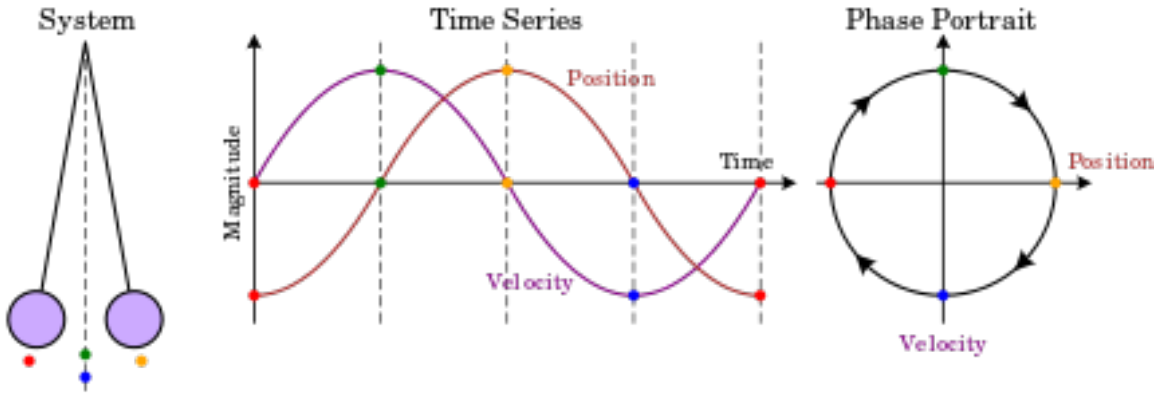


FIGURE 3.1 – Illustration de la construction de l'espace de phase pour un système simple : le pendule. Image extraite de `https://en.wikipedia.org/wiki/Phase_space`.

avec $X(.)$ une mesure de u et t_n les moments discrets de la mesure de u. Dans le cas où $X(.)$ est une fonction de S vers $\mathbb{R}^1$, on obtient alors une série temporelle uni-dimensionnelle $x = (x_1 = x(u(t_1)), x_2, ..., x_l)$ comme mesure du système dynamique.

Il est possible à partir de cette mesure x de reconstruire l'espace de phase caché. Dans cet objectif, on peut s'appuyer sur le Delay Coordinate Embedding [55, 56, 57]. L'idée est ici de s'intéresser à des tranches de taille m (appelé dimension d'"embedding") d'éléments de x décalés d'un temps τ (appelé retard ou "delay") :

$$\pi(x_t, \tau, m) = [x_t, x_{t+\tau}, ..., x_{t+m\tau}]$$

Cette méthode est utilisée dans beaucoup de domaines [58, 59]. Ces tranches déterminent l'évolution d'un point dans $\mathbb{R}^m$. C'est cette trajectoire qui nous intéresse maintenant pour déterminer la dynamique de la série temporelle. Ainsi, à partir de la série temporelle x, on définit ce qu'on appelle une matrice de trajectoire, dépendante des deux paramètres m et τ :

$$Tr_x = \begin{pmatrix} x_0 & x_\tau & \cdots & x_{(m-1)\tau} \\ x_1 & x_{1+\tau} & & \vdots \\ \vdots & \vdots & & \vdots \\ x_k & x_{k+\tau} & \cdots & x_{k+(m-1)\tau} \\ \vdots & \vdots & & \vdots \\ x_{l-1-(m-1)\tau} & x_{l-1-(m-2)\tau} & \cdots & x_{l-1} \end{pmatrix} \tag{3.1}$$

Cette matrice de trajectoire est donc dans l'espace ambiant $\mathbb{R}^{n\times p}$ avec $p = m$ et $n = l - (m-1)\tau$

Une analyse mathématiques de cette méthode a été réalisée dans [60], connue sous le nom de théorème de Takens. Soit une série temporelle x issue d'un système dynamique continu et déterministe qui présente un attracteur de dimension d. Ce théorème assure qu'à partir de cette série temporelle x, il est possible de reconstruire cet attracteur. Cette reconstruction est à un difféomorphisme près [61]. La topologie de l'attracteur est donc bien récupérée. Pour réaliser cette reconstruction, il faut des tranches d'au plus $m = 2d$ éléments de X décalé d'un temps τ (souvent une valeur de m strictement inférieure à $2d$ peut suffire [62]).
Prenons l'exemple d'un système dynamique bien connu : le système de Lorenz. Un point P dans $\mathbb{R}^3$ qui a une trajectoire issue de ce système vérifie les équations suivantes :

$$\begin{cases} \dot{x}(t) = & \sigma(y(t) - x(t)) \\ \dot{y}(t) = x(t)(\rho - z(t)) - y(t) \\ \dot{z}(t) = & x(t)y(t) - \beta.z(t) \end{cases} \tag{3.2}$$

avec ρ, σ et β des paramètres (scalaires). C'est donc un système d'équations différentielles couplées du premier ordre. Pour $\rho = 28$, $\sigma = 10$ et $\beta = 8/3$, une trajectoire issue de ce système va présenter la forme d'un papillon (voir l'attracteur de Lorenz sur la Fig. 3.2). Supposons que l'on mesure seulement la composante x de P avec un pas de temps de 0.01 et 5000 itérations. On a donc comme représentation de ce système une série temporelle X de longueur 5000. Une illustration de cette série temporelle est proposée dans la Fig. 3.2. À partir de cette série temporelle, on construit la matrice de trajectoire avec le Delay Coordinate Embedding avec comme paramètres $\tau = 10$ et $m = 3$. Comme annoncé par le théorème de Takens, on remarque que la trajectoire issue des lignes de cette matrice est fortement ressemblante à la trajectoire de l'attracteur de Lorenz. En particulier, la forme en papillon est bien récupérée, ce qui indique que la topologie originale est respectée.

En utilisant le Delay Coordinate Embedding, les séries temporelles sont maintenant étudiées à travers leurs matrices de trajectoire. Ce choix de représentation des séries temporelles a un impact sur la définition d'une distance, un des deux aspects fondamentaux pour définir un résultat de

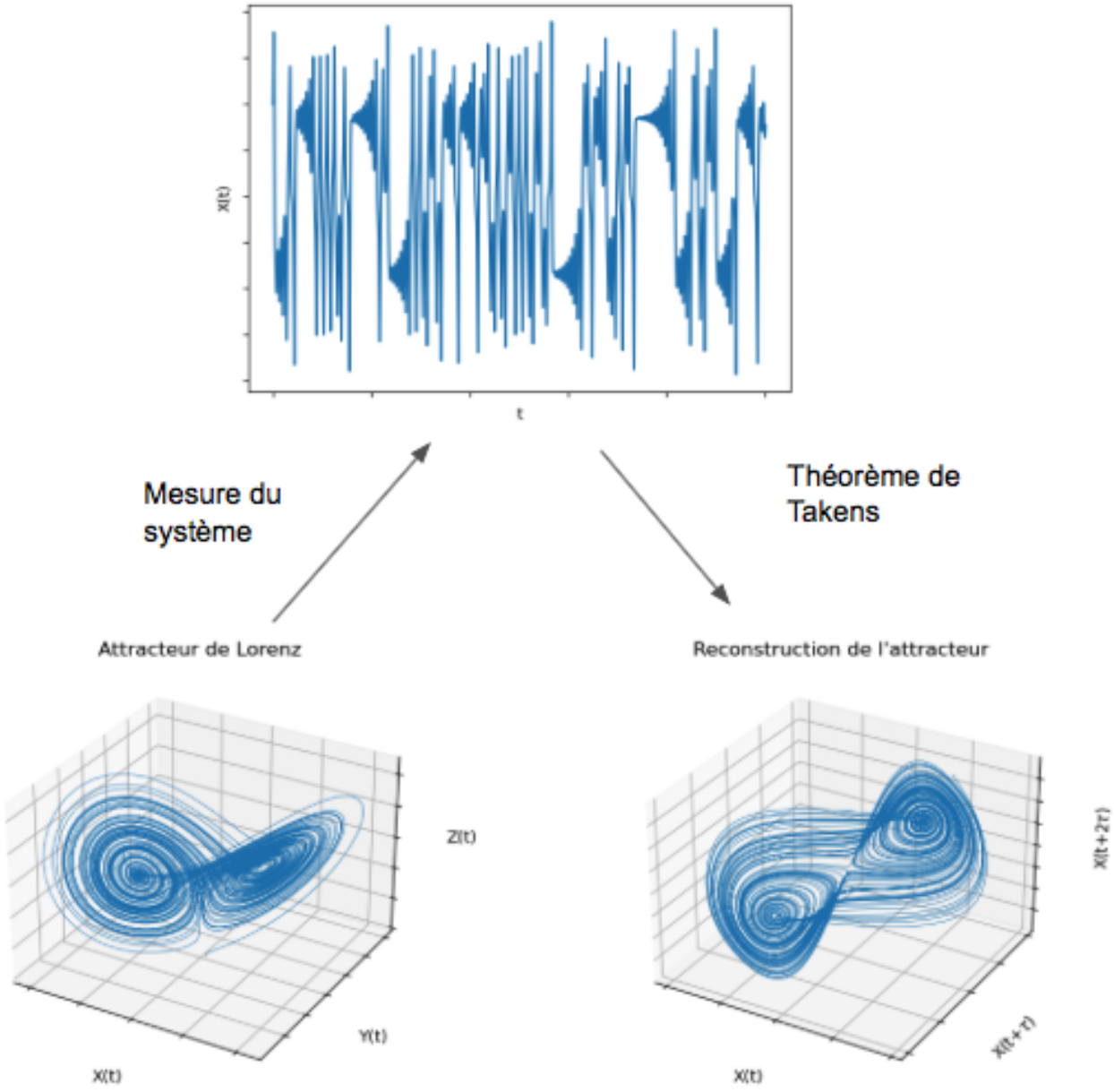


FIGURE 3.2 – Exemple d'application du théorème de Takens sur le système de Lorenz. À partir de la trajectoire réel, une mesure du système est réalisée qui extrait la composante x. On reconstruit ensuite la trajectoire via le Delay Coordinate Embedding.

clustering. En effet, lors du chapitre précédent, on a ouvert des pistes pour appliquer une distance directement sur les séries temporelles uni-dimensionnelles, éléments de $\mathbb{R}^l$. Or, on a maintenant besoin d'une métrique pour comparer deux matrices de trajectoire Tr_x, Tr_y, représentantes de deux séries temporelles x, y. Dans cette optique, on va d'abord proposer plusieurs variétés sur lesquelles représenter ces matrices. Le choix de variété, de représentation géométrique des matrices, amène à différentes distances.

3.3 Représentation des matrices de trajectoire

Pour une base de données de séries temporelles $x_i \in \mathbb{R}^l$, on a donc associé une matrice de trajectoire $T_{x_i} \in \mathbb{R}^{n\times p}$ (n, p dépendants de l et des paramètres m et τ) à chacune de ces séries temporelles. Quelle variété choisir ensuite pour analyser ces matrices ? Plusieurs méthodes existent pour les étudier [63, 64]. Durant cette thèse, nous nous sommes penchés sur quatre possibilités : directement $\mathbb{R}^{n\times p}$, la variété de Grassmann, la Sphère unité et l'ensemble des matrices strictement positives symétriques.

3.3.1 Norme de Frobenius

La représentation géométrique la plus basique de ces matrices de trajectoire est de les considérer dans leur espace de départ $\mathbb{R}^{n\times p}$. On utilise alors directement la norme de Frobenius définie par :

$$d(A,B) = (\sum_{k=0}^{n-1}\sum_{l=0}^{p-1}(a_{k,l} - b_{k,l})^2)^{\frac{1}{2}} \ \forall\ A, B \in\ \mathbb{R}^{n\times p}.$$

comme distance entre les différentes matrices de trajectoire. Dans ce contexte, la moyenne admet l'expression basique : $\overline{A} = \frac{1}{k}\sum_{i=1}^{k} A_i, \forall\ A_1, ..., A_k\ \in \mathbb{R}^{n\times p}$.

Pour approfondir l'extraction d'information contenue dans les matrices de trajectoire, on propose d'aller plus loin que la distance "naïve" liée à la norme de Frobenius. On se rapproche alors de l'apprentissage sur variété. Dans ce but, on choisit un sous-espace de $\mathbb{R}^{n\times p}$ auquel appartient les matrices de trajectoire.

3.3.2 Variété de Grassmann

La variété de Grassmann $Gr_{n,p}$ est l'ensemble des p sous-espaces de $\mathbb{R}^n$. Si une matrice de trajectoire est placée dans $\mathbb{R}^{n\times p}$ avec $p < n$, l'espace engendré par ses colonnes est alors un élément de la variété de Grassmann. C'est directement le cas pour des petites valeurs de m et τ. Si on est dans le cas où $p > n$, il est possible de se retrouver dans le cas $p < n$ via une réduction de dimension [65].

Avec cette représentation des matrices de trajectoire, on ne regarde plus localement les éléments des matrices (comme avec la norme de Frobenius), mais on s'intéresse de manière plus globale aux espaces engendrées par ces dernières.

Pour clarifier ces sous-espaces engendrés, nous proposons d'orthogonaliser ces matrices de trajectoire. Elles sont alors projetées sur la variété de Stiefel. La variété de Stiefel est un sous-espace de $\mathbb{R}^{n\times p}$ défini par $V_{n,p} = \{U \in \mathbb{R}^{n\times p} : U^T U = I_p\}$.

Les séries temporelles sont alors vues comme des éléments de la variété de Grassmann, représentés par des éléments de la variété de Stiefel. Ce choix de variété a déjà fait ses preuves en traitement du signal et en analyse de données [66, 67].

3.3.3 Matrice de Gramm

Dans cette section, nous proposons d'utiliser la matrice de Gramm pour analyser la matrice de trajectoire. Cette dernière est définie à partir de la matrice de trajectoire $Tr_x \in \mathbb{R}^{n\times p}$:

$$G_{Tr_x} = \frac{Tr_x Tr_x^T}{\|Tr_x Tr_x^T\|} \tag{3.3}$$

Cette méthode est proposée dans [68] et les auteurs démontrent que cela permet de réduire un éventuel bruit présent dans la série temporelle. Cette nouvelle matrice G_{Tr_x} est par définition sur l'espace des matrices positives symétriques de rang p noté $P(n,p)$. Ces matrices de $P(n,p)$ peuvent être approximées comme des éléments des matrices strictement positives symétriques $SPD(n)$ en ajoutant à chaque élément P de $P_{n,p}$ ϵI_n (avec ϵ d'une petite valeur) [68].

3.3.4 Sphère unité

Sur les précédentes géométries présentées pour analyser la matrice de trajectoire, on a supposé que les matrices de trajectoire étaient de même taille sur l'ensemble des séries temporelles d'une base de données. On peut alors placer l'ensemble des séries temporelles sur le même espace (que ce soit $\mathbb{R}^{n\times p}$, $V_{n,p}$ et $P(n,p)$). Il est cependant possible d'avoir besoin de comparer des matrices de trajectoire de tailles différentes. En effet, on peut par exemple décider d'appliquer une réduction de dimension sur chacune des matrices de trajectoire qui garderait un certain nombre de colonnes selon le niveau de bruit. Dans ce cas-là, on obtient un ensemble de matrices de trajectoire dont le nombre de colonnes varie. Pour replacer ces matrices sur un même espace, on peut tout d'abord les orthogonaliser et obtenir des éléments de différentes variétés de Stiefel. Ensuite, [69] propose de replacer l'ensemble de ces éléments sur un même espace : la Sphère unité. Soit $U_1, U_2, ..., U_k$ des éléments de $(V_{n,p_1} \times V_{n,p_2} ... \times V_{n,p_k})$. Pour comparer ces éléments, on calcule les matrices de projection $P_{U_i} = U_i U_i^T$ (projection car $P_{U_i}^2 = P_{U_i}$) qui sont alors des éléments de $\mathbb{R}^{n\times n}$ symétriques. Chacune de ces matrices est alors transformée en un vecteur de $\mathbb{R}^{n(n+1)/2}$ contenant tous les éléments différents de ces matrices une unique fois. On peut ainsi garder l'ensemble de l'information contenue dans la matrice tout en la transformant en un vecteur. Par exemple, une matrice de projection symétrique dans $\mathbb{R}^{2\times 2}$ est transformée en un vecteur ainsi :

$$\begin{pmatrix} a & b \\ b & d \end{pmatrix} \rightarrow \begin{pmatrix} a \\ b \\ d \end{pmatrix}. \tag{3.4}$$

Grâce à l'orthogonalité des matrices $U_i, i = 0, ..., k$, tous les vecteurs associés v_i de $\mathbb{R}^{(n(n+1)/2)}$ sont sur la même Sphère de rayon r et de centre c dans $\mathbb{R}^{(n(n+1)/2)}$. Pour simplifier, ces vecteurs sont ensuite tous replacés sur la Sphère unité de $\mathbb{R}^{(n(n+1)/2)}$ avec $v_i' = \frac{v_i - c}{||v_i - c||}$. Les séries temporelles d'une base de données sont donc maintenant analysées comme des éléments de la Sphère unité.

3.4 Synthèse

Pour récupérer au mieux la dynamique des séries temporelles, nous avons décidé de nous appuyer sur le Delay Coordinate Embedding. On obtient alors pour chaque série temporelle une matrice que l'on peut représenter sur différents espaces ($\mathbb{R}^{n\times p}$, Sphère unité, Variété de Stiefel/Grassmann, $P(n,p)/SPD(n)$). Sur $\mathbb{R}^{n\times p}$, l'expression de la distance (norme de Frobenius) et de la moyenne sont directes. Sur les autres géométries proposées, déterminer la distance et la moyenne demande plus de travail. Ces espaces nommés variétés sont des espaces particuliers. En effet, ils ne sont pas forcément "plat" comme un espace Euclidien classique. Pour appliquer les algorithmes de clustering, il est donc nécessaire de déterminer une distance et une moyenne qui respectent la géométrie, la courbure, de ces différents espaces. Dans la partie suivante, on propose une introduction à la géométrie de Riemann qui va permettre de définir cette distance et cette moyenne sur tous ces espaces.

Chapitre 4

Géométrie Riemannienne

La géométrie Riemannienne se distingue de la géométrie Euclidienne par l'étude des géométries non planes. Elle permet de définir une métrique sur les variétés et donc de réaliser des apprentissages sur variétés [70, 71]. Cette caractéristique nous permet de comparer les séries temporelles une fois qu'elles ont été injectées sur les différentes variétés.

4.1 Introduction à la notion de variété

4.1.1 Définition d'une variété

Une variété est en ensemble de points qui ressemble localement à un espace Euclidien (géométrie plane) qui existe dans un espace ambiant Euclidien mais qui présente globalement une courbure. On peut prendre pour exemple la surface de la Terre, une Sphère dans l'espace ambiant de $\mathbb{R}^3$. Localement, on peut considérer cette surface comme étant un plan dans $\mathbb{R}^2$ sans perte de précision. On peut ainsi se déplacer en utilisant une carte 2D. Cependant, sur de longues distances, par exemple lors de trajets en avion, il n'est plus possible d'utiliser ces cartes 2D. La représentation Mercator (planisphère), représentation globale de la Terre en plan 2D, est seulement une approximation de la vraie géographie avec une échelle qui varie selon la distance à l'équateur. Ainsi, même si localement notre Sphère ressemble à un espace plat (Euclidien), il n'est pas possible de la représenter comme tel globalement sans tordre la géométrie naturelle. Pour avoir une vision globale de cette géométrie, il est possible de réunir dans un atlas l'ensemble des cartes locales.

4.1.2 Carte et atlas

Définition 1 (Carte)**.** *Pour une variété M de dimension d, une carte est un homéomorphisme (bijectif et continu) depuis un ensemble ouvert $U \in M$ vers un ensemble ouvert $\tilde{U} \subseteq \mathbb{R}^d$.*

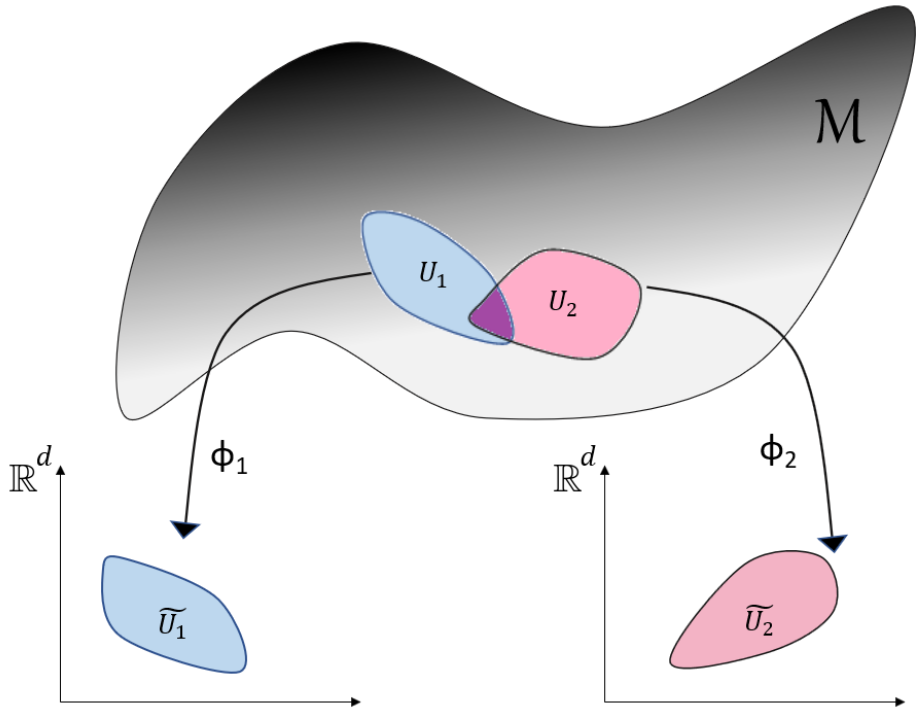


FIGURE 4.1 – La variété M est recouverte par des ouverts U_i sur lesquels on peut définir un homéomorphisme ϕ_i envoyant les éléments de U_i vers un ouvert $\tilde{U}_i$ de $\mathbb{R}^d$.

En utilisant une carte sur un ouvert $U \in M$, la géométrie de cet ouvert peut être déterminée en s'appuyant sur la géométrie de l'espace Euclidien $\tilde{U} \subseteq \mathbb{R}^d$. On a donc un ensemble d'ouvert permettant de déterminer localement la géométrie, il faut pouvoir les réunir en gardant de la cohérence pour pouvoir déterminer la géométrie globale. Une famille de cartes permettant d'avoir cette cohérence est appelé un atlas. Sur une variété M, une telle famille de cartes $(\phi_i)_{1\leq i\leq N}$ vérifie certaines conditions :

Définition 2 (Atlas)**.** *Un atlas d'une variété M est une famille de cartes $(\phi_i)_{1\leq i\leq N}$ avec $\phi_i : U_i \to \tilde{U}_i$ tel que :*

- *Cette famille recouvre tout M : $\forall x \in M, \exists i \in \{1, ..., N\} | x \in U_i$*
- *Cette famille est compatible : Pour toute paire $i, j \in \{1, ..., N\}$ tel que $U_i \cap U_j \neq \emptyset$ la composition (appelée carte de transition) $\phi_i \circ \phi_j^{-1} : \phi_j(U_i \cap U_j) \to \mathbb{R}^d$ est un homéomorphisme.*

Une visualisation de la notion des cartes et atlas est proposée en Fig. 4.1.

4.1.3 Variété différentielle

Cependant, ces cartes Euclidiennes locales regroupées en un atlas ne sont pas forcément suffisante pour définir une métrique et une distance sur la variété. En effet, pour appliquer la géométrie de Riemann, on a besoin d'avoir une structure locale suffisamment ressemblante à un espace Euclidien pour généraliser le calcul différentielle. Pour cela, il est nécessaire que les cartes $(\phi_i)_{1\leq i\leq N}$ soient des difféomorphismes (bijectif et différentiable) ainsi que les cartes de transitions. On est alors dans le cas des variétés différentielles (ou "smooth"). On peut montrer qu'une variété est bien différentielle comme sous-partie d'une variété qui l'est :

Définition 3 ("Embedded" Variété)**.** *Soit une fonction $F : \mathbb{R}^k \to \mathbb{R}^m$ dérivable telle que la matrice Jacobienne $dF(x) = (\frac{\partial}{\partial x^j}F^i(x))^i_j$ a un rang constant de $k-d$ pour tout $x \in F^{-1}(0)$. L'ensemble M défini par $M = F^{-1}(0)$ est alors une "embedded" variété dans $\mathbb{R}^k$ de dimension d.*

Avec cette définition, la variété est intégrée dans $\mathbb{R}^k$ (espace Euclidien, cas particulier d'une variété différentielle) et cette fonction F assure que l'on peut appliquer les méthodes classiques issues de la géométrie Riemannienne. Par la suite, par abus de langage, le terme variété sera utilisé pour parler de variété différentielle.

On peut vérifier avec cette définition que la Sphère unité $S(3)$ de $\mathbb{R}^3$ est bien une variété. Elle est définie par :

$$S(3) = \{(x_1, x_2, x_3) \in \mathbb{R}^3 | x_1^2 + x_2^2 + x_3^2 = 1\}$$

Soit la fonction F définie :

$$F : \begin{cases} \mathbb{R}^3 \to \mathbb{R} \\ (x_1, x_2, x_3) \to x_1^2 + x_2^2 + x_3^2 - 1 \end{cases}$$

On a ainsi F qui est (infiniment) dérivable et $S(3) = F^{-1}(0)$. S_3 est donc bien une variété dont on

peut déduire directement la dimension avec le théorème du rang :

$$\begin{aligned} dim(S(3)) &= dim(Ker(F)) \\ &= dim(\mathbb{R}^3) - rg(F) \\ &= 3 - 1 = 2 \end{aligned}$$

Au chapitre précédent, nous proposions de représenter les séries temporelles comme des éléments de $S(n)$ et de $V_{n,p}$. Vérifions que ce sont bien des variétés ainsi que $\mathcal{O}(n)$ l'ensemble des matrices orthonormales de $\mathbb{R}^{n\times n}$:

— $S(n) = F^{-1}(0)$ avec $F : \begin{cases} \mathbb{R}^n \to \mathbb{R} \\ x \to \|x\|^2 - 1 \end{cases}$

— $V_{n,p} = F^{-1}(0)$ avec $F : \begin{cases} \mathbb{R}^{n\times p} \to \mathbb{R}^{p\times p} \\ A \to A^T A - I_p \end{cases}$

— $\mathcal{O}(n) = F^{-1}(0)$ avec $F : \begin{cases} \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n} \\ A \to A^T A - I_p \end{cases}$

4.1.4 Quotient de variété

Soit G un groupe tel que pour tout $x, y \in M$, il existe $g \in G$ tel que $gx = y$. Pour un $x \in M$, on note H le sous espace $H = \{g \in G, g.x = x\}$. On dit alors que M est un espace homogène au quotient G/H. H est aussi appelé la classe d'équivalence de x et est notée $[x]$
On peut montrer qu'un ensemble est une variété comme quotient de deux variétés en utilisant le théorème suivant issu de [72] :

Théorème 1 (Quotient de deux variétés)**.** *Soit M_1 et M_2 deux variétés. L'ensemble $M = M_1/M_2$ est alors aussi une variété.*

4.1.5 Espace Tangent

Soit une courbe paramétrée dérivable $\gamma : [0,1] \to M \in \mathbb{R}^k$, passant par p à t_0. Le vecteur v tangent de γ à p est défini par :

$$v = \frac{d\gamma}{dt})_{t_0} = \begin{bmatrix} \frac{d}{dt}\gamma_1(t_0) \\ \frac{d}{dt}\gamma_2(t_0) \\ \vdots \\ \frac{d}{dt}\gamma_k(t_0) \end{bmatrix}$$

v est alors défini dans l'espace Euclidien $\mathbb{R}^k$ ambiant à M. Il est tangent à γ en p et donc naturellement tangent aussi à M en p. L'ensemble des vecteurs tangents à M en p issus de l'ensemble des courbes sur M passant par p engendre un espace affine (de la dimension d de M) de $\mathbb{R}^k$ qui réalise une approximation au premier ordre de M à p. Cet ensemble est appelé l'espace tangent T_pM à M en p. Il peut être défini explicitement par :

$$T_pM = \{v \in \mathbb{R}^k \ v = p + Ker(dF(p))\}$$

avec F tel que $M = F^{-1}(0)$.

4.2 Adaptation des éléments classiques de la géométrie Euclidienne à la géométrie de Riemann

4.2.1 Métrique de Riemann et distance

Les précédentes définitions et notions vont nous permettre de développer une métrique. Pour un espace Euclidien, la distance entre deux points est définie par la longueur de la ligne droite entre ces deux points. Il faut donc définir maintenant une notion de "droiture" d'une courbe sur une variété pour déterminer la courbe "droite" appelée géodésique.

Pour tout $p \in M$, on appelle $g_p : T_pM \times T_pM \to \mathbb{R}$ le produit scalaire (aussi noté $\langle ., . \rangle_p$) défini sur l'espace tangent. Il en découle naturellement une norme $\|.\|_p : T_pM \to \mathbb{R}$ définie par $\|v\|^2 = g_p(v,v) = \langle v, v \rangle_p$. L'ensemble de ces produits scalaires $g = \{g_p \ p \in M\}$ est ce qu'on appelle la métrique de Riemann. Le couple $\{M, g\}$ est une variété de Riemann.

Cette métrique de Riemann permet de mesurer la longueur d'une courbe sur la variété. Soit une courbe $\gamma : [0;1] \to M$ (avec $\{M, g\}$ une variété de Riemann). On définit la longueur de cette courbe par :

$$L(\gamma) = \int_0^1 \sqrt{g(\gamma'(t), \gamma'(t)}\,\mathrm{d}t = \int_0^1 \sqrt{g_{\gamma(t)}(\gamma'(t), \gamma'(t)}\,\mathrm{d}t = \int_0^1 ||\gamma'(t)||_{\gamma(t)}\,\mathrm{d}t$$

Cette métrique permet de donner une mesure intrinsèque à la variété de la longueur de la courbe. Une autre manière de mesurer la longueur d'une courbe sur une variété M dans un espace ambiant $\mathbb{R}^k$ est de la considérer comme un élément de $\mathbb{R}^k$. C'est une mesure extrinsèque à la variété. Dans ce cas-là, on a :

$$L_{ext.}(\gamma) = \int_0^1 ||\gamma'(t)||\,\mathrm{d}t$$

avec $||.||$ le produit scalaire classique sur l'espace ambiant. La distance entre $x, y \in M$ est alors la plus petite longueur de courbe permettant de relier ces deux points :

$$d(x, y) = \min_{\gamma(0)=x, \gamma(1)=y} L(\gamma)$$

La courbe réalisant cette minimisation est appelée la géodésique. On peut remarquer que quelque soit le choix de mesure (extrinsèque ou intrinsèque), on retrouve la même courbe qui réalise la géodésique.

Sur la Fig. 4.2, extrait de [52], les éléments définis dans ce début de chapitre sont représentés.

À ce stade-là, on peut déjà appliquer la plupart des algorithmes de clustering (HDBSCAN, Hiérarchique, ...) car ils n'ont besoin que de la distance entre tous les éléments de l'ensemble des données. Cependant, on a besoin d'aller plus loin pour définir une moyenne et appliquer un algorithme comme K-means.

4.2.2 Exponentielle et logarithme de Riemann

Soit $p \in M$ et $v \in T_pM$. Une géodésique a la propriété d'avoir une accélération nulle. Elle vérifie donc une équation du second degré. Or, pour une équation différentielle du second ordre avec une double condition initiale (vitesse et position), il y a existence et unicité de la solution. Il existe donc

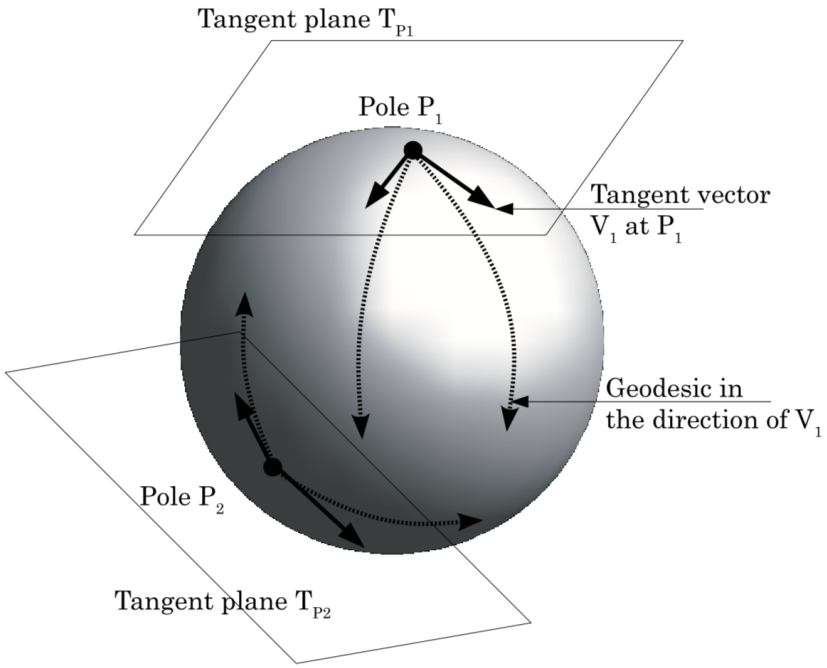


FIGURE 4.2 – Illustration des éléments de la géométrie Riemannienne sur la variété de la Sphère dans $\mathbb{R}^3$. On retrouve des espaces tangents en P_1 et P_2, ainsi que les courbes "droites" (géodésiques) sur cette variété.

une unique géodésique, notée $\gamma_{p,v}$ ayant pour conditions initiales $\gamma_{p,v}(0) = p$ et $\dot{\gamma}_{p,v}(0) = v$. À partir d'un élément de la variété et d'un élément de l'espace tangent, on définit donc de manière unique une courbe qui renvoie à chaque instant un élément de la variété. Cette définition nous permet de construire une fonction qui part de l'espace tangent et qui renvoie un élément de la variété. Par convention, on s'intéresse au point atteint à $t = 1$ et on appelle l'exponentielle de Riemann la fonction définie par :

$$Exp_p : \begin{cases} T_pM \to M \\ v \to \gamma_{p,v}(1) \end{cases}$$

Inversement, on peut noter que l'on peut définir la géodésique à partir de cette fonction avec $\gamma_{p,v}(t) = Exp_p(tv)$

Comme pour les fonctions classiques (définies pour les scalaires ou les matrice carrées) exponentielle et logarithme (noté exp et log), la fonction logarithme de Riemann (noté Log) est définie

comme la fonction inverse à l'exponentielle de Riemann :

$$Log_{p_1} : \begin{cases} M \to T_{p_1}M \\ p_2 \to v | Exp_{p_1}(v) = p_2 \end{cases}$$

Pour un point de base p_1 de la variété, Log_{p_1} prend donc en entrée un autre point de la variété et renvoie le vecteur tangent partant de p_1 dans la direction de p_2. Pour un espace Euclidien (cas particulier des variétés), l'unique géodésique vérifiant $\gamma(0) = X$ et $\dot{\gamma}(0) = Y$ est tout simplement $\gamma(t) = X + tY$. On en déduit que sur un espace plat, on a $Exp_X(Y) = X + Y$ et $Log_X(Y) = Y - X$.

Sur la Fig. 4.2, extrait de [52], on représente deux vecteurs tangents à la Sphère dans $\mathbb{R}^3$ replacés sur la variété via l'exponentielle de Riemann.

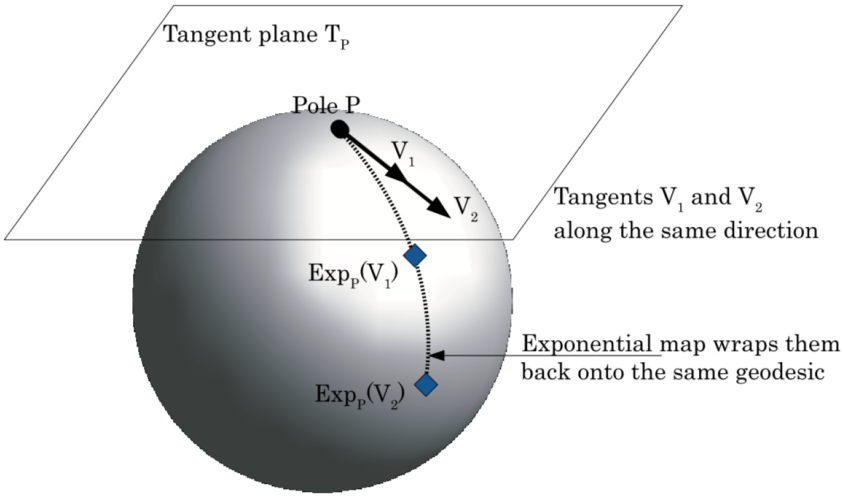


FIGURE 4.3 – Illustration de l'exponentielle de Riemann sur la variété de la Sphère dans $\mathbb{R}^3$. Depuis un point de départ P et une vitesse initiale V_1, on obtient $Exp_P(V_1) = \gamma_{P,V_1}(1)$. Pour une vitesse initiale V_2 dans la même direction mais de norme plus importante, on retrouve la même trajectoire mais avec une vélocité plus importante.

L'exponentielle de Riemann est très utile pour adapter l'algorithme de descente de gradient à une variété. En effet, pour déterminer le minimum d'une fonction f sur un espace Euclidien, il est possible d'appliquer une descente de gradient avec comme itération $x_{t+1} = x_t - \alpha grad(f)_{x_t}$ (avec $\alpha > 0$ le pas). De même sur une variété, $grad(f)_{x_t}$ est un vecteur tangent qui nous donne la direction à suivre pour minimiser la fonction. Il suffit donc d'appliquer l'exponentielle de Riemann pour retourner sur la variété et la précédente itération devient $x_{t+1} = Exp_{x_t}(-\alpha grad(f)_{x_t})$.

4.2.3 Moyenne de Fréchet

Pour une variété M, on a défini une distance. Il faut maintenant définir une moyenne cohérente avec cette distance. Cette moyenne, appelée moyenne de Fréchet (ou Karcher) est définie par :

$$\overline{U} = \underset{X \in M}{\arg\min} f_{moy}(X) = \underset{X \in M}{\arg\min} \sum_{i=1}^{n} d_M(X, U_i)^2, \ \forall\, U_1, ..., U_n \in M. \tag{4.1}$$

En dérivant cette équation, comme proposé dans [73], on obtient une nouvelle condition sur la moyenne de Fréchet $\overline{U}$:

$$-\, grad(f_{moy})(\overline{U}) = \sum_{i=1}^{n} Log_{\overline{U}}(U_i) = 0, \ \forall U_1, ..., U_n \in M. \tag{4.2}$$

Grâce à la dérivation de la fonction f_{moy} (qui fait apparaître le logarithme de Riemann), on peut donc appliquer une descente de gradient (Algo. 1) sur n'importe quelle variété pour définir une moyenne de Fréchet.

Algorithm 1: Moyenne de Fréchet.

Entrée : $X_1, X_2, ..., X_N \in M$;
$\mu_0 = X_1$
while $\mu_k \neq \mu_{k+1}$ **do**
  $\Delta\mu = \frac{1}{N} \sum_{i=1}^{N} Log_{\mu_k}(X_i)$;
  $\mu_{k+1} = Exp_{\mu_k}(\Delta\mu)$;
end
Sortie : μ

On propose en Fig. 4.4 une illustration de cet algorithme pour quatre éléments. On calcule le vecteur directionnel (donné par le logarithme de Riemann) entre l'approximation de la moyenne à t et chacun des éléments. L'exponentielle de Riemann permet ensuite de remettre la résultante de ces vecteurs sur la variété. On peut vérifier rapidement ce que donne cet algorithme sur un espace Euclidien E. Soit $X_1, X_2, ..., X_N \in E$. On a :

$$\Delta\mu = \frac{1}{N} \sum_{i=1}^{N} Log_{\mu_k}(X_i) = \frac{1}{N} \sum_{i=1}^{N} (X_i - \mu_0)$$

Et,

$$\mu_1 = Exp_{\mu_0}(\Delta\mu) = \mu_0 + \frac{1}{N}\sum_{i=1}^{N}(X_i - \mu_0) = \frac{1}{N}\sum_{i=1}^{N} X_i = \overline{X}$$

On retrouve donc directement dès la première itération l'expression classique d'une moyenne sur un espace Euclidien.

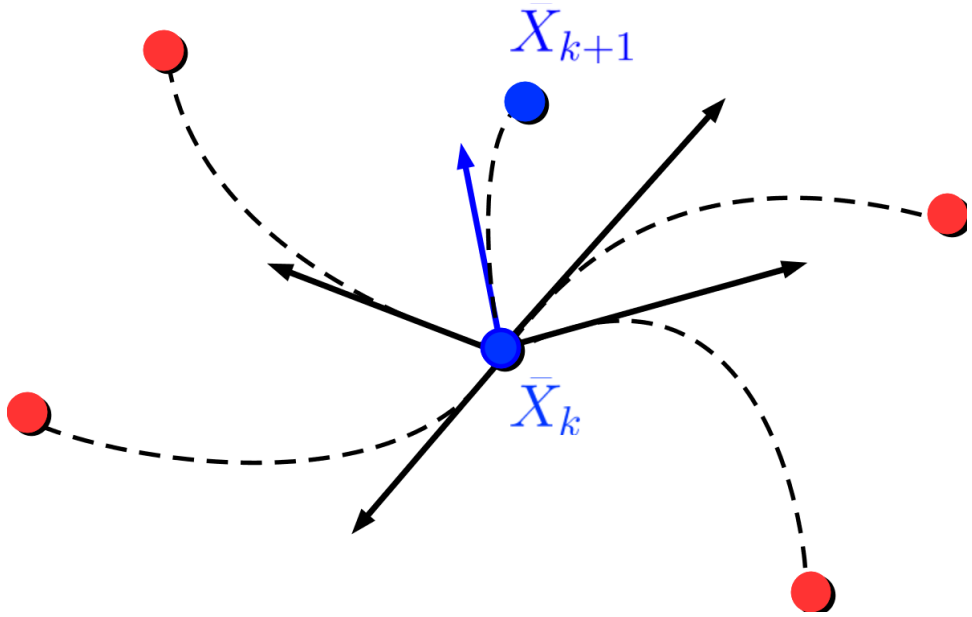


FIGURE 4.4 – Illustration d'une itération de la descente de gradient de Riemann. Les vecteurs tangents (en noirs) donnent la direction dans l'espace tangent $T_{X_k}M$ pour atteindre les quatre éléments de la variété. On en déduit la résultante (en bleu). Ensuite, via l'exponentielle de Riemann, on obtient l'itération suivante de la moyenne estimée.

4.2.4 Résumé

On propose ici la Table 4.1 qui résume les différentes notions abordées. Une analogie est réalisée entre les notions sur l'espace Euclidien et sur une variété de Riemann. Cette analogie permet d'adapter de nombreux algorithmes définis classiquement sur un espace Euclidien à une variété.

4.3 Géométrie des variétés proposées

4.3.1 Géométrie de $S(n)$

La Sphère (unité) [74] dans l'espace ambiant $\mathbb{R}^n$ est définie par $S(n) = \{X \in \mathbb{R}^n :< X, X >= X^T X = 1\}$ avec $< ., . >$ le produit scalaire usuel sur $\mathbb{R}^n$. Une courbe $\gamma : \mathbb{R} \to S(n)$ vérifie donc à

x, y appartenant à	**Espace Euclidien**	**Variété de Riemann**
Exemple	$\mathbb{R}^n, \mathbb{R}^{n\times p}$,...	$S_n, V_{n,p}, \mathbb{O}(n)$,...
Soustraction	$\overrightarrow{xy} = y - x$	$\overrightarrow{xy} = Log_x(y)$
Addition	$x + y$	$Exp_x(y)$
Distance	$d(x, y) = \|y - x\|$	$d(x, y) = \|\overrightarrow{xy}\|_x$
Définition de la moyenne	$\sum_{i=1}^n (U_i - \overline{U}) = 0$	$\sum_{i=1}^n Log_{\overline{U}}(U_i) = 0$
Descente de gradient pour f	$x_{t+1} = x_t - \alpha grad(f)_{x_t}$	$x_{t+1} = Exp_{x_t}(-\alpha grad(f)_{x_t})$.
Géodésique de p_1 à p_2	$\gamma(t) = p_1 + t\overrightarrow{p_1p_2}$	$\gamma(t) = Exp_{p_1}(t\overrightarrow{p_1p_2})$

TABLE 4.1 – Analogie entre la géométrie Euclidienne et la géométrie de Riemann.

tout instant t :

$$< \gamma(t), \gamma(t) >= 1$$

En dérivant, on obtient,

$$< \gamma'(t), \gamma(t) > + < \gamma(t), \gamma'(t) >= 0$$

Par symétrie du produit scalaire, on en déduit l'espace tangent à $X \in S(n)$, défini par :

$$T_X S = \{U \in \mathbb{R}^n :< U, X >= U^T X = 0\}$$

La distance entre X, Y de $S(n)$ est définie par :

$$d(X, Y) = arccos(X^T Y)$$

Pour $(X, U) \in S(n) \times T_X S(n)$, on peut définir une géodésique γ sur $[0, 1]$ vers $S(n)$ par :

$$\gamma(t) = cos(t\|U\|)X + \frac{sin(t\|U\|)}{\|U\|}U$$

On vérifie facilement qu'on a bien pour tout t, $\gamma(t)^T\gamma(t) = 1$, et on a $\gamma(0) = X, \gamma'(0) = U$. On en déduit donc directement par définition l'exponentielle de Riemann définie par :

$$Exp_X(U) = \gamma(1) = cos(\|U\|)X + \frac{sin(\|U\|)}{\|U\|}U$$

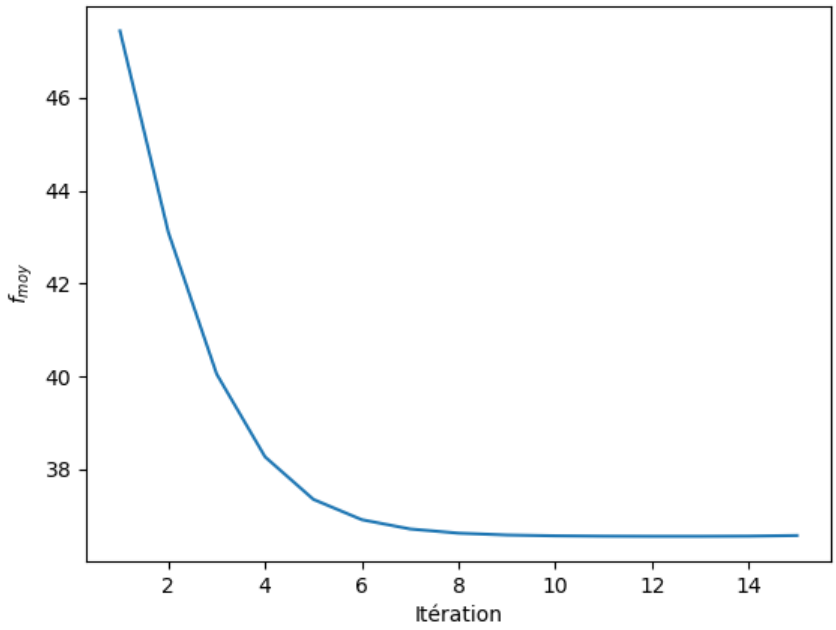


FIGURE 4.5 – Évolution de la quantité à minimiser f_{moy} suivant le nombre d'itérations de l'algorithme 1, appliqué à 20 éléments de la Sphère. Comme attendu, on remarque une décroissance de f_{moy}, puis une stagnation dès la $8^{ème}$ itération.

Le logarithme de Riemann est défini comme inverse de cette fonction par :

$$Log_X(Y) = \frac{d(X,Y)}{\|P_X(Y-X)\|} P_X(Y-X)$$

avec $P_X(H) = H - X^T H$, pour tout $(X, H) \in S(n) \times \mathbb{R}^n$.

Grâce à ces deux fonctions, on peut déterminer une moyenne sur la Sphère. en utilisant la distance en plus, on peut donc appliquer l'ensemble des algorithmes de clustering.

Pour obtenir une validation rapide de l'efficacité de l'algorithme 1 pour déterminer la moyenne de Fréchet, nous l'avons appliqué à 20 éléments pris au hasard sur la Sphère $S(101)$ (d'une dimension de 100). En Fig. 4.5, la quantité à minimiser f_{moy} (déterminée par l'Éq. 4.1) est tracée en fonction du nombre d'itérations de l'algorithme 1. On remarque une décroissance rapide jusqu'à la $8^{ème}$ itération puis une stagnation.

4.3.2 Géométrie de la variété de Grassmann représentée par des éléments de la variété de Stiefel

La variété de Stiefel $V_{n,p}$ est l'ensemble de toutes les familles des p orthonormaux vecteurs de $\mathbb{R}^n$. Pour $A = (a_1|a_2|...|a_p) \in V_{n,p}$, on a $A^TA = (< a_i, a_j >)_{1\leq i,j\leq p}$. On peut donc noter $V_{n,p} = \{U \in \mathbb{R}^{n\times p} : U^TU = I_p\}$.

Cette variété est dans l'espace ambiant $\mathbb{R}^{n\times p}$ qui est de dimension np. Par rapport à cette espace de dimension np, un élément $V = (v_1|v_2|...|v_p)$ de la variété de Stiefel est soumis à plusieurs contraintes : le premier vecteur colonne v_1 doit être normé (une contrainte), le deuxième doit être orthogonal au premier et normé (deux contraintes), le troisième doit être orthogonal aux deux premiers vecteurs et normé (trois contraintes),... Ainsi, on en déduit la dimension de cette variété :

$$\begin{aligned} d_{Stiefel} &= np - (1 + 2 + ... + p) \\ &= np - \frac{1}{2}p(p+1) \end{aligned}$$

Soit $X \in V_{n,p}$ et une courbe Y sur cette variété qui démarre de X. On a donc :

$$\begin{aligned} &Y(t)^TY(t) = I_p \\ &\dot{Y}(t)^TY(t) + Y(t)^T\dot{Y}(t) = 0 \text{ en dérivant} \\ &\dot{Y}(0)X = -X^T\dot{Y}(0) \text{ en } t = 0 \end{aligned}$$

On en déduit que l'espace tangent à $X \in V_{n,p}$, noté $T_XV_{n,p}$ est défini par :

$$T_XV_{n,p} = \{X\Omega + X_\perp K, \Omega^T = -\Omega \in \mathbb{R}^{p\times p}, K \in \mathbb{R}^{(n-p)\times p}\}$$

On peut remarquer que pour tout élément U de la variété de Stiefel, il existe $Q \in O(n)$ tel que $U = QI_{n,p}$. En effet, il suffit de compléter U en une base orthonormale de $\mathbb{R}^n$ pour obtenir un

élément de $O(n)$ tel que ses p premières colonnes soient les mêmes que celles de U. De plus, on a :

$$Q^* = Q \begin{pmatrix} I_p & 0 \\ 0 & Q'_{n,p} \end{pmatrix}$$

qui vérifie aussi $A = Q^* I_{n,p}$ pour tout $Q'_{n,p} \in O(n-p)$. Il vient donc que $V_{n,p}$ est un espace homogène à $O(n)/O(n-p)$.

Une autre façon d'interpréter un élément U de la variété de Stiefel est de s'intéresser au sous-espace de $\mathbb{R}^n$ engendré par les vecteurs colonnes de V. Dans ce cas-là, on se place sur la variété de Grassmann $Gr_{n,p}$ (l'ensemble des p sous-espaces de $\mathbb{R}^n$). Cependant, un élément P de la variété de Grassmann n'est pas représenté de manière unique par un élément V de la variété de Stiefel. En effet, le sous-espace engendré par les colonnes de U est stable par l'action à gauche de la variété $O(p)$: $P = [U] = UO :\in O(p)$. La variété de Grassmann peut donc être vue comme le quotient de variété $V_{n,p}/O(p) = O(n)/O(n-p)O(p)$. Cette structure quotient confirme que la variété $Gr_{n,p}$ est bien une variété différentielle. On en déduit directement sa dimension :

$$\begin{aligned} d_{Gr_{n,p}} &= d_{V_{n,p}} - d_{O(p)} \\ &= np - \frac{1}{2}p(p+1) - \frac{1}{2}p(p-1) \\ &= p(n-p) \end{aligned}$$

Une illustration de la variété de Grassmann comme quotient de $V_{n,p}/O(p)$ est proposée en Fig. 4.6.

L'avantage de cette interprétation d'un élément de la variété de Stiefel comme représentant d'un élément de la variété de Grassmann est la définition intuitive de la géodésique qui en découle. Prenons l'exemple de deux vecteurs $u_1, u_2 \in \mathbb{R}^n$. Si on s'intéresse aux droites $D_1 = vect(u_1), D_2 = vect(u_1) \in Gr_{1,n}$ engendrées par ces vecteurs, une distance naturelle entre ces deux droites est alors l'angle formé par ces deux vecteurs. On peut d'ailleurs remarquer qu'en normalisant u_1 et u_2, on obtient deux éléments de la Sphère $S(n)$ et on obtient la même distance. De la même manière, le

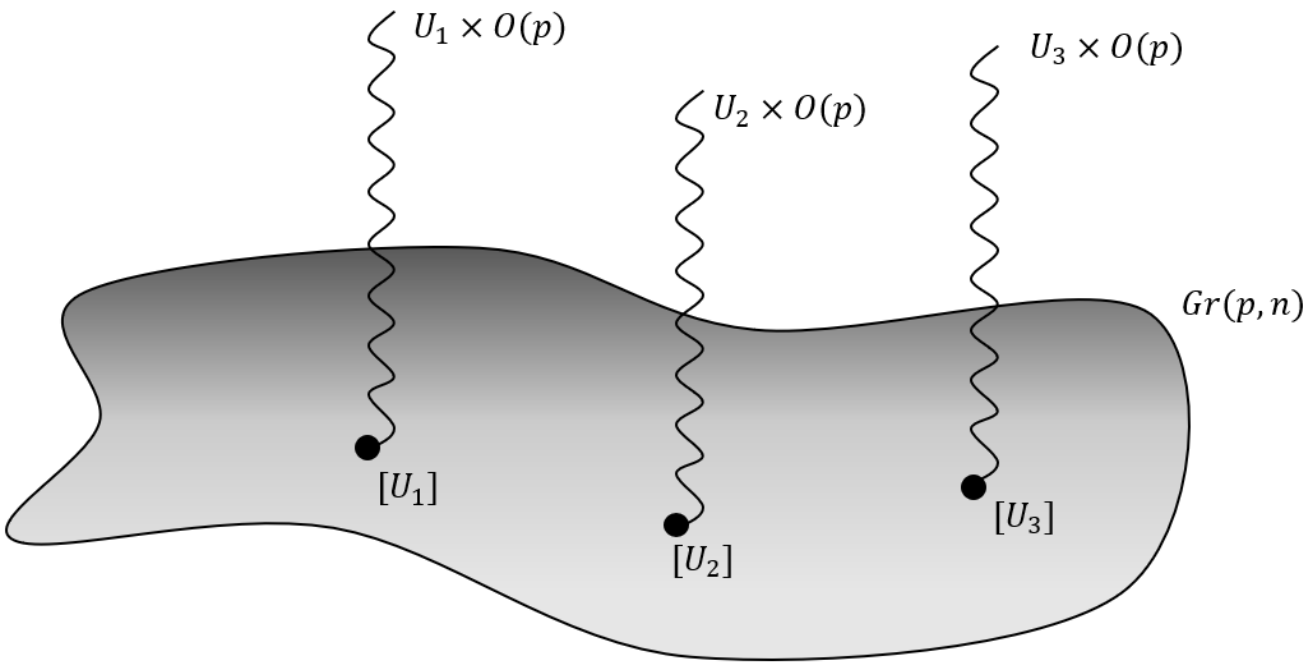


FIGURE 4.6 – La variété de Grassmann est représentée comme des classes d'équivalences sur la variété de Stiefel. Un point P de cette variété est donc caractérisé par un élément U de la variété de Stiefel avec $P = [U] = \{UO : O \in O(p)\}$.

chemin le plus court sur $Gr_{p,n}$ est issu de rotations. Pour passer d'un p sous-espace G_1 défini par les colonnes de $U_1 \in V_{n,p}$ à un autre G_2 (défini par U_2), il suffit de réaliser une série de rotations sur les colonnes de U_1 jusqu'à obtenir U_2. Pour assurer que ce chemin de rotations successives soit bien une géodésique, il faut que ce soit les rotations à l'énergie la plus faible. Pour cela on peut utiliser les angles principaux entre U_1 et U_2 [75]. Ainsi, pour $U_1, U_2 \in V_{n,p}$, on a :

$$distance_{Gr}([U_1],[U_2]) = \left(\sum_i^d \theta_i^2\right)^{\frac{1}{2}} \tag{4.3}$$

avec θ_i les angles principaux entre U_1 et U_2.

Pour déterminer les angles principaux, on calcule le spectre $\lambda_i, i = 1,..,p$ de $U_1^T U_2$. Alors, les $\theta_i = arccos(\lambda_i)$ sont les angles principaux de U_1 et U_2.

L'algorithme rapide 2 résume comment calculer cette distance.

La Fig. 4.7 illustre un exemple de distance entre deux éléments $A = (a_1|a_2)$ et $B = (b_1|b_2)$ de $V_{3,2}$. Une base du sous-espace $\mathcal{R}(A)$ engendrée par A est définie par les deux vecteurs de $\mathbb{R}^3$ a_1, a_2 (en vert dans la figure). Pour B, le sous-espace $\mathcal{R}(B)$ est engendré par les deux vecteurs

Algorithm 2: Calcul de la distance sur la variété de Grassmann représentée par la variété de Stiefel.

Entrée : $U_1, U_2 \in V_{n,p}$
$U_1^T U_2 = R\Sigma T$ (Singular Value Decomposition, SVD)
$\theta_i = arccos(\lambda_i), i = 1, \ldots, p$
Sortie : $d_{Gr}([U_1], [U_2]) = \left(\sum_{i=1}^{p} \theta_i^2\right)^{\frac{1}{2}}$

b_1, b_2 (en rouge sur la figure). Une possibilité de chemin de $\mathcal{R}(A)$ vers $\mathcal{R}(B)$ est alors une première rotation d'angle $\theta_1 = (a_1, b_1)$ (qui fait coïncider les vecteurs a_1 et b_1), suivie d'une rotation d'angle $\theta_2 = (a_2, b_2)$ (qui fait coïncider les vecteurs a_2 et b_2). Un autre chemin possible est une première rotation d'angle $\theta_1 = (a_1, b_2)$ (qui fait coïncider les vecteurs a_1 et b_2), suivie d'une rotation d'angle $\theta_2 = (a_2, b_1)$ (qui fait coïncider les vecteurs a_2 et b_1). Il est donc nécessaire de déterminer les rotations qui demandent le moins d'énergie.

En utilisant cette distance, on peut donc appliquer la majorité des algorithmes de clustering et de réduction de dimension : Hiérarchique, HDBSCAN, et UMAP. Cependant, pour appliquer K-means, on a besoin de définir le logarithme et l'exponentielle de Riemann qui permettent de déterminer la moyenne de Fréchet. Les algorithmes 3 et 4, extrait de [76], permettent de calculer ces fonctions.

Algorithm 3: Exponentielle de Riemann sur la variété de Grassmann représentée par la variété de Stiefel.

Entrée : $U \in V_{n,p}$, $\Delta \in T_{[U]}Gr_{n,p}$
SVD de Δ : $\Delta = Q \sum V$
Sortie : $Exp_U^{Gr}(\Delta) = UVcos(\sum)V^T + Qsin(\sum)V^T$;

Algorithm 4: Logarithme de Riemann sur la variété de Grassmann représentée par la variété de Stiefel.

Entrée : $U \in V_{n,p}$, $\overline{U} \in V_{n,p}$;
$M = \overline{U}^T U$
SVD de M : $M = QSR^T$
$\overline{U}' = \overline{U}(QR^T)$
$N = (I_n - UU^T)\overline{U}'$
SVD de N : $N = QSR^T$
Sortie : $Log_U^{Gr}(\overline{U}) = Q \arcsin(S) R^T$

Comme sur la variété de la Sphère, on propose d'appliquer à 20 éléments pris au hasard sur

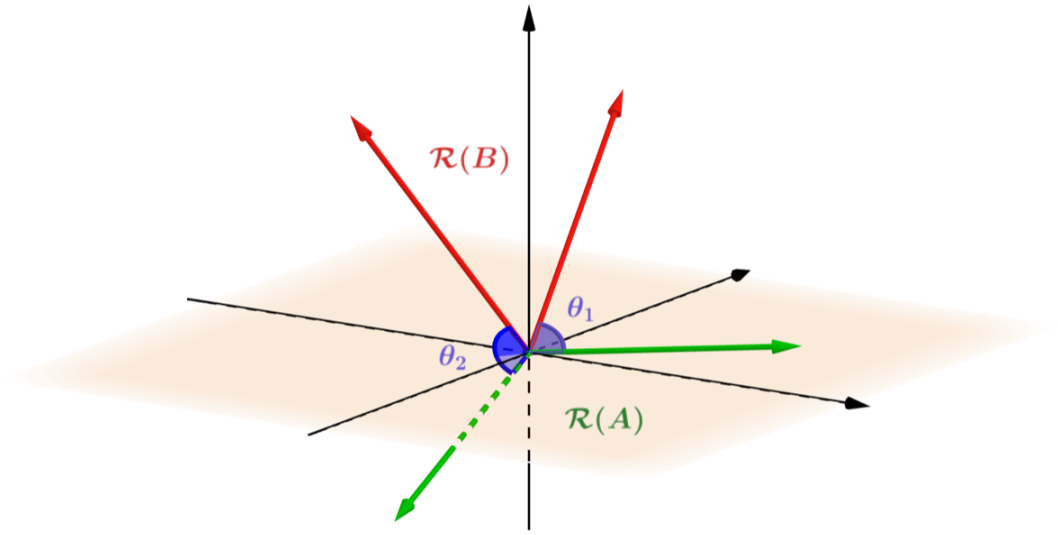


FIGURE 4.7 – Exemple d'angles principaux entre deux éléments A et B de $V_{3,2}$. Les angles principaux caractérisent la distance entre les sous-espaces engendrés par les colonnes de A et B, notés $\mathcal{R}(A)$ et $\mathcal{R}(B)$ respectivement. Seulement deux rotations d'angle sont nécessaires pour passer de $\mathcal{R}(A)$ à $\mathcal{R}(B)$.

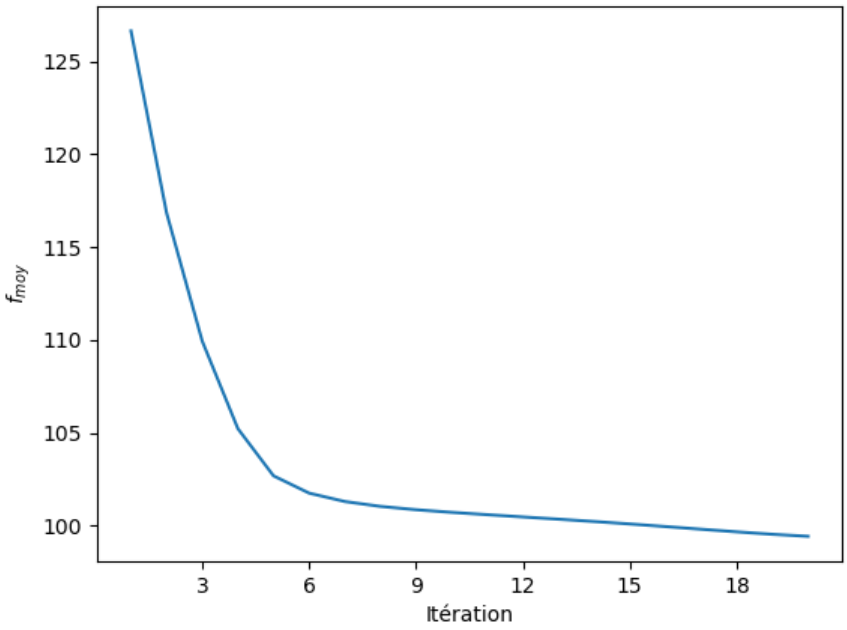


FIGURE 4.8 – Évolution de la quantité à minimiser f_{moy} suivant le nombre d'itérations de l'algorithme 1, appliqué à 20 éléments de la variété de Grassmann. Comme attendu, on remarque une décroissance de f_{moy}.

la variété de Stiefel $V_{16,9}$ (vues comme éléments de $Gr_{16,9}$). En Fig. 4.8, la quantité à minimiser f_{moy} (déterminée par l'Éq. 4.1) est tracée en fonction du nombre d'itération de l'algorithme 1. On remarque une décroissance rapide jusqu'à la $6^{ème}$ itération puis une décroissance légère.

4.3.3 Géométrie de $SPD(n)$

Pour comparer les matrices de $P_{n,p}$ obtenues à partir des matrices de trajectoires, nous avons décidé de les transformer en éléments de la variété des matrices symétriques strictement définies positives (SPD) (valeurs propres > 0). On rappelle que dans cette optique, on ajoute à chaque élément P de $P_{n,p}$ ϵI_n (avec ϵ faible) pour s'assurer que $P' := P+\epsilon I_n$ soit bien une matrice inversible et donc un élément de $SPD(n)$. Nous allons donc préciser la géométrie de la variété SPD. Cette variété est de forme conique. Par exemple, un élément de $SPD(2)$ est à l'intérieur d'un cône dans

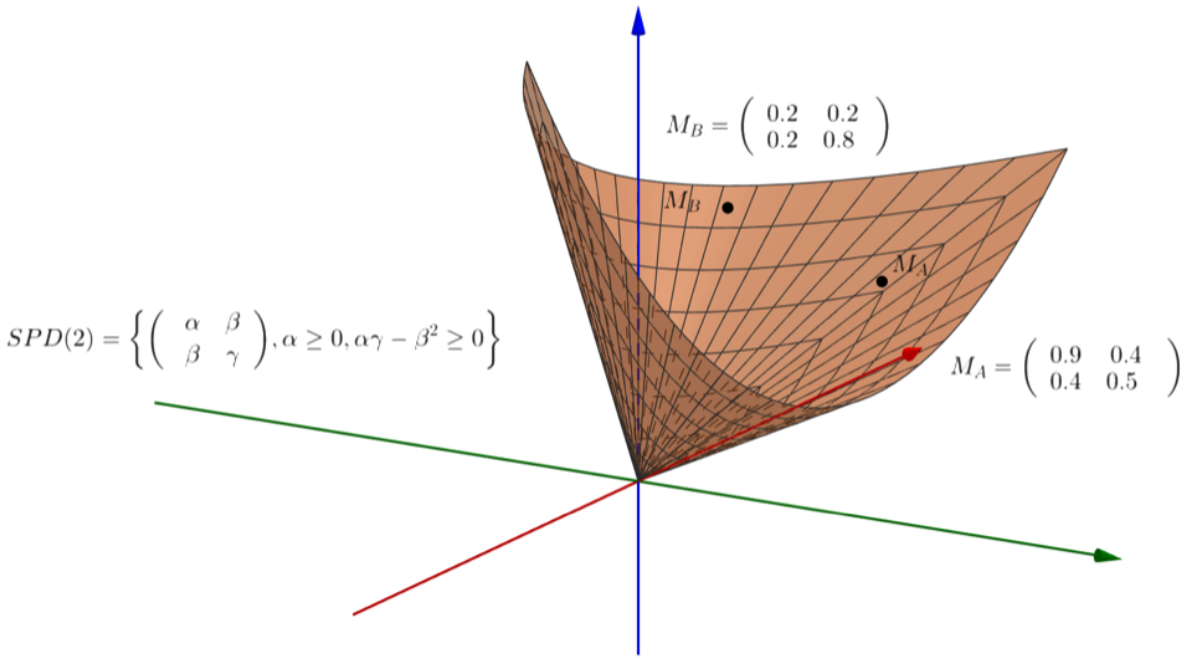


FIGURE 4.9 – Représentation de 2 matrices de $SPD(2)$ dans $\mathbb{R}^3$.

$\mathbb{R}^3$. En effet, on a :

$$SPD(2) = \left\{ M = \begin{pmatrix} \alpha & \beta \\ \\ \beta & \gamma \end{pmatrix}, \alpha \geq 0, \alpha\gamma - \beta^2 > 0 \right\}$$

La frontière de cet ensemble est alors donnée par l'équation $xy - z^2 = 0$ ce qui nous donne l'équation d'un cône dans $\mathbb{R}^3$. Ainsi, une matrice $M \in SPD(2)$ peut être représentée dans $\mathbb{R}^3$ avec α, β, γ comme coordonnées. Cette matrice est alors à l'intérieur du cône. La représentation de 3 éléments de $SPD(2)$ est proposée en Fig. 4.9.

La distance sur la variété SPD [77] est définie par :

$$d(A, B) = \|log(A^{-\frac{1}{2}} B A^{-\frac{1}{2}})\|$$

avec $\|.\|$ la norme de Frobenius.

L'exponentielle de Riemann est définie par :

$$Exp_A(B) = A^{\frac{1}{2}} exp(A^{-\frac{1}{2}} B A^{-\frac{1}{2}}) A^{\frac{1}{2}}, (A, B) \in (SPD(n), T_A SPD)$$

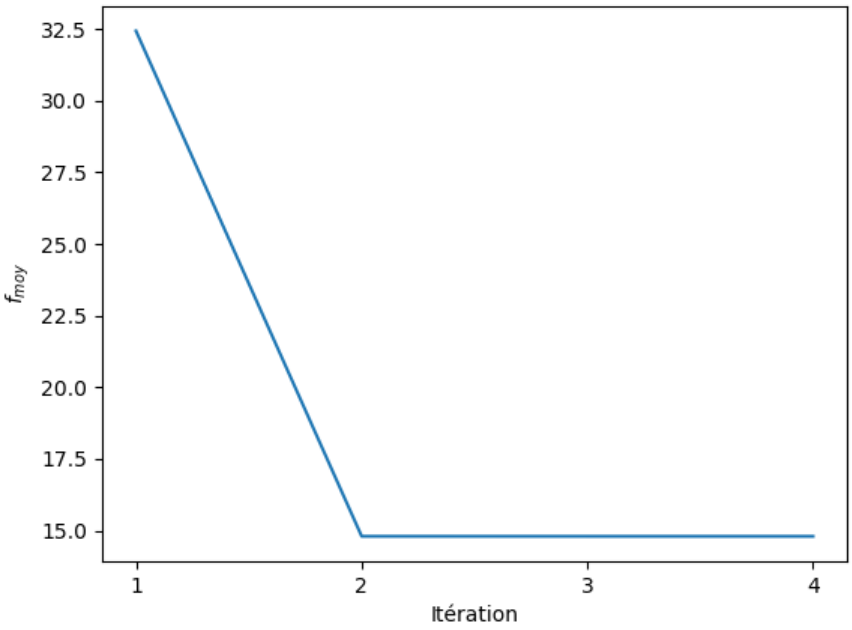


FIGURE 4.10 – Évolution de la quantité à minimiser f_{moy} suivant le nombre d'itérations de l'algorithme 1, appliqué à 20 éléments de la variété P_{14}. Comme attendu, on remarque une décroissance de f_{moy}.

avec $T_A SPD$ l'espace tangent à la variété SPD en A.

Cette expression admet une unique application inverse définie par :

$$Log_A(B) = A^{\frac{1}{2}} exp(A^{-\frac{1}{2}} B A^{-\frac{1}{2}}) A^{\frac{1}{2}}, (A, B) \in SPD(n)$$

Comme pour les deux autres géométries, nous avons appliqué l'algorithme 1 à 20 éléments pris au hasard sur la variété P_{14} (d'une dimension de 105). En Fig. 4.10, la quantité à minimiser f_{moy} (déterminée par l'Éq. 4.1) est tracée en fonction du nombre d'itérations de l'algorithme 1. On remarque une décroissance puis une stagnation dès la $2^{ème}$ itération.

4.3.4 Résumé

On propose, en Table 4.2, un résumé des différents éléments présentés dans cette section.

Géométrie, $X, Y \in M, v \in T_X M$	$Gr_{n,p}$	$S(n)$	$SPD(n)$
Distance	See algo 2	$d(X,Y) = arccos(X^T P)$	$d(X,Y) = \|log(X^{-\frac{1}{2}} Y X^{-\frac{1}{2}})\|$
Exponentielle de Riemann	See algo 3	$Exp_X(v) = cos(\|v\|) \times x + \frac{sin(\|v\|)}{\|v\|} \times y$	$Exp_X(v) = X^{1/2} exp(X^{-\frac{1}{2}} v X^{-\frac{1}{2}}) X^{1/2}$
Logarithme de Riemann	See algo 4	$Log_X(Y) = \frac{d(X,Y)}{\sqrt{1-(X^T Y)^2}}(Y - (X^T P)X$	$Log_X(Y) = X^{1/2} log(X^{-\frac{1}{2}} Y X^{-\frac{1}{2}}) X^{1/2}$
Moyenne de Fréchet	See algo 1	See algo 1	See algo 1

TABLE 4.2 – Résumé des différents éléments de géométrie Riemannienne sur plusieurs variétés.

4.4 Amélioration UMAP

Comme nous allons le montrer dans le chapitre suivant, UMAP a permis de nettement améliorer les résultats de clustering. Nous avons donc cherché à améliorer cet algorithme. En particulier, vu que nous proposons plusieurs espaces de départ différents ($\mathbb{R}^l$, $\mathbb{R}^{n\times p}$, $S(n)$, $Gr_{n,p}$, $SPD(n)$), nous nous sommes posé la question du choix de la géométrie d'arrivée d'UMAP. Normalement, UMAP projette sur un espace Euclidien. Nous avons donc voulu l'adapter pour pouvoir projeter sur n'importe quel espace et en particulier sur la même variété (mais de dimension réduite) que la variété d'origine choisie pour analyser les matrices de trajectoire. Ainsi, au lieu d'avoir une réduction de dimension qui projette d'une variété M de dimension p vers un espace Euclidien de dimension réduite d $\mathbb{R}^d$ ($d < p$), nous souhaitons déterminer une réduction de dimension qui va de la variété M de dimension p vers la même variété M de dimension d.

Comme vu précédemment au chapitre 2, UMAP initialise les éléments sur l'espace Euclidien en utilisant les vecteurs propres du Laplacien comme coordonnées pour chaque élément de la base de données. Nous avons décidé d'opter pour une initialisation aléatoire sur la variété de notre choix. Ensuite, UMAP réalise un "forced directed graph layout" pour minimiser la cross-entropy, ce qui est l'objectif final de cet algorithme. Dans ce but, des forces d'attractions et de répulsions dérivant de la cross-entropy sont utilisées. Ainsi, pour deux éléments X, Y de l'espace d'arrivée, X est attiré (resp. répoussé) dans la direction de Y avec une magnitude dépendant de la distance entre X et Y (dans l'espace d'arrivée) et du poids entre ces deux éléments dans le graphe de l'espace d'origine. Si l'on souhaite maintenant aller de M vers M, il faut adapter ce "forced directed graph layout" :

— La distance entre X et Y doit maintenant être la distance sur la variété M.

— La direction entre X et Y doit être adaptée pour prendre en compte la courbure de la variété M. Pour cela, on peut utiliser le logarithme et l'exponentielle de Riemann.

Nous avions pour l'espace Euclidien X qui était attiré par Y à chaque itération par :

$$X \leftarrow X + F_{att}(X,Y)(Y - X) \text{ avec } F_{att}(X,Y) > 0$$

qui devient sur la variété M :

$$X \leftarrow Exp_X(F_{att}(X,Y)Log_X(Y)) \tag{4.4}$$

en suivant les analogies proposées dans le chapitre 4 entre géométrie Euclidienne et géométrie de Riemann.

De même, on a pour la répulsion à chaque itération :

$$X \leftarrow X + F_{rep}(X,Y)(Y-X) \text{ avec } F_{rep}(X,Y) < 0$$

$$\Downarrow$$

$$X \leftarrow Exp_X(F_{rep}(X,Y)Log_X(Y)) \tag{4.5}$$

L'objectif d'UMAP est de réduire la cross-entropy entre le graphe originel et le graphe dans l'espace réduit. Pour valider notre nouvelle réduction de dimension allant de M vers M, on a donc calculé la cross-entropy à chaque itération du "forced directed graph layout". On montre le type de courbe que l'on obtient en Fig. 4.11. Cette figure est pour une base de données (issue de UCR Time Series Archive) de 70 éléments placés sur $S(n)$ via le Delay Coordinate Embedding. Nous avons ensuite réduit ses éléments vers $S(11)$ avec notre adaptation. On remarque que malgré quelques irrégularités la cross-entropy diminue clairement au fil des itérations.

Nous proposons en Table 4.3 quelques résultats obtenus avec cette adaptation. Pour plusieurs bases de données de UCR Time Series Archive, nous avons placé les séries temporelles sur la Sphère (resp. la variété de Grassmann) (via le Delay Coordinate Embedding) puis nous avons appliqué la réduction de dimension classique UMAP vers $\mathbb{R}^{10}$ et notre adaptation vers la Sphère réduite $S(11)$ de dimension 10 (resp. la variété de Grassmann $Gr_{7,2}$ de dimension 10).

Grâce à notre adaptation, on retrouve des cross-entropy très similaires (que ce soit sur la Sphère ou sur la variété de Grassmann) bien que légèrement supérieures à la version classique d'UMAP. Cela s'explique par le problème de l'initialisation, où pour le moment nous n'avons pas trouvé d'autres solutions que de faire un placement aléatoire. Cela complique la vitesse de convergence du "forced directed graph layout" et on peut converger vers un minimum local de moins bonne qualité

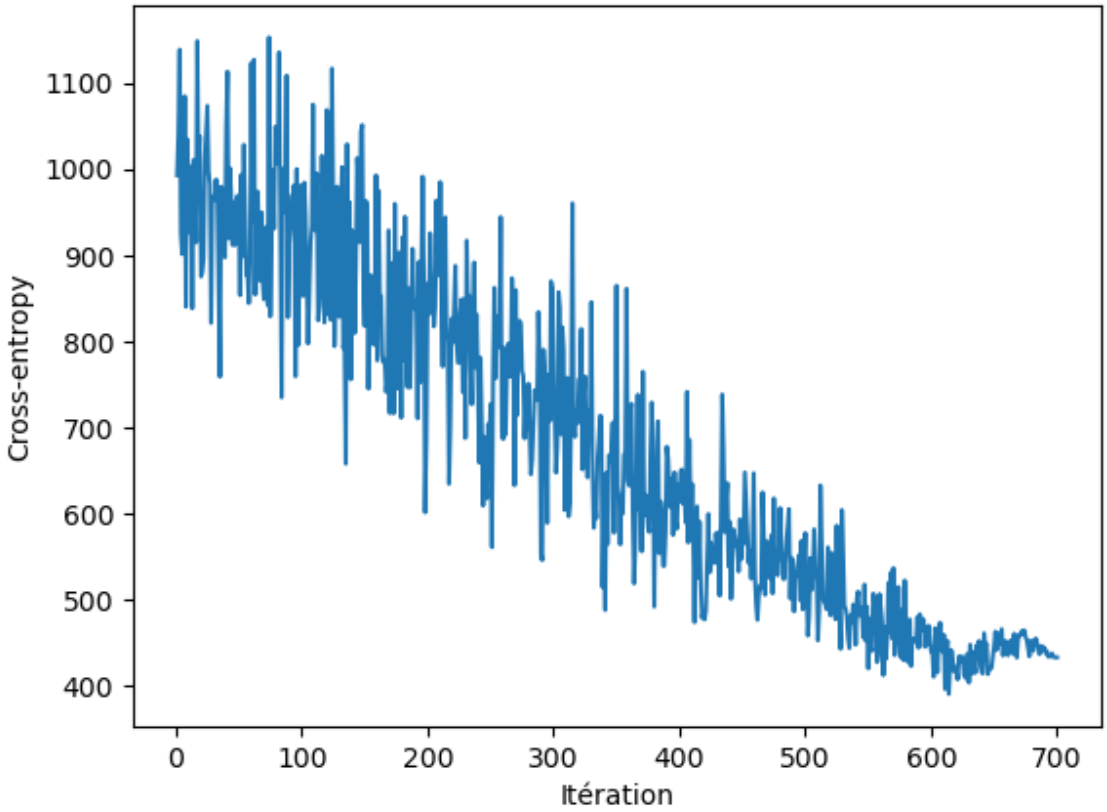


FIGURE 4.11 – Évolution de la cross-entropy pour 70 éléments sur la Sphère. On remarque une décroissance globale au fil des itérations.

que ce qu'on obtiendrait avec une meilleure initialisation.

De plus, le temps de calcul est lent ce qui nous a empêché de lancer cette adaptation de UMAP sur l'ensemble des bases de données de UCR Time Series Archive.

4.5 Synthèse

Dans ce chapitre, nous nous sommes penché sur la géométrie des espaces sur lesquels on définit nos matrices de trajectoires. Ces matrices sont placées sur des espaces particuliers, des variétés. Ce sont des espaces non-Euclidiens qui sont localement assimilables à un espace Euclidien. Même si ils sont localement assimilables à un espace plat, il n'est pas possible de directement appliquer la géométrie Euclidienne sans perdre la notion de courbure de la variété. Sur les variétés

TABLE 4.3 – Comparaison de la cross-entropy avec UMAP sur l'espace Euclidien classique et notre adaptation sur la même variété que celle de départ.

Géométrie de départ	Géométrie d'arrivée	Nb bases de données	Cross-entropy moy.
Grassmann	Euclidien	7	258.64
Grassmann	Grassmann	7	265.32
Sphère	Euclidien	30	477.10
Sphère	Sphère	30	478.19

Riemanniennes (sous-catégorie des variétés), il est possible d'appliquer la géométrie de Riemann. Cette théorie assure qu'il est possible d'analyser localement la géométrie via un ensemble de cartes (un atlas), puis de les réunir pour obtenir la géométrie globale de la variété. En effet, les cartes permettent de définir localement un espace tangent Euclidien. On peut alors définir des produits scalaires locaux sur ces espaces tangents. L'atlas nous assure alors de réunir avec cohérence ces produits scalaires dans une métriques g, métrique de Riemann. Cette métrique nous permet alors de définir une distance globale d sur la variété. On définit ensuite la moyenne de Fréchet dépendante de d. Avec tous ces éléments-là, il est possible d'appliquer les différents algorithmes de clustering. Ainsi, les séries temporelles sont clusterisées comme éléments d'une variété ($Gr_{n,p}$, $S(n)$, $SPD(n)$,$\mathbb{R}^{n\times p}$).

Nous avons aussi adapté UMAP pour rester sur la géométrie de départ. On obtient des résultats intéressants avec des limitations encore importantes. Le temps de calcul, en particulier dû aux calculs des exponentielles et logarithmes de Riemann, est lourd. De plus, l'initialisation est pour le moment réaliser aléatoirement ce qui ralentit aussi la vitesse de convergence du Forced Directed Graph Layout.

Chapitre 5

Analyse des différentes pistes proposées

Lors des précédents chapitres, nous avons mis en en avant plusieurs pistes pour réaliser le clustering. Nous avons étudié plusieurs algorithmes de clustering (K-means, Hiérarchique Agglomerative, HDBSCAN), avec la possibilité d'utiliser UMAP comme réduction de dimension en étape préliminaire. Nous avons aussi proposé plusieurs variétés pour analyser les matrices issues du Delay Coordinate Embedding :

— L'espace Euclidien classique $\mathbb{R}^{n\times p}$ associé à la norme de Frobenius

— La Sphère $S(n)$

— La variété de Grassmann $Gr_{n,p}$ représentée par la variété de Stiefel $V_{n,p}$

— La variété des matrices symétriques strictement positives $SPD(n)$

Nous souhaitons maintenant évaluer ces méthodes pour déterminer le meilleur moyen de réaliser du clustering de séries temporelles uni-dimensionnelles, à travers une analyse comparative détaillée.

5.1 Création de l'analyse comparative

Pour réaliser notre analyse comparative, nous avons utilisé les bases de données disponibles sur le site `https://www.cs.ucr.edu/~eamonn/time_series_data/` (UCR Time Series Archive). Un résumé rapide de ces bases de données est présenté dans la Table 5.1. Le nombre de groupes dans une base de données varie entre 2 et 60. De plus, tandis que certaines bases de données ont un nombre important de séries temporelles, d'autres sont de taille beaucoup plus modeste. La longueur des séries temporelles (constante au sein d'une même base de données) varie aussi fortement entre 24 et 2709 occurrences.

Une fois que les résultats de clustering ont été déterminés, nous proposons de les comparer en utilisant le v-measure score puisque les résultats attendus sont connus.

TABLE 5.1 – Présentation des bases de données.

	Min	**Moy.**	**Max**
Nb de groupes	2	7.5	60
Nb de séries temporelles	16	432	8926
Longueur des séries temporelles	24	422	2709

5.2 Analyse des algorithmes de clustering

Nous proposons d'abord de comparer les algorithmes de clustering en utilisant la distance directe Euclidienne sans transformation des séries temporelles. Notre but ici est de mettre en évidence l'efficacité d'UMAP, en particulier quand il est couplé avec HDBSCAN. Dans cette optique, nous appliquons les algorithmes de clustering avec K-means, Hiérarchique et HDBSCAN, avec et sans UMAP en étape préliminaire. Nous comparons donc 6 résultats de clustering. En Fig. 5.1, le résumé de la création de cette analyse est proposé.

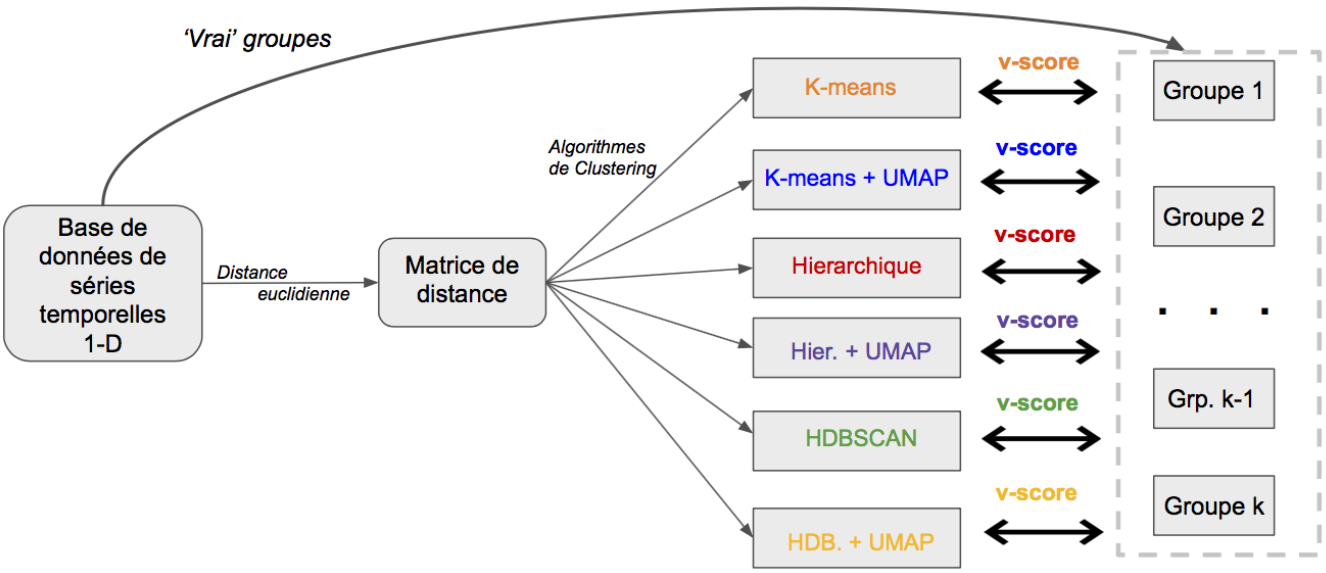


FIGURE 5.1 – Résumé de notre analyse comparative détaillée sur les algorithmes de clustering. Pour une base de données, 6 résultats de clustering ont été déterminés selon le choix d'algorithme de clustering et d'utilisation ou non d'UMAP. Ensuite, en utilisant le v-measure score, on compare ces résultats aux vrais groupes attendus dans les données.

On rappelle qu'ici deux des algorithmes de clustering (K-means et Hiérarchique) ont besoin de déterminer en avance le nombre de groupes qui doit être formé. Pour cela, comme précisé dans le chapitre 2, il est possible d'utiliser le silhouette score. Nous avons donc calculé les résultats de clustering pour un nombre de groupes qui varient entre 2 et le nombre de groupes réels de la base de données + 10. Le nombre de groupes maximisant le silhouette score est alors gardé.

Les résultats sont résumés dans la Table 5.2. On retrouve dans cette table :

— La moyenne des v-measure score sur l'ensemble des bases de données pour chacune des méthodes,

— L'écart-type de ce v-measure score,

— Le nombre de bases de données pour lesquelles une méthode est la meilleure. Par exemple, HDBSCAN sans UMAP obtient le meilleur résultat pour 12 bases de données (parmi 85).

— Pour un algorithme de clustering donné (K-means, Hiérarchique ou HDBSCAN), le pourcentage de bases de données pour lesquelles l'utilisation d'UMAP améliore le résultat. Par exemple, UMAP avant l'utilisation de K-means donne des meilleurs résultats que K-means seul dans 61% des bases de données.

Les résultats complets sont disponibles dans l'annexe A.

On peut tout d'abord remarquer que UMAP est une bonne étape préliminaire quelque soit l'algorithme de clustering. En effet, UMAP augmente le v-measure score pour la majorité des bases de données pour les 3 algorithmes. Cette amélioration se retrouve aussi dans une augmentation de la moyenne du v-measure score. Cependant, l'écart-type augmente légèrement pour K-means et pour Hiérarchique. Les résultats sont donc un peu moins stables dans ces cas. Pour HDBSCAN, UMAP améliore les résultats pour tous les indicateurs. Étant donné que ce sont deux algorithmes de même nature (basés sur la densité), il est logique de les voir particulièrement bien fonctionner ensemble.

Pour visualiser ces résultats, nous proposons de comparer les méthodes 2 à 2. Pour cela, on représente les bases de données comme des points dans le plan $[0, 1] \times [0, 1]$ avec comme coordonnées le v-measure score d'une première méthode et le v-measure score d'une deuxième méthode. On ajoute aussi une droite rouge $y = x$ pour clairement séparer l'espace en deux zones où une méthode domine l'autre.

	K-means	**Hiérarchique**	**HDBSCAN**	**UMAP+K.**
Moyenne	0.253	0.212	0.218	0.284
Écart-type	0.258	0.256	0.236	0.272
Nb meilleurs	15	10	12	15
Nb avec UMAP				61%

	UMAP+Hiér.	**UMAP+HDB.**
Moyenne	0.273	**0.292**
Écart-type	0.282	**0.250**
Nb meilleurs 15	8	**25**
Nb avec UMAP	66%	75%

TABLE 5.2 – Comparaison des algorithmes de clustering avec une distance Euclidienne directement appliquée aux séries temporelles uni-dimensionnelles.

Les Fig. 5.2, Fig. 5.3 et Fig. 5.4 donnent une représentation visuelle des bons résultats obtenu par UMAP. Dans la Fig. 5.2, nous pouvons remarquer que UMAP améliore clairement HDBSCAN dans une large majorité des cas. Pour un certain nombre de bases de données, UMAP + HDBSCAN donne des bons résultats alors que HDBSCAN est complètement faux : on le voit par le nombre de base de données sur la droite $x = 0$.

Seulement 26 bases de données sur 85 présentent des meilleurs résultats avec UMAP pour les 3 algorithmes de clustering à la fois. En particulier, UMAP améliore K-means pour 52 bases de données (groupe G1 des bases de données) et améliore Hiérarchique pour 56 bases de données (G2). Or, seulement 35 bases de données sont communes à G1 et G2. Cela indique que les résultats ne sont pas intrinsèques aux bases de données. En effet, il n'y a pas un groupe de bases de données particulièrement adapté à UMAP mais plutôt une amélioration globale des résultats.

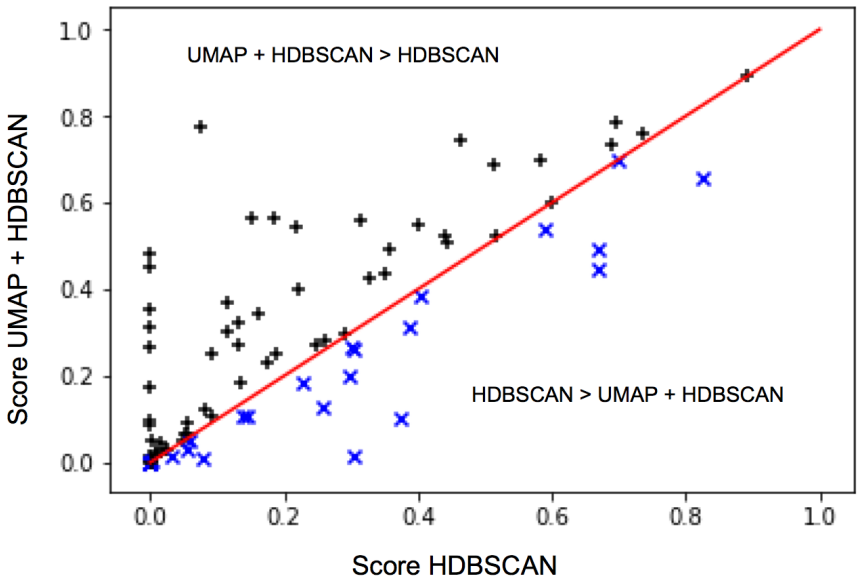


FIGURE 5.2 – Les bases de données sont représentées par le v-measure score de UMAP + HDBSCAN en fonction du v-measure score d'HDBSCAN. Le plan est séparé en deux zones : une où la première méthode a un meilleur score et l'autre où c'est la deuxième méthode qui a un meilleur score.

Enfin, parmi les six méthodes, UMAP associé à HDBSCAN propose clairement les meilleurs résultats. Cette méthode a la meilleure moyenne, un bon écart-type et un meilleur score pour 25 bases de données sur 85. En comparaison avec l'algorithme le plus connu K-means, on obtient une augmentation du v-measure score moyen de 15% et un meilleur résultat sur 62% des bases de données. En Fig. 5.5 une comparaison a été réalisée entre ces deux méthodes (K-means et UMAP + HDBSCAN).

On a donc mis en lumière dans cette section l'efficacité de UMAP en étape préliminaire, en particulier quand il est utilisé avec HDBSCAN. Ces résultats ont fait l'objet d'une publication [78] dans la conférence internationale ICPR 2020.

Dans la section suivante, nous nous posons maintenant la question de l'intérêt du Delay Coordinate Embedding et ensuite de la meilleure variété à adopter pour analyser les matrices de trajectoire. On vérifie aussi que UMAP associé à HDBSCAN reste la meilleure méthode de clustering, même pour des données issues de variétés avec des géométries différentes que l'espace Euclidien classique.

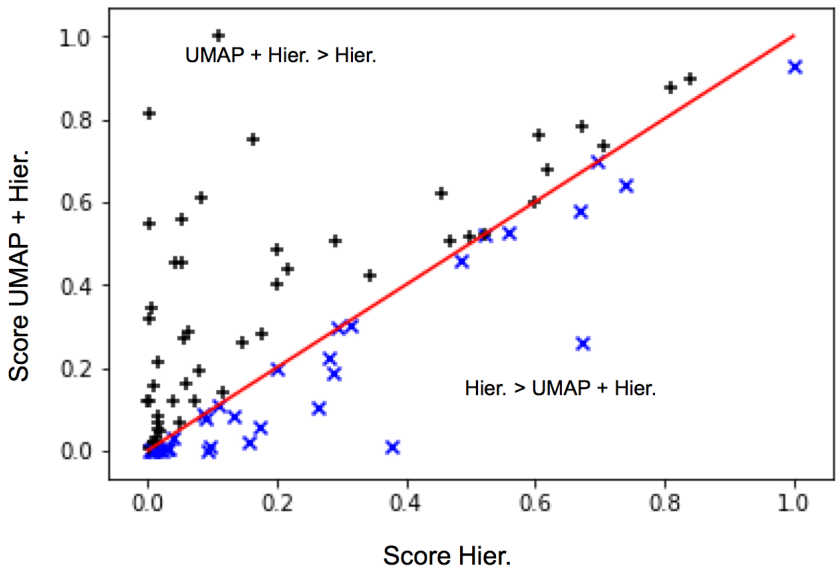


FIGURE 5.3 – Les bases de données sont représentées par le v-measure score de UMAP + Hiérarchique en fonction du v-measure score d'Hiérarchique. Le plan est séparé en deux zones : une où la première méthode a un meilleur score et l'autre où c'est la deuxième méthode qui a un meilleur score.

5.3 Comparaison des différentes géométries

5.3.1 Résultats sur des séries temporelles directement extraites de systèmes dynamiques

Pour avoir une première idée de l'efficacité du Delay Coordinate Embedding, nous proposons de l'utiliser sur des bases de données créées à partir de systèmes dynamiques. En effet, les séries temporelles uni-dimensionnelles issues de systèmes dynamiques devraient être particulièrement bien adaptées à l'utilisation du Delay Coordinate Embedding, car comme vu précédemment dans le chapitre 3, on se retrouve dans le cas du théorème de Takens qui assure une reconstruction pertinente du système dynamique réel derrière la série temporelle uni-dimensionnelle.

On crée donc une première base de données utilisant quatre systèmes dynamiques connus :

— Le système de Lorenz : Il est défini par le système d'équations différentielles de $\mathbb{R}^3$ vu précédemment (Éq. 3.2). Une visualisation de ce système est présentée en Fig. 5.6a. Cette courbe est la trajectoire d'un point soumis aux équations (3.2) avec comme paramètres $\rho = 28$, $\sigma = 10$ et $\beta = 8/3$.

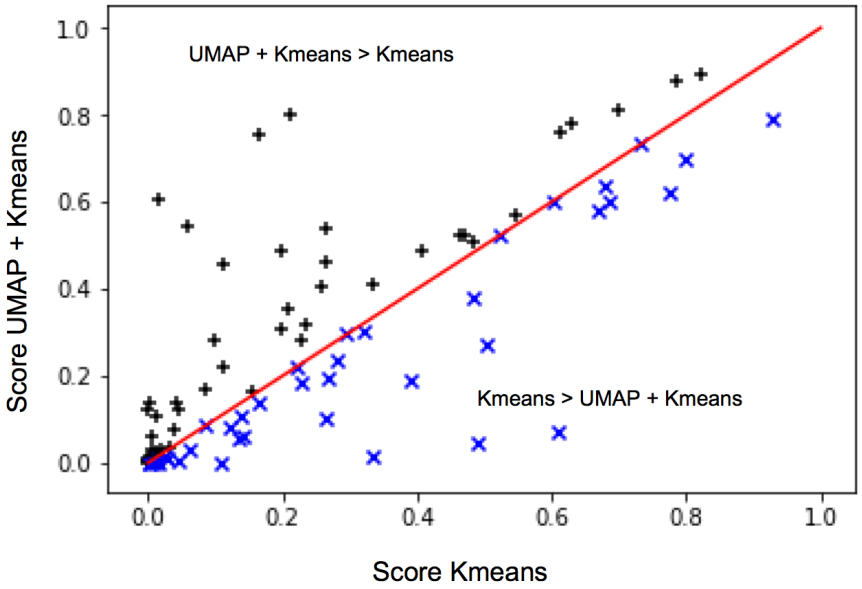


FIGURE 5.4 – Les bases de données sont représentées par le v-measure score de UMAP + K-means en fonction du v-measure score de K-means. Le plan est séparé en deux zones : une où la première méthode a un meilleur score et l'autre où c'est la deuxième méthode qui a un meilleur score.

— Le système de Rössler : C'est un système dynamique de $\mathbb{R}^3$. Le mouvement d'un point de $\mathbb{R}^3$ soumis à ce système est déterminé par le système (5.1) de trois équations différentielles couplées suivant :

$$\begin{cases} \dot{x}(t) = & -y(t) - z(t) \\ \dot{y}(t) = & x(t) + ay(t) \\ \dot{z}(t) = & b + z(t)(x(t) - c) \end{cases} \tag{5.1}$$

La Fig. 5.6b est obtenue avec $a = 0.1$, $b = 0.1$ et $c = 14$ comme paramètres.

— Le système d'Ikeda : C'est un système de $\mathbb{R}^2$ déterminé par le système d'équations suivant :

$$\begin{cases} x_{n+1} = 1 + u(x_n \cos(\theta_n) - y_n \sin(\theta_n)) \\ y_{n+1} = u(x_n \sin(\theta_n) + y_n \cos(\theta_n)) \\ \theta_n = 0.4 - \frac{6}{1+x_n^2+y_n^2} \end{cases} \tag{5.2}$$

On a tracé la trajectoire de 100 points avec différentes conditions initiales dans la Fig. 5.6c et avec comme paramètre $u = 0.7$.

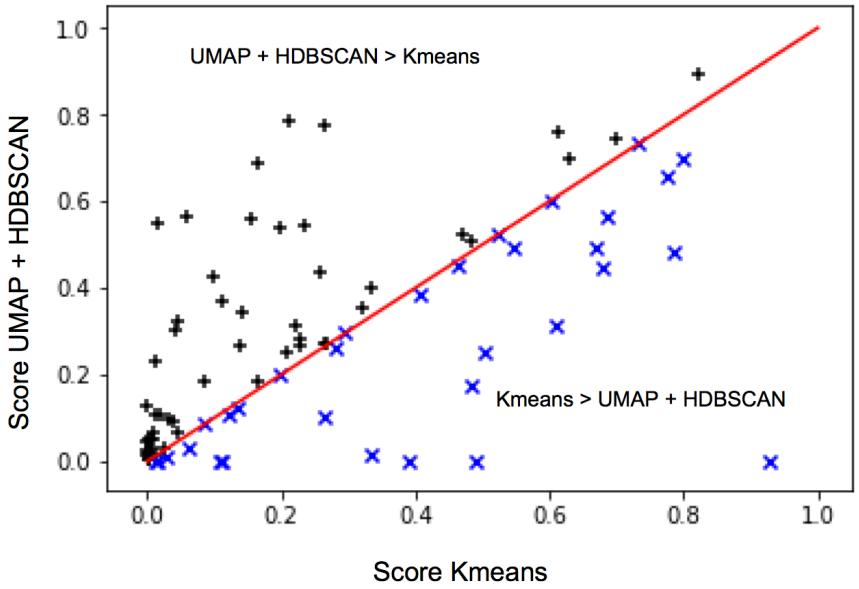


FIGURE 5.5 – Les bases de données sont représentées par le v-measure score de UMAP + HDBSCAN en fonction du v-measure score de K-means. Le plan est séparé en deux zones : une où la première méthode a un meilleur score et l'autre où c'est la deuxième méthode qui a un meilleur score.

— Le système d'Hénon : Ce système dynamique est défini par le système d'équations suivant :

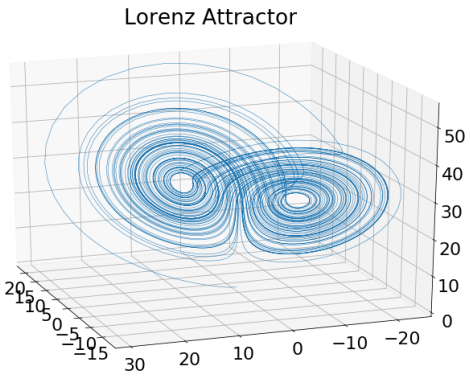
$$\begin{cases} x_{n+1} = y_n + 1 - a.x_n^2 \\ y_{n+1} = \quad b.x_n \end{cases} \tag{5.3}$$

C'est un système de $\mathbb{R}^2$ et nous avons tracé la trajectoire de 100 points (avec différentes conditions initiales) dans la Fig. 5.6d avec comme paramètres $a = 1.4$ et $b = 0.3$. On peut remarquer que ce système d'équations est cette fois discret.

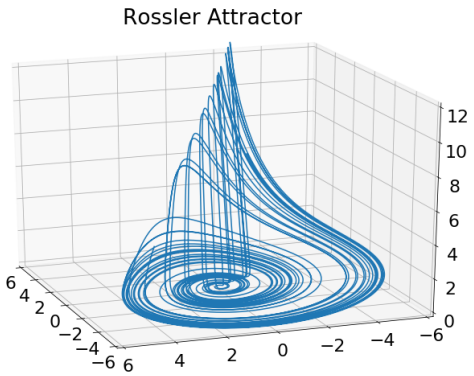
Il nous est possible de construire une base de données de séries temporelles uni-dimensionnelles avec des groupes réels connus à partir de ces systèmes dynamiques.

Résultats de clustering sur une base de données obtenue à partir de plusieurs systèmes dynamiques

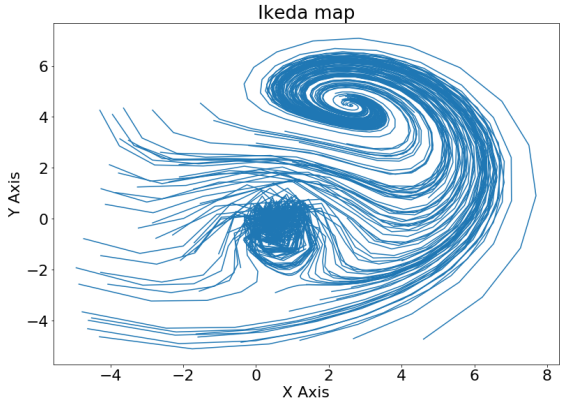
On propose de créer une première base de données avec quatre groupes. Pour chaque système dynamique, nous déterminons les trajectoires de 100 points soumis aux équations différentielles précédentes. Chaque trajectoire correspond à des conditions initiales particulières, prises aléatoirement



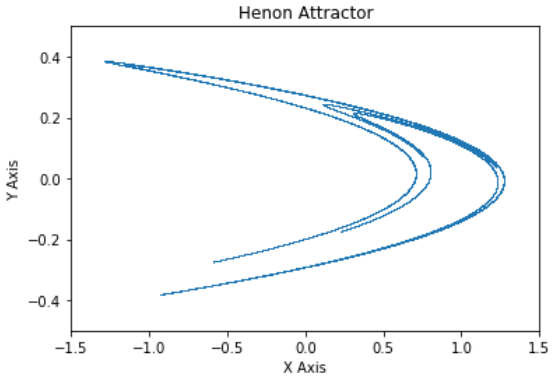
(a) Attracteur de Lorenz avec $\rho = 28$, $\sigma = 10$, $\beta = 8/3$.



(b) Attracteur de Rössler avec $a = 0.1$, $b = 0.1$,$c = 14$.



(c) Attracteur d'Ikeda pour $a = 1.4$,$b = 0.3$, avec les trajectoires de 100 points dans $\mathbb{R}^2$.



(d) Attracteur d'Hénon pour $u = 0.7$, avec les trajectoires de 100 points dans $\mathbb{R}^2$.

FIGURE 5.6 – Visualisation de chaque attracteur issu des quatre systèmes dynamique.

entre 0 et 1. Pour calculer l'évolution des points, la résolution de ces équations a été discrétisée avec un pas de 0.01 et 10000 itérations ont été réalisées. On obtient donc des coordonnées de $\mathbb{R}^2$ ou $\mathbb{R}^3$ (selon le système dynamique) x, y et z. Pour simuler une mesure de ces systèmes dynamiques, nous ne gardons que la coordonnée x. On obtient donc une base de données de 400 séries temporelles uni-dimensionnelles de 10000 occurrences, appartenant à 4 groupes de 100 séries temporelles issues de chaque système dynamique.

Ces séries temporelles sont donc ensuite transformées en matrices de trajectoire en utilisant le Delay Coordinate Embedding avec comme paramètres m (dimension d'embedding) et τ (retard). On garde $m = 3$ pour toutes les séries temporelles puis on prend le τ qui permettait de récupérer au mieux visuellement la structure des trajectoires originales. Cela nous donne :

— $\tau = 10$ pour le système de Lorenz,

— $\tau = 1$ pour le système d'Ikeda et celui d'Hénon,

— $\tau = 100$ pour le système de Rössler.

Les valeurs différentes de τ amènent à des matrices de différentes dimensions qui peuvent être compliquées à comparer. On rappelle qu'une matrice de trajectoire a m colonnes et $l-(m-1)\tau$ lignes avec l la longueur de la série temporelle (ici, 10000). On profite d'avoir des longues séries temporelles issues d'attracteurs (trajectoires qui se répètent) pour réduire les matrices de trajectoire directement sans perte d'information. Nous avons donc décidé de garder seulement $l-1-(m-1)100$ vecteurs de $\mathbb{R}^m$ ($l-1-(m-1)100$ lignes), car c'est le système de Rössler qui donnait les plus petites matrices de trajectoire à cause d'un τ de 100. Ainsi, toutes les matrices appartiennent au même espace $\mathbb{R}^{n\times p}$. Elles sont ensuite analysées comme éléments de la variété de Grassmann (décomposition QR), de la Sphère, de $SPD(n)$ (en ajoutant $\epsilon * I_n$), ou encore directement $\mathbb{R}^{n\times p}$ comme on a pu voir dans le chapitre 3.

On compare alors avec le v-measure score les différentes variétés en utilisant ensuite le même algorithme de clustering (UMAP associé à HDBSCAN). Les résultats sont présentés dans la Table 5.3.

Ces résultats mettent en évidence la pertinence d'utiliser le Delay Coordinate Embedding dans ce cas (séries temporelles issues de différents systèmes dynamiques). De plus, l'utilisation de la

TABLE 5.3 – V-measure score des résultats de clustering en fonction du choix de variété sur une base de données dont les séries temporelles sont extraites de 4 systèmes dynamiques.

Choix de variété :	Grassmann $Gr_{n,p}$	Sphère $S(n)$	$SPD(n)$	$\mathbb{R}^{n\times p}$
V-measure score :	0.8456	0.837	0.8176	0.471

variété de Grassmann pour analyser ces matrices de trajectoire est le choix le plus pertinent avec un très bon résultat de 0.8456. À titre de comparaison, une comparaison directe en utilisant la distance Euclidienne entre chaque série temporelle donne le plus mauvais résultat avec un v-measure score de 0.460. Le Delay Coordinate Embedding permet donc de très bien récupérer la géométrie intrinsèque des séries temporelles, mais il est aussi nécessaire de bien choisir la variété pour représenter les matrices de trajectoire.

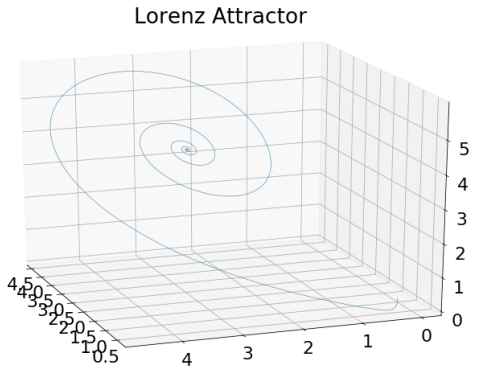
Résultat de clustering à partir d'un unique modèle

Dans la partie précédente, nous avons montré l'intérêt de cette méthode pour détecter des groupes dans des séries temporelles issues de systèmes dynamiques différents. Ici, la même méthode est appliquée mais pour comparer des séries temporelles issues d'un même système.

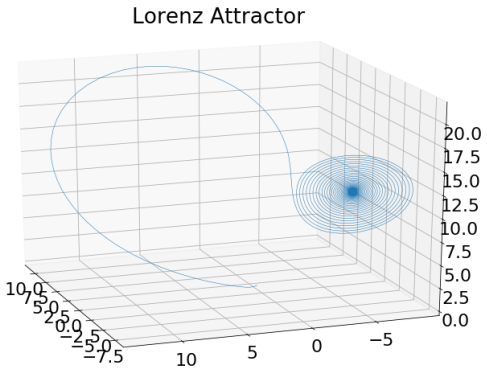
Pour cela, on utilise le système de Lorenz avec une variation de paramètres pour créer différentes trajectoires. Dans la Fig. 5.7, on a tracé quatre trajectoires obtenues pour quatre valeurs différentes de ρ. De la même manière que précédemment, on obtient une base de données de 400 séries temporelles uni-dimensionnelles (coordonnées x des trajectoires) séparées en quatre groupes, un groupe pour chaque valeur de ρ.

Comme précédemment, les paramètres pour appliquer le Delay Coordinate Embedding sont $\tau = 10$ et $m = 3$ pour le système de Lorenz. Les résultats sont présentés dans la Table 5.4.

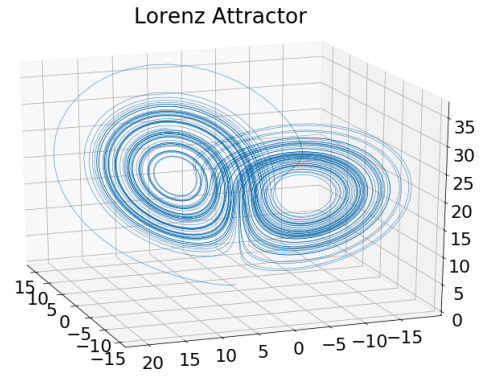
Encore une fois, le Delay Coordinate Embedding permet de très bien clusteriser ce type de base



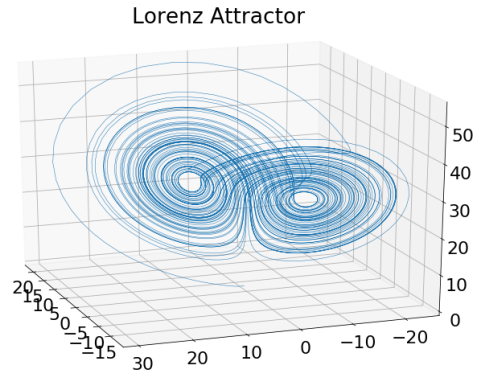
(a) Système de Lorenz avec $\rho = 5$.



(b) Système de Lorenz avec $\rho = 13$.



(c) Système de Lorenz avec $\rho = 20$.



(d) Système de Lorenz avec $\rho = 28$.

FIGURE 5.7 – Trajectoires obtenues avec le système de Lorenz pour différentes valeurs de ρ.

TABLE 5.4 – Résultat de clustering en fonction du choix de géométrie sur une base de données dont les séries temporelles sont extraites d'un unique système dynamique (système de Lorenz).

Choix de variété :	Grassmann $Gr_{n,p}$	Sphère $S(n)$	$SPD(n)$	$\mathbb{R}^{n \times p}$
V-measure score :	1	1	1	0.721

de données. À titre de comparaison, une application directe de la distance Euclidienne sur les séries temporelles donne un score faible de 0.648.

On a donc montré la pertinence d'utiliser le Delay Coordinate Embedding dans le cas de systèmes dynamiques, que ce soit pour clusteriser au sein d'un même système dynamique ou pour clusteriser différents systèmes dynamiques. Cela parait logique au vu du théorème de Takens. On peut aussi noter qu'une analyse par la variété de Grassmann semble être la plus pertinente.

5.3.2 Résultats sur les bases de données de UCR time Series Archive

On s'interroge maintenant sur la pertinence d'utiliser cette méthode dans des cas réels. Cela implique des séries temporelles potentiellement plus courtes, avec un certain niveau de bruit, un paramètre τ moins évident à trouver, etc.

Comment nous avons réalisé notre analyse comparative

Sur 79 bases de données issues de UCR Time Series Archive, nous avons appliqué quatre géométries et quatre algorithmes de clustering (Fig. 5.8). La géométrie des séries temporelles est exploitée par plusieurs variétés : la variété de Grassmann, la Sphère et $\mathbb{R}^{n\times p}$, avec la distance Euclidienne comme référence. Pour les quatre algorithmes de clustering, nous avons comparé HDBSCAN avec et sans UMAP, K-means et Hiérarchique. Cela nous donne 16 résultats de clustering par base de données qui sont ensuite comparés au résultat attendu via le v-measure score. Il nous est apparu pertinent de garder une comparaison entre les algorithmes de clustering. En effet, même si nous avons mis en évidence une hiérarchie entre ces algorithmes dans le cas de bases de données analysées avec une distance Euclidienne, cela n'est pas forcément le cas sur d'autres géométries comme la variété de Grassmann ou la Sphère.

On rappelle qu'avec le Delay Coordinate Embedding on obtient des matrices de trajectoire dépendantes de deux paramètres m et τ. Suite à des premiers tests concluants, il nous est apparu que le paramètre m était nettement moins déterminant sur le résultat de clustering que le paramètre τ. Nous avons donc décidé de garder m fixé à 5 (valeur faible pour réduire le temps de calcul). Pour le paramètre τ, nous avons fait varier sa valeur entre 1 et 20. Nous avons ensuite gardé le meilleur

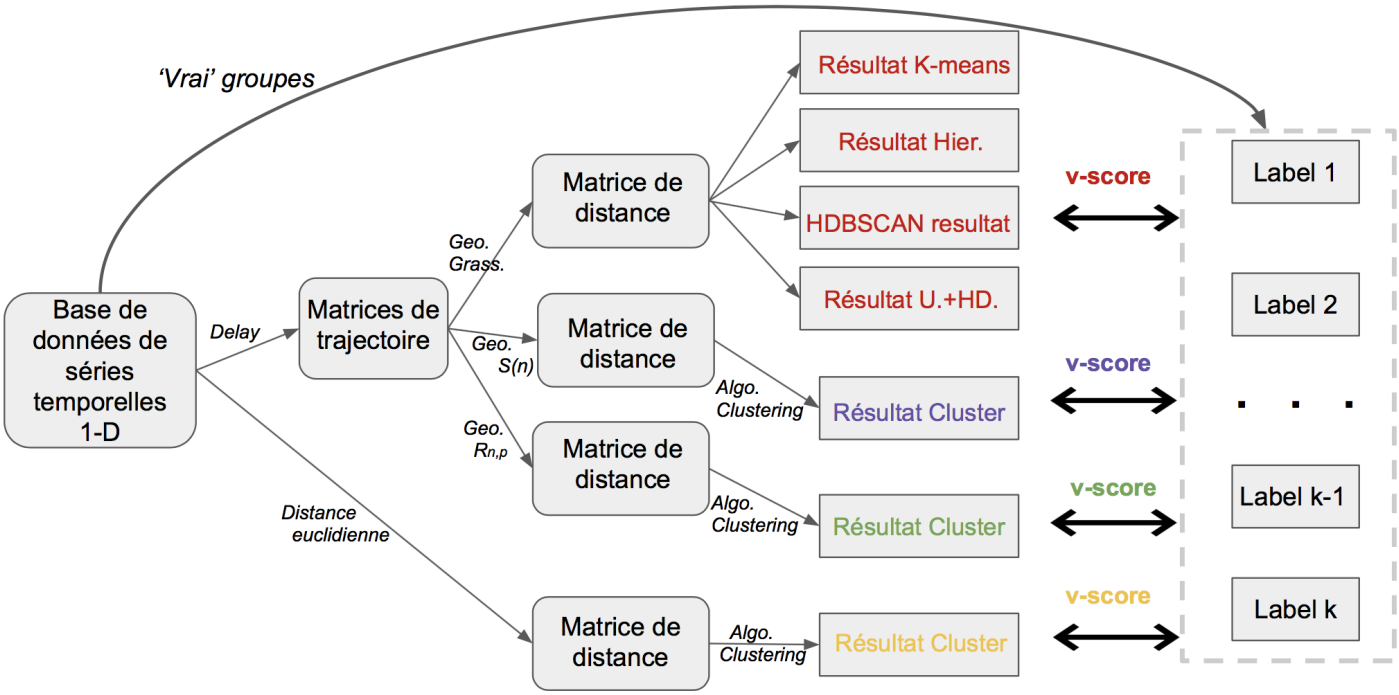


FIGURE 5.8 – Résumé de notre analyse comparative détaillée. Pour une base de données, 16 résultats de clustering ont été déterminés selon le choix de géométrie et du choix d'algorithme de clustering. En utilisant le v-measure score, on compare alors les résultats obtenus et les résultats attendus.

résultat de clustering (selon le v-measure score) obtenu pour une des valeurs de τ.

Pour les algorithmes de clustering nécessitant de déterminer à l'avance le nombre de groupes, nous avons calculé le silhouette score pour un nombre de groupes variant de 2 au nombre de groupes attendus + 10.

Analyse des résultats obtenus

Comme précédemment sur la distance Euclidienne, les principaux indicateurs sont résumés dans la Table 5.5 avec :

— La moyenne du v-measure score pour chaque méthode de clustering sur 79 bases de données,

— L'écart type du v-measure score,

— Le nombre de bases de données pour lesquelles une méthode est la meilleure,

— Le pourcentage de bases de données pour lesquelles une méthode est la meilleure.

On présente aussi entre parenthèses les mêmes éléments mais avec le score ARI. Les résultats complets sont disponibles dans l'annexe B.

TABLE 5.5 – Principaux indicateurs des v-measure scores obtenus. Pour 4 géométries et 4 algorithmes de clustering (donc 16 résultats), cette table donne la moyenne et l'écart-type des v-measure scores. On ajoute aussi le nombre et le pourcentage de bases de données pour lesquelles une méthode est meilleure que toutes les autres.

Géométrie	$\mathbb{R}^{n\times p}$				$S(n)$			
Algo. clustering	**K-means**	**Hiér.**	**HDB.**	**UMAP+HDB.**	**K-means**	**Hiér.**	**HDB.**	**UMAP+HDB.**
Moyenne	.240 (.220)	.224 (.152)	.217 (.151)	.300 (.226)	.303 (.228)	.254 (.199)	.227 (.178)	.322 (.252)
Écart-type	.241 (.252)	.253 (.208)	.232 (.202)	.272 (.234)	.260 (.235)	.290 (.270)	.250 (.232)	.270 (.230)
Nb Meilleur	3 (4)	2 (3)	3 (4)	9 (7)	7 (8)	10 (5)	6 (6)	12 (6)
Pourcentage Meilleur	3.8% (5.1)	2.5% (3.8)	3.8% (5.1)	11.4% (8.9)	8.9% (10.1)	12.7% (6.3)	7.6% (7.6)	15.2% (7.6)
Géométrie	$Gr_{n,p}$				**Euclidien, $\mathbb{R}^l$**			
Algo. clustering	**K-means**	**Hiér.**	**HDB.**	**UMAP+HDB.**	**K-means**	**Hiér.**	**HDB.**	**UMAP+HDB.**
Moyenne	.350 (.239)	.220 (.223)	.190 (.164)	**.376 (.270)**	.281 (.210)	.242 (.138)	.215 (.131)	.293 (.183)
Écart-type	.254 (.261)	.263 (.265)	0.219 (.224)	0.271 (.242)	.256 (.228)	.260 (.219)	.233 (.201)	.259 (.217)
Nb Meilleur	14 (9)	5 (5)	4 (4)	**21 (14)**	8 (8)	6 (0)	3 (0)	4 (1)
Pourcentage Meilleur	16.4% (10.6)	6.3% (6.3)	5.1% (5.1)	**26.6% (17.8)**	17.7% (10.1)	7.6% (0)	3.8% (0)	5.1% (1.3)

Nous pouvons remarquer qu'on obtient des résultats semblables avec le score ARI et le v-measure score. Pour le reste de l'analyse, nous nous sommes focalisés sur le v-measure score. À travers ses résultats, la meilleure méthode est clairement l'application de UMAP + HDBSCAN sur la variété de Grassmann. En effet, cette méthode présente la meilleure moyenne de v-measure score (0.376) et donne le meilleur résultat pour 21 bases de données parmi 79. C'est assez nettement mieux que la deuxième meilleure méthode (K-means sur la Grassmann) qui présente une moyenne de 0.350 et qui obtient le meilleur score pour 14 bases de données. UMAP + HDBSCAN sur la Sphère se démarque aussi avec une moyenne de 0.322 et obtient le meilleur score pour 12 bases de données. Les autres méthodes donnent de moins bons résultats avec une moyenne comprise entre 0.2 et 0.3. Pour mettre en lumière l'efficacité de cette méthode, nous proposons de la comparer plus en détail avec la méthode dite de "référence" (K-means directement appliquée sur les séries

temporelles). Comme précédemment, en Fig. 5.9, les bases de données sont placées dans le plan $[0,1] \times [0,1]$ en fonction de leur v-measure score de UMAP + HDBSCAN appliqué aux matrices de la variété de Stiefel (avec la distance de la variété de Grassmann) et leur v-measure score de K-means directement sur les séries temporelles.

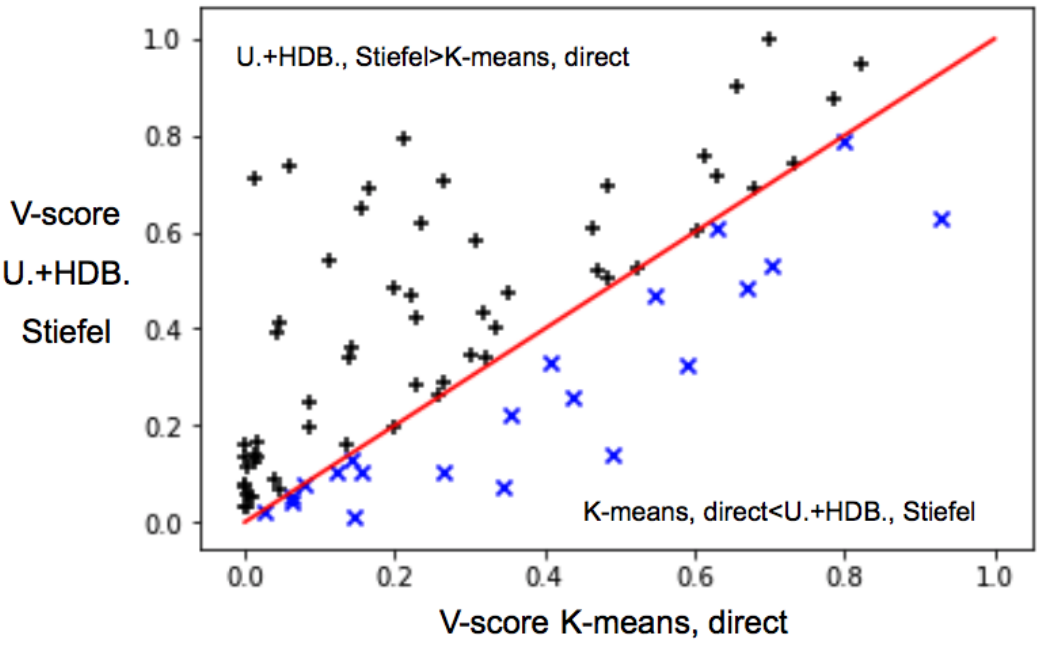


FIGURE 5.9 – Les bases de données sont représentées par le v-measure score de UMAP + HDBSCAN appliqué aux matrices de la variété de Stiefel (avec la distance de la variété de Grassmann) en fonction du v-measure score de K-means directement appliqué aux séries temporelles. Le plan est séparé en deux zones : une où la première méthode a un meilleur score et l'autre où c'est la deuxième méthode qui a un meilleur score.

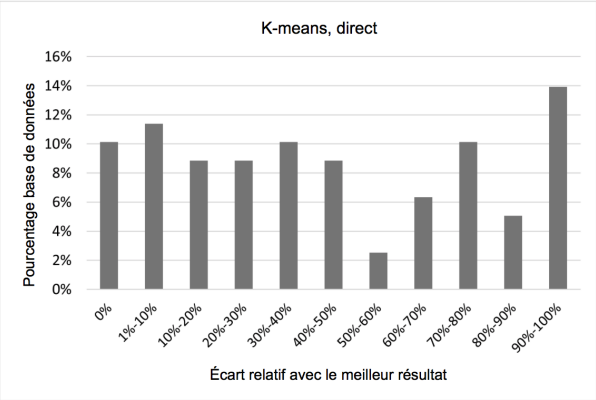
Une large partie des bases de données sont au-dessus de la ligne rouge (où UMAP + HDBSCAN appliqué aux matrices de la variété de Stiefel -avec la distance de la variété de Grassmann- est meilleur que K-means appliqué directement). De plus, les éléments qui sont dans la partie inférieure, sont très proches de la ligne rouge. Cette figure met donc en évidence que, quand K-means sur les séries temporelles est meilleur (seulement 21 bases de données sur 79), il l'est de peu. Sur ces 21 bases de données, on a en moyenne un écart de 0.117 avec l'autre méthode et même 0.08 en terme de médiane.

On propose en Fig. 5.10 une autre visualisation de l'efficacité de UMAP + HDBSCAN sur la variété de Grassmann comparée à celle de la méthode de référence. Pour chaque base de donnée, l'écart

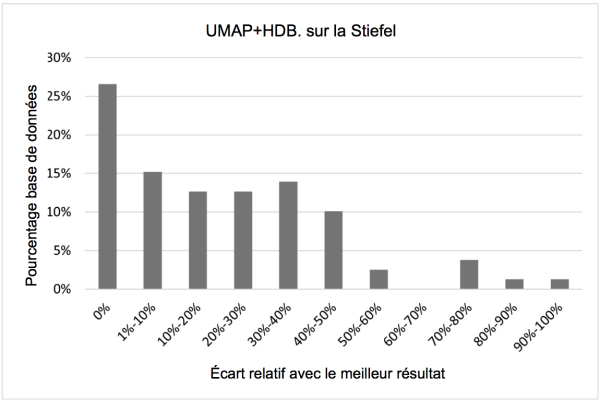
relatif

$$e = \frac{v_{max} - v_{method}}{v_{max}}$$

entre le résultat d'une méthode donnée (Fig. 5.10a : K-means directement sur les séries temporelles, Fig. 5.10b : UMAP + HDBSCAN sur les matrices de la variété de Stiefel) et le meilleur résultat parmi les 16 méthodes est calculé. Ensuite, pour chaque intervalle de 10% de l'écart relatif, le pourcentage de bases de données concernées est tracé. Par exemple, pour 10% des bases de données, K-means directement sur les séries temporelles donne des résultats très éloignés du meilleur résultat possible avec un écart relatif entre 70% et 80%. On remarque que UMAP + HDBSCAN sur la variété de Stiefel est nettement plus souvent la meilleure méthode (26% comparé à 10%, première colonne). De plus, quand cette méthode n'est pas la meilleure, le v-measure score n'en est pas très éloigné non plus. En effet, le pourcentage de bases de données concernées diminue fortement quand l'écart relatif augmente. Au contraire, la méthode de référence a un pourcentage de bases de données qui reste très élevé quand l'écart relatif est élevé. Ainsi, sur 14% des bases de données, cette méthode est complètement fausse avec un écart relatif entre 0.9 et 1.



(a) Comparaison entre K-means directement appliqué aux séries temporelles et la meilleure méthode de chaque base de données.



(b) Comparaison entre UMAP + HDBSCAN sur les matrices de la variété de Stiefel -avec la distance de la variété de Grassmann- et la meilleure méthode de chaque base de données.

FIGURE 5.10 – L'écart relatif entre le v-measure score d'un résultat de clustering d'une méthode donnée et le meilleur résultat est calculé pour chaque base de données. Ensuite, le pourcentage de bases de données concernées par intervalle d'écart relatif est tracé pour deux méthodes : K-means, directement sur les séries temporelles et UMAP + HDBSCAN sur les matrices de la variété de Stiefel.

Si on s'intéresse plus particulièrement aux choix de variété, on remarque que :

— La variété de Grassmann est la meilleure représentation pour 55.7% des bases de données (pour 44 bases de données parmi 79, un des quatre algorithmes de clustering sur la variété de Grassmann donne les meilleurs résultats),

— La Sphère est la meilleure géométrie pour 44.3% des bases de données,

— L'espace Euclidien (directement sur les séries temporelles) $\mathbb{R}^l$ est la deuxième moins bonne géométrie avec seulement 26.6% de bases de données pour lesquelles les résultats sont supérieurs aux autres géométries,

— L'espace Euclidien (après Delay Coordinate Embedding) $\mathbb{R}^{n\times p}$ donne les moins bons résultats avec le meilleur score pour seulement 21.5% des bases de données.

La distance Euclidienne directement sur les séries temporelles est clairement une métrique moins efficace que celle induite par la variété de Grassmann et de la Sphère. Encore une fois, ces résultats mettent en évidence la pertinence d'utiliser le Delay Coordinate Embedding pour obtenir les caractéristiques géométriques de la série temporelle via la matrice de trajectoire. De plus, il est utile de travailler sur des variétés bien choisies (Grassmann, Sphère,...) plutôt que d'utiliser directement la norme de Frobenius sur $\mathbb{R}^{n\times p}$. Enfin, la Sphère est une alternative intéressante à la variété de Grassmann, car cette géométrie donne des résultats légèrement moins bons mais offre plus de flexibilité en permettant la comparaison entre éléments issus de variétés de Stiefel de différentes dimensions.

Après avoir analysé l'impact des géométries sur la qualité des résultats de clustering, on s'intéresse maintenant aux algorithmes de clustering. Dans cette optique, on peut remarquer à partir de la Table 5.5 en prenant en compte les quatre résultats obtenus par algorithme et par base de données :

— UMAP + HDBSCAN est le meilleur algorithme pour 58.2% des bases de données (46/79). Il présente la meilleure moyenne de v-measure score avec 0.323,

— K-means donne le meilleur résultat pour 44.3% des bases de données (35/79) et a un score moyen de 0.293,

— Hiérarchique donne des meilleurs résultats dans 29.1% des cas avec un score moyen de 0.235,

— HDBSCAN seule obtient le meilleur résultat dans 20.2% des bases de données avec un score moyen faible de 0.212.

L'algorithme HDBSCAN seul présente donc les plus mauvais résultats. On peut d'ailleurs noter que 16% des bases de données (13/79) présente un score de 0 pour les quatre résultats (un sur chaque géométrie) liés à cet algorithme. Cependant, en combinant UMAP avec HDBSCAN, les résultats sont très largement améliorés, devenant l'algorithme de clustering le plus efficace. De plus, un certain nombre de paramètres entre en jeu entre UMAP et HDBSCAN. Nous avons décidé de laisser par défaut tous ces paramètres et ne pas traiter cette problématique. En effet, il serait compliqué d'optimiser ces paramètres pour chacune des bases de données. Cependant, pour un utilisateur sur une base unique, il est possible de s'y intéresser. On peut en déduire que UMAP + HDBSCAN donne déjà les meilleurs résultats et peut être encore amélioré en s'intéressant à ces paramètres.

On a aussi lancé des résultats de clustering utilisant la géométrie $SPD(n)$ pour analyser la matrice de trajectoire. Cependant, le calcul de distance entre deux éléments sur cette géométrie - utilisant du *log* matricielle, de l'inverse de matrice,... - est très chronophage. Or, les algorithmes de clustering prennent en entrée une matrice de distance composée de l'ensemble des distances deux à deux entre chaque élément de la base de données. Il devient donc très rapidement impossible à calculer quand le nombre d'éléments dans la base de données augmente. Nous avons pu lancer les algorithmes de clustering avec cette géométrie sur seulement 15 bases de données (les 15 bases de données avec le moins d'éléments). Sur ces bases de données, on propose une comparaison dans la Table 5.6 avec les résultats sur la variété de Grassmann sur ces mêmes bases de données.

TABLE 5.6 – Comparaison des résultats entre $SPD(n)$ et $V_{n,p}$ à travers la moyenne du v-measure score sur les 15 bases de données.

	K-means	**Hiérarchique**	**HDB.**	**U.+HDB.**
$V_{n,p}$	0.251	0.231	0.207	0.382
$SPD(n)$	0.261	0.221	0.198	0.354

On remarque que les résultats sont assez similaires avec un léger avantage pour la variété de Grassmann, à part pour K-means où $SPD(n)$ présente des meilleurs résultats. Il est donc a priori plus pertinent de passer par la variété de Grassmann au vu du gain de temps et du gain d'efficacité

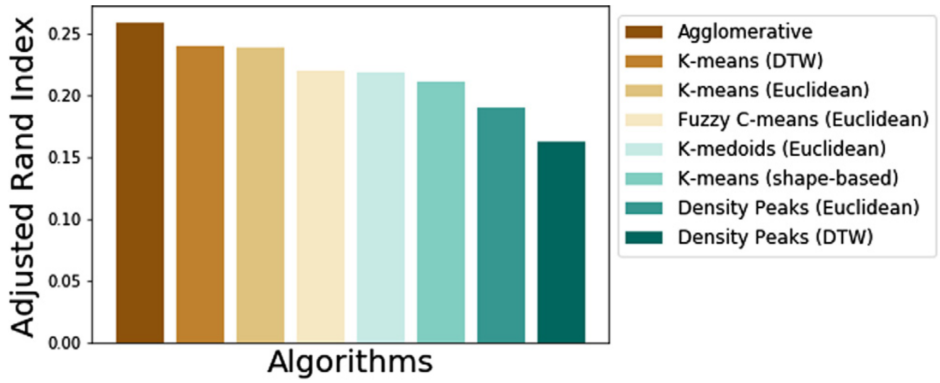


FIGURE 5.11 – Résultat obtenu dans l'article [54]. Les algorithmes Hiérarchique Agglomerative et K-means avec une distance Euclidienne ont bien performé (1er et 3ème de leur classement).

global.

D'autres études ont été réalisées sur ces bases de données. En particulier, les auteurs de [54] ont réalisé une analyse comparative similaire à la nôtre. Ils ont testé deux distances (Euclidienne et DTW) et 5 algorithmes de clustering. En Fig. 5.11, extraite de cet article, on retrouve le score moyen qu'ils ont obtenu pour chaque méthode de clustering. Il en ressort que K-means avec la distance Euclidienne et Hiérachique Agglomerative (aussi avec la distance Euclidienne) donnent de bons résultats. Or, de notre côté, on a montré que UMAP + HDBSCAN sur la variété de Stiefel présente clairement de meilleurs résultats que ces méthodes-là. On peut en déduire que cette méthode surpasse aussi l'ensemble des méthodes proposées dans [54].

Ces résultats ont fait l'objet d'un article [79] "Improved Time Series Clustering Based on New Geometric Frameworks" publié dans "Pattern Recognition, Elsevier".

En conclusion, nous obtenons d'excellents résultats en utilisant le Delay Coordinate Embedding. Ensuite, l'association d'un placement sur la variété de Stiefel vu comme des éléments de la variété de Grassmann d'une part et de la méthode de clustering UMAP + HDBSCAN d'autre part est très efficace. On se focalise donc sur la variété de Grassmann associée à cette méthode de clustering par la suite. Cependant, deux paramètres rentrent en jeu dans le Delay Coordinate Embedding. En particulier, le paramètre τ a une influence primordiale sur le résultat. Pour le moment, nous

l'avons fait varier entre 1 et 20 pour trouver la bonne valeur. Il est donc nécessaire maintenant de déterminer le "bon" τ à l'avance pour avoir une méthode complètement non supervisée.

5.4 Détermination des paramètres du Delay Coordinate Embedding

Pour réaliser notre clustering, nous nous sommes appuyé sur le Delay Coordinate Embedding. En général, cette méthode est utilisée dans le domaine de la mécanique des systèmes dynamiques où le but est de reconstruire la dynamique d'un système à partir d'une série temporelle. Des outils ont déjà été mis en place dans la littérature de ce domaine pour déterminer les paramètres m et τ. Nous nous sommes donc appuyé sur ces outils en première instance. Cependant, ici, nous voulons réaliser du clustering sur un ensemble de séries temporelles. Le choix de m et τ pour chaque série temporelle doit donc être compatible sur l'ensemble des matrices de trajectoire. C'est une première contrainte qui nous éloigne du domaine de la mécanique. De plus, notre but final est avant tout de bien séparer les séries temporelles (clustering) plutôt que de reconstruire au mieux le système dynamique. Nous avons donc aussi chercher dans le domaine du clustering pour obtenir des outils plus adaptés à notre cas d'application. Enfin, nous nous sommes penchés aussi sur la formes des trajectoires obtenues en fonction de τ.

5.4.1 Détermination du paramètre m

Il ne nous est pas apparu intéressant de creuser fortement sur la détermination de ce paramètre. En effet, l'influence de τ était nettement plus importante pour la détermination du résultat de clustering. Cependant, si on veut aller plus loin qu'une détermination arbitraire de ce paramètre à $m = 5$, des méthodes existent.

Indicateur pour m pour les systèmes dynamiques : Dans l'utilisation du Delay Coordinate Embedding pour les systèmes dynamiques, l'algorithme du "False Nearest Neighbor" [80] est privilégié dans la littérature [81]. Le but de cet algorithme est de déterminer le paramètre m qui garde au mieux la topologie initiale. Pour une série temporelle issue d'un système dynamique caché de dimension n, la reconstruction de ce système via le Delay Coordinate Embedding (de paramètre m et τ), nous donne des éléments de $\mathbb{R}^m$ ($m < n$). Il peut donc y avoir une perte d'in-

formations due à la projection dans $\mathbb{R}^m$. Pour quantifier cette perte d'information sans connaître le système caché derrière, cet algorithme utilise la notion de faux voisins. La Fig. 5.12, extraite de `https://personal.egr.uri.edu/chelidz/documents/mce567_Chapter_7.pdf`, illustre cette notion.

Un exemple simple de faux voisins : un avion volant au-dessus de nous (éloigné dans $\mathbb{R}^3$) apparaîtrait très proche dans une projection au sol ($\mathbb{R}^2$). L'avion est donc un faux voisin et il y a une perte d'informations lors de cette projection. Au contraire, un couple de vrais voisins reste proche quelque soit la projection. L'algorithme "False Nearest Neighbors" propose donc de faire varier les valeurs de m et garde la valeur minimale m_{min} telle que tous les voisins dans cette projection le restent avec $m_{min} + 1$.

Avec cette méthode, on obtient donc une valeur m intrinsèque à chaque série temporelle. L'ensemble des matrices de trajectoire peuvent alors avoir des nombres de colonnes différentes. Nous obtenons donc des valeurs de m non compatibles pour la variété de Grassmann -on ne sait comparer que des éléments venant d'une même variété de Grassmann-. Nous sommes donc rester sur notre valeur de $m = 5$.

5.4.2 Détermination du paramètre τ

Indicateur pour τ pour les systèmes dynamiques : De manière similaire à m, on peut s'intéresser à ce que nous dit la littérature sur les systèmes dynamiques pour déterminer τ. Dans ce domaine, la méthode la plus couramment utilisée [82, 83] est l'information mutuelle [84]. L'information mutuelle mesure la dépendance statistique entre deux variables. L'avantage de cette quantité plutôt que l'utilisation de l'auto-corrélation est que l'on peut mesurer aussi les corrélations non linéaires. On peut donc déterminer l'information mutuelle notée $I(\tau)_X$ entre la série temporelle $X(t)$ et elle-même décalée d'un temps τ, $X(t + \tau)$. En prenant le τ_{min} qui minimise de cette fonction, on s'assure que le couple $(X(t), X(t + \tau_{min})$ est le couple qui apporte le plus d'information. En pratique, [85] suggère de prendre le premier minimum de $I(\tau)$. Cependant, cette méthode est adaptée pour les systèmes dynamiques et il est compliqué de l'unifier sur l'ensemble des séries temporelles. En effet, si on définit un paramètre τ pour chaque série temporelle X via $I(\tau)_X$, on obtient

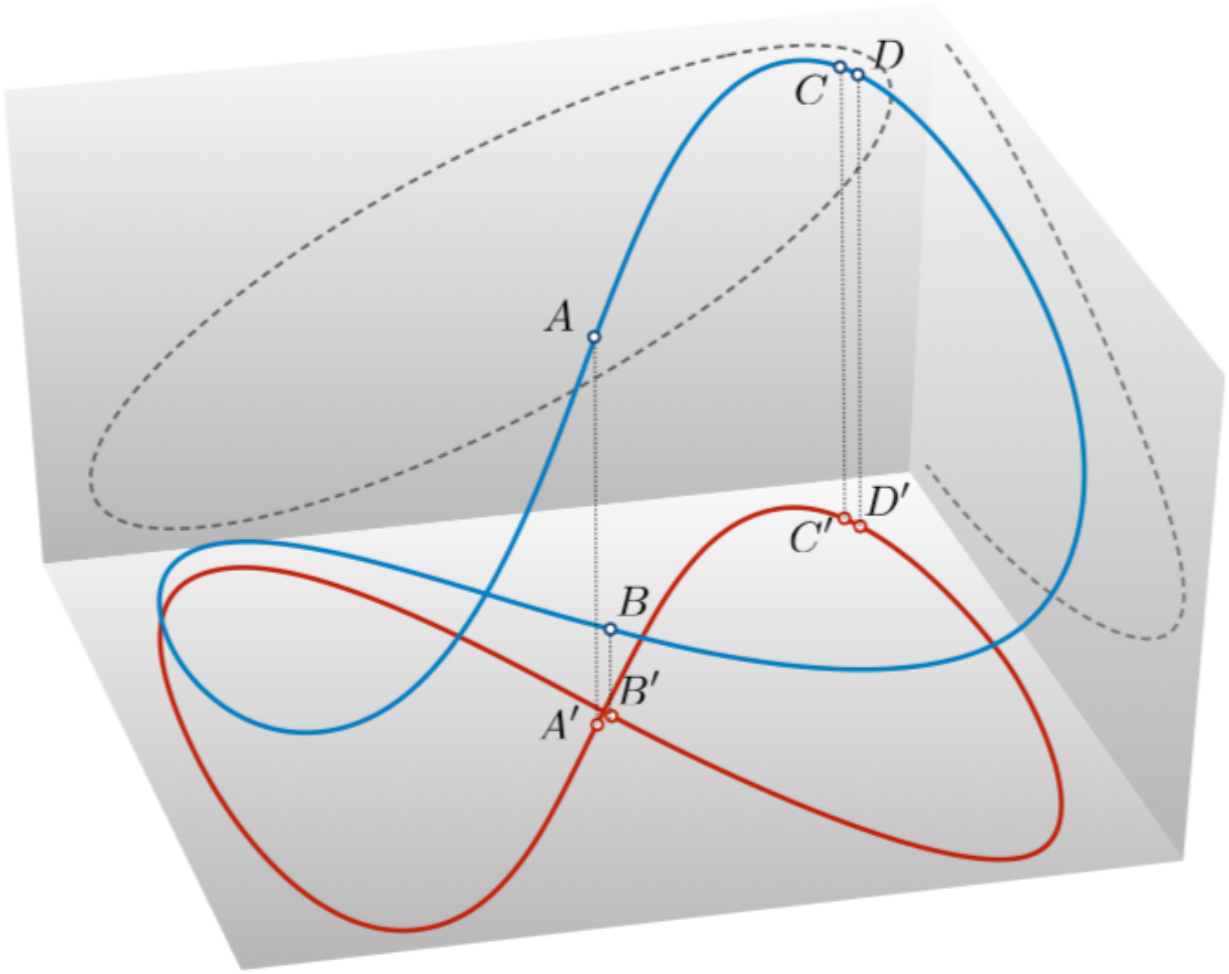
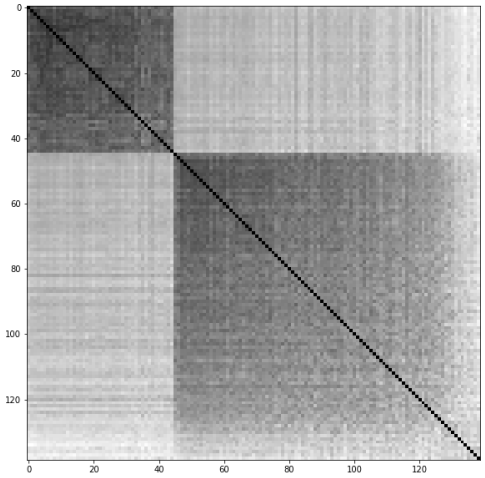


FIGURE 5.12 – Exemple de la notion de faux voisins sur une courbe dans $\mathbb{R}^3$ en bleue et sa projection dans $\mathbb{R}^2$ en rouge. On remarque que B' (projeté de B) est voisin de A' (projeté de A). Or, dans $\mathbb{R}^3$, A et B étaient des points éloignés. On appelle donc A' et B' des faux voisins. Au contraire, les points C et D, proches dans $\mathbb{R}^3$ ont leurs projections respectives C' et D' qui restent proches dans $\mathbb{R}^2$.

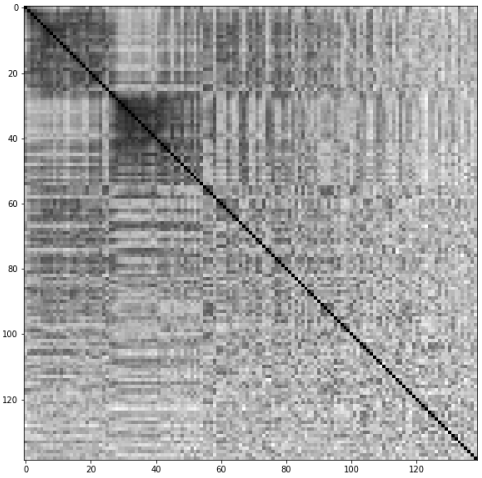
des matrices appartenant à des variétés de Stiefel différentes, car un changement de τ entraîne un changement sur le nombre de lignes dans la matrice de trajectoire. Dans ce cas-là, on ne peut pas se replacer sur la Sphère. En effet, on peut seulement comparer des variétés de Stiefel avec un nombre différent de colonnes et non avec un nombre différent de lignes. Ces différentes valeurs de τ ne sont donc a priori pas compatibles sur l'ensemble de la base de données.

Pour régler ce problème, nous avons pensé à définir une notion d'angle entre des vecteurs de taille différente. Cela permettrait de définir une nouvelle distance en angle principal pour des variétés de Stiefel de différentes dimensions. De manière abrupte, il est possible de compléter le vecteur le plus court avec des 0 pour obtenir deux vecteurs de même taille. Cela a été testé sur quelques bases de données et les résultats étaient très mauvais. Nous avons donc creusé d'autres pistes. En remarquant que DTW réalise une extension de la distance Euclidienne sur des séries temporelles de longueurs différentes, nous avons essayé d'adapter la quantité à optimiser pour travailler avec des angles. Pour cela, au lieu de chercher à minimiser la distance Euclidienne en modifiant le matching, nous proposions de modifier le matching pour maximiser le produit scalaire (et donc minimiser l'angle). Cependant, il était compliqué de s'assurer d'avoir un produit scalaire inférieur au produit des normes de chaque vecteur. On obtenait ainsi des angles θ entre les vecteurs tel que $cos(\theta) > 1$ ce qui était absurde. Pour régler ce problème, nous pouvons utiliser une information mutuelle multivariée [86, 87] qui va nous permettre de comparer l'information entre l'ensemble des séries temporelles et elle-même décaler d'un temps τ. Nous aurons ainsi un paramètre τ pour l'ensemble de la base de données ce qui permet de placer l'ensemble des séries temporelles sur la même variété de Stiefel (et donc sur la même variété de Grassmann). Cette dernière idée (l'information mutuelle multivariée) a été essayée sur l'ensemble des bases de données.

Indicateur pour τ pour le clustering : On s'est posé aussi la question de trouver un paramètre τ adapté au clustering plutôt qu'aux systèmes dynamiques. Dans ce cas, on voudrait obtenir un τ qui sépare au mieux les groupes dans l'ensemble des données plutôt que de s'appliquer à reconstruire parfaitement les systèmes dynamiques cachés derrière les séries temporelles. Pour cela, nous nous sommes intéressés à la distribution de l'ensemble des distances entre deux éléments de l'ensemble des données. L'idée est que, dans le cas de données non clusterisable, la distribution des distances sera globalement uniforme. Au contraire, dans le cas de données séparables nettement



(a) VAT représentation de la matrice de distance obtenue pour la base de données DistalPhalanxTW avec le τ donnant les meilleurs résultats de clustering et $m = 5$.



(b) VAT représentation de la matrice de distance obtenue pour la base de données DistalPhalanxTW avec le τ donnant les moins bons résultats de clustering et $m = 5$.

FIGURE 5.13 – Visualisation de l'effet de τ sur la matrice de distance. On remarque que pour le meilleur τ deux groupes semblent clairement émergés. Pour le moins bon τ la distribution des distances semble beaucoup moins uniforme.

en deux groupes, on obtiendra une distribution des distances marquée par deux pics de densité, un de faible distance et un de grande distance. Pour confirmer cette idée, nous proposons de visualiser l'effet de τ sur la matrice des distances (par pairs d'éléments) via une visualisation VAT. La visualisation VAT donne une représentation d'une matrice de distance qui souligne la présence ou non de groupe. Les éléments de la matrice sont plus ou moins noircis selon l'amplitude de ces éléments : plus une distance est grande entre deux éléments, plus son emplacement dans la matrice sera claire et plus une distance est faible plus son emplacement sera foncé. Ensuite, une permutation sur l'ordre des éléments de la base de données est réalisée pour mettre en lumière la présence de groupe. En Fig. 5.13, les visualisations VAT obtenues pour le meilleur et le moins bon τ (en terme de qualité de clustering) de la base de données DistalPhalanxTW (UCR TimeSeries Archive) sont présentées. Pour le meilleur τ, on remarque apparaître nettement deux carrés foncés. Ce τ-là permet donc de mettre en valeur la présence de deux groupes distincts dans la base de données. Au contraire, pour le moins bon τ, aucun groupe ne semble se dégager. Suite à cette confirmation visuelle de notre idée de départ, nous proposons d'utiliser l'indicateur d'Hopkins [88].

Cet indicateur a pour but de mesurer la tendance de la base de données à être séparée en groupes cohérents ("clusterisable"). Pour une base de données de n éléments $(X_1, ..., X_n)$ dans M, on prend m_H $(m < n)$ éléments $(Y_1, ...Y_{m_H})$ uniformément distribués dans M. On note ensuite u_i la distance de Y_i à son plus proche voisin dans $(X_1, ..., X_n)$ et w_i la distance entre un élément au hasard dans $(X_1, ..., X_n)$ et son plus proche voisin. L'indicateur d'Hopkins H est alors donné par :

$$H = \frac{\sum_{i=1}^{m_H} u_i^d}{\sum_{i=1}^{m_H} u_i^d + \sum_{i=1}^{m_H} w_i^d}$$

Une base de données sera alors dite "clusterisable" si son indicateur d'Hopkins est proche de 1. En effet, cela voudra dire que les w_i sont négligeables en comparaison des u_i. Ainsi, les m_H éléments pris au hasard dans $(X_1, ..., X_n)$ ont tous trouvé un voisin nettement plus proche que les $(Y_1, ...Y_{m_H})$ distribués uniformément. Cela nous indique la présence de poches d'éléments avec une densité importante entouré d'espace moins dense. Au contraire, une base de données sera considérée comme uniformément distribuée pour un indicateur d'Hopkins de 0.5. En effet, dans ce cas, les distances u_i et w_i sont similaires. Notre base de données serait donc proche d'une distribution uniforme. Un paramètre m_H intervient dans cet indicateur. Dans [89], il est montré de manière heuristique que prendre $m_H = 0.1n$ est pertinent. Nous sommes donc restés sur cette valeur. Pour récapituler cette méthode, pour une valeur de τ donnée, on détermine la matrice des distances entre chaque élément représenté par une matrice de la variété de Stiefel et comparé comme élément de la variété de Grassmann. On calcule ensuite l'indicateur d'Hopkins en utilisant cette matrice de distance. Le τ qui maximise l'indicateur d'Hopkins est donc à priori un très bon candidat pour avoir un bon clustering. Cependant, pour calculer l'indicateur d'Hopkins, il faut pouvoir tirer m_H éléments depuis une distribution uniforme sur la variété, ici la variété de Grassmann vue par des matrices de la variété de Stiefel. Pour cela, on peut s'appuyer sur les théorème 2.2.1 et 2.2.2 proposés dans [90]. D'après le théorème 2.2.1, Il est ainsi possible de générer facilement des matrices distribuées uniformément sur $V_{n,p}$ en suivant cette méthode :

1. Créer une matrice X de $\mathbb{R}^{n\times p}$ avec tous les éléments $(n \times p)$ de cette matrice issue de la loi normale $\mathbf{N}(0, 1)$.
2. Calculer $Y = X(X^T X)^{-1/2}$

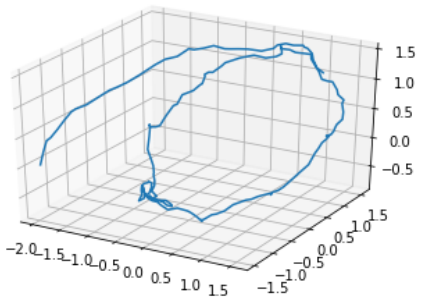
Y est alors un élément de $V_{n,p}$ issu d'une distribution uniforme sur cette variété. Le théorème 2.2.2

assure ensuite que l'élément P_Y de la variété de Grassmann (sous-espace engendré par les colonnes de Y) est aussi issu d'une distribution uniforme sur la variété de Grassmann. On peut donc utiliser l'indicateur d'Hopkins sur la variété de Grassmann.

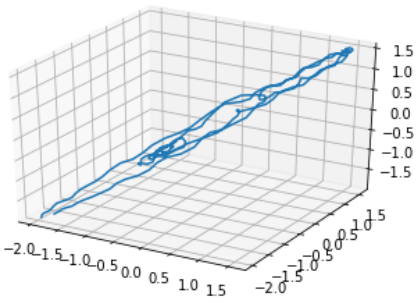
Indicateur pour τ en s'intéressant aux trajectoires : Enfin, nous avons aussi exploré l'effet de τ sur la forme de la trajectoire. Pour une série temporelle d'une base de données de UCR Time Series Archive, nous avons déterminé la matrice de trajectoire avec $m = 3$ et le τ maximisant les résultats de clustering et la matrice de trajectoire avec $m = 3$ et le τ minimisant les résultats de clustering. Ce choix de $m = 3$ nous permet d'avoir une visualisation 3D de la trajectoire. En Fig. 5.14, les trajectoires obtenues pour les bases de données ArrowHead et uWaveGestureLibraryY sont présentées. ArrowHead est un exemple type que l'on a retrouvé fréquemment parmi l'ensemble des bases de données. Les moins bons résultats de clustering sont obtenus pour un τ de 1 qui a tendance à "écraser" la trajectoire sur la droite $x = y = z$. Les trajectoires de toutes les séries temporelles sont alors très ressemblantes ce qui donne des mauvais résultats de clustering. Au contraire, le meilleur τ donne des trajectoires plus "aérées" permettant de mieux les différencier. Le problème c'est qu'il peut être compliqué de donner un critère objectif mathématique pour qualifier une trajectoire d'"aérée" et une d'"écrasée". De plus, pour une partie des bases de données, cette observation ne tient plus comme pour la base de données uWaveGestureLibraryY.

5.4.3 Utilisation de la réduction de dimension

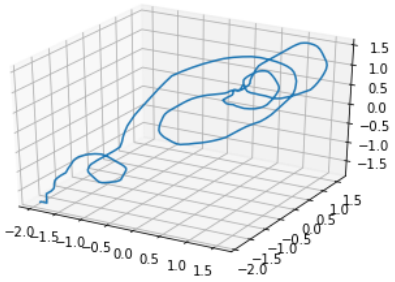
Une autre solution pour mettre en lumière la dynamique des séries temporelles est d'utiliser des techniques de réduction de dimension sur la matrice de trajectoire avec un τ de 1 plutôt que de chercher à optimiser le τ. Pour une série temporelle de longueur l, on choisit comme paramètres pour le Delay Coordinate Embedding $m = \lfloor l/2 \rfloor$ et $\tau = 1$ comme proposé par exemple dans [91, 63]. Via un algorithme de réduction de dimension, on peut alors réduire le nombre de colonne ce qui permet de réduire le bruit et la redondance. En effet, le bruit et la redondance ont tendance à avoir une faible variance et vont donc être limités si on garde les caractéristiques avec le plus de variance. On fait ainsi ressortir la dynamique de la série temporelle sur la matrice de trajectoire. Ce qui nous est apparu le plus pertinent est d'utiliser le Dynamic Mode Decomposition (DMD) [91] car c'est une réduction de dimension adaptée pour faire ressortir les principaux modes d'une série temporelle.



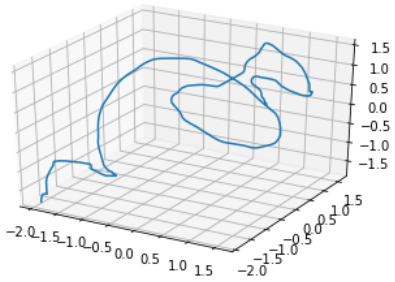
(a) Trajectoire d'une série temporelle de la base de données ArrowHead obtenue avec $m = 3$ et avec le τ donnant les meilleurs résultats.



(b) Trajectoire d'une série temporelle de la base de données ArrowHead obtenue avec $m = 3$ et avec le τ donnant les moins bons résultats.



(c) Trajectoire d'une série temporelle de la base de données uWaveGestureLibraryY obtenue avec $m = 3$ et avec le τ donnant les meilleurs résultats.



(d) Trajectoire d'une série temporelle de la base de données uWaveGestureLibraryY obtenue avec $m = 3$ et avec le τ donnant les moins bons résultats.

FIGURE 5.14 – Visualisation de l'effet de τ sur la trajectoire et sur la qualité du clustering. Pour la base de données ArrowHead, on obtient les moins bons résultats pour un τ de 1 ce qui donne des trajectoires très ramassées sur l'axe $x = y = z$. Au contraire, la trajectoire pour le meilleur τ est plus aérée. Ce comportement est retrouvé dans une partie importante des bases de données utilisées. Cependant, on remarque que ce n'est pas du tout vérifié pour la base de données uWaveGestureLibraryY.

5.4.4 Analyse des pistes explorées

Selon la méthode utilisée pour déterminer τ (puis une analyse avec la variété de Grassmann et UMAP + HDBSCAN pour le clustering), nous obtenons un v-measure score moyen de :

— 0.295 avec la réduction de dimension DMD.

— 0.313 avec l'indicateur Hopkins.

— 0.302 avec l'information mutuelle multivariée.

Pour rappel, on a un v-measure score moyen de :

— 0.281 avec la méthode classique (distance Euclidienne associée avec K-means).

— 0.376 avec une recherche exhaustive sur le τ (variation entre 1 et 20) et une analyse avec la variété de Grassmann et UMAP + HDBSCAN pour le clustering.

Parmi les pistes explorées, l'indicateur d'Hopkins est donc le plus pertinent pour avoir une méthode complètement non-supervisée. L'information mutuelle multivariée donne des résultats intéressant (tout comme DMD) mais on peut penser que cet indicateur reste limité pour des séries temporelles qui présentent des informations mutuelles très différentes. En effet, dans ce cas, en les unifiant via du multivariée, on peut passer à côté de beaucoup d'informations intrinsèques à chacune des séries temporelles. Cela nous offre une méthode plus intéressante que l'association la plus classique - Euclidien et K-means -. Cependant, on reste assez éloigné des scores optimaux qu'on obtient en variant les valeurs de τ.

5.5 Synthèse

Dans ce chapitre, nous avons comparé plusieurs méthodes de clustering. En s'appuyant sur le Delay Coordinate Embedding, on obtient des matrices de trajectoire comparables sur différentes variétés sur lesquelles on peut appliquer différents algorithmes de clustering. Le clustering étant une méthode non supervisée, il peut vite être compliqué de valider une méthode par rapport à une autre. Il nous a donc paru important de réaliser une analyse comparative détaillée et complète. Pour cela, nous avons utilisé les bases de données présentes à UCR Time Series Archive où les

résultats de clustering attendus sont connus. Nous avons ensuite comparé les différents résultats de clustering obtenus aux résultats attendus en utilisant le v-measure score.

On a tout d'abord mis en évidence que UMAP était un algorithme de réduction de dimension efficace avant d'utiliser n'importe quel algorithme de clustering (sur un espace Euclidien). En particulier, cela fonctionne particulièrement bien avec HDBSCAN.

On a ensuite comparer plusieurs géométries et plusieurs algorithmes de clustering en utilisant le Delay Coordinate Embedding. À ce stade, on réalise une recherche exhaustive pour déterminer le meilleur paramètre τ (variation entre 1 et 20). Il ressort de cette comparaison que l'utilisation du Delay Coordinate Embedding est particulièrement pertinent vu que quelque soit la variété utilisée ensuite, nous obtenons des résultats nettement meilleurs qu'avec des algorithmes de clustering directement appliqués sur les séries temporelles. En particulier, la variété de Grassmann est particulièrement efficace pour comparer les matrices de trajectoire. Au niveau des algorithmes de clustering, on trouve une hiérarchie assez similaires entre eux quelque soit la géométrie. Ainsi, UMAP couplé à HDBSCAN est globalement le meilleur algorithme de clustering. Logiquement, la méthode de clustering qui donne les meilleurs résultats est donc le Delay Coordinate Embedding enchaîné avec la variété de Grassmann et UMAP couplé à HDBSCAN.

Pour obtenir une méthode complètement non-supervisée, nous avons cherché à déterminer le meilleur τ sans utiliser les résultats de clustering attendus. Pour cela, nous avons exploré plusieurs pistes, issues de la théorie des systèmes dynamiques, de la réduction de dimension et des indicateurs de clustering. Utiliser le τ qui maximise l'indicateur d'Hopkins (qui détermine si une base de donnée est "clusterisable") est la méthode qui a donné les meilleurs résultats.

Au vu des bons résultats obtenus, nous proposons d'appliquer le Delay Coordinate Embedding, avec une analyse des matrices de trajectoire avec la variété de Grassmann puis une application d'UMAP couplé avec HDBSCAN à notre cas d'étude médicale.

Chapitre 6

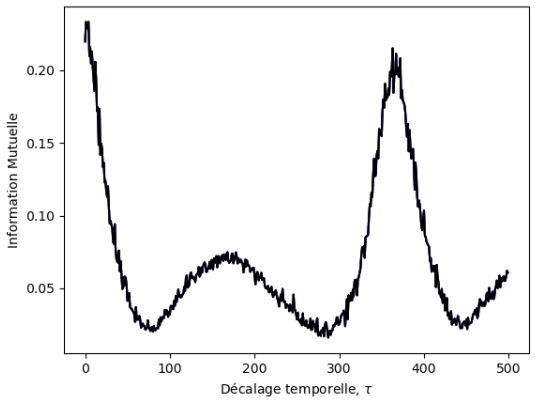
Application au cas de l'hôpital

Suite aux bons résultats obtenus sur les bases de données de UCR Time Series Archive, nous voulons maintenant appliquer ses méthodes sur les bases de données hospitalières. En particulier, nous avons choisi d'utiliser la projection sur la variété de Stiefel de chaque série temporelle (via le Delay Coordinate Embedding) suivie de l'algorithme de clustering UMAP + HDBSCAN. Pour chaque hôpital (Saint-Étienne et Grenoble), en utilisant cette méthode de clustering de série temporelle uni-dimensionnelle, nous voulons réunir la base de données virologique et la base de données du service des urgences. Cela nous permet de mettre en lumière les codes diagnostiques dûs aux virus et donc de suivre l'évolution de la population présentant ces diagnostiques. Un patient arrivant aux urgences présentant ces symptômes/diagnostiques est donc un patient fortement à risque probablement porteur de ces virus. On peut ainsi suivre l'évolution de ces épidémies à travers le nombre de patients arrivant aux urgences avec ces symptômes.

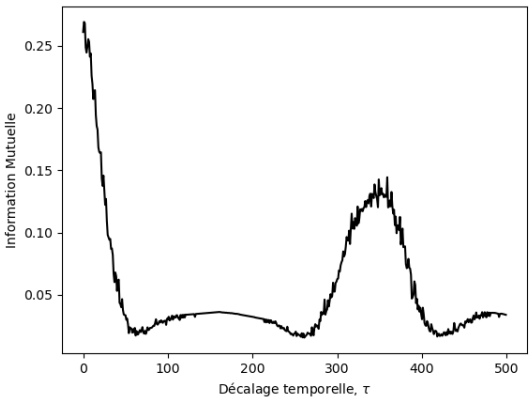
6.1 Détermination des paramètres pour le Delay Coordinate Embedding

Pour appliquer notre méthode, il faut tout d'abord déterminer les paramètres m et τ lors de la création de la matrice de trajectoire. Ici, on est dans un cas assez particulier du clustering. En effet, on ne cherche pas à clusteriser correctement l'ensemble de la base de données. Nous voulons seulement regrouper correctement chaque virus à des diagnostiques aux comportements similaires. Il nous est donc apparu pertinent d'utiliser ici l'information mutuelle sur la série temporelle $x(t)$ du virus (donc entre $x(t)$ et $x(t+\tau)$) pour déterminer le τ. Ainsi, on est sûr de bien récupérer la dynamique de la série temporelle des virus quitte à perdre de l'information sur des séries temporelles de diagnostiques aux dynamiques différentes. Nous proposons en Fig. 6.1 le tracé des informations

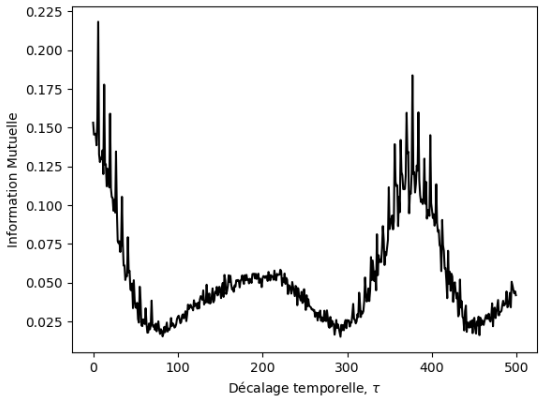
mutuelles pour chaque virus et chaque hôpital.



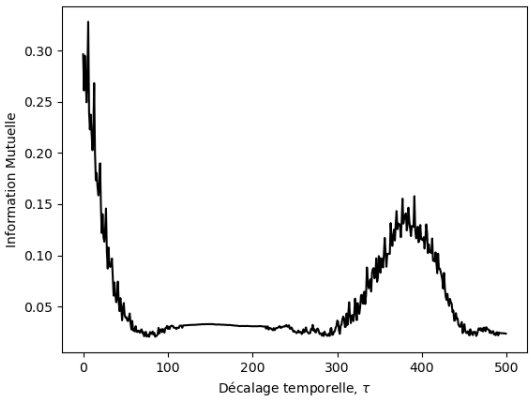
(a) Information mutuelle entre la série temporelle du VRS de l'hôpital de Saint-Étienne et la même série temporelle décalée de τ.



(b) Information mutuelle entre la série temporelle de la grippe de l'hôpital de Saint-Étienne et la même série temporelle décalée de τ.



(c) Information mutuelle entre la série temporelle du VRS de l'hôpital de Grenoble et la même série temporelle décalée de τ.



(d) Information mutuelle entre la série temporelle de la grippe de l'hôpital de Grenoble et la même série temporelle décalée de τ.

FIGURE 6.1 – Évolution de l'information mutuelle en fonction du τ. pour chaque série temporelle de virus de chaque hôpital. Les quatre séries temporelles suivent une évolution assez similaire. Après une décroissance jusqu'à $\tau = 60$, un plateau est atteint hormis un pic important aux alentours de $\tau = 365$. La pseudo-période de 365 jours apparaît donc clairement.

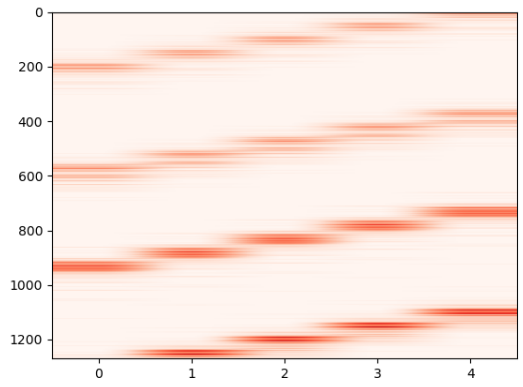
On remarque tout d'abord le comportement extrêmement similaire quelque soit le virus et l'hôpital. On observe ainsi une décroissance rapide jusqu'à atteindre un plateau (aux alentours de $\tau = 60$) puis un nouveau pic aux alentours de $\tau = 365$. Les virus, revenant chaque année mais pas tout à fait à la même date, il est logique de voir apparaître cette pseudo-période d'environ 365 jours. Nous décidons de nous servir de ça pour nos choix de paramètres sur la matrice de trajectoire. On

garde $m = 5$ (car peu impactant). Ensuite, pour garder un maximum d'information sur chacune des tranches de $\mathbb{R}^5$ (les lignes de la matrice de trajectoire), nous voulons que chaque colonne de la matrice de trajectoire ait le moins d'information mutuelle entre elle. Pour cela,un τ de 60 est pertinent. Ainsi, sur l'ensemble des 5 colonnes, on retrouve des décalages de 60,120,180 et 240. L'ensemble de ces décalages se situent tous dans les plateaux bas que l'on retrouve sur les courbes d'information mutuelle de Fig. 6.1. Nous gardons donc un maximum d'information sur chacune des tranches de $\mathbb{R}^5$ (les lignes de la matrice de trajectoire).

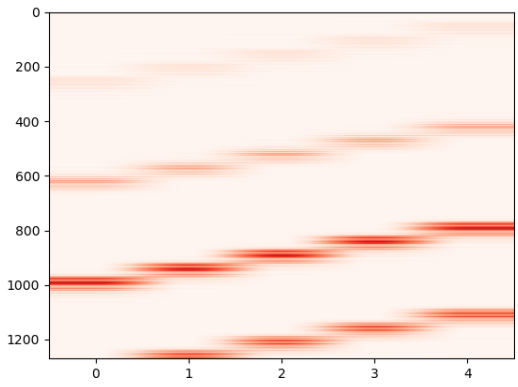
Nous proposons en Fig. 6.2 une illustration des matrices de trajectoire obtenues pour chaque virus et chaque hôpital. Chaque élément de la matrice est représenté par une case qui devient de plus en plus rouge selon l'amplitude de cet élément. On voit nettement apparaître l'épidémie saisonnière. Pour chaque ligne, une seule colonne porte l'information de l'épidémie (valeur nettement plus élevée). Puis, cette information se transmet de colonne en colonne par vague.

Pour la distance Euclidienne, nous avions mis dans une base de données commune les deux virus avec le reste des séries temporelles, puis une analyse rapide des éléments les plus proches avait été faite. Nous proposons de faire la même chose avec ces paramètres-là et la distance sur la variété de Grassmann représentée par la variété de Stiefel. Les résultats sont présentés en Table 6.1. Nous remarquons que quelque soit l'hôpital, un virus est l'élément le plus proche ou le deuxième plus proche de l'autre virus. Nous obtenons aussi une forte présence de symptômes respiratoires (codes J) et de symptômes cardiaques (codes I). Un autre code qui apparaît souvent est le B34 qui correspond à une infection virale. Cette table est donc beaucoup plus cohérente avec les résultats attendus par les médecins que la Table 2.3 précédente obtenue avec la distance Euclidienne. Cela confirme donc le bon choix des paramètres m et τ.

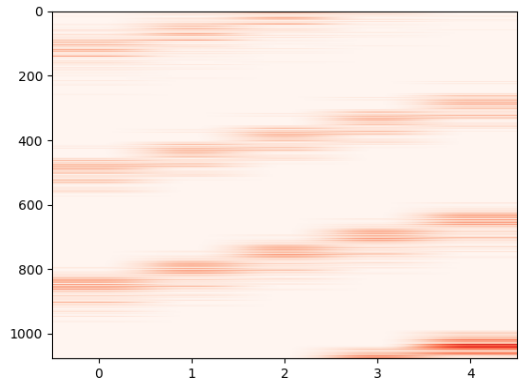
Maintenant que l'on a défini les paramètres du Delay Coordinate Embedding et qu'on a vérifié rapidement la cohérence médicale de la distance sur la Grassmann, on peut appliquer naturellement la suite de la méthode de clustering (UMAP + HDBSCAN).



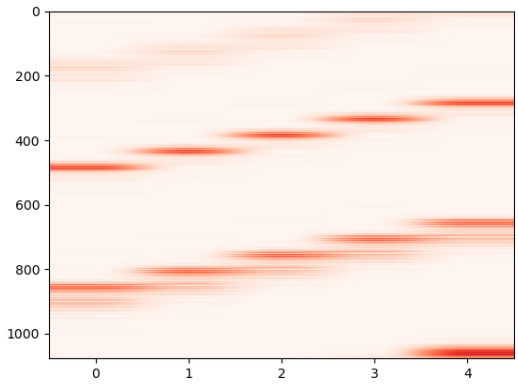
(a) Matrice de trajectoire obtenue avec $m = 5$ et $\tau = 60$ pour la série temporelle du VRS de l'hôpital de Saint-Étienne.



(b) Matrice de trajectoire obtenue avec $m = 5$ et $\tau = 60$ pour la série temporelle de la grippe de l'hôpital de Saint-Étienne.



(c) Matrice de trajectoire obtenue avec $m = 5$ et $\tau = 60$ pour la série temporelle du VRS de l'hôpital de Grenoble.



(d) Matrice de trajectoire obtenue avec $m = 5$ et $\tau = 60$ pour la série temporelle de la grippe de l'hôpital de Grenoble.

FIGURE 6.2 – Illustration des matrices de trajectoire des virus obtenues pour $m = 5$ et $\tau = 60$. Plus un élément de la matrice a une valeur élevée, plus il est rouge foncé. On remarque donc des vagues d'épidémies qui glissent sur les 5 colonnes. Cela permet de récupérer à chaque ligne une image assez complète de la dynamique de la série temporelle.

TABLE 6.1 – Table des éléments les plus proches au sens de la variété de Grassmann (avec $m = 5$ et $\tau = 60$). Pour un virus et un hôpital donné, les éléments proches sont dans la dernière colonne (classés par ordre de proximité).

Virus	**Hôpital**	**Éléments les plus proches**
Grippe	Grenoble	J11 VRS J18 B34 T14 J44
	Saint-Étienne	VRS J20 J18 R04 J15 I50
VRS	Grenoble	Grippe J11 J18 B34 I20 J44
	Saint-Étienne	Grippe J20 J18 B34 I50 J15

6.2 Résultats de clustering

Pour un couple (virus x hôpital), nous déterminons les codes CIM à surveiller en créant une base de données commune avec la série temporelle du virus concerné issue de la base virologique et les séries temporelles des diagnostiques (codes CIM) issues de la base du service d'urgence. Par exemple, pour l'hôpital de Grenoble, nous créons une base commune de 151 séries temporelles : 150 liées à l'arrivée de patients aux urgences pour un code CIM et une série temporelle associée à l'un des deux virus. On pourrait aussi créer une base de données avec les deux virus en même temps. Mais, les séries temporelles étant très proches, elles se retrouveraient forcément dans le même groupe (ou alors le résultat de clustering est mauvais). Les séparer permet de mettre en lumière des différences dans les diagnostiques avec un comportement similaire à ces virus. Les résultats de clustering sont présentés dans la Table 6.2.

On obtient une forte présence de codes J (symptômes respiratoires). On trouve

TABLE 6.2 – Résultats de clustering obtenus avec notre méthode. Pour un coule (hôpital x virus) donné, les codes CIM à surveiller sont dans la dernière colonne (classés par ordre alphabétique).

Virus	**Hôpital**	**Code**
Grippe	Grenoble	B34 E11 G35 I25 I48 I50 J00 J01 J11 J15 J18 J40 J42 J44 J45 J90 J96 K35 R57 Y44
	Saint-Étienne	I10 I20 I25 I48 I50 J18 J20 J44 J45 J96 R04 R57
VRS	Grenoble	B34 E11 G35 I10 I25 I48 I50 J00 J01 J11 J15 J18 J40 J42 J44 J90 J96 K35 R57 Y44
	Saint-Étienne	A09 D50 D64 F22 I10 I20 I25 I30 I48 I50 J15 J18 J20 J44 J45 J90 J96 R04 R07

aussi des codes I (symptômes cardiaques) qui sont des complications habituelles des virus hivernaux. On retrouve également B34 (infection virale) et A09 (gastro-entérite). Trouver ce type d'élément est très cohérent avec la présence de virus hivernaux.

La présence des autres codes peut être plus surprenante. Cependant, au vu de ces résultats de clustering, il nous semble pertinent de s'y intéresser car ces symptômes sont marqueurs d'un groupe de patients qui participe à l'engorgement hivernale même si leurs symptômes ne sont pas directement liés aux virus.

On peut aussi remarquer une forte base commune aux virus pour un même hôpital. En effet, on a 17 codes en commun entre les deux virus sur l'hôpital de Grenoble et 11 pour l'hôpital de Saint-Étienne. C'est logique de retrouver de nombreux codes en communs car la Grippe et le VRS ont une dynamique similaire. Réaliser un clustering pour chacun des virus permet de mettre en lumière les légères différences entre ces deux virus.

Ces résultats mettent aussi en lumière la différence d'habitude de codage entre deux hôpitaux. En effet, on a seulement 6 codes en commun entre les deux hôpitaux pour le même virus de la Grippe. De même, il y en a 8 pour le VRS. Même si les codes CIM reposent sur une classification mondiale, il y a une certaine flexibilité sur le choix des codes ce qui peut entraîner des différences entre deux hôpitaux. **Il est donc important de réaliser ce travail pour chaque hôpital pour avoir une surveillance précise de la présence d'un virus.** On a obtenu ces résultats sans s'intéresser aux paramètres d'HDBSCAN, laissés par défaut. En particulier, le paramètre "min_samples" permet d'obtenir des groupes de différentes tailles. Plus la valeur de ce paramètre est faible, plus le nombre de groupe aura tendance à être grand. Cela nous permet de mettre en lumière une hiérarchie de ces diagnostiques à surveiller. Par exemple, pour l'hôpital de Grenoble et la Grippe, avec "min_samples" à 1, nous obtenons un groupe avec seulement B34, J01 et J11. Avec "min_samples" à 2, nous obtenons en plus de ces 3 éléments J15, J18, J44, J45 et J96. À partir de "min_samples" à 3, nous retrouvons les résultats vu plus tôt dans la Table 6.2. En jouant avec ce paramètre, on peut donc garder différents niveaux de précision pour surveiller la présence des virus.

Les résultats obtenus ici sont utilisés pour la surveillance des virus dans les deux hôpitaux (Grenoble et Saint-Étienne) via le site web `http://predaflu.chu-st-etienne.fr/`**.**

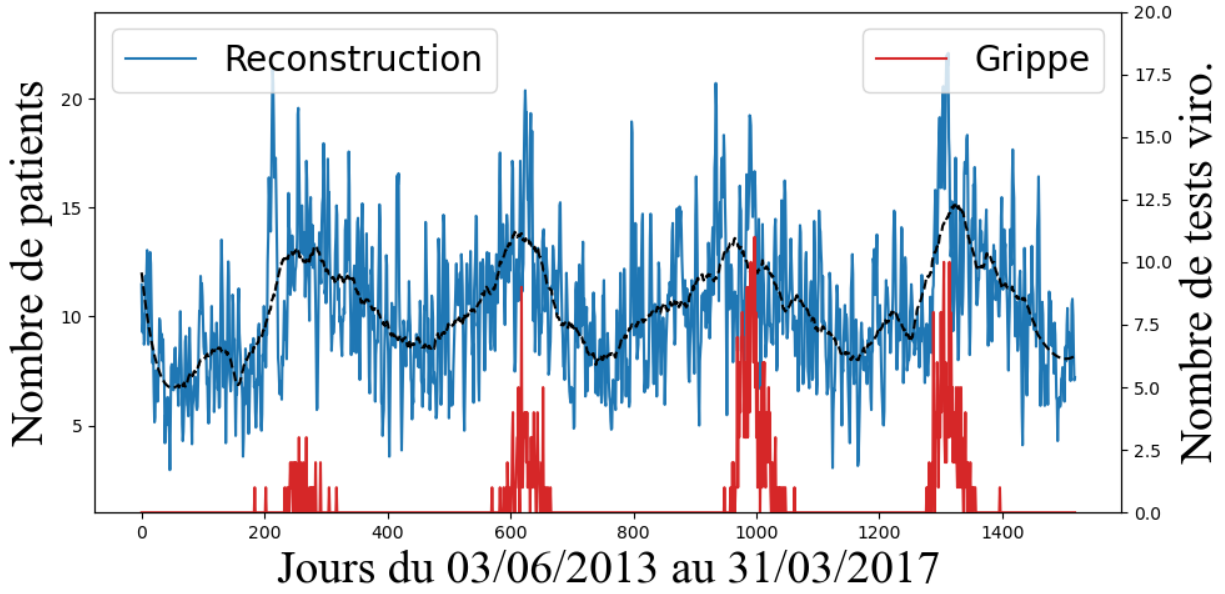
6.3 Reconstruction des séries temporelles des virus

En utilisant les résultats de la Table 6.2, nous pouvons, comme précédemment, reconstruire une série temporelle sensée marquer la présence d'un virus. Pour un virus et un hôpital donné, nous calculons la moyenne des séries temporelles de diagnostiques présents dans le groupe du virus. Pour chacun des deux virus, cette série temporelle signature du virus est tracée en même temps que la série temporelle de ce virus en Fig. 6.3 (pour Saint-Étienne) et Fig. 6.4 (pour Grenoble).

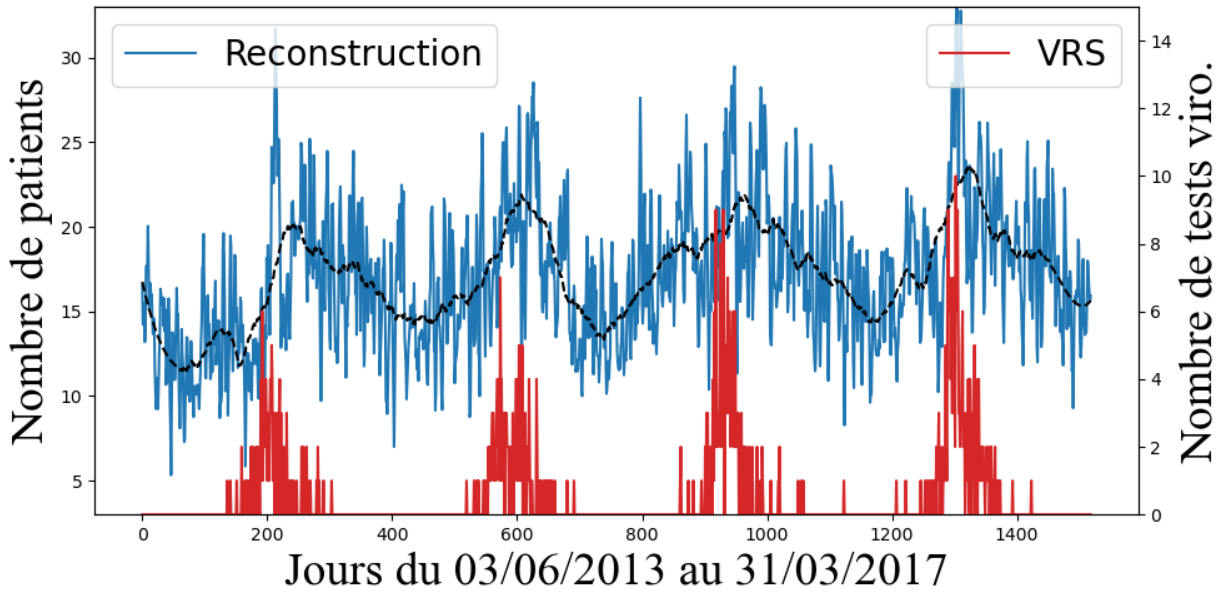
Les reconstructions sont imparfaites mais on voit clairement une reproduction des pics d'activités des virus sur la série temporelle de la reconstruction. Nous avons donc obtenu pour chaque virus et chaque hôpital une représentation journalière de la présence des virus en utilisant uniquement les codes CIM. On rappelle qu'avec la distance Euclidienne associée à K-means, il était impossible de déceler la dynamique particulière des virus avec une admission quasiment constante au cours de l'année.

6.4 Synthèse

Suite aux bons résultats obtenus sur les bases de données de UCR Time Series Archive, nous avons lancé notre méthode sur les bases de données des CHU. Avec un τ de 60 pour maximiser l'information sur les matrices de trajectoire, nous obtenons des résultats cohérents (avec par exemple une forte présence de codes J). Les résultats de clustering obtenus permettent d'avoir une vue en temps réel de la présence du virus. En particulier, nous avons pu obtenir des reconstructions pertinentes des virus à partir des codes diagnostiques. Avec ces reconstructions, on peut appliquer des méthodes pour anticiper une activité anormale. De plus, cela permet de mettre en lumière des patients dits à risques. Par exemple, un patient arrivant au CHU de Saint-Étienne avec un code A09 est vu comme un porteur potentiel du VRS. Il est donc possible de l'isoler, d'anticiper ses besoins médicaux, et de réaliser en priorité sur lui un test PCR. Enfin, nos résultats mettent en lumière le besoin d'effectuer ce travail pour chaque hôpital. En effet, les habitudes locales de codage rendent impossible de déterminer globalement (à l'échelle nationale par exemple) des codes marqueurs d'un virus hivernal. Ce chapitre a fait l'objet d'un article "Using a manifold-based approach to extract

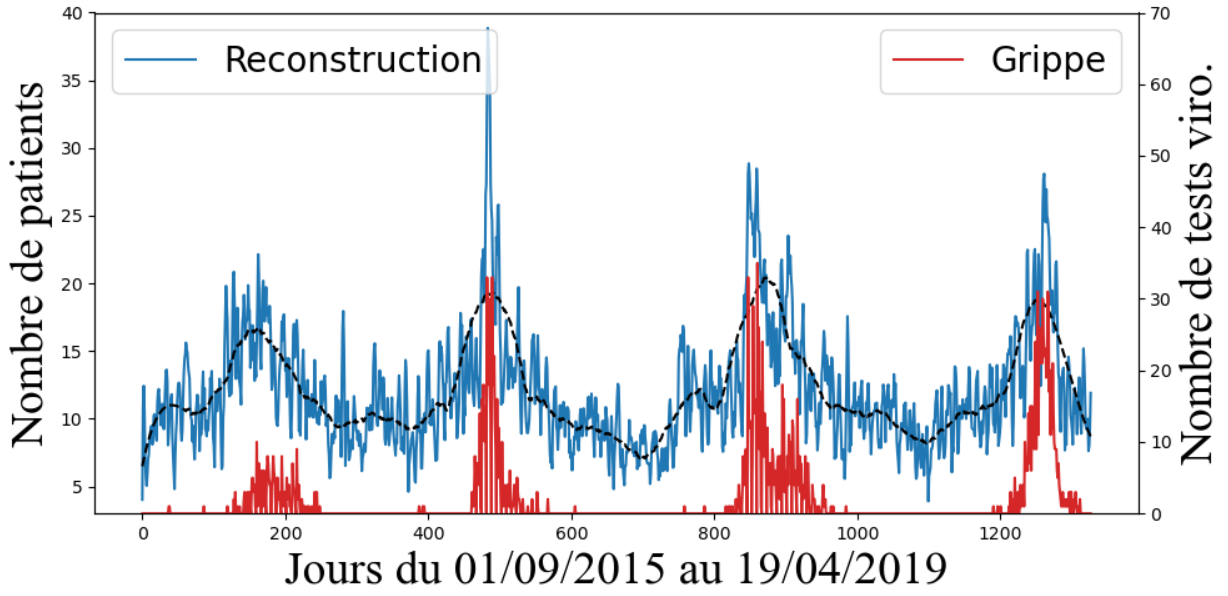


(a) Reconstruction de la Grippe.

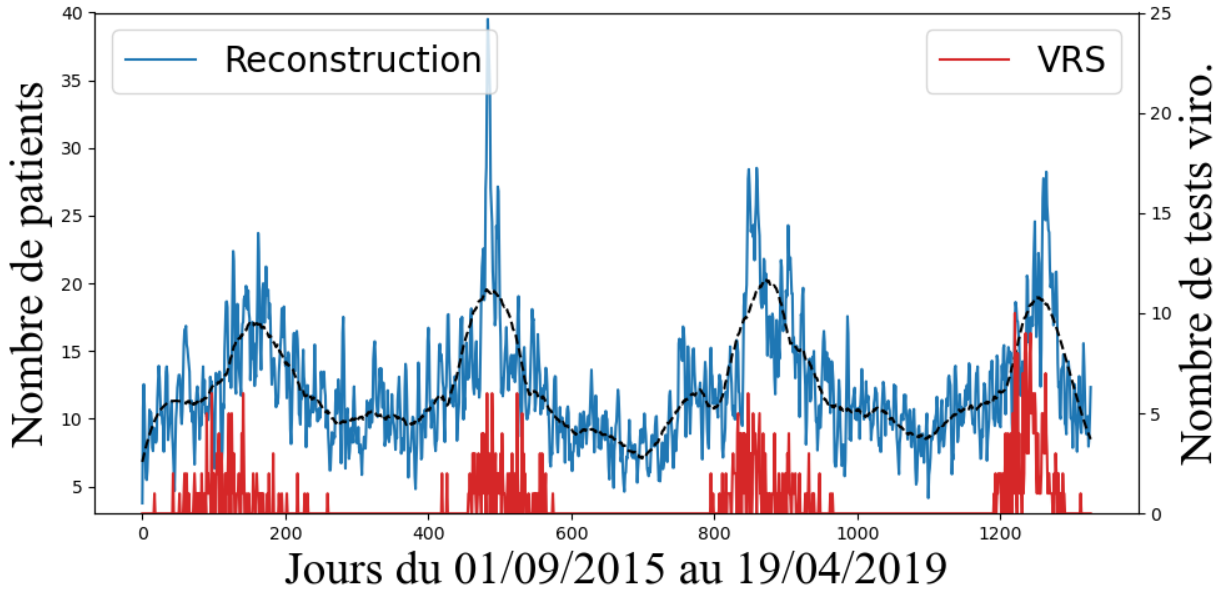


(b) Reconstruction du VRS.

FIGURE 6.3 – Reconstruction pour l'hôpital de Saint-Étienne. La série temporelle d'un virus - (a) pour la Grippe et (b) pour le VRS - est tracée en rouge en même temps que la série temporelle moyenne des séries temporelles des symptômes présents dans le groupe du virus (en bleu).Pour mettre en évidence la tendance globale de cette reconstruction, nous soulignons en pointillé noir la série temporelle de reconstruction lissée.



(a) Reconstruction de la Grippe.



(b) Reconstruction du VRS.

FIGURE 6.4 – Reconstruction pour l'hôpital de Grenoble. La série temporelle d'un virus - (a) pour la Grippe et (b) pour le VRS - est tracée en rouge en même temps que la série temporelle moyenne des séries temporelles des symptômes présents dans le groupe du virus (en bleu). Pour mettre en évidence la tendance globale de cette reconstruction, nous soulignons en pointillé noir la série temporelle de reconstruction lissée.

clinical codes associated with winter respiratory viruses at an emergency department " soumis au journal "Health Care Management Science".

Chapitre 7

Conclusions et perspectives de recherche

7.1 Résumé des contributions

Nous avons souhaité aborder le problème du clustering de séries temporelles uni-dimensionnelles pour aider à la prise de décisions des hôpitaux. Pour réaliser un clustering, il est nécessaire de définir une distance entre les éléments à comparer et ensuite d'appliquer un algorithme de clustering. Sur ces deux étapes, nous avons voulu apporter un nouveau regard par rapport à ce qu'il se fait de plus classique.

Sur les séries temporelles uni-dimensionnelles, les deux distances les plus souvent utilisées sont la distance Euclidienne et DTW. Cependant, dans notre cas, cela ne nous permettait pas de bien récupérer la dynamique des séries temporelles. Dans la littérature sur les systèmes dynamiques, le Delay Coordinate Embedding est souvent utilisé pour analyser une série temporelle. Une série temporelle uni-dimensionnelle est alors vue comme une mesure discrète d'un système dynamique. Le théorème de Takens nous assure qu'on peut récupérer ce système dynamique en utilisant le Delay Coordinate Embedding en créant une matrice de trajectoire à partir de la série temporelle. Cette méthode, très utilisée pour l'analyse des systèmes dynamiques, ne l'est pas -à notre connaissance- pour le clustering de série temporelle. En utilisant cette méthode, chaque série temporelle d'une base de donnée est alors transformée en matrice de trajectoire.

Au niveau des algorithmes de clustering, les algorithmes les plus classiques sont le clustering Hiérarchique et le K-means. Là-aussi, on a voulu creuser plus loin et utiliser des algorithmes de

clustering par densité.

7.1.1 Représentation des matrices de trajectoire

Pour utiliser le Delay Coordinate Embedding pour du clustering, il faut déterminer une distance pour comparer les matrices de trajectoire. Pour cela, en s'appuyant sur l'apprentissage sur variété, qu'on retrouve dans la littérature sur les séries temporelles multidimensionnelles, nous nous sommes penchés sur la notion de variété. Nous avons alors proposés plusieurs géométries différentes pour analyser les matrices de trajectoire. Ensuite, en utilisant la géométrie de Riemann, on a pu définir une distance sur ces variétés et ainsi appliquer des algorithmes de clustering. À travers une profonde analyse comparative, nous avons mis en évidence la pertinence d'utiliser la variété de Grassmann représentée par des éléments de la variété de Stiefel pour analyser les matrices de trajectoire. Sur cette variété, nous avons utilisé les angles principaux comme métrique. De plus, en appliquant la méthode de la descente de gradient pour minimiser la somme des distances au carré, nous avons pu déterminer une moyenne sur cette géométrie. Parmi les autres géométries explorées, nous avons proposé une représentation sur la Sphère. Celle-ci propose des résultats similaires à la représentation par la variété de Stiefel avec l'avantage de pouvoir comparer des variétés de Stiefel de différentes dimensions.

7.1.2 Comparaison des algorithmes de clustering

Une fois que l'on a défini une distance sur nos séries temporelles, quels algorithmes de clustering utiliser ? Nous avons proposé ici aussi une analyse comparative détaillée pour cela. Dans cette analyse, nous avons intégré une méthode qui n'a(vait) pas encore fait ses preuves : UMAP + HDBSCAN. L'utilisation de UMAP + HDBSCAN est discutable car en théorie UMAP peut faire apparaître des pseudo-groupes dans la donnée. Cependant, nous avons montré que sur de multiples cas réels et pour plusieurs distances, cette méthode est nettement supérieure à des algorithmes plus classiques comme K-means.

7.1.3 Adaptation de UMAP sur n'importe quelle variété

Nos travaux sur les géométries et sur la réduction de dimension nous ont poussé à adapter UMAP sur n'importe quelle variété. L'idée de départ est que pour représenter au mieux des éléments d'une

certaine variété de dimension d, il est plus pertinent d'utiliser la même variété de dimension $p, p < d$ plutôt qu'un espace Euclidien de dimension p. Pour permettre à UMAP de projeter sur n'importe quelle variété, nous avons transformé la dernière étape de cet algorithme - le forced graph layout - pour prendre en compte la structure de la variété. En s'appuyant sur ce que nous avons vu dans la géométrie de Riemann, les forces d'attractions et de répulsions ont été modifiées pour prendre en compte la courbure de la variété.

7.1.4 Pistes pour déterminer les paramètres du Delay Coordinate Embedding

Nous avons obtenu de très bons résultats avec le Delay Coordinate Embedding avec une méthode semi-supervisée. Les paramètres m et τ étaient alors optimisés en fonction des résultats de clustering. Pour obtenir une vraie méthode de clustering complètement non supervisée, nous avons cherché à déterminer à l'avance ces deux paramètres. Nous avons montré que m n'est pas un élément important pour la performance de cette méthode contrairement à τ qui est primordial. Pour déterminer τ, nous avons exploré plusieurs pistes issues du domaine des systèmes dynamiques ou du domaine du clustering. En particulier, nous avons mis en évidence que le meilleur moyen de déterminer τ dans le cas général était l'indicateur d'Hopkins.

7.1.5 Application à l'analyse des flux dans les services d'urgences

Suite aux bons résultats obtenus, nous avons utilisé le Delay Coordinate Embedding sur la variété de Stiefel avec UMAP + HBDSCAN pour déterminer les diagnostiques à risques, marqueurs de la présence du virus. Pour la question du τ, nous étions ici dans un cas particulier où nous avions une série temporelle "cible" : celle du virus. Nous nous sommes donc appuyés sur l'information mutuelle pour obtenir un τ qui récupère au mieux la dynamique de cette série temporelle. Le résultat de clustering obtenu, cohérent avec ce qui est attendu, nous permet de mettre en lumière les patients à risques et de mieux comprendre le flux de patients aux urgences. C'est un outil d'aide à la décision pertinent pour l'hôpital.

7.2 Perspectives

Nous envisageons encore plusieurs pistes d'améliorations sur les recherches que nous avons entreprises :

— Nous avons obtenu avec l'indicateur d'Hopkins un τ qui donne des bons résultats. Cependant, la qualité de ces résultats reste assez éloignée de ce que l'on obtenait en faisant varier la valeur de τ entre 1 et 20. Il est donc peut-être possible de trouver un meilleur critère pour déterminer τ.

— Nous pourrions aussi vouloir utiliser un τ propre à chaque série temporelle d'une base de donnée. Cela nous permettrait d'utiliser des méthodes issues des systèmes dynamiques comme l'information mutuelle. Cependant, dans ce cas, il faut trouver une méthode pour analyser des matrices de tailles différdentes.

— Pour l'analyse des matrices de trajectoire, on peut imaginer d'autres variétés à utiliser plutôt que celle que nous avons proposé.

— Notre proposition d'amélioration de UMAP a le problème d'être lent. L'utilisation du logarithme et de l'exponentielle de Riemann prend beaucoup de temps de calcul. Pour obtenir un algorithme plus rapide, il faut trouver des méthodes plus efficaces pour calculer ces deux fonctions.

— Sur notre amélioration de UMAP, une initialisation aléatoire sur l'espace réduit est utilisée pour le moment. Cela entraîne une perte de précision et ralentit la convergence du forced graph layout. Il est donc nécessaire de trouver une méthode pour obtenir une première réduction de dimension sur l'espace réduit qui donnera la position initiale des éléments avant le forced graph layout.

Annexe A

Résultats complets pour l'analyse des algorithmes de clustering

Bases	Kmeans	Hier.	HDB.	U.+Kme.	U.+Hier.	U.+HDB.
50words	0.619	0.618	0.186	0.576	0.640	0.576
Adiac	0.167	0.060	0.315	0.164	0.164	0.569
ArrowHead	0.476	0.051	0.000	0.466	0.466	0.531
Beef	0.278	0.265	0.376	0.104	0.104	0.110
BeetleFly	0.234	0.122	0.000	0.167	0.122	0.007
BirdChicken	0.265	0.301	0.000	0.300	0.300	0.283
Car	0.152	0.116	0.000	0.139	0.139	0.293
CBF	0.234	0.281	0.000	0.221	0.224	0.106
Chlo...tion	0.018	0.006	0.023	0.001	0.001	0.025
CinC...orso	0.405	0.469	0.388	0.460	0.572	0.350
Coffee	0.708	0.110	0.464	1.000	1.000	0.559
Computers	0.025	0.016	0.003	0.006	0.019	0.063
CricketX	0.060	0.071	0.132	0.050	0.050	0.308
CricketY	0.154	0.174	0.162	0.058	0.058	0.291
CricketZ	0.054	0.038	0.116	0.153	0.040	0.249
Diat...tion	0.941	1.000	0.000	0.928	0.928	0.007
Dist...roup	0.242	0.176	0.262	0.282	0.282	0.289
Dist...rect	0.041	0.009	0.078	0.012	0.012	0.019
Dist...nxTW	0.483	0.521	0.439	0.521	0.521	0.528
Earthquakes	0.014	0.003	0.008	0.001	0.002	0.050
ECG200	0.153	0.062	0.299	0.185	0.185	0.289
ECG5000	0.693	0.740	0.671	0.745	0.745	0.424
ECG...ays	0.028	0.015	0.000	0.000	0.000	0.065
Elec...ices	0.198	0.006	0.188	0.340	0.343	0.293
FaceAll	0.071	0.054	0.152	0.500	0.342	0.546
FaceFour	0.474	0.486	0.000	0.381	0.343	0.007
FacesUCR	0.363	0.058	0.326	0.283	0.281	0.287
FISH	0.125	0.042	0.114	0.179	0.204	0.462

TABLE A.1 – Résultats complets pour 6 algorithmes de clustering (avec la distance Euclidienne) sur 85 bases de données (1/3)

Bases	Kmeans	Hier.	HDB.	U.+Kme.	U.+Hier.	U.+HDB.
FordA	0.013	0.010	0.015	0.000	0.000	0.052
FordB	0.014	0.001	0.010	0.003	0.009	0.026
GunPoint	0.061	0.089	0.305	0.013	0.013	0.019
Ham	0.039	0.018	0.000	0.035	0.009	0.058
Hand...ines	0.227	0.008	0.134	0.140	0.134	0.201
Haptics	0.134	0.017	0.090	0.094	0.091	0.162
Herring	0.014	0.021	0.000	0.000	0.002	0.007
InlineSkate	0.146	0.133	0.083	0.058	0.082	0.188
Inse...ound	0.540	0.557	0.357	0.529	0.543	0.533
Ital...emand	0.027	0.083	0.402	0.607	0.607	0.560
Larg...ances	0.042	0.097	0.057	0.082	0.088	0.076
Lighting2	0.017	0.039	0.000	0.029	0.039	0.078
Lighting7	0.211	0.198	0.297	0.308	0.198	0.205
MALLAT	0.813	0.697	0.700	0.697	0.697	0.788
Meat	0.747	0.707	0.691	0.734	0.734	0.740
Med...ages	0.236	0.015	0.131	0.168	0.171	0.183
Midd...roup	0.118	0.111	0.139	0.020	0.020	0.117
Midd...rect	0.015	0.016	0.003	0.039	0.015	0.009
Midd...nxTW	0.497	0.470	0.443	0.506	0.506	0.512
MoteStrain	0.248	0.081	0.000	0.278	0.192	0.372
NonI...rax1	0.277	0.001	0.075	0.531	0.548	0.789
NonI...rax2	0.224	0.001	0.696	0.800	0.800	0.844
OliveOil	0.213	0.200	0.589	0.486	0.486	0.516
OSULeaf	0.088	0.064	0.304	0.259	0.104	0.288
Phal...rect	0.016	0.002	0.048	0.000	0.000	0.009
Phoneme	0.513	0.484	0.091	0.485	0.495	0.274
Plane	0.836	0.841	0.893	0.831	0.831	0.902
Prox...roup	0.537	0.522	0.516	0.524	0.524	0.530

TABLE A.2 – Résultats complets pour 6 algorithmes de clustering (avec la distance Euclidienne) sur 85 bases de données (2/3)

Bases	Kmeans	Hier.	HDB.	U.+Kme.	U.+Hier.	U.+HDB.
Prox...rect	0.076	0.012	0.055	0.034	0.034	0.039
Prox...nxTW	0.615	0.599	0.599	0.602	0.602	0.608
Refr...ices	0.015	0.030	0.010	0.005	0.005	0.054
ScreenType	0.041	0.009	0.025	0.007	0.007	0.037
ShapeletSim	0.020	0.147	0.000	0.000	0.031	0.007
ShapesAll	0.179	0.165	0.512	0.389	0.419	0.652
Smal...nces	0.040	0.011	0.053	0.016	0.019	0.071
Sony...face	0.287	0.157	0.000	0.018	0.018	0.007
Sony...ceII	0.308	0.295	0.291	0.295	0.295	0.290
Star...rves	0.625	0.760	0.736	0.760	0.760	0.766
Strawberry	0.099	0.004	0.228	0.009	0.008	0.169
SwedishLeaf	0.247	0.004	0.218	0.329	0.328	0.562
Symbols	0.800	0.810	0.000	0.843	0.843	0.636
Synt...trol	0.785	0.456	0.825	0.623	0.623	0.590
ToeS...ion1	0.015	0.002	0.010	0.007	0.007	0.015
ToeS...ion2	0.013	0.002	0.059	0.010	0.010	0.033
Trace	0.682	0.669	0.669	0.669	0.669	0.500
TwoPatterns	0.021	0.031	0.174	0.027	0.060	0.229
TwoLeadECG	0.099	0.086	0.000	0.086	0.086	0.093
uWav...aryX	0.277	0.218	0.249	0.498	0.437	0.478
uWav...aryY	0.270	0.200	0.350	0.410	0.405	0.464
uWav...aryZ	0.424	0.292	0.405	0.527	0.548	0.432
uWav...yAll	0.646	0.673	0.582	0.789	0.782	0.730
Wafer	0.013	0.000	0.257	0.000	0.000	0.137
Wine	0.014	0.091	0.001	0.001	0.001	0.030
Wor...yms	0.305	0.344	0.221	0.352	0.349	0.388
Worms	0.027	0.048	0.144	0.033	0.079	0.108
Wor...lass	0.013	0.012	0.032	0.007	0.010	0.019
Yoga	0.021	0.010	0.054	0.001	0.001	0.048

TABLE A.3 – Résultats complets pour 6 algorithmes de clustering (avec la distance Euclidienne) sur 85 bases de données (3/3)

Annexe B

Résultats complets de clustering pour l'analyse comparative des variétés

	Frobenius				Sphere				Grassmann				Euclidien			
Bases	Kmeans	Hiér.	HDB.	U.+HDB.	Kmeans	Hiér.	HDB.	U.+HDB.	Kmeans	Hiér.	HDB.	U.+HDB.	Kmeans	Hiér.	HDB.	U.+HDB.
50words	0.442	0.621	0.187	0.546	0.190	0.036	0.298	0.545	0.468	0.045	0.245	0.532	0.704	0.695	0.186	0.531
Adiac	0.162	0.060	0.306	0.208	0.186	0.005	0.311	0.580	0.468	0.092	0.465	0.651	0.154	0.060	0.315	0.534
ArrowHead	0.390	0.051	0.378	0.451	0.445	0.540	0.553	0.489	0.532	0.051	0.319	0.607	0.463	0.051	0.000	0.371
Beef	0.265	0.265	0.376	0.104	0.182	0.238	0.214	0.104	0.352	0.265	0.160	0.104	0.265	0.265	0.376	0.104
BeetleFly	0.174	0.206	0.000	0.000	0.356	0.399	0.000	0.243	0.456	0.423	0.000	0.347	0.302	0.206	0.000	0.000
BirdChicken	0.313	0.307	0.000	0.221	0.170	0.283	0.000	0.158	0.3	0.283	0.000	0.342	0.320	0.313	0.000	0.233
Car	0.149	0.277	0.000	0.139	0.448	0.576	0.440	0.303	0.348	0.040	0.000	0.343	0.139	0.116	0.000	0.303
CBF	0.649	0.211	0.000	0.281	0.274	0.420	0.000	0.274	0.512	0.431	0.000	0.467	0.221	0.281	0.000	0.264
Chlo...tion	0.005	0.015	0.018	0.021	0.005	0.015	0.020	0.012	0.012	0.032	0.029	0.030	0.005	0.006	0.023	0.017
CinC...orso	0.599	0.655	0.388	0.485	0.902	0.902	0.687	0.902	0.902	0.902	0.662	0.902	0.655	0.637	0.388	0.198
Coffee	0.812	0.524	0.476	1.000	1.000	0.812	0.871	1.000	1	0.842	0.000	1.000	0.700	0.110	0.464	0.871
Computers	0.018	0.009	0.018	0.037	0.011	0.008	0.024	0.062	0.072	0.054	0.066	0.070	0.046	0.086	0.003	0.052
CricketX	0.049	0.062	0.135	0.122	0.198	0.011	0.060	0.381	0.247	0.296	0.098	0.413	0.047	0.071	0.132	0.315
CricketY	0.174	0.095	0.167	0.110	0.119	0.005	0.140	0.311	0.266	0.021	0.102	0.364	0.141	0.174	0.162	0.191
CricketZ	0.043	0.041	0.143	0.164	0.285	0.022	0.109	0.326	0.284	0.036	0.114	0.391	0.041	0.038	0.116	0.235
Diat...tion	0.648	1.000	0.000	0.789	0.641	0.884	0.000	0.606	0.873	0.504	0.000	0.629	0.928	1.000	0.000	0.000
Dist...roup	0.194	0.012	0.258	0.282	0.185	0.279	0.311	0.207	0.282	0.334	0.374	0.282	0.229	0.176	0.262	0.282
Dist...rect	0.051	0.009	0.065	0.024	0.046	0.009	0.120	0.023	0.096	0.070	0.115	0.021	0.028	0.009	0.078	0.012
Dist...nxTW	0.474	0.521	0.437	0.521	0.521	0.521	0.523	0.521	0.521	0.551	0.562	0.521	0.470	0.521	0.439	0.521
Earthquakes	0.006	0.022	0.008	0.020	0.000	0.007	0.000	0.009	0.028	0.019	0.007	0.043	0.064	0.077	0.008	0.025
ECG200	0.228	0.289	0.314	0.289	0.350	0.224	0.000	0.268	0.231	0.319	0.239	0.423	0.230	0.289	0.299	0.185
ECG5000	0.672	0.005	0.669	0.601	0.760	0.819	0.782	0.692	0.666	0.005	0.640	0.691	0.680	0.740	0.671	0.559
ECG...ays	0.000	0.015	0.000	0.002	0.028	0.231	0.000	0.059	0.123	0.015	0.003	0.136	0.015	0.015	0.000	0.007
FaceAll	0.055	0.004	0.456	0.559	0.727	0.121	0.549	0.772	0.466	0.136	0.224	0.737	0.059	0.054	0.152	0.565
FaceFour	0.381	0.537	0.000	0.588	0.687	0.736	0.000	0.601	0.657	0.813	0.000	0.698	0.484	0.486	0.000	0.417
FacesUCR	0.115	0.022	0.252	0.416	0.251	0.102	0.253	0.637	0.493	0.102	0.185	0.607	0.629	0.058	0.326	0.188

TABLE B.1 – Résultats complets pour 16 méthodes et 79 bases de données (1/3)

	Frobenius				Sphere				Grassmann				Euclidien			
Bases	Kmeans	Hiér.	HDB.	U.+HDB.	Kmeans	Hiér.	HDB.	U.+HDB.	Kmeans	Hiér.	HDB.	U.+HDB.	Kmeans	Hiér.	HDB.	U.+HDB.
FISH	0.125	0.124	0.113	0.479	0.456	0.309	0.299	0.336	0.452	0.011	0.232	0.542	0.112	0.042	0.114	0.433
FordA	0.000	0.000	0.019	0.042	0.187	0.016	0.016	0.028	0.035	0.030	0.047	0.135	0.000	0.010	0.015	0.051
FordB	0.001	0.005	0.013	0.023	0.036	0.010	0.020	0.009	0.013	0.002	0.012	0.077	0.000	0.001	0.010	0.023
GunPoint	0.363	0.352	0.315	0.429	0.208	0.253	0.078	0.358	0.374	0.013	0.438	0.475	0.351	0.379	0.305	0.013
Ham	0.020	0.018	0.000	0.103	0.020	0.018	0.000	0.046	0.116	0.018	0.000	0.103	0.155	0.018	0.000	0.066
Haptics	0.098	0.017	0.070	0.156	0.079	0.017	0.081	0.182	0.088	0.017	0.000	0.106	0.121	0.017	0.090	0.091
Herring	0.003	0.021	0.000	0.005	0.017	0.021	0.000	0.005	0.023	0.039	0.000	0.030	0.001	0.021	0.000	0.005
InlineSkate	0.082	0.101	0.083	0.230	0.024	0.064	0.074	0.054	0.056	0.064	0.074	0.159	0.135	0.133	0.083	0.186
Inse...ound	0.520	0.548	0.360	0.293	0.475	0.423	0.288	0.290	0.511	0.324	0.279	0.467	0.547	0.557	0.357	0.552
Ital...emand	0.014	0.083	0.035	0.551	0.458	0.474	0.308	0.390	0.386	0.491	0.205	0.713	0.014	0.083	0.402	0.607
Lighting2	0.039	0.039	0.000	0.051	0.035	0.019	0.022	0.059	0.056	0.026	0.000	0.051	0.007	0.039	0.000	0.067
Lighting7	0.138	0.198	0.297	0.198	0.442	0.033	0.210	0.291	0.377	0.198	0.152	0.198	0.198	0.198	0.297	0.198
MALLAT	0.800	0.697	0.700	0.781	0.644	0.716	0.774	0.698	0.767	0.796	0.751	0.791	0.800	0.697	0.700	0.781
Meat	0.734	0.707	0.691	0.734	0.734	0.707	0.691	0.721	0.542	0.031	0.617	0.740	0.734	0.707	0.691	0.734
Med...ages	0.290	0.004	0.126	0.140	0.070	0.012	0.084	0.078	0.212	0.298	0.112	0.222	0.355	0.015	0.131	0.143
Midd...roup	0.019	0.112	0.139	0.129	0.051	0.139	0.139	0.139	0.139	0.113	0.078	0.139	0.013	0.111	0.139	0.111
Midd...rect	0.023	0.003	0.002	0.003	0.091	0.027	0.005	0.010	0.212	0.003	0.022	0.115	0.002	0.016	0.003	0.003
Midd...nxTW	0.461	0.466	0.454	0.506	0.506	0.506	0.506	0.506	0.506	0.424	0.477	0.506	0.484	0.470	0.443	0.506
MoteStrain	0.204	0.081	0.000	0.097	0.179	0.081	0.000	0.196	0.392	0.275	0.000	0.258	0.438	0.475	0.000	0.293
NonI...rax1	0.261	0.001	0.075	0.774	0.249	0.001	0.478	0.658	0.431	0.001	0.159	0.708	0.264	0.001	0.075	0.765
NonI...rax2	0.214	0.001	0.683	0.816	0.253	0.001	0.616	0.788	0.557	0.001	0.609	0.794	0.211	0.001	0.696	0.839
OliveOil	0.516	0.200	0.589	0.486	0.583	0.200	0.568	0.486	0.492	0.679	0.459	0.486	0.200	0.200	0.589	0.486
OSULeaf	0.072	0.322	0.273	0.317	0.372	0.485	0.440	0.402	0.26	0.390	0.248	0.434	0.318	0.424	0.304	0.285
Phal...rect	0.005	0.009	0.065	0.003	0.000	0.002	0.071	0.001	0.069	0.007	0.072	0.057	0.003	0.002	0.048	0.002
Phoneme	0.505	0.500	0.000	0.254	0.528	0.010	0.138	0.255	0.462	0.026	0.097	0.327	0.589	0.569	0.091	0.290
Plane	0.796	0.841	0.893	0.891	0.798	0.923	0.929	0.948	0.866	0.870	0.853	0.948	0.823	0.841	0.893	0.831

TABLE B.2 – Résultats complets pour 16 méthodes et 79 bases de données (2/3)

	Frobenius				Sphere				Grassmann				Euclidien			
Bases	Kmeans	Hiér.	HDB.	U.+HDB.	Kmeans	Hiér.	HDB.	U.+HDB.	Kmeans	Hiér.	HDB.	U.+HDB.	Kmeans	Hiér.	HDB.	U.+HDB.
Prox...roup	0.524	0.005	0.516	0.524	0.524	0.524	0.525	0.524	0.524	0.524	0.524	0.524	0.524	0.522	0.516	0.524
Prox...rect	0.056	0.012	0.066	0.033	0.038	0.057	0.097	0.034	0.085	0.050	0.088	0.055	0.063	0.012	0.055	0.034
Prox...nxTW	0.602	0.599	0.599	0.602	0.581	0.581	0.599	0.602	0.602	0.602	0.602	0.602	0.602	0.599	0.599	0.602
Refr...ices	0.025	0.030	0.010	0.021	0.019	0.004	0.054	0.059	0.035	0.045	0.004	0.086	0.038	0.063	0.010	0.041
ShapeletSim	0.278	0.042	0.000	0.000	0.126	0.287	0.000	0.000	0.126	0.027	0.000	0.012	0.147	0.165	0.000	0.000
ShapesAll	0.169	0.465	0.514	0.634	0.717	0.660	0.493	0.666	0.669	0.007	0.478	0.693	0.166	0.165	0.512	0.636
Smal...nces	0.056	0.043	0.053	0.082	0.024	0.005	0.085	0.091	0.019	0.065	0.072	0.076	0.079	0.063	0.053	0.068
Sony...face	0.115	0.157	0.000	0.070	0.115	0.157	0.000	0.025	0.557	0.416	0.000	0.140	0.490	0.157	0.000	0.038
Star...rves	0.610	0.604	0.736	0.656	0.760	0.760	0.471	0.750	0.603	0.760	0.525	0.760	0.612	0.607	0.736	0.760
Strawberry	0.084	0.062	0.224	0.143	0.145	0.017	0.056	0.293	0.184	0.048	0.164	0.249	0.086	0.004	0.228	0.195
SwedishLeaf	0.234	0.042	0.218	0.550	0.262	0.008	0.357	0.631	0.519	0.008	0.313	0.620	0.234	0.004	0.218	0.546
Symbols	0.000	0.651	0.000	0.000	0.691	0.872	0.000	0.547	0.897	0.654	0.000	0.875	0.787	0.810	0.000	0.580
ToeS...ion1	0.002	0.032	0.042	0.003	0.072	0.163	0.129	0.031	0.238	0.202	0.000	0.132	0.144	0.055	0.010	0.035
ToeS...ion2	0.280	0.360	0.017	0.043	0.593	0.430	0.000	0.616	0.743	0.126	0.000	0.585	0.309	0.317	0.059	0.125
Trace	0.669	0.669	0.600	0.669	0.014	0.457	0.599	0.237	0.476	0.588	0.462	0.484	0.669	0.669	0.669	0.498
TwoPatterns	0.012	0.017	0.170	0.209	0.036	0.020	0.022	0.168	0.01	0.002	0.015	0.126	0.013	0.031	0.174	0.198
TwoLeadECG	0.086	0.086	0.000	0.086	0.082	0.082	0.175	0.082	0.174	0.252	0.000	0.197	0.086	0.086	0.000	0.086
uWav...aryX	0.261	0.271	0.246	0.476	0.321	0.003	0.119	0.205	0.128	0.002	0.098	0.290	0.264	0.218	0.249	0.438
uWav...aryY	0.257	0.406	0.336	0.411	0.214	0.002	0.074	0.269	0.097	0.003	0.068	0.261	0.257	0.200	0.350	0.415
uWav...aryZ	0.383	0.447	0.410	0.548	0.320	0.007	0.057	0.237	0.245	0.011	0.113	0.329	0.408	0.292	0.405	0.540
uWav...yAll	0.537	0.668	0.583	0.800	0.673	0.656	0.304	0.643	0.451	0.699	0.198	0.719	0.630	0.758	0.582	0.755
Wafer	0.000	0.000	0.225	0.143	0.263	0.006	0.268	0.143	0.542	0.042	0.214	0.161	0.000	0.000	0.257	0.123
Wine	0.001	0.091	0.019	0.034	0.193	0.091	0.016	0.009	0.01	0.091	0.004	0.071	0.344	0.091	0.001	0.023
Wor...yms	0.326	0.345	0.176	0.356	0.395	0.514	0.158	0.337	0.422	0.037	0.228	0.400	0.335	0.344	0.221	0.368
Worms	0.003	0.199	0.190	0.116	0.209	0.031	0.141	0.136	0.177	0.270	0.149	0.166	0.018	0.252	0.144	0.095
Wor...lass	0.000	0.061	0.092	0.004	0.086	0.037	0.000	0.062	0.075	0.033	0.000	0.076	0.001	0.080	0.032	0.009
Yoga	0.008	0.014	0.056	0.041	0.020	0.000	0.054	0.035	0.022	0.006	0.066	0.051	0.008	0.010	0.054	0.040

TABLE B.3 – Résultats complets pour 16 méthodes et 79 bases de données (3/3)

Bibliographie

[1] "Discours de Madame Roselyne Bachelot-Narquin, Ministre de la santé, de la jeunesse, des sports et de la vie associative". In : *Bulletin de l'Académie Nationale de Médecine* 192.5 (mai 2008), p. 853-860. DOI : 10.1016/S0001-4079(19)32748-7.

[2] Farid KADRI. "Contribution à la conception d'un système d'aide à la décision pour la gestion de situations de tension au sein des systèmes hospitaliers. Application à un service d'urgence." Thèse de doct. Université de Valenciennes et de Hainaut Cambrésis, 2014. URL : https://hal.inria.fr/tel-01131642/.

[3] M. S. AINI et A. FAKHRUL-RAZI. "Development of socio-technical disaster model". In : *Safety Science* 48.10 (déc. 2010), p. 1286-1295. DOI : 10.1016/j.ssci.2010.04.007.

[4] Dena L. SCHANZER et Brian SCHWARTZ. "Impact of Seasonal and Pandemic Influenza on Emergency Department Visits, 2003–2010, Ontario, Canada". In : *Academic Emergency Medicine* 20.4 (2013), p. 388-397. DOI : 10.1111/acem.12111.

[5] Robert W. DERLET et John R. RICHARDS. "Overcrowding in the nation's emergency departments : Complex causes and disturbing effects". In : *Annals of Emergency Medicine* 35.1 (jan. 2000), p. 63-68. DOI : 10.1016/S0196-0644(00)70105-3.

[6] Florence HY YAP et al. "Excess hospital admissions for pneumonia, chronic obstructive pulmonary disease, and heart failure during influenza seasons in Hong Kong". In : *Journal of medical virology* 73.4 (2004), p. 617-623. DOI : 10.1002/jmv.20135.

[7] Spencer S. JONES et al. "An Independent Evaluation of Four Quantitative Emergency Department Crowding Scales". In : *Academic Emergency Medicine* 13.11 (2006), p. 1204-1211. DOI : 10.1197/j.aem.2006.05.021.

[8] Dena L. SCHANZER, Joanne M. LANGLEY et Theresa W. S. TAM. "Role of influenza and other respiratory viruses in admissions of adults to Canadian hospitals". In : *Influenza and Other Respiratory Viruses* 2.1 (2008), p. 1-8. DOI : 10.1111/j.1750-2659.2008.00035.x.

[9] Guillaume BOULEUX, Eric MARCON et Olivier MORY. "Early Index for Detection of Pediatric Emergency Department Crowding". In : *IEEE Journal of Biomedical and Health Informatics* 19.6 (nov. 2015), p. 1929-1936. DOI : 10.1109/JBHI.2014.2350996.

[10] S Ben OTHMAN et S. HAMMADI. "Driven Workflow Orchestration of Patient pathway in Hospital Emergencies". In : *META 2016*. Marrakech, Morocco, oct. 2016.

[11] Nathan R. HOOT et al. "Forecasting Emergency Department Crowding : A Discrete Event Simulation". In : *Annals of Emergency Medicine* 52.2 (août 2008), p. 116-125. DOI : 10.1016/j.annemergmed.2007.12.011.

[12] Marek Laskowski et al. “Models of emergency departments for reducing patient waiting times”. In : *PloS one* 4.7 (2009), e6127. doi : 10.1371/journal.pone.0006127.

[13] Krishan L. Khatri et Lakshman S. Tamil. “Early Detection of Peak Demand Days of Chronic Respiratory Diseases Emergency Department Visits Using Artificial Neural Networks”. In : *IEEE Journal of Biomedical and Health Informatics* 22.1 (jan. 2018), p. 285-290. doi : 10.1109/JBHI.2017.2698418.

[14] Maël Dugast. “Géométrie et topologie des processus périodiquement corrélés induit par la dilation : Application à l’étude de la variabilité des épidémies pédiatriques saisonnières.” Thèse de doct. INSA Lyon, p. 160. url : https://hal.archives-ouvertes.fr/DISP/tel-02124588v1.

[15] Charles Truong, Laurent Oudre et Nicolas Vayatis. “A review of change point detection methods.” In : (jan. 2018). url : https://openreview.net/forum?id=4_eVexSJ8B.

[16] “Hospitaliser les patients âgés en chambre individuelle : un rempart contre la COVID-19 et les infections nosocomiales”. In : *Bulletin de l’Académie Nationale de Médecine* 204.7 (juill. 2020), p. 735-736. doi : 10.1016/j.banm.2020.06.001.

[17] Dr Christine Lawrence. *Faut-il isoler tous les patients ayant une infection à virus influenza ou virus respiratoire syncytial ?* 5eme journées du GREPI. 2017.

[18] Don Wunsch et Rui Xu. *Clustering*. John Wiley & Sons, nov. 2008. isbn : 978-0-470-38278-3.

[19] Adel Mellit, A Massi Pavan et Mohamed Benghanem. “Least squares support vector machine for short-term prediction of meteorological time series”. In : *Theoretical and applied climatology* 111.1 (2013), p. 297-307. doi : 10.1007/s00704-012-0661-7.

[20] Hossein Hassani, Saeed Heravi et Anatoly Zhigljavsky. “Forecasting UK industrial production with multivariate singular spectrum analysis”. In : *Journal of Forecasting* 32.5 (2013), p. 395-408. doi : 10.1002/for.2244.

[21] Maël Dugast et al. “Improving Health Care Management Through Persistent Homology of Time-Varying Variability of Emergency Department Patient Flow”. In : *IEEE Journal of Biomedical and Health Informatics* 23.5 (sept. 2019), p. 2174-2181. doi : 10.1109/JBHI.2018.2882748.

[22] Peter J. Rousseeuw. “Silhouettes : A graphical aid to the interpretation and validation of cluster analysis”. In : *Journal of Computational and Applied Mathematics* 20 (nov. 1987), p. 53-65. doi : 10.1016/0377-0427(87)90125-7.

[23] Andrew Rosenberg et Julia Hirschberg. “V-Measure : A Conditional Entropy-Based External Cluster Evaluation Measure”. In : *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*. Prague, Czech Republic : Association for Computational Linguistics, juin 2007, p. 410-420. url : https://aclanthology.org/D07-1043.

[24] Lawrence HUBERT et Phipps ARABIE. "Comparing partitions". In : *Journal of Classification* 2.1 (déc. 1985), p. 193-218. DOI : 10.1007/BF01908075.

[25] Christos FALOUTSOS, Mudumbai RANGANATHAN et Yannis MANOLOPOULOS. "Fast subsequence matching in time-series databases". In : *ACM Sigmod Record* 23.2 (1994), p. 419-429. DOI : 10.1145/191843.191925.

[26] Johnpaul C et al. "Trendlets : A novel probabilistic representational structures for clustering the time series data". In : *Expert Systems with Applications* 145 (mai 2020), p. 113119. DOI : 10.1016/j.eswa.2019.113119.

[27] Thanawin RAKTHANMANON et al. "Searching and mining trillions of time series subsequences under dynamic time warping". In : *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2012, p. 262-270.

[28] Talal Bin AMIN et Iftekhar MAHMOOD. "Speech Recognition using Dynamic Time Warping". In : *2008 2nd International Conference on Advances in Space Technologies*. Nov. 2008, p. 74-79. DOI : 10.1109/ICAST.2008.4747690.

[29] Charu C. AGGARWAL, Alexander HINNEBURG et Daniel A. KEIM. "On the Surprising Behavior of Distance Metrics in High Dimensional Space". In : *Database Theory — ICDT 2001*. Sous la dir. de Gerhard GOOS et al. T. 1973. Series Title : Lecture Notes in Computer Science. Berlin, Heidelberg : Springer Berlin Heidelberg, 2001, p. 420-434. ISBN : 978-3-540-41456-8 978-3-540-44503-6. DOI : 10.1007/3-540-44503-X_27.

[30] Carla S. MÖLLER-LEVET et al. "Fuzzy Clustering of Short Time-Series and Unevenly Distributed Sampling Points". In : *Advances in Intelligent Data Analysis V*. Sous la dir. de Michael R. BERTHOLD et al. Berlin, Heidelberg : Springer Berlin Heidelberg, 2003, p. 330-340. ISBN : 978-3-540-45231-7.

[31] Xavier GOLAY et al. "A new correlation-based fuzzy logic clustering algorithm for FMRI". In : *Magnetic resonance in medicine* 40.2 (1998), p. 249-260. DOI : 10.1002/mrm.1910400211.

[32] Anil K. JAIN. "Data clustering : 50 years beyond K-means". In : *Pattern Recognition Letters*. Award winning papers from the 19th International Conference on Pattern Recognition (ICPR) 31.8 (juin 2010), p. 651-666. DOI : 10.1016/j.patrec.2009.09.011.

[33] Leonard KAUFMAN et Peter J. ROUSSEEUW. *Finding Groups in Data : An Introduction to Cluster Analysis*. John Wiley & Sons, sept. 2009. ISBN : 978-0-470-31748-8.

[34] James C. BEZDEK. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Springer Science & Business Media, mars 2013. ISBN : 978-1-4757-0450-1.

[35] Anna GROSSWENDT, Heiko RÖGLIN et Melanie SCHMIDT. "Analysis of Ward's method". In : *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2019, p. 2939-2957.

[36] T. Warren Liao. “Clustering of time series data—a survey”. In : *Pattern Recognition* 38.11 (nov. 2005), p. 1857-1874. DOI : 10.1016/j.patcog.2005.01.025.

[37] Hans-Peter Kriegel et al. “Density-based clustering”. In : *WIREs Data Mining and Knowledge Discovery* 1.3 (2011), p. 231-240. DOI : 10.1002/widm.30.

[38] Erich Schubert et al. “DBSCAN revisited, revisited : why and how you should (still) use DBSCAN”. In : *ACM Transactions on Database Systems (TODS)* 42.3 (2017), p. 1-21. DOI : 10.1145/3068335.

[39] Leland McInnes, John Healy et Steve Astels. “HDBSCAN : Hierarchical density based clustering”. In : *Journal of Open Source Software* 2.11 (2017), p. 205.

[40] David L. Donoho. “High-dimensional data analysis : The curses and blessings of dimensionality”. In : *Ams Conference on Math Challenges of the 21st Century*. 2000.

[41] Ian Jolliffe. “Principal component analysis”. In : *Encyclopedia of statistics in behavioral science* (2005). DOI : 10.1002/0470013192.bsa501.

[42] Michael E Wall, Andreas Rechtsteiner et Luis M Rocha. “Singular value decomposition and principal component analysis”. In : *A practical approach to microarray data analysis*. Springer, 2003, p. 91-109.

[43] Peter J Schmid. “Dynamic mode decomposition of numerical and experimental data”. In : *Journal of fluid mechanics* 656 (2010), p. 5-28. DOI : 10.1017/S0022112010001217.

[44] Bernhard Schölkopf, Alexander Smola et Klaus-Robert Müller. “Kernel principal component analysis”. In : *International conference on artificial neural networks*. Springer. 1997, p. 583-588.

[45] Michael AA Cox et Trevor F Cox. “Multidimensional scaling”. In : *Handbook of data visualization*. Springer, 2008, p. 315-347.

[46] Leland McInnes, John Healy et James Melville. “UMAP : Uniform Manifold Approximation and Projection for Dimension Reduction”. In : *arXiv :1802.03426 [cs, stat]* (sept. 2020). URL : http://arxiv.org/abs/1802.03426.

[47] Mikhail Belkin et Partha Niyogi. “Laplacian eigenmaps for dimensionality reduction and data representation”. In : *Neural computation* 15.6 (2003), p. 1373-1396. DOI : 10.1162/089976603321780317.

[48] Etienne Becht et al. “Dimensionality reduction for visualizing single-cell data using UMAP”. In : *Nature biotechnology* 37.1 (2019), p. 38-44. DOI : 10.1038/nbt.4314.

[49] Erich Schubert et Michael Gertz. “Intrinsic t-Stochastic Neighbor Embedding for Visualization and Outlier Detection”. In : *Similarity Search and Applications*. Sous la dir. de Christian Beecks et al. Lecture Notes in Computer Science. Cham : Springer International Publishing, 2017, p. 188-203. ISBN : 978-3-319-68474-1. DOI : 10.1007/978-3-319-68474-1_13.

[50] Leonid Sigal et Michael J Black. "Humaneva : Synchronized video and motion capture dataset for evaluation of articulated human motion". In : *Brown Univertsity TR* 120.2 (2006).

[51] Rui Li, Tai-Peng Tian et Stan Sclaroff. "Simultaneous learning of nonlinear manifold and dynamical models for high-dimensional time series". In : *2007 IEEE 11th International Conference on Computer Vision*. IEEE. 2007, p. 1-8.

[52] P. Turaga et al. "Statistical Computations on Grassmann and Stiefel Manifolds for Image and Video-Based Recognition". In : *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33.11 (nov. 2011), p. 2273-2286. doi : 10.1109/TPAMI.2011.52.

[53] Luke Melas-Kyriazi. "The Mathematical Foundations of Manifold Learning". In : *arXiv preprint arXiv :2011.01307* (2020).

[54] Ali Javed, Byung Suk Lee et Donna M Rizzo. "A benchmark study on time series clustering". In : *Machine Learning with Applications* 1 (2020), p. 100001. doi : 10.1016/j.mlwa.2020.100001.

[55] Norman H Packard et al. "Geometry from a time series". In : *Physical review letters* 45.9 (1980), p. 712. doi : 10.1103/PhysRevLett.45.712.

[56] Tim Sauer, James A Yorke et Martin Casdagli. "Embedology". In : *Journal of statistical Physics* 65.3 (1991), p. 579-616. doi : 10.1007/BF01053745.

[57] Floris Takens. "Detecting strange attractors in turbulence". In : *Dynamical Systems and Turbulence, Warwick 1980*. Sous la dir. de David Rand et Lai-Sang Young. Lecture Notes in Mathematics. Berlin, Heidelberg : Springer, 1981, p. 366-381. isbn : 978-3-540-38945-3. doi : 10.1007/BFb0091924.

[58] Kathleen T Alligood, Tim D Sauer et James A Yorke. *Chaos.* Springer, 1996.

[59] Steven H Strogatz. *Nonlinear dynamics and chaos with student solutions manual : With applications to physics, biology, chemistry, and engineering*. CRC press, 2018.

[60] Jaroslav Stark et al. "Takens embedding theorems for forced and stochastic systems". In : *Nonlinear Analysis : Theory, Methods & Applications* 30.8 (1997), p. 5303-5314. doi : 10.1016/S0362-546X(96)00149-6.

[61] Cosma Rohilla Shalizi. "Methods and techniques of complex systems science : An overview". In : *Complex systems science in biomedicine* (2006), p. 33-114. doi : 10.1007/978-0-387-33532-2_2.

[62] Cosma Rohilla Shalizi. "Methods and Techniques of Complex Systems Science : An Overview". In : *Complex Systems Science in Biomedicine*. Sous la dir. de Thomas S. Deisboeck et J. Yasha Kresh. Topics in Biomedical Engineering International Book Series. Boston, MA : Springer US, 2006, p. 33-114. doi : 10.1007/978-0-387-33532-2_2.

[63] Colin O'Reilly, Klaus Moessner et Michele Nati. "Univariate and multivariate time series manifold learning". In : *Knowledge-Based Systems* 133 (2017), p. 1-16. doi : 10.1016/j.knosys.2017.05.026.

[64] Zhongke Gao et Ningde Jin. "Complex network from time series based on phase space reconstruction". In : *Chaos : An Interdisciplinary Journal of Nonlinear Science* 19.3 (2009), p. 033137. doi : 10.1063/1.3227736.

[65] Clément Pealat, Guillaume Bouleux et Vincent Cheutet. "Extracting Most Impacting Emergency Department Patient Flow By Embedding Laboratory-confirmed and Clinical Diagnosis on The Stiefel Manifold". In : *2019 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI)*. IEEE. 2019, p. 1-4.

[66] Kyle A Gallivan et al. "Efficient algorithms for inferences on grassmann manifolds". In : *IEEE Workshop on Statistical Signal Processing, 2003.* IEEE. 2003, p. 315-318.

[67] Inam Ur Rahman et al. "Multiscale representations for manifold-valued data". In : *Multiscale Modeling & Simulation* 4.4 (2005), p. 1201-1232. doi : 10.1137/050622729.

[68] Xikang Zhang et al. "Efficient Temporal Sequence Comparison and Classification Using Gram Matrix Embeddings on a Riemannian Manifold". In : *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA : IEEE, juin 2016, p. 4498-4507. isbn : 978-1-4673-8851-1. doi : 10.1109/CVPR.2016.487.

[69] Brian St Thomas et al. "Learning subspaces of different dimension". In : *arXiv preprint arXiv :1404.6841* (2014).

[70] Salem Said et al. "Exact principal geodesic analysis for data on SO (3)". In : *2007 15th European Signal Processing Conference*. IEEE. 2007, p. 1701-1705.

[71] Paolo Zanini et al. "Transfer learning : A Riemannian geometry framework with applications to brain–computer interfaces". In : *IEEE Transactions on Biomedical Engineering* 65.5 (2017), p. 1107-1116. doi : 10.1109/TBME.2017.2742541.

[72] John M. Lee et John Michael Lee. *Introduction to Smooth Manifolds*. Springer Science & Business Media, 2003. isbn : 978-0-387-95448-6.

[73] H. Karcher. "Riemannian center of mass and mollifier smoothing". In : *Communications on Pure and Applied Mathematics* 30.5 (1977), p. 509-541. doi : 10.1002/cpa.3160300502.

[74] Sigmundur Gudmundsson. *An Introduction to Riemannian Geometry*. Lund University, mai 2021.

[75] Jianming Miao et Adi Ben-Israel. "On principal angles between subspaces in Rn". In : *Linear Algebra and its Applications* 171 (juill. 1992), p. 81-98. doi : 10.1016/0024-3795(92)90251-5.

[76] T Bendokat, R Zimmermann et P-A Absil. “A Grassmann Manifold Handbook : Basic Geometry and Computational Aspects”. In : (), p. 35.

[77] Raviteja Vemulapalli et David W. Jacobs. “Riemannian Metric Learning for Symmetric Positive Definite Matrices”. In : *arXiv :1501.02393 [cs]* (jan. 2015). URL : http://arxiv.org/abs/1501.02393.

[78] Clément Pealat, Guillaume Bouleux et Vincent Cheutet. “Improved Time-Series Clustering with UMAP dimension reduction method”. In : *2020 25th International Conference on Pattern Recognition (ICPR)*. 2021, p. 5658-5665. DOI : 10.1109/ICPR48806.2021.9412261.

[79] Clément Péalat, Guillaume Bouleux et Vincent Cheutet. “Improved Time Series Clustering Based on New Geometric Frameworks”. In : *Pattern Recognition* (2021), p. 108423. DOI : 10.1016/j.patcog.2021.108423.

[80] Matthew B Kennel, Reggie Brown et Henry DI Abarbanel. “Determining embedding dimension for phase-space reconstruction using a geometrical construction”. In : *Physical review A* 45.6 (1992), p. 3403. DOI : 10.1103/PhysRevA.45.3403.

[81] Carl Rhodes et Manfred Morari. “The false nearest neighbors algorithm : An overview”. In : *Computers & Chemical Engineering* 21 (1997), S1149-S1154. DOI : 10.1016/S0098-1354(97)87657-0.

[82] Ai-Hua Jiang et al. “Mutual information algorithms”. In : *Mechanical Systems and Signal Processing* 24.8 (2010), p. 2947-2960. DOI : 10.1016/j.ymssp.2010.05.015.

[83] Solyman Ashrafi et al. “Solar flux forecasting using mutual information with an optimal delay”. In : *NASA STI/Recon Technical Report A* 95 (1993), p. 901-913.

[84] Solomon Kullback. *Information theory and statistics*. Courier Corporation, 1997.

[85] Andrew M Fraser et Harry L Swinney. “Independent coordinates for strange attractors from mutual information”. In : *Physical review A* 33.2 (1986), p. 1134. DOI : 10.1103/PhysRevA.33.1134.

[86] William McGill. “Multivariate information transmission”. In : *Transactions of the IRE Professional Group on Information Theory* 4.4 (1954), p. 93-111.

[87] Sunil Srinivasa. “A review on multivariate mutual information”. In : *Univ. of Notre Dame, Notre Dame, Indiana* 2.1 (2005).

[88] GR Cross et AK Jain. “Measurement of clustering tendency”. In : *Theory and Application of Digital Control*. Elsevier, 1982, p. 315-320.

[89] Amit Banerjee et Rajesh N Dave. “Validating clusters using the Hopkins statistic”. In : *2004 IEEE International conference on fuzzy systems (IEEE Cat. No. 04CH37542)*. T. 1. IEEE. 2004, p. 149-153.

[90] Yasuko Chikuse. *Statistics on special manifolds*. T. 174. Springer Science & Business Media, 2003.

[91] Seth M Hirsh et al. "Structured Time-Delay Models for Dynamical Systems with Connections to Frenet-Serret Frame". In : *arXiv preprint arXiv :2101.08344* (2021).

Résumé

Chaque année, lors de la période hivernale, les hôpitaux sont profondément impactés par l'arrivée des virus hivernaux. Ces virus hivernaux, la grippe et le VRS sont difficiles à anticiper. En effet, ces phénomènes épidémiques ne sont pas parfaitement périodiques et ont un impact principalement sur le temps de séjour des patients plutôt que sur le nombre d'arrivée. Il n'est donc pas possible d'anticiper ces épidémies en analysant directement le nombre de patients arrivant au service des urgences par jour. A posteriori, pour avoir une image de l'épidémie, des tests PCR sont réalisés sur les patients de l'hôpital. De plus, un patient arrivant aux urgences est immédiatement classé selon ses symptômes. On propose alors de réunir les tests PCR positifs et le nombre d'arrivée par symptôme via du clustering de série temporelle. Cela met en lumière les symptômes liés aux virus. Ainsi, pour analyser l'arrivée probable d'une épidémie, on peut utiliser le nombre d'arrivée pour les symptômes marqueurs des virus plutôt que le nombre d'arrivée totale au service des urgences. Pour réaliser ce clustering, nous proposons une méthode innovante reposant sur une représentation géométrique des séries temporelles. En particulier, on met en lumière l'efficacité d'utiliser la géométrie de Riemman appliquée à la variété de la Grassmann (via une représentation sur la variété de la Stiefel) pour analyser les séries temporelles.

English version : Every year, during the winter period, hospitals are deeply impacted by the arrival of winter viruses. These winter viruses, influenza and RSV, are difficult to anticipate. Indeed, these epidemic phenomena are not perfectly periodic and have an impact mainly on the length of stay of patients rather than on the number of arrivals. It is therefore not possible to anticipate these epidemics by directly analyzing the number of patients arriving in the emergency department per day. A posteriori, in order to have an image of the epidemic, PCR tests are carried out on the hospital's patients. In addition, a patient arriving at the emergency department is immediately classified according to his symptoms. We then propose to gather the positive PCR tests and the number of arrivals per symptom via time series clustering. This highlights the symptoms related to viruses. Thus, to anticipate an arrival of an epidemic in a near future, we can use the number of arrivals for the virus marker symptoms rather than the total number of arrivals at the emergency department.

To achieve this clustering, we propose an innovative method based on a geometric representation of time series. In particular, we highlight the efficiency of using the Riemmannian geometry applied to the Grassmann manifold (via a representation on the Stiefel manifold) to analyze time series.

FOLIO ADMINISTRATIF

THESE DE L'UNIVERSITE DE LYON OPEREE AU SEIN DE L'INSA LYON

NOM : PEALAT DATE de SOUTENANCE : 22/03/2022

Prénoms : Clément

TITRE : Modélisation du flux de patients aux urgences liés aux maladies respiratoires par analyse géométrique de séries temporelles

NATURE : Doctorat Numéro d'ordre : 2022LYSEI017

Ecole doctorale : InfoMaths ED512

Spécialité : Mathématiques et applications

RESUME : Chaque année, lors de la période hivernale, les hôpitaux sont profondément impactés par l'arrivée des virus hivernaux. Ces virus hivernaux, la grippe et le VRS sont difficiles à anticiper. En effet, ces phénomènes épidémiques ne sont pas parfaitement périodiques et ont un impact principalement sur le temps de séjour des patients plutôt que sur le nombre d'arrivée. Il n'est donc pas possible d'anticiper ces épidémies en analysant directement le nombre de patients arrivant au service des urgences par jour. A posteriori, pour avoir une image de l'épidémie, des tests PCR sont réalisés sur les patients de l'hôpital. De plus, un patient arrivant aux urgences est immédiatement classé selon ses symptômes. On propose alors de réunir les tests PCR positifs et le nombre d'arrivée par symptôme via du clustering de série temporelle. Cela met en lumière les symptômes liés aux virus. Ainsi, pour analyser l'arrivée probable d'une épidémie, on peut utiliser le nombre d'arrivée pour les symptômes marqueurs des virus plutôt que le nombre d'arrivée totale au service des urgences. Pour réaliser ce clustering, nous proposons une méthode innovante reposant sur une représentation géométrique des séries temporelles. En particulier, on met en lumière l'efficacité d'utiliser la géométrie de Riemman appliquée à la variété de la Grassmann (via une représentation sur la variété de la Stiefel) pour analyser les séries temporelles.

MOTS-CLÉS : Clustering, Séries temporelles, Delay Coordinate Embedding, Variété, Géométrie de Riemann

Laboratoire (s) de recherche : Laboratoire DISP

Directeur de thèse: Vincent CHEUTET, Guillaume BOULEUX

Président de jury :

Composition du jury : Absil, Pierre-Antoine, Professeur, UC Louvain, Rapporteur
Le Bihan, Nicolas, Directeur de Recherche, GIPSA Lab Grenoble INP, Rapporteur
Berthoumieu, Yannick, Professeur des universités, IMS
Poisson Caillault, Émilie, Maitre de Conférence HDR, Université du littoral Côte d'Opale
Cheutet, Vincent, Professeur des universités, DISP INSA-LYON, Directeur de thèse
Bouleux, Guillaume, Maître de Conférence, Université Jean Monnet, Co-directeur de thèse