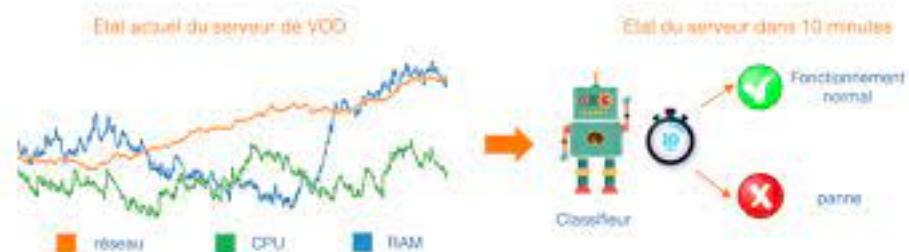


Classification de série temporelle

Projet Fin d'étude

15 mars 2024



Étudiant :

Hengshuo LI hengshuo.li@insa-rouen.fr

Tuteur :

Bruno PORTIER

Mot-clé : Classification non supervisé , Série temporelle

Table des matières

Remerciements	1
Introduction	2
I Partie Théorique	6
1 Méthodes de représentation	6
1.1 Introduction	6
1.1.1 Adaptative des données	6
1.1.2 Non Adaptatives	7
1.1.3 Modèle	8
1.1.4 dicté	8
1.2 R package	8
1.3 Les Méthodes dans R	8
1.4 Nos principales méthodes de recherche	8
1.4.1 DWT	9
1.4.2 SAX	12
1.4.3 Données dictées	13
1.4.4 Réprésentation basé sur modèle	14
2 Mesures de similarité/dissimilité	15
2.0.1 Recherche de séries temporelles similaires dans le temps	15
2.0.2 Recherche de séries temporelles similaires en termes de forme	15
2.0.3 Recherche de séries temporelles similaires dans le changement (similarité structurelle)	16
2.1 Considérations sur la sélection de la mesure de dissimilité	16
2.2 Paquet R	16
2.3 Les Méthode dans R	17

2.4 Résumé des mesures de distance pour les séries temporelles implémentées dans R.	17
2.4.1 DTW	17
2.4.2 GAK	25
2.4.3 DTW Doux	29
2.4.4 PDC	32
2.4.5 D'autre distance	34
3 Prototypes des clusters	35
3.1 Nos principales méthodes de recherche	35
3.1.1 DBA	35
3.1.2 K-shape	36
4 Algorithme	41
4.1 Regroupement hiérarchique de séries temporelles	41
4.2 Partitionnement en groupe	41
4.3 Regroupement basé sur un modèle	41
4.4 Regroupement basé sur la densité	41
4.5 Regroupement basé sur une grille	41
4.6 Regroupement en plusieurs étapes	41
4.7 Nos principales méthodes de recherche	41
4.7.1 TADpole	41
5 Évaluation des clusters	48
5.1 Les indices internes	48
5.1.1 Notation	48
5.1.2 L'indice de Dunn	48
5.1.3 L'indice de Calinski–Harabasz	49
5.1.4 L'indice de Davies–Bouldin	49
5.1.5 Indice de Davies–Bouldin*	49
5.1.6 L'indice de Silhouette	49

5.1.7	L'indice de Fonction du score	50
5.1.8	L'indice de COP	50
5.2	Les indices externes	50
5.2.1	L'indice de Rand	51
5.2.2	L'indice de Rand ajusté (ARI)	51
5.2.3	L'indice de Jaccard	51
5.2.4	L'indice de Similarité (TSclust)	51
5.2.5	L'indice de Fowlkes-Mallows (FM)	52
5.2.6	La Variation d'Information (VI)	52
II	Partie Pratique	53
6	R instruction	53
6.1	Contexte	53
6.2	Préparer les données et Visualisation des données	53
6.2.1	synthetic.tseries [Pablo Montero, 2014]	55
6.2.2	Shape complexities [Brandmaier, 2015]	55
6.2.3	paired.tseries [Pablo Montero, 2014]	57
6.2.4	Consommation d'électricité des appareils ménagers.[Chen et al., 2015]	58
6.2.5	Consommation d'électricité [Laurinec]	62
6.2.6	interest.rates [Pablo Montero, 2014]	63
6.3	Approches basées sur la forme	64
6.4	Approches basées sur les caractéristiques	68
6.5	Approches basées sur un modèle	69
6.6	Evaluation	69
7	Problème Consommation d'électricité	70
8	Problème Les signatures de forme des objets	74
9	Problème 18 paires de séries temporelles couvrant différents domaines	83

10 une collection de séries temporelles synthétiques	85
11 les taux d'intérêt mensuels à long terme	86
Conclusion	88

Remerciements

Je tiens à remercier mon tuteur, M. Bruno Portier, pour ses conseils avisés et son soutien. Il a toujours été extrêmement généreux de son temps à chaque étape de mon projet. Il a écouté patiemment mon français approximatif à chaque rendez-vous. Je le remercie d'avoir examiné mon rapport et de m'avoir fait part de ses commentaires et suggestions utiles.

Je tiens également à remercier INSA-Rouen Normandie et l'Université de Rouen Normandie pour leur accès aux principales revues scientifiques. Et merci au département du génie mathématique de l'INSA-ROUEN pour avoir mis à ma disposition un ordinateur pour travailler.

Merci à mes amis (Yuchen MO) (Liuxin Yang) de m'avoir écouté me plaindre de la charge de travail, et pour leurs encouragements, leur soutien, leur réconfort et leur compréhension.

Introduction

Avant de débuter notre étude, nous définissons les concepts clés. Une série temporelle, de dimension mm et composée de valeurs réelles, peut être représentée comme $T = \{T_i\}_1^m \in \mathbf{R}^m$.

Nous considérons un ensemble D composé de n séries temporelles, $D = \{T_1, T_2, \dots, T_n\}$. Le but est de partitionner D de façon non supervisée en k clusters $C = C_1, C_2, \dots, C_k$, de manière à regrouper les séries temporelles similaires selon une mesure de similarité définie. Un cluster Ci est défini de sorte que $D = \bigcup_{i=1}^k C_i$ et $C_i \cap C_j = \emptyset \quad \forall i \neq j$.

L'objectif est de répartir ces n séries temporelles en k clusters distincts, en minimisant la distance intra-cluster et en maximisant la distance inter-cluster, pour aboutir à une séparation nette entre les groupes.

La présente étude se concentre sur le "Classification non supervisé de séries temporelles". Nous commençons par classifier les problématiques liées au clustering de séries temporelles, en considérant divers critères tels que leur nature univariée ou multivariée, l'homogénéité de leur longueur au sein de l'ensemble, etc. Cette classification, bien qu'elle prenne en compte les propriétés de la série temporelle elle-même, ne tient pas compte des difficultés rencontrées dans le processus de regroupement. Par exemple, nous examinons l'impact du timing d'événements spécifiques dans une série ou comment la similarité entre séries est définie. Cela nous mène à explorer la notion d'invariance, c'est-à-dire comment des modifications appliquées à une série temporelle peuvent ne pas affecter la distance perçue avec une autre. Cette approche est cruciale pour comprendre et traiter l'invariance dans nos analyses.

Nous explorons les invariances des séries temporelles, se basant sur les travaux de [G. E. Batista and de Souza, 2013].

Invariance d'Amplitude et de Décalage : La capacité à reconnaître des séquences comme similaires, malgré leurs variations d'amplitude ou de décalage, est essentielle. Cela signifie que transformer une séquence x en $x = ax + b$, où a et b sont des constantes, ne modifie pas sa similarité avec d'autres séquences. Cette invariance est utile, par exemple, pour analyser les variations saisonnières dans les marchés financiers sans être affecté par les changements de valeur des monnaies. La normalisation des données facilite l'obtention de ces invariances.

Invariance de Phase : Cruciale pour l'alignement de séries temporelles périodiques, elle assure que différentes phases d'une fonction sinusoïdale, par exemple, n'altèrent pas la mesure de similarité.

Invariance aux Déformations Locales : Cette invariance permet de faire correspondre des formes similaires même en présence de déphasages locaux dans le temps, illustré par la capacité de l'algorithme DTW à aligner des séries temporelles montrant des comportements d'insectes malgré des déformations, voir la figure 1

Invariance d'Échelle Globale : Elle est nécessaire pour comparer efficacement des séries de différentes longueurs, en ajustant la taille des séquences plus courtes ou plus longues. Cette adaptation est essentielle, par exemple, pour comparer des Électrocardiographie de différents fréquences du pouls sur la durée d'un battement. Nous pouvons garantir cette propriété par une ré-interpolation.

Invariance d'Occlusion : Permet la comparaison de séquences même en l'absence de sous-séquences, en ignorant les parties qui ne correspondent pas bien. Utile, notamment dans l'analyse d'écritures manuscrites imparfaites.

Invariance de Complexité : Reconnaît des séquences comme similaires malgré des différences de complexité, comme des enregistrements audio effectués dans des environnements variés qui peuvent être considérés comparables malgré le bruit de fond extérieur.

L'adaptation aux invariances nécessaires dépend de la tâche à accomplir et du prétrai-

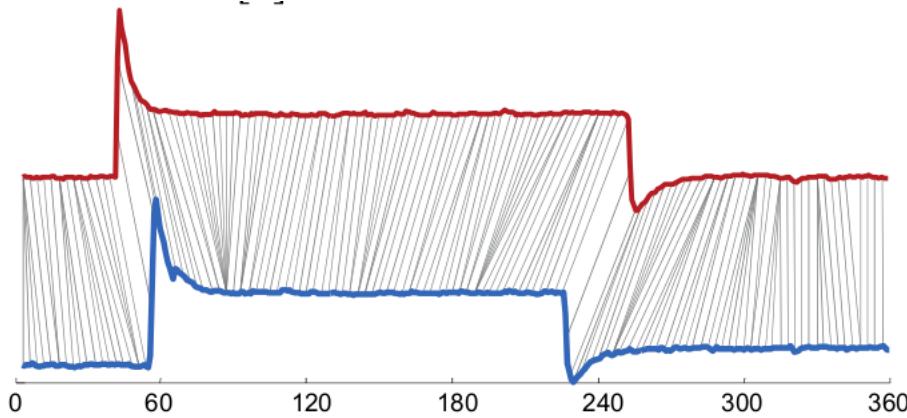


FIGURE 1 – Deux séries temporelles de comportements d'insectes avec invariance à la déformation. L'alignement a été calculé par l'algorithme DTW.

tement des données pour éliminer les distorsions indésirables avant le clustering. La normalisation standard des données peut respecter les invariances d'échelle et de décalage. Pour celles ne pouvant être corrigées simplement par un prétraitement, des mesures de distance sophistiquées, offrant des invariances spécifiques, seront discutées dans les sections suivantes.

Suite à notre examen des différentes catégories de problèmes liés au clustering de séries temporelles, nous concluons en proposant des solutions pour différents problèmes de regroupement.

La littérature [Saeed Aghabozorgi, 2015] révèle une diversité de techniques recommandées pour le clustering de séries temporelles. Ces techniques se regroupent en deux aspects.

- **Adaptation des algorithmes de clustering traditionnels :** Cette stratégie consiste à modifier les algorithmes existants, conçus initialement pour des données statiques, pour qu'ils puissent traiter les spécificités des séries temporelles. Cela implique souvent de changer la façon dont la distance entre les données est calculée pour mieux s'adapter aux caractéristiques temporelles.
- **Transformation des séries temporelles :** Cette approche transforme les séries temporelles en formats simplifiés, via des méthodes de caractérisation ou de modélisation, pour les rendre compatibles avec les algorithmes de clustering standards.

Nous pouvons intégrer quatre approches distinctes (dans le Figure 2) en prenant en compte ces deux aspects. Les trois premières sont Clustering basé sur la forme, les caractéristiques, ou le modèle, il sont mutuellement exclusives, tandis que la quatrième représente une combinaison des deux premières. Nous détaillons les trois premiers.

Les approches basées sur la forme visent à aligner les séries temporelles par ajustements non linéaires de leurs axes temporels, ce qui permet une comparaison directe des données. Cette méthode est parfois appelée approche sur données brutes, car elle manipule directement ces dernières. Les algorithmes de cette catégorie modifient les mesures de distance/similarité traditionnelles pour s'adapter aux spécificités temporelles.

Les méthodes basées sur les caractéristiques transforment les séries temporelles en vecteurs de caractéristiques de dimension réduite, permettant l'application d'algorithmes de clustering standard sur ces vecteurs. Cette transformation génère pour chaque série un vecteur de caractéristiques de longueur fixe, et la distance entre ces vecteurs est souvent mesurée par la distance euclidienne.

Les approches basées sur un modèle convertissent les séries temporelles en un ensemble de paramètres modélisant ces dernières. La similarité entre séries est alors évaluée à travers la

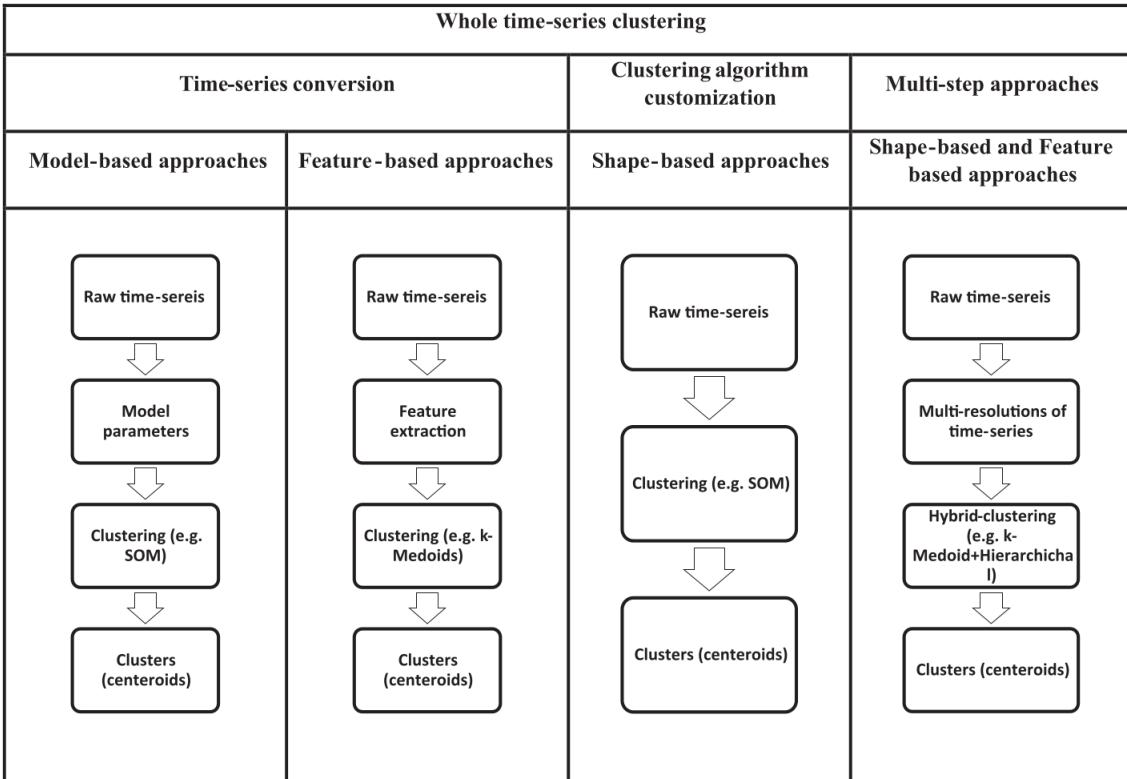


FIGURE 2 – Approches de classification de série temporelle

distance entre leurs modèles paramétriques, et le clustering est réalisé à l'aide d'algorithmes standards adaptés à ces modèles.

Ces approches révèlent que le clustering de séries temporelles se décompose en cinq éléments clés [3], selon la littérature [Saeed Aghabozorgi, 2015] : la représentation ou réduction de dimensionnalité, la mesure de distance, le choix de l'algorithme de clustering, la définition de prototypes, et l'évaluation des résultats.

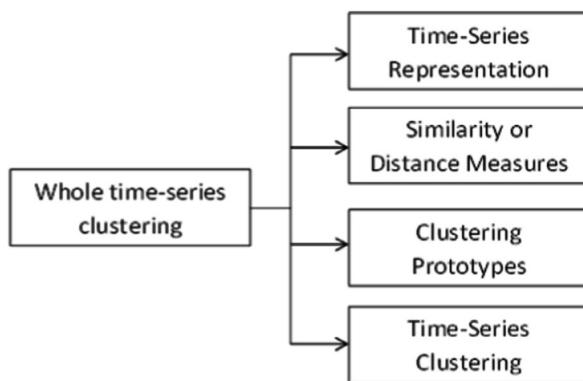


FIGURE 3 – Vue d'ensemble des quatre composantes du regroupement de séries temporelles entières.

Le processus général des méthodes de regroupement des séries temporelles utilise tout ou partie de ces composants en fonction du problème. En général, les données sont approximées à l'aide d'une méthode de représentation de manière à ce qu'elles puissent tenir dans la mémoire. Ensuite, un algorithme de regroupement est appliqué aux données en utilisant

une mesure de distance. Dans le processus de regroupement, un prototype est généralement nécessaire pour résumer les séries temporelles. Enfin, les sont évalués à l'aide de critères.

Dans les sections suivantes, chaque composante est examinée et plusieurs travaux et méthodes connexes sont présentés. En outre, nous nous concentrerons sur la description de l'existence de méthodes mises en œuvre dans les parcelles de R. La raison en est que l'objectif de ce projet est de développer mon expérience dans le domaine du clustering des séries temporelles de manière pratique.

Le processus de clustering des séries temporelles s'articule généralement autour de plusieurs étapes clés, adaptées selon les spécificités du problème à résoudre. Typiquement, les données sont d'abord réduites ou représentées de manière à être gérables en mémoire. Un algorithme de clustering est ensuite appliqué, exploitant une mesure de distance spécifique pour organiser les données en groupes. Au cours de ce processus, un prototype est souvent élaboré pour résumer chaque groupe de séries temporelles. L'évaluation des résultats s'effectue en dernier lieu, via des critères définis.

Les sections suivantes détaillent chacune de ces composantes, présentent diverses approches et travaux pertinents, et mettent en lumière les méthodes implémentées dans l'environnement R. Cette focalisation sur R est motivée par le but de ce projet : enrichir mon expérience pratique dans le clustering de séries temporelles.

Première partie

Partie Théorique

1 Méthodes de représentation

1.1 Introduction

Le premier élément crucial dans le clustering de séries temporelles est la réduction de dimension, une étape prédominante dans de nombreuses méthodologies de clustering de séries complètes, comme souligné dans divers travaux [J. Lin and Chiu, 2003] [Keogh, 2005] [E. Ghysels, 2006] [G. Duan, 2006]. Cette section examine les techniques de réduction de dimension, ou représentation, des séries temporelles. L'idée est de transformer les données brutes en un espace de moindre dimension, soit par transformation, soit par extraction de caractéristiques. La réduction de dimension est essentielle pour plusieurs raisons :

- **Économie de mémoire** : Les données brutes de série temporelle peuvent être volumineuses et ne pas tenir entièrement en mémoire.
- **Efficacité de calcul** : Calculer des distances entre données brutes est coûteux en termes de temps de calcul. Réduire la dimension des données peut considérablement accélérer le processus de clustering.
- **Pertinence des mesures de distance** : Mesurer la distance entre séries temporelles brutes peut conduire à des résultats contre-intuitifs, en raison de la sensibilité de certaines mesures à des distorsions spécifiques dans les données. Ainsi, sans réduction de dimension, les séries pourraient être regroupées sur la base de similitudes de bruit plutôt que de forme.

Le choix de la méthode de réduction de dimension et de son application est délicat, car il implique un compromis entre vitesse et précision. Il est crucial de trouver un équilibre optimal entre la qualité du clustering et le temps d'exécution.

La classification des méthodes de représentation des séries temporelles se divise en quatre catégories principales selon [Saeed Aghabozorgi, 2015] : adaptatives, non adaptatives, basées sur des modèles et dictées par les données. Les techniques clés sont résumées dans le tableau 1.

1.1.1 Adaptative des données

Les méthodes de représentation adaptative segmentent chaque série temporelle en sous-séquences de longueurs variables, visant à minimiser l'erreur de reconstruction globale. Chaque segment est analysé pour sélectionner des caractéristiques spécifiques, qui sont ensuite assemblées en un vecteur de caractéristiques représentant la série temporelle. Bien que les sous-séquences varient en longueur au sein de chaque série, les vecteurs de caractéristiques résultants ont une longueur uniforme pour toutes les séries. Des techniques telles que l'interpolation polynomiale par morceaux (PPI)[Y. Morinaka, 2001], la régression polynomiale par morceaux (PPR)[Zdonik, 1996], l'approximation linéaire par morceaux (PLA), l'approximation constante par morceaux (PCA), l'approximation adaptative par morceaux (APCA)[E. Keogh, 2001], la décomposition en valeurs singulières (SVD)[C. Faloutsos, 1994],[F. Korn, 1997], le langage naturel symbolique (NLG)[F. Portet, 2009], l'approximation d'agrégats symboliques (SAX) et iSAX[J. Lin, 2007] illustrent cette approche. Ces représentations adaptatives offrent une approximation précise des séries individuelles,

Representation Method	Year	Type	Complexity
Auto-regressive (AR) model	1971	T4	
Discrete Fourier Transform (DFT)	1993	T1	$O(n \log(n))$
Approximation linéaire par morceaux indexable (LLAI)	1994	T1	
Discrete Wavelet Transform (DWT)	1999	T1	$O(n)$
Singular Value Decomposition (SVD)	1997	T2	
Discrete Cosine Transformation (DCT)	1997	T1	—
Piecewise Linear Approximation (PLA)	1998	T2	$O(n \log(n))$
Hidden Markov models (HMMs)	1998	T4	—
Piecewise Aggregate Approximation (PAA) or Segmented Means	2000	T1	$O(n)$
Piecewise Constant Approximation (PCA)	2000	T2	—
Adaptive Piecewise Constant Approximation (APCA)	2002	T2	$O(n)$
Perceptually important point (PIP)	2001	T1	—
Chebyshev Polynomials (CHEB)	2004	T1	—
Symbolic Aggregate Approximation (SAX)	2003	T2	$O(n)$
HOT SAX	2005	T2	
Clipped Data	2005	T3	
Group SAX	2006	T2	
Extended SAX	2006	T2	
Combining SAX and Piecewise Linear Approximation	2007	T2	
Indexable Piecewise Linear Approximation (IPLA)	2007	T1	
1d-SAX	2013	T2	
Move-Split-Merge (MSM)	2013		
SAX-VSM	2013	T2	
SAX-EFG	2014	T2	
Tree-based Representations	2015		
SC-DTW	2015	T1	
Representation based on Local Autopatterns	2016		
Grid Representation	2019		
SAX Navigator	2019	T2	
SAX-ARM	2020	T2	
SAX-BD	2020	T2	
Data-driven Kernel-based Probabilistic SAX	2021	T2	

TABLE 1 – Principales méthodes de représentation pour les données de séries temporelles (pas indiqué par les auteurs ; n est la longueur des séries temporelles ; T1 non-data adaptive ; T2 data-adaptive ; T3 data dictated ; T4 model-based).

mais peuvent rendre la comparaison entre séries temporelles plus complexe.

1.1.2 Non Adaptatives

Les approches non adaptatives aux données sont des représentations qui conviennent aux séries temporelles avec une segmentation de taille fixe (longueur égale), et la comparaison des représentations de plusieurs séries temporelles est directe. Les méthodes de ce groupe sont les ondelettes : HAAR, DAUBECHIES, Coiflets, Symlets, transformée en ondelettes discrète (DWT), polynômes de Chebyshev spectraux DFT spectrale , mappages aléatoires , l'approximation agrégée par morceaux (PAA) et l'approximation linéaire par morceaux indexable (LLAI) . etc. Les méthodes non adaptatives traitent les séries temporelles par segmentation de taille constante, facilitant la comparaison directe entre séries. Cette catégorie inclut des approches comme les ondelettes[K. Chan, 1999] (HAAR, DAUBECHIES, Coiflets, Symlets, transformée en ondelettes discrète (DWT)), les polynômes de Chebyshev spectraux[Cai and Ng, 2004],, la transformée de Fourier discrète (DFT)[C. Faloutsos, 1994], les mappages aléa-

toires[Bingham, 2001], l'approximation agrégée par morceaux (PAA) [E. Keogh, 2000], et l'approximation linéaire par morceaux indexable (LLAI)[C. Faloutsos, 1994]Indexable Piecewise Linear Approximation (IPLA) ,[Q. Chen, 2007], entre autres. Ces techniques permettent une analyse uniforme et directe des séries temporelles, en rendant la comparaison entre multiples séries plus simple et intuitive.

1.1.3 Modèle

Les approches basées sur des modèles traitent les séries temporelles à travers des cadres stochastiques, tels que les modèles de Markov, les modèles de Markov cachés (HMM)[D. Minnen, 2006],[D. Minnen, 2007] ,[A. Panuccio and Murino, 2002], les analyses statistiques[N. Kumar, 2005], les modèles autorégressifs intégrés à des moyennes mobiles (ARIMA), et les moyennes mobiles autorégressives (ARMA)[M. Corduas, 2008],[K. Kalpakis, 2001]. . Ces techniques offrent une perspective probabiliste pour représenter la dynamique temporelle.

Dans les approches adaptatives aux données, non adaptatives aux données et basées sur un modèle, l'utilisateur peut définir le taux de compression en fonction de l'application concernée.

1.1.4 dicté

En revanche, dans les approches dicté, le taux de compression est défini automatiquement sur la base des séries temporelles brutes, comme dans le cas de Clipped [C. Ratanamahatana, 2005] [A.A.J. Bagnall, 2006].

1.2 R package

Le package TSrepr est conçu pour faciliter la représentation des séries temporelles, englobant la réduction de dimensionnalité, le prétraitement et l'extraction de caractéristiques, afin d'optimiser l'analyse des données temporelles. Il intègre des méthodes de représentation adaptatives, non adaptatives, basées sur des modèles et dictées par les données (comme l'écrêtage), ainsi que des normalisations min-max et z-score, et propose des outils d'évaluation de la précision des prévisions.

Quant au package TSclust, il offre une gamme de mesures de distance adaptées à différentes méthodes de représentation, telles que la distance basée sur la transformée en ondelettes discrètes (dist_dwt) et MINDIST pour SAX, facilitant ainsi le clustering précis et efficace des séries temporelles.

1.3 Les Méthodes dans R

Ces méthodes de représentation des séries temporelles sont implémentées dans le paquet TSrepr (les noms des fonctions sont entre parenthèses) :

1.4 Nos principales méthodes de recherche

Bien que certaines sections pas encore vu aient été mentionnées dans l'introduction de cette méthode principale, leur inclusion vise à assurer une compréhension complète et à faciliter l'intégration de cette approche dans la classification de séries temporelles. Elles servent également à établir des liens cohérents entre les différentes parties du texte. Cette logique s'applique aussi aux autres sections introduisant des méthodes spécifiques.

TABLE 2 – Données non adaptatives :

PAA - Approximation agrégée par morceaux	(repr_paa)
DWT - Transformée en ondelettes discrète	(repr_dwt)
DFT - Transformée de Fourier discrète	(repr_dft)
DCT - Transformée en cosinus discrète	(repr_dct)
SMA - Moyenne mobile simple	(repr_sma)
PIP - Points importants pour la perception	(repr_pip)

TABLE 3 – Données adaptatives :

SAX - Approximation agrégée symbolique	(repr_sax)
PLA - Approximation linéaire par morceaux	(repr_pla)

TABLE 4 – Basé sur un modèle :

Profil saisonnier moyen, Profil saisonnier médian, etc.	
Représentations saisonnières basées sur un modèle linéaire (additif)	(repr_lm, repr_gam)
Lissage exponentiel des coefficients saisonniers	(repr_exp)

TABLE 5 – Données dictées :

FeaClip - Extraction de caractéristiques	
à partir d'une représentation découpée	(repr_feaclip, clipping)
FeaTrend - Extraction de caractéristiques	
à partir d'une représentation avec tendance	(repr_featrend, trending)
FeaClipTrend - Extraction de caractéristiques	
à partir d'une représentation découpée et en tendance	(repr_feacliptrend)

Nous détaillons ici deux méthodes emblématiques, disponibles dans le package TSclust, pour l'extraction de caractéristiques et les mesures de distance associées. Les détails supplémentaires sont abordés dans le chapitre consacré aux applications expérimentales.

1.4.1 DWT

La transformée en ondelettes discrète (DWT) est une technique d'extraction de caractéristiques souvent utilisée pour mesurer la dissimilarité entre des séries temporelles. La DWT effectue une décomposition à l'échelle de la série temporelle de manière à ce que la majeure partie de l'énergie de la série puisse être représentée par quelques coefficients seulement. L'idée principale est de remplacer les séries originales par leurs coefficients d'approximation d'ondelettes à une échelle appropriée, puis de mesurer la dissimilarité entre ces approximations. Contrairement à la transformation de Fourier discrète (TFD), qui convertit la série originale du domaine temporel au domaine fréquentiel, la DWT la transforme du domaine temporel au domaine temps-fréquence.

Contexte de la Transformée en Ondelettes Orthogonale : La transformée en ondelettes est une technique de transformation de domaine permettant la décomposition hiérarchique de séquences. Elle permet de décrire une séquence en termes d'une approximation de la séquence originale, plus un ensemble de détails allant du grossier au fin.

L'ondelette est une fonction oscillante lisse et rapidement décroissante avec une bonne

localisation à la fois en fréquence et en temps. Une famille d'ondelettes $\psi_{j,k}$ est l'ensemble de fonctions générées par dilatations et translations d'une ondelette mère unique

$$\psi_{j,k} = 2^{j/2}\psi(2^j t - k), j, k \in \mathbb{Z}$$

Une fonction $\psi \in L^2(\mathbb{R})$ est une ondelette orthogonale si la famille $\psi_{j,k}$ est une base orthogonale de $L^2(\mathbb{R})$, c'est-à-dire

$$\langle \psi_{j,k}, \psi_{l,m} \rangle = \delta_{j,l} \cdot \delta_{k,m}, j, k, l, m \in \mathbb{Z}$$

Le théorème de Parseval stipule que l'énergie est préservée sous la transformée en ondelettes orthogonales, soit,

$$\sum_{j,k \in \mathbb{Z}} |\langle f(t), \psi_{j,k} \rangle|^2 = \|f(t)\|^2, f(t) \in L^2(\mathbb{R})$$

[Mallat., 1999] a introduit l'Analyse Multirésolution (MBA) de $L^2(\mathbb{R})$ comme une chaîne de sous-espaces $\{V_j : j \in \mathbb{Z}\}$ satisfaisant aux conditions suivantes :

- (i) $\dots \subset V_{-2} \subset V_{-1} \subset V_0 \subset V_1 \subset \dots \subset L^2(\mathbb{R})$
- (ii) $\bigcap_{j \in \mathbb{Z}} V_j = \{0\}$, $\bigcup_{j \in \mathbb{Z}} V_j = L^2(\mathbb{R})$
- (iii) $f(t) \in V_j \Leftrightarrow f(2t) \in V_{j+1}; \forall j \in \mathbb{Z}$
- (iv) Il existe $\phi(t)$ fonction d'échelle telle que $\{\phi(t - k) : k \in \mathbb{Z}\}$ est une base orthogonale de V_0 .

Ainsi $\phi_{j,k} = 2^{j/2}\phi(2^j t - k)$, est la base orthogonale de V_j . Considérons l'espace W_{j-1} , qui est le complément orthogonal de V_{j-1} dans V_j : $V_j = V_{j-1} \oplus W_{j-1}$. En définissant $\psi_{j,k}$ formant la base orthogonale de W_j , la base couvre l'espace V_j :

$$V_0 \oplus W_0 \oplus W_1 \oplus \dots \oplus W_{j-1} = V_j$$

Notez que puisque W_{j-1} est orthogonal à V_{j-1} , le ψ est orthogonal à ϕ .

Transformée en Ondelettes de Haar C'est l'ondelette orthogonale la plus couramment utilisée, proposée par Haar.

L'ondelette de Haar a pour fonction mère

$$\psi_{Haar}(t) = \begin{cases} 1 & \text{si } 0 < t < 0.5 \\ -1 & \text{si } 0.5 < t < 1 \\ 0 & \text{sinon} \end{cases}$$

et pour fonction d'échelle

$$\phi_{Haar}(t) = \begin{cases} 1 & \text{si } 0 \leq t < 1 \\ 0 & \text{sinon} \end{cases}$$

Une série temporelle $Q = \{q_1, q_2, \dots, q_n\}$ située à l'échelle $J = \log_2(n)$ peut être décomposée en une partie d'approximation $A_{J-1} = \{(q_1 + q_2)/\sqrt{2}, (q_3 + q_4)/\sqrt{2}, \dots, (q_{n-1} + q_n)/\sqrt{2}\}$ et une partie de détail $D_{J-1} = \{(q_1 - q_2)/\sqrt{2}, (q_3 - q_4)/\sqrt{2}, \dots, (q_{n-1} - q_n)/\sqrt{2}\}$. Les coefficients d'approximation et les coefficients d'ondelettes à une échelle j , A_j et D_j , ayant tous deux une longueur $n/2^{J-j}$, peuvent être décomposés à partir de A_{j+1} , les coefficients d'approximation à l'échelle $j + 1$ de manière récursive. Le i ème élément de A_j est calculé comme :

$$a_j^i = \frac{1}{\sqrt{2}}(a_{j+1}^{2i-1} + a_{j+1}^{2i}), \quad \in [1, 2, \dots, n/2^{J-j}]$$

Le i ème élément de D_j est calculé comme :

$$d_j^i = \frac{1}{\sqrt{2}}(a_{j+1}^{2i-1} - a_{j+1}^{2i}), \quad \in [1, 2, \dots, n/2^{J-j}]$$

A_0 a un seul élément indiquant la moyenne globale de Q . Le i ème élément de A_j correspond au segment dans la série Q commençant à la position $(i-1) * 2^{J-j} + 1$ jusqu'à la position $i * 2^{J-j}$. Le a_j^i est proportionnel à la moyenne de ce segment et peut donc être vu comme l'approximation de ce segment. Il est clair que les coefficients d'approximation à différentes échelles fournissent une compréhension des tendances majeures dans les données à un niveau particulier de granularité.

L'idée de l'Algorithme Voici un point clé lorsque nous souhaitons utiliser la technique DWT pour le clustering : le choix d'une échelle appropriée pour obtenir un clustering précis. Dans TSclust, un algorithme automatique pour déterminer cette échelle est implémenté. L'algorithme, proposé par [Zhang and Ho, 2006], vise à sélectionner l'échelle en équilibrant deux exigences contradictoires : une réduction efficace de la dimensionnalité tout en préservant autant d'informations que possible des données originales. Spécifiquement, l'algorithme fonctionne comme suit.

Considérez un ensemble de séries temporelles $\{Q_n^1, \dots, Q_n^m\}$ situées à une échelle $J = \log_2(n)$. Notons par $H_j(Q_n^i) = \{A_j, D_j, D_{j+1}, \dots, D_{J-1}\}$ les coefficients correspondant à la transformée en ondelettes discrète de Q_n à l'échelle j . Les A_j^i sont appelés coefficients d'approximation et représentent le comportement lisse des données. Les $D_k^i, k = j, j+1, \dots, J-1$, sont appelés coefficients de détail car ils représentent la nature plus fine et à haute fréquence des données. Comme vecteur de caractéristiques, **Zhang et al.** proposent de retenir les coefficients d'approximation A_j^i pour une échelle particulière j^* correspondant à l'échelle la plus élevée qui satisfait à :

$$\sum_{i=1}^m E(D_{j^*}^i) < \sum_{i=1}^m E(D_{j^*-1}^i) \quad (1)$$

où $E(Z) = \sum_{k=1}^s z_k^2$ désigne l'énergie associée à un vecteur $Z \in \mathbb{R}^s$. L'argument derrière ce critère est que la somme des erreurs quadratiques entre Q_n^i et la série d'approximation reconstruite \hat{Q}_n^i est donnée par

$$SSE(Q_n^i, \hat{Q}_n^i) = E(Q_n^i) - E(A_j^i) = \sum_{k=j}^{J-1} E(D_k^i).$$

Ainsi, supprimer les coefficients de détail à une échelle j^* satisfaisant à l'Équation 1 et à des échelles supérieures signifie atteindre un compromis entre une dimensionnalité plus faible et un $SSE(Q_n^i, \hat{Q}_n^i)$ plus bas (c'est-à-dire, une perte d'énergie plus faible). De plus, [Zhang and Ho, 2006] montrent que l'algorithme proposé est efficace lors de l'utilisation de l'ondelette de Haar. Une fois l'échelle appropriée j^* déterminée, la dissimilarité entre deux séries Q_n^u et Q_n^v , avec $u, v \in \{1, \dots, m\}$, $u \neq v$, est donnée par

$$d_{DWT}(Q_n^u, Q_n^v) = \sqrt{\sum_k (a_{k,j^*}^u - a_{k,j^*}^v)^2},$$

où $a_{k,j}^u$ et $a_{k,j}^v$ sont les éléments de $A_{j^*}^u$ et $A_{j^*}^v$, respectivement.

1.4.2 SAX

La symbolisation consiste à transformer les séries temporelles en séquences de symboles discrétisés qui peuvent être traités efficacement pour extraire des informations sur les processus sous-jacents. La représentation symbolique SAX (approximation agrégée symbolique), introduite par [Lin J, 2003], ne souffre pas de ces défauts. L'approche SAX transforme d'abord les données originales en la représentation d'approximation agrégée par morceaux (PAA) ([Eamonn Keogh, 2000]) puis symbolise la représentation PAA en une chaîne discrète. La représentation intermédiaire PAA garantit une forte puissance de réduction de dimensionnalité et permet de prouver qu'une dissimilarité basée sur les chaînes symboliques est une borne inférieure de la véritable dissimilarité entre les séries temporelles originales (Lin et al. 2003). L'approche SAX est décrite ci-dessous.

Normalisation : Les séries temporelles sont transformées pour avoir une moyenne nulle et une variance unitaire.

Représentation PAA : Chaque série normalisée Q_n est représentée dans un espace de dimension ω par le vecteur $\bar{Q}_\omega = (\bar{Q}_1, \dots, \bar{Q}_\omega)$, dont l'élément i -ème est calculé par

$$\bar{Q}_i = \frac{\omega}{n} \sum_{j=\frac{n}{\omega}(i-1)+1}^{\frac{n}{\omega}i} Q_j, \quad (1)$$

Ainsi, les données sont d'abord divisées en ω segments de longueur égale. Ensuite, la i -ème composante de la représentation PAA est donnée par la valeur moyenne des données se trouvant dans le i -ème segment. L'équation 1 suppose que n/ω est un entier, mais cette hypothèse peut être assouplie en pondérant la contribution des points de données placés dans des segments adjacents (voir les détails sur cette généralisation dans la section 3.5 de [Lin J, 2007]

Représentation SAX Une fois le nombre a de symboles (lettres de l'alphabet) sélectionné, l'ensemble des a^{-1} -quantiles de la densité $N(0, 1)$, $\{z_{1/a}, z_{2/a}, \dots, z_{(a-1)/a}\}$, est calculé pour déterminer des zones de taille égale sous la courbe gaussienne. Maintenant, si l_i désigne la i -ème lettre de l'alphabet, pour $i = 1, \dots, a$, alors chaque représentation PAA \bar{Q}_ω est mappée à une concaténation de a symboles (appelée "mot"), disons $\hat{Q}_a = (\hat{Q}_1, \dots, \hat{Q}_a)$, comme suit.

$$\hat{Q}_i = \begin{cases} l_1 & \text{si } \bar{Q}_i < z_{1/a} \\ l_j & \text{si } \bar{Q}_i \in [z_{(j-1)/a}, z_{j/a}) \\ l_a & \text{si } \bar{Q}_i \geq z_{(a-1)/a} \end{cases}$$

De cette manière, chaque symbole du mot résultant X , c'est-à-dire de la représentation symbolique SAX, est généré avec approximativement la même probabilité.

Mesure de dissimilarité MINDIST : Bien que de nombreuses dissimilarités puissent être définies sur la représentation SAX, une qui approxime la distance euclidienne peut être utile. Tout d'abord, la distance entre une paire de symboles l_i et l_j , $i, j \in \{1, \dots, a\}$, est définie par

$$d_a(l_i, l_j) = \begin{cases} 0 & \text{si } |i - j| \leq 1 \\ z_{\max(i,j)/a} - z_{\min(i,j)/a} & \text{sinon} \end{cases}$$

Notez que la matrice carrée contenant toutes les distances entre paires de symboles n'a besoin d'être calculée qu'une seule fois. Sur la base de cette matrice, la dissimilarité est directement calculée comme suit

$$d_{MINDIST.SAX}(\hat{Q}_a, \hat{C}_a) = \sqrt{\frac{n}{\omega} \sum_{i=1}^w d_a(\hat{Q}_i, \hat{C}_i)^2}$$

Une information complète sur l'approche SAX et les questions connexes peut être vue sur la page web **Keogh (2014)** (Keogh E (2014). "The SAX (Symbolic Aggregate approXimation)." Consulté le 2014-04-22, URL <http://www.cs.ucr.edu/~eamonn/SAX.htm>.) et les références y afférentes. TSclust intègre des routines pour générer les représentations PAA et SAX et calculer la dissimilarité $d_{MINDIST.SAX}$.

1.4.3 Données dictées

Le dernier type de méthodes de représentation implémentées est celui des **données dictées** - l'écrêtage. J'ai développé deux méthodes dans cette catégorie - **FeaClip** et **FeaTrend**. Ces deux méthodes créent une représentation au niveau des bits (binaire) à partir de la série temporelle originale et calculent la longueur des valeurs par RLE (Run Length Encoding). Les caractéristiques interprétables sont ensuite extraites des longueurs d'exécution.

Je vais maintenant décrire la première des méthodes mentionnées - **FeaClip**. La représentation écrêtée est créée très facilement - si une valeur d'une série temporelle est supérieure à sa valeur moyenne, alors la valeur est transformée en 1 et sinon en 0. Elle peut être définie formellement comme suit :

$$\hat{x}_t = \begin{cases} 1 & \text{if } x_t > \mu \\ 0 & \text{otherwise} \end{cases},$$

où μ est la valeur moyenne d'une série temporelle. Sur la représentation **clipped** (au niveau des bits) \hat{x} , la méthode de compression des séries binaires appelée Run Length Encoding (**RLE**) est appliquée. Une série est une séquence continue de uns, respectivement de zéros. Le nombre de uns ou de zéros dans une série est appelé longueur de la série. À partir des longueurs de passage comptées par RLE, huit caractéristiques simples et interprétables sont extraites pour former la représentation finale et sont définies comme suit :

repr = {
 max_1 = max. from run lengths of ones,
 sum_1 = sum of run lengths of ones,
 max_0 = max. from run lengths of zeros,
 $crossings$ = length of RLE encoding - 1,
 f_0 = number of first zeros,
 l_0 = number of last zeros,
 f_1 = number of first ones,
 l_1 = number of last ones, }.

La deuxième méthode dictée par les données est **FeaTrend**. Elle permet d'extraire des caractéristiques à partir d'une représentation "tendancielle" (à nouveau binaire). La représentation de la tendance est définie comme suit :

$$\hat{x}_t = \begin{cases} 1 & \text{if } x_t - x_{t+1} < 0 \\ 0 & \text{otherwise} \end{cases} .$$

1.4.4 Répresentation basé sur modèle

Un autre groupe de représentations de séries temporelles sont des méthodes basées sur un modèle. En raison de la nature saisonnière de nos données, dans une saison (1 jour), nous avons 96 ou 48 mesures, ce qui nous permet d'extraire exactement le même nombre de paramètres du modèle statistique qui représentera les séries temporelles. Avec la création de cette représentation, les soi-disant profils quotidiens des consommateurs sont dérivés. Ils sont par la suite utilisés dans le clustering des consommateurs. Définissons la fréquence pendant une saison comme *seas*. Alors le nombre de points dans le modèle basé sur la représentation saisonnière est également *seas*.

$$\hat{x}_k = \frac{1}{d-1} \sum_{j=0}^{d-1} x_{k+seas \times j}$$

$$\hat{x}_k = \text{médiane}(x_k, x_{k+seas}, \dots, x_{k+seas \times (d-1)})$$

La première représentation est basée sur la régression linéaire multiple (nous la référons par LM). Similaire aux autres méthodes de régression, elle vise à modéliser la variable dépendante par des variables indépendantes. Formellement, le modèle peut être écrit comme suit :

$$x_i = \beta_1 u_{i1} + \beta_2 u_{i2} + \dots + \beta_{seas} u_{iseas} + \varepsilon_i, \quad (1)$$

pour $i = 1, \dots, n$, où u_{ij} est la consommation d'électricité i -ème, $\beta_1, \dots, \beta_{seas}$ sont les coefficients de régression. Les u_{i1}, \dots, u_{iseas} sont des variables binaires (dummy) indépendantes représentant les numéros de séquence dans le modèle de régression. Cela signifie juste dans le cas où ils pointent vers la valeur j -ème de la saison, $j = 1, 2, \dots, seas$. L'erreur ε_i est une erreur aléatoire avec la distribution normale $N(0, \sigma^2)$ et sont indépendants. La méthode la plus répandue pour obtenir une estimation du vecteur $\beta = (\beta_1, \dots, \beta_{seas})$ est la méthode des moindres carrés ordinaires (MCO). Pour la représentation des RLM, GAM, etc., voir [Laurinec and Lucká, 2016].

2 Mesures de similarité/dissimilitude

Le clustering de séries temporelles dépend fortement de la mesure de distance utilisée. Certaines mesures de similarité sont conçues pour des représentations spécifiques des séries temporelles, comme MINDIST en lien avec SAX [Lin J, 2007], tandis que d'autres sont indépendantes de la méthode de représentation, ou adaptées aux séries temporelles brutes. Contrairement aux mesures de distance dans le clustering standard, qui se basent sur des correspondances exactes entre objets statiques, les mesures dans le clustering de séries temporelles sont souvent calculées de manière approximative.

Plusieurs mesures de distance ont été développées pour quantifier la similarité entre séries temporelles, incluant la distance de Hausdorff, la distance de Hausdorff modifiée (MODH), la distance basée sur les modèles de Markov cachés (HMM), le Dynamic Time Warping (DTW), la distance euclidienne, la distance euclidienne dans un sous-espace PCA, et la plus longue sous-séquence commune (LCSS). Ces méthodes sont largement utilisées pour analyser les données de séries temporelles.

Les efforts de recherche sur la mesure de distance tenant compte de la forme des séries temporelles rencontrent souvent des défis tels que le bruit, les variations d'amplitude, les décalages, les changements d'échelle temporelle, la dérive linéaire, les discontinuités et la dérive temporelle, autant de caractéristiques fréquentes dans les données de séries temporelles.

Le choix d'une mesure de distance adéquate dépend des propriétés spécifiques des séries temporelles, de leur longueur, de la méthode de représentation choisie, et bien sûr, de l'objectif du clustering.

En général, trois grands objectifs nécessitent l'adoption d'approches distinctes en matière de mesure de distance.

2.0.1 Recherche de séries temporelles similaires dans le temps

Étant donné que cette similarité s'applique à chaque pas de temps, les distances basées sur la corrélation ou la mesure de la distance euclidienne conviennent à cet objectif. Cependant, comme ce type de mesure de distance est coûteux sur les séries temporelles brutes, le calcul est effectué sur des séries temporelles transformées, telles que les transformées de Fourier, les ondelettes ou l'approximation agrégée par morceaux (AAP). Regroupement de séries temporelles qui sont corrélées (par exemple, pour regrouper les séries temporelles du prix des à de nombreuses entreprises pour déterminer quelles actions changent ensemble et comment elles sont corrélées). est classé comme un regroupement basé sur la similarité dans le temps [C. Ratanamahatana, 2005] [A.J. Bagnall, 2005].

2.0.2 Recherche de séries temporelles similaires en termes de forme

L'heure d'apparition des motifs n'est pas importante pour trouver des séries temporelles similaires en termes de forme. Par conséquent, des méthodes élastiques [R. Agrawal, 1993],[?] telles que le Dynamic time Warping (DTW) sont utilisées pour calculer la dissimilitude. Selon cette définition, Cette définition permet de créer des groupes de séries temporelles présentant des modèles de changement similaires, indépendamment des points temporels. (par exemple pour regrouper les cours des actions de différentes entreprises qui ont un modèle commun dans leurs actions, indépendamment de son occurrence dans les séries temporelles [A.J. Bagnall, 2005].)

2.0.3 Recherche de séries temporelles similaires dans le changement (similarité structurelle)

Cette approche utilise des méthodes de modélisation telles que les modèles de Markov cachés (HMM) ou un processus ARMA pour mesurer la similarité des paramètres de modèles ajustés aux séries temporelles. L'idée est de regrouper les séries ayant une structure d'autocorrélation semblable, par exemple, des actions dont le prix tend à augmenter après une chute. Bien adaptée aux séries temporelles longues, cette méthode est moins efficace pour des séries plus courtes.

2.1 Considérations sur la sélection de la mesure de dissimilarité

Les techniques de clustering peuvent être catégorisées selon la longueur des séries temporelles en deux niveaux : "forme" et "structure". Le niveau "forme" convient pour le clustering de séries temporelles courtes, mesurant la similarité basée sur des aspects immédiats et visuels, tandis que le niveau "structure" se focalise sur la similarité fondée sur la structure globale, appropriée pour des données temporelles de longue durée.

Nous examinons d'abord le clustering basé sur la forme pour des séries temporelles de courte durée, utilisant des mesures de similarité à ce niveau. Selon l'objectif et la longueur des séries, un type spécifique de mesure de distance est choisi.

Les mesures de similarité basées sur la forme, telles que la distance Euclidienne, DTW, LCSS, et MVM, sont optimales pour des séries temporelles courtes. À l'inverse, la similarité basée sur la compression est adaptée tant aux séries courtes que longues, avec des méthodes comme CDM, l'autocorrélation, Permuttaion distribution et la distance pour courtes séries temporelles, ainsi que le coefficient de corrélation de Pearson et les distances associées, le Cepstrum, la normalisation par morceaux, et les ondelettes de Cosinus.

Pour les séries temporelles longues, les mesures de similarité basées sur les caractéristiques ou sur des modèles spécifiques, comme les HMM et ARMA, sont préférées.

Nous verrons plus loin que nous pouvons analyser conjointement "temps" et "structure", et qu'elles sont toutes deux très dépendantes du temps.

2.2 Paquet R

TSclust propose une collection de mesures de dissimilarité pour le clustering de séries temporelles, incluant des approches basées sur les données brutes, la modélisation et l'analyse prédictive. Il offre également des outils supplémentaires pour le clustering de séries temporelles, comme des algorithmes de clustering et des indicateurs pour évaluer la qualité des clusters.

dtwclust fournit des méthodes optimisées pour le clustering de séries temporelles en utilisant la distance Dynamic Time Warping (DTW) et ses bornes inférieures. Il supporte le clustering partitionnel, hiérarchique, flou, k-Shape et TADPole, avec la possibilité d'étendre les fonctionnalités via des mesures de distance et des définitions de centroïdes personnalisées. Le paquet inclut des implémentations de la moyenne de barycentre DTW, une distance basée sur des noyaux d'alignement global, et des routines pour soft-DTW. Les fonctions de distance profitent d'optimisations pour calculer efficacement les matrices de distances croisées, avec support pour la parallélisation. Divers indices de validité des clusters sont également intégrés.

LPWC calcule une mesure de distance pour le clustering de séries temporelles basée sur la corrélation pondérée, avec l'introduction de décalages pour capturer les réponses retardées dans les données. Les points temporels doivent être spécifiés explicitement.

TSdist fournit un ensemble de mesures de distance fréquemment utilisées pour les séries temporelles, ainsi que quelques fonctions additionnelles qui, bien que non conçues initialement à cet effet, peuvent servir à évaluer la dissimilarité entre séries temporelles. Ces mesures sont applicables à des tâches de clustering, classification ou autres analyses exploratoires nécessitant une définition de distance entre les séries temporelles.

2.3 Les Méthode dans R

Fig.3 montre que l'on sait sur la mesure des distances entre les séries temporelles dans R

	proxy	longitudinal Data	TSclust	dtw	pdc	TSdist
<i>Shape based distances</i>						
<i>Lock-step measures</i>						
L_p distances		✓				
DISSIM						✓
Short Time Series Distance (STS)						✓
Cross-correlation based						✓
Pearson correlation based			✓			
CORT distance			✓			
<i>Elastic measures</i>						
Frechet distance		✓				
Dynamic Time Warping (DTW)				✓		
Keogh_LB for DTW						✓
Edit Distance for Real Sequences (EDR)						✓
Edit Distance with Real Penalty (ERP)						✓
Longest Common Subsequence (LCSS)						✓
<i>Feature-based distances</i>						
(Partial) Autocorrelation based			✓			
Fourier Decomposition based						✓
TQuest						✓
Wavelet Decomposition based			✓			
(Integrated) Periodogram based			✓			
SAX representation based			✓			
Spectral Density based			✓			
<i>Structure-based distances</i>						
<i>Model based</i>						
Piccolo distance			✓			
Maharaj distance			✓			
Cepstral based distances			✓			
<i>Compression based</i>						
Compression based distances			✓			
Complexity invariant distance			✓			
Permutation distribution based distance						✓
<i>Prediction based</i>						
Non Parametric Forecast based			✓			

FIGURE 4 – Résumé des mesures de distance pour les séries temporelles implémentées dans R.

2.4 Résumé des mesures de distance pour les séries temporelles implémentées dans R.

2.4.1 DTW

Le Dynamic Time Warping (DTW) est une mesure de distance robuste pour les séries temporelles, permettant d'apparier des formes similaires même lorsqu'elles sont déphasées sur l'axe temporel. Toutefois, le DTW ne respecte pas l'inégalité triangulaire, ce qui a limité les tentatives d'indexation exacte.

Supposons deux séries temporelles, Q et C , de longueur n et m respectivement, où

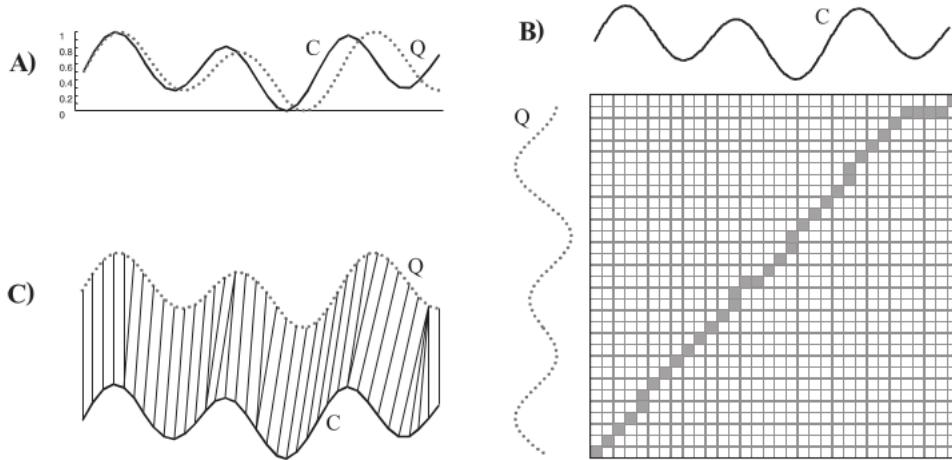


FIGURE 5 – **A)** Deux séquences Q et C similaires mais déphasées. **B)** Pour aligner les séquences, nous construisons une matrice de guerre et recherchons le chemin de guerre optimal, montré par les carrés pleins. **C)** L’alignement résultant

$$Q = q_1, q_2, \dots, q_i, \dots, q_n \quad C = c_1, c_2, \dots, c_j, \dots, c_m \quad (1)$$

Pour aligner deux séquences avec DTW, la première étape consiste à créer une matrice des coûts locaux (LCM ou lcm), en calculant la norme l_p entre q_i et c_j . Nous construisons une matrice n -par- m où l’élément (i, j) contient la distance $d_p(q_i, c_j)$ entre les deux points q_i et c_j (i.e. $d_p(q_i, c_j) = (\sum_v |q_i^v - c_j^v|^p)^{1/p}$). q_i^v représente la v -ième variable pour les points q_i . Chaque élément de la matrice (i, j) correspond à l’alignement entre les points q_i et c_j . Ceci est illustré dans la figure 5. Un chemin de guerre W est un ensemble contigu (au sens ci-dessous) d’éléments de matrice qui définit un mappage entre Q et C. Le k -ième élément de W est défini comme $w_k = (i, j)_k$. Ainsi, nous avons

$$W = w_1, w_2, \dots, w_k, \dots, w_K \quad \max(m, n) \leq K < m + n - 1 \quad (2)$$

Le chemin de guerre est soumis à plusieurs contraintes.

- **Conditions aux limites :** $w_1 = (1, 1)$ et $w_K = (m, n)$. Cela exige que le chemin de guerre commence et se termine dans les cellules d’angle opposées de la matrice.
- **Continuité :** Étant donné $w_k = (a, b)$, alors $w_{k-1} = (a', b')$, où $a - a' \leq 1$ et $b - b' \leq 1$. Cela limite les étapes admissibles dans le chemin de guerre aux cellules adjacentes (y compris les cellules adjacentes en diagonale).
- **Monotonie :** Étant donné $w_k = (a, b)$, alors $w_{k-1} = (a', b')$, où $a - a' \geq 0$ et $b - b' \geq 0$. Cela oblige les points dans W à être espacés de manière monotone dans le temps.

Il existe exponentiellement de nombreux chemins de guerre qui satisfont aux conditions ci-dessus. Cependant, nous sommes seulement intéressés par le chemin qui minimise le coût de guerre :

$$DTW_p(Q, C) = \min\left(\sum_{k=1}^K \frac{d_p(w_k)^p m_k}{M_w}\right)^{1/p} \quad (3)$$

où $d_p(w_k) = d_p(q_a, c_b)$. m_k est un coefficient de pondération par étape et M_w est la constante de normalisation correspondante.

Dans cette définition, le choix de la norme l_p joue deux fois durant l’algorithme DTW.

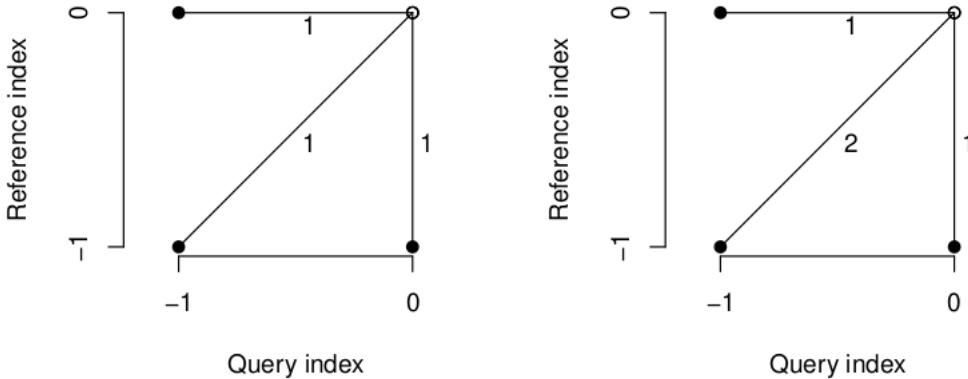


FIGURE 6 – Deux motifs de pas bien connus, de gauche à droite : **symétrique 1**, **symétrique 2**

Cependant, comme mentionné précédemment, la norme n'affecte la LCM que si des séries multivariées sont utilisées.

Cette revue de DTW est nécessairement brève ; nous renvoyons le lecteur intéressé à [Kruskall and Liberman] pour un traitement plus détaillé.

1.1 Motifs de pas et contraintes locales sur le warping temporel Les alignements sont généralement réalisés en dupliquant des éléments, c'est-à-dire en faisant correspondre un seul point temporel dans X avec plusieurs éléments consécutifs dans Y, ou vice-versa. Le nombre d'éléments pouvant être appariés consécutivement ou omis détermine la pente locale de la courbe de warping. Cette propriété peut être contrôlée par un schéma très flexible appelé motifs de pas. Les motifs de pas listent des ensembles de transitions autorisées entre les paires appariées et les poids correspondants.

Le chemin dans LCM peut être trouvé en utilisant la programmation dynamique pour évaluer la récurrence suivante, qui définit la distance cumulative $g(i, j)$ comme la distance $d(i, j)$ trouvée dans la cellule actuelle et le minimum des distances cumulatives des éléments adjacents :

$$g(i, j) = d(q_i, c_j) + \min\{g(i - 1, j - 1), g(i - 1, j), g(i, j - 1)\}. \quad (4)$$

ou sous une autre forme

$$g(i, j) = \min\{g(i - 1, j - 1) + 2d(q_i, c_j), g(i - 1, j) + d(q_i, c_j), g(i, j - 1) + d(q_i, c_j)\}. \quad (5)$$

Les deux formules de la distance cumulative sont symétriques et correspondent aux différents motifs de pas de la 6.

Pour la récursion **symétrique 2**, le coût moyen par étape (normalisation) est calculé en divisant la distance cumulative par $n + m$,

L'idée de base des contraintes locales est de limiter les chemins de warping possibles, en fournissant des restrictions locales sur l'ensemble des étapes alternatives considérées. Par exemple, l'Eq. (4) peut être visualisée comme un diagramme de motifs de pas admissibles, comme dans la Fig. 3a. Les lignes illustrent les étapes permises que le chemin de warping peut prendre à chaque étape. Nous pourrions remplacer l'Eq. (4) par $g(i, j) = d(i, j) + \min\{g(i - 1, j - 1) + 2d(q_i, c_j), g(i - 1, j) + d(q_i, c_j), g(i, j - 1) + d(q_i, c_j)\}$.

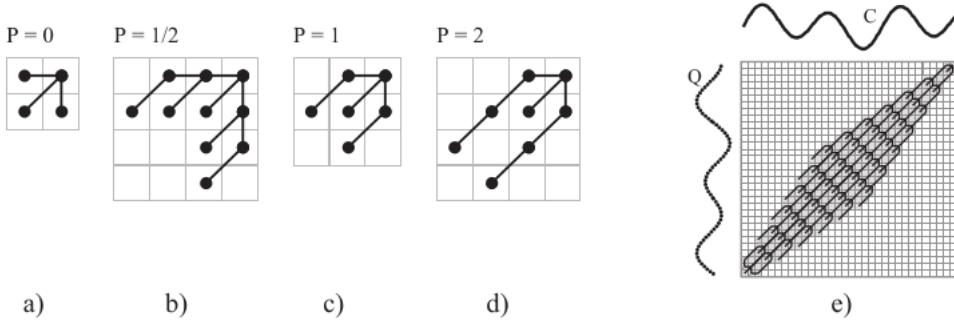


FIGURE 7 – a à d Quatre contraintes locales sur le Dynamic Time Warping, comme suggéré par Sakoe et Chiba. a) correspond au cas trivial sans contrainte, et est donc équivalent à l’Éq. (4),

$1, j - 1), g(i - 1, j - 2), g(i - 2, j - 1)\}$, qui correspond au motif de pas montré dans la 7. En utilisant cette équation, le chemin de warping est contraint de se déplacer d’un pas diagonal pour chaque pas parallèle à un axe. L’intensité effective de la contrainte de pente peut être mesurée par $P = n_1/m_1$. (c'est-à-dire, si le point $w(k)$ avance dans la direction de l'axe i (ou j) consécutivement m_1 fois, alors le point $w(k)$ n'est pas autorisé à avancer davantage dans la même direction avant de faire au moins n_1 pas dans la direction diagonale.) Les 7 illustrent les quatre contraintes originales suggérées par [SAKOE and CHIBA, 1978]. De plus, beaucoup d'autres ont été suggérées, y compris asymétriques. L'implication importante des contraintes locales pour notre travail est le fait qu'elles peuvent être réinterprétées comme des contraintes globales. La ?? montre un exemple.

Dans le 8, les algorithmes sont résumés pour les formes symétriques et asymétriques, avec diverses conditions de contrainte de pente.

La distance euclidienne entre deux séquences peut être vue comme un cas spécial de DTW où le k -ième élément de W est contraint de telle sorte que $w_k = (i, j)_k, i = j = k$. Notez que cela n'est défini que dans le cas spécial où les deux séquences ont la même longueur. La complexité en temps et en espace du DTW est $O(nm)$.

Cette revue des motifs de pas et des contraintes locales sur le warping temporel est nécessairement brève ; nous renvoyons le lecteur intéressé à [SAKOE and CHIBA, 1978].

1.2 Contraintes globales sur le warping temporel En plus des contraintes sur le chemin de warping énumérées dans la Sect. 1, presque tous les praticiens utilisant le DTW imposent également des contraintes globales en limitant l'écart possible du chemin par rapport à la diagonale. L'ensemble de la matrice que le chemin de warping est autorisé à visiter est appelé la fenêtre de warping. 2.4.1 illustre deux des contraintes globales les plus fréquemment utilisées, la bande de Sakoe-Chiba (**Sakoe et Chiba 1978**) et le parallélogramme d'Itakura (**Itakura 1975**).

— **Condition de la fenêtre d'ajustement (Bande de Sakoe-Chiba)** : Étant donné $w_k = (a, b)$, où $|a - b| \leq r$, r étant un entier positif approprié appelé longueur de la fenêtre.

Il convient de noter que la bande de Sakoe-Chiba fonctionne bien lorsque $n \sim m$, mais est inappropriée lorsque les longueurs des deux entrées diffèrent significativement. En particulier, lorsque $|n - m| > r$, le point de terminaison (n, m) se trouve hors de la bande, et donc aucune solution n'existe pour un alignement global. En général, lors de l'application de contraintes

P	Schematic explanation	Symmetric Asymmetric	DP-equation $g(i, j) =$
0		Symmetric	$\min \begin{bmatrix} g(i, j-1) + d(i, j) \\ g(i-1, j-1) + 2d(i, j) \\ g(i-1, j) + d(i, j) \end{bmatrix}$
		Asymmetric	$\min \begin{bmatrix} g(i, j-1) \\ g(i-1, j-1) + d(i, j) \\ g(i-1, j) + d(i, j) \end{bmatrix}$
1/2		Symmetric	$\min \begin{bmatrix} g(i-1, j-3) + 2d(i, j-2) + d(i, j-1) + d(i, j) \\ g(i-1, j-2) + 2d(i, j-1) + d(i, j) \\ g(i-1, j-1) + 2d(i, j) \\ g(i-2, j-1) + 2d(i-1, j) + d(i, j) \\ g(i-3, j-1) + 2d(i-2, j) + d(i-1, j) + d(i, j) \end{bmatrix}$
		Asymmetric	$\min \begin{bmatrix} g(i-1, j-3) + (d(i, j-2) + d(i, j-1) + d(i, j))/3 \\ g(i-1, j-2) + (d(i, j-1) + d(i, j))/2 \\ g(i-1, j-1) + d(i, j) \\ g(i-2, j-1) + d(i-1, j) + d(i, j) \\ g(i-3, j-1) + d(i-2, j) + d(i-1, j) + d(i, j) \end{bmatrix}$
1		Symmetric	$\min \begin{bmatrix} g(i-1, j-2) + 2d(i, j-1) + d(i, j) \\ g(i-1, j-1) + 2d(i, j) \\ g(i-2, j-1) + 2d(i-1, j) + d(i, j) \end{bmatrix}$
		Asymmetric	$\min \begin{bmatrix} g(i-1, j-2) + (d(i, j-1) + d(i, j))/2 \\ g(i-1, j-1) + d(i, j) \\ g(i-2, j-1) + d(i-1, j) + d(i, j) \end{bmatrix}$
2		Symmetric	$\min \begin{bmatrix} g(i-2, j-3) + 2d(i-1, j-2) + 2d(i, j-1) + d(i, j) \\ g(i-1, j-2) + 2d(i, j) \\ g(i-3, j-2) + 2d(i-2, j-1) + 2d(i-1, j) + d(i, j) \end{bmatrix}$
		Asymmetric	$\min \begin{bmatrix} g(i-2, j-3) + 2(d(i-1, j-2) + d(i, j-1) + d(i, j))/3 \\ g(i-1, j-2) + d(i, j) \\ g(i-3, j-2) + d(i-2, j-1) + d(i-1, j) + d(i, j) \end{bmatrix}$

FIGURE 8 – Algorithmes DP symétriques et asymétriques avec condition de contrainte de pente $P = 0, \frac{1}{2}, 1, 2$

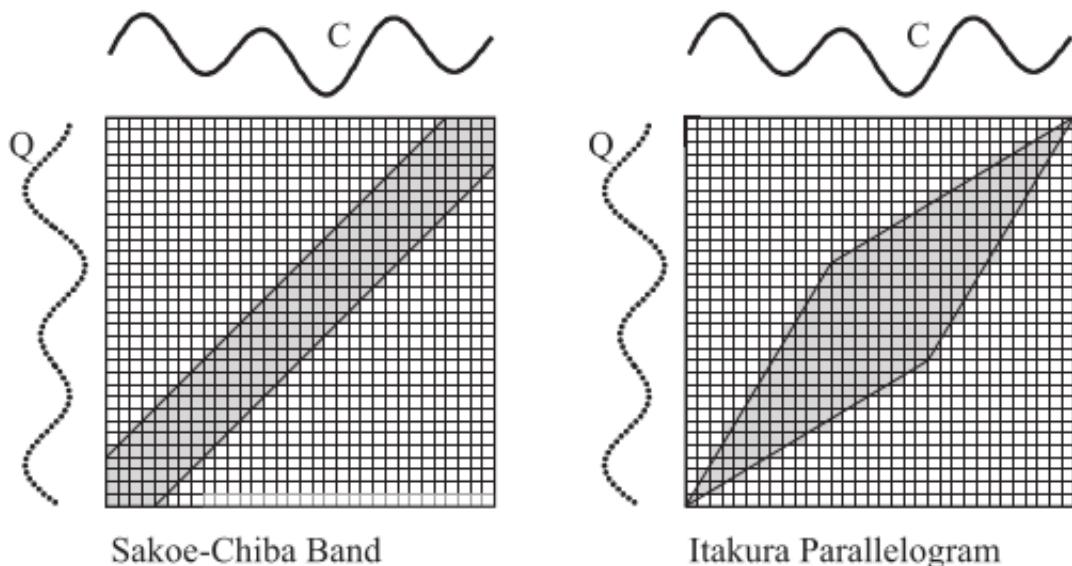


FIGURE 9 – Les contraintes globales limitent le champ d'action du chemin de warping, le restreignant aux zones grises. Les deux contraintes les plus courantes dans la littérature sont la bande de Sakoe-Chiba et le parallélogramme d'Itakura

globales et/ou locales (mentionnées ci-dessous), il faut veiller à ce qu'elles soient compatibles entre elles et avec les longueurs des séries temporelles.

Itakura a proposé une limitation de pente par morceaux. Cela équivaut graphiquement à créer un parallélogramme englobant les régions de début et de fin du warping temporel pour restreindre le chemin de warping comme montré dans la 6. Cette contrainte est appelée parallélogramme d'Itakura. Cette contrainte offre une amélioration pratique de la performance en plus d'augmenter la précision de l'alignement mais n'affecte pas le temps d'exécution. Les sommets sont obtenus par l'intersection des lignes définies par les équations 6 et 7, respectivement.

$$i - 1 = 2(j - 1)i - n = (j - m)/2 \quad (6)$$

$$i - 1 = (j - 1)/2i - n = 2(j - m) \quad (7)$$

Il existe plusieurs raisons d'utiliser des contraintes globales, l'une d'entre elles est qu'elles accélèrent légèrement le calcul de la distance DTW (Dynamic Time Warping). Cependant, la raison la plus importante est d'éviter les déformations pathologiques, où une section relativement petite d'une séquence est mappée sur une section relativement grande d'une autre.

Cette revue des contraintes globales sur le décalage temporel est nécessairement brève ; nous renvoyons le lecteur Vuppala intéressé vers une traitement plus détaillé.

1.3.1 Enveloppes de déformation Nos séries temporelles sont maintenant de longueur égale et univariées.

Nous définissons $U(Q)_i = \max_{\{|k-i| \leq r\}} q_k$ et $L(Q)_i = \min_{\{|k-i| \leq r\}} q_k$, la paire $U(Q)$ et $L(Q)$ forme l'enveloppe de déformation de Q (voir 10). Nous laissons la contrainte globale (c'est-à-dire la longueur de la fenêtre) r implicite. Et nous notons $NDTW_p$ lorsque la monotonie n'est pas requise.

Théorème. Étant donné deux séries temporelles de même longueur Q et C et $1 \leq p < \infty$, alors pour toute série temporelle H satisfaisant $Q_i \geq H_i \geq U(C)_i$ ou $Q_i \leq H_i \leq L(C)_i$ ou $Q_i = H_i$ pour tous les indices i, nous avons

$$DTW_p(Q, C)^p \geq NDTW_p(Q, C)^p \geq \|Q - H\|_p^p + NDTW_p(H, C)^p$$

Pour $p = \infty$, un résultat similaire est vrai : $DTW_\infty(Q, C) \geq NDTW_\infty(Q, C) \geq \max(\|Q - H\|_\infty, NDTW_\infty(H, C))$.

Pour la preuve, veuillez trouver le détail dans [Lemire, 2009]

Alors que le Théorème 1 définit une borne inférieure ($\|Q - H\|_p$), la proposition suivante montre que cette borne inférieure doit être une approximation serrée tant que H est proche de C dans la norme l_p .

Proposition : Étant donné deux séries temporelles de même longueur Q et C , et $1 \leq p \leq \infty$ avec H comme dans le Théorème, nous avons que $\|Q - H\|_p$ approxime à la fois $DTW_p(Q, C)$ et $NDTW_p(Q, C)$ à $\|H - C\|_p$. (pour la preuve, veuillez trouver le détail dans [Lemire, 2009])

Le calcul de l'enveloppe de déformation $U(Q), L(Q)$ nécessite $O(nw)$ temps en utilisant l'approche naïve de calculer répétitivement le maximum et le minimum sur des fenêtres. Au lieu de cela, nous calculons l'enveloppe avec au plus $3n$ comparaisons en utilisant l'Algorithm WE.

Algorithm WE : Algorithm de streaming pour calculer l'enveloppe

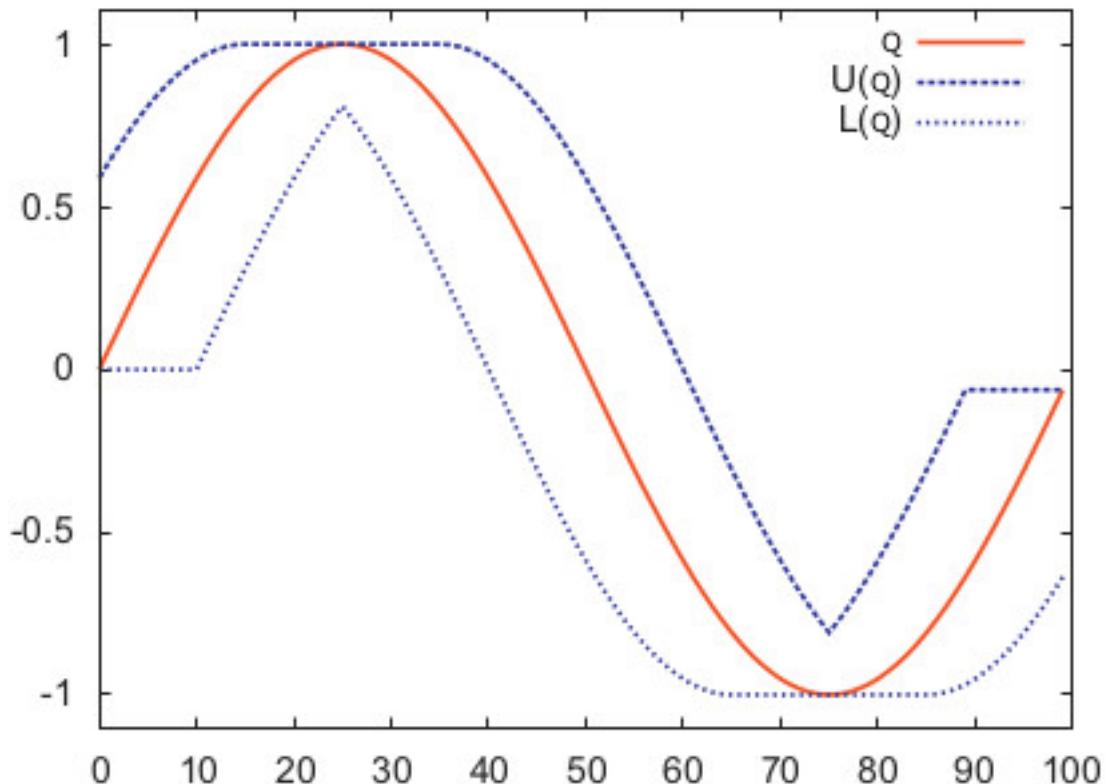


FIGURE 10 – Exemple d'enveloppe de déformation

de déformation en utilisant pas plus de $3n$ comparaisons.
 Entrée une série temporelle Q indexée de 1 à n
 Entrée une contrainte globale DTW r
 retourne enveloppe de déformation U , L (deux séries temporelles de longueur n)

début

```

u, l ← files doublement terminées vides, nous ajoutons à "l'arrière"
ajouter 1 à u et l
pour i dans {2, ..., n} faire
    si i >= r + 1 alors
        U[i-r] <- Q[avant(u)] , L[i-r] <- Q[avant(l)]
        Finsi
        si Q[i] > Q [i-1] alors
            pop u de l'arrière
        tant que Q[i] > Q[arrière(u)] faire
            pop u de l'arrière
            Fintantque
    sinon
        pop l de l'arrière
        tant que Q[i] < Q[arrière(l)] faire
            pop l de l'arrière
            Fintantque
    
```

```

        Finsi
        ajouter i à u et l
        si i = 2r + 1 + avant(u) alors
pop u de l'avant
sinon si i = 2r + 1 + avant(l) alors
pop l de l'avant
        Finsi
        Finsi
    FinPour
pour i dans {n + 1, ..., n + r} faire
U[i-r] <- Q[avant(u)], L[i-r] <- Q[avant(l)]
si i-avant(u) >= 2r + 1 alors
pop u de l'avant
        Finsi
si i-avant(l) >= 2r + 1 alors
pop l de l'avant
        Finsi
    Finpour
FIN
    
```

chaque élément $Q[i]$ est impliqué dans au plus 3 comparaisons : une fois lorsqu'il est considéré pour être ajouté à u ou l , et au plus deux fois lorsqu'il est considéré pour être retiré de l'une ou l'autre file d'attente. Comme il y a n éléments dans la série temporelle Q , le calcul de l'enveloppe de déformation nécessite au plus $3n$ comparaisons.

LB_Keogh Soit $H(Q, C)$ la projection de Q sur C définie comme

$$H(Q, C)_i = \begin{cases} U(C)_i & \text{si } Q_i \geq U(C)_i \\ L(C)_i & \text{si } Q_i \leq L(C)_i \\ Q_i & \text{sinon} \end{cases} \quad (8)$$

pour $i = 1, 2, \dots, n$. Nous avons que $H(Q, C)$ est dans l'enveloppe de C . Par le Théorème et en posant $H = H(Q, C)$, nous avons que $NDTW_p(Q, C)^p \geq \|Q - H(Q, C)\|_p^p + NDTW_p(H(Q, C), C)^p$ pour $1 \leq p < \infty$. Écrivons $LB_{Keogh_p}(Q, C) = \|Q - H(Q, C)\|_p$ (voir 11), alors $LB_{Keogh_p}(Q, C)$ est une borne inférieure à $NDTW_p(Q, C)$ et donc à $DTW_p(Q, C)$. Le corollaire suivant découle du Théorème et de la Proposition.

Corollaire Étant donné deux séries temporelles de même longueur Q et C et $1 \leq p \leq \infty$ alors

- $LB_{Keogh_p}(Q, C)$ est une borne inférieure au DTW :
 - $DTW_p(Q, C) \geq NDTW_p(Q, C) \geq LB_{Keogh_p}(Q, C)$
 - la précision de LB_{Keogh} est limitée par la distance à l'enveloppe :
 - $DTW_p(Q, C) - LB_{Keogh_p}(Q, C) \leq \|\max\{U(C)_i - C_i, C_i - L(C)_i\}_i\|_p$ pour tout Q .
- (pour la preuve, veuillez trouver le détail dans Lemire [2009])

Cette revue des contraintes globales sur le décalage temporel est nécessairement brève ; nous renvoyons le lecteur intéressé à Eamonn Keogh [2005] pour un traitement plus détaillé.

LB_Amélioré Dans la section précédente, nous avons vu que $NDTW_p(Q, C)^p \leq LB_{Keogh_p}(Q, C)^p + NDTW_p(H(Q, C), C)^p$ pour $1 \leq p < \infty$.

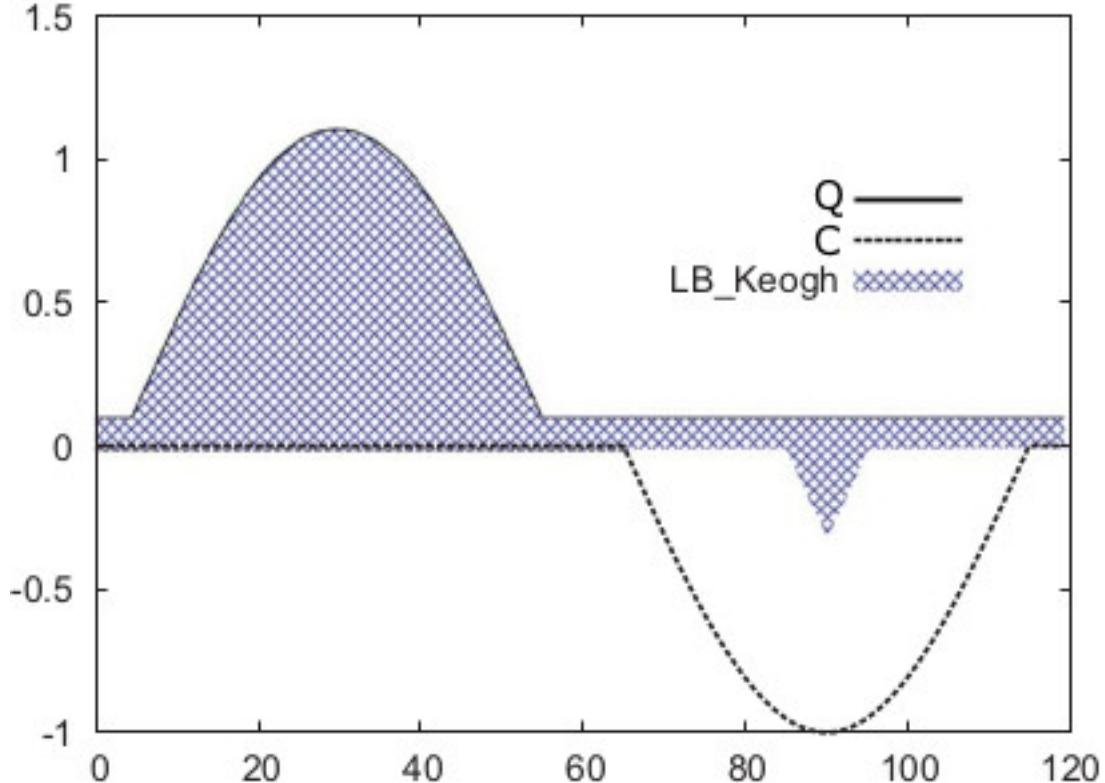


FIGURE 11 – Exemple de LB_Keogh : l’aire de la région marquée est $\text{LB_Keogh}_1(Q, C)$

De plus, nous avons $NDTW_p(H(Q, C), C) \geq \text{LB_Keogh}_p(C, H(Q, C))$. Par conséquent, écrivons

$$\text{LB_Amélioré}_p(Q, C)^p = \text{LB_Keogh}_p(Q, C)^p + \text{LB_Keogh}_p(C, H(Q, C))^p$$

pour $1 \leq p < \infty$. Par définition, nous avons $\text{LB_Amélioré}_p(Q, C) \geq \text{LB_Keogh}_p(Q, C)$. Intuitivement, alors que $\text{LB_Keogh}_p(Q, C)$ mesure la distance entre Q et l’enveloppe de C , $\text{LB_Keogh}_p(C, H(Q, C))$ mesure la distance entre C et l’enveloppe de la projection de Q sur C (voir 12). Le corollaire suivant montre que LB_Amélioré est une borne inférieure au DTW

Corollaire 2. Étant donné deux séries temporelles de même longueur Q et C et $1 \leq p < \infty$, alors $\text{LB_Amélioré}_p(Q, C)$ est une borne inférieure au DTW :

$$DTW_p(Q, C) \geq NDTW_p(Q, C) \geq \text{LB_Amélioré}_p(Q, C).$$

(pour la preuve, veuillez trouver le détail dans Lemire [2009])

Cette revue des contraintes globales sur le décalage temporel est nécessairement brève ; nous renvoyons le lecteur intéressé à Lemire [2009] pour un traitement plus détaillé.

2.4.2 GAK

[Cuturi, 2007] proposed a family of kernels to handle times series, based on the DTW, a distance has taken the local dependencies into account. These kernels are positive definite kernels under favorable conditions, which is an important requirement of kernel machines (for example SVM), but most importantly, they incorporate more information on the compared

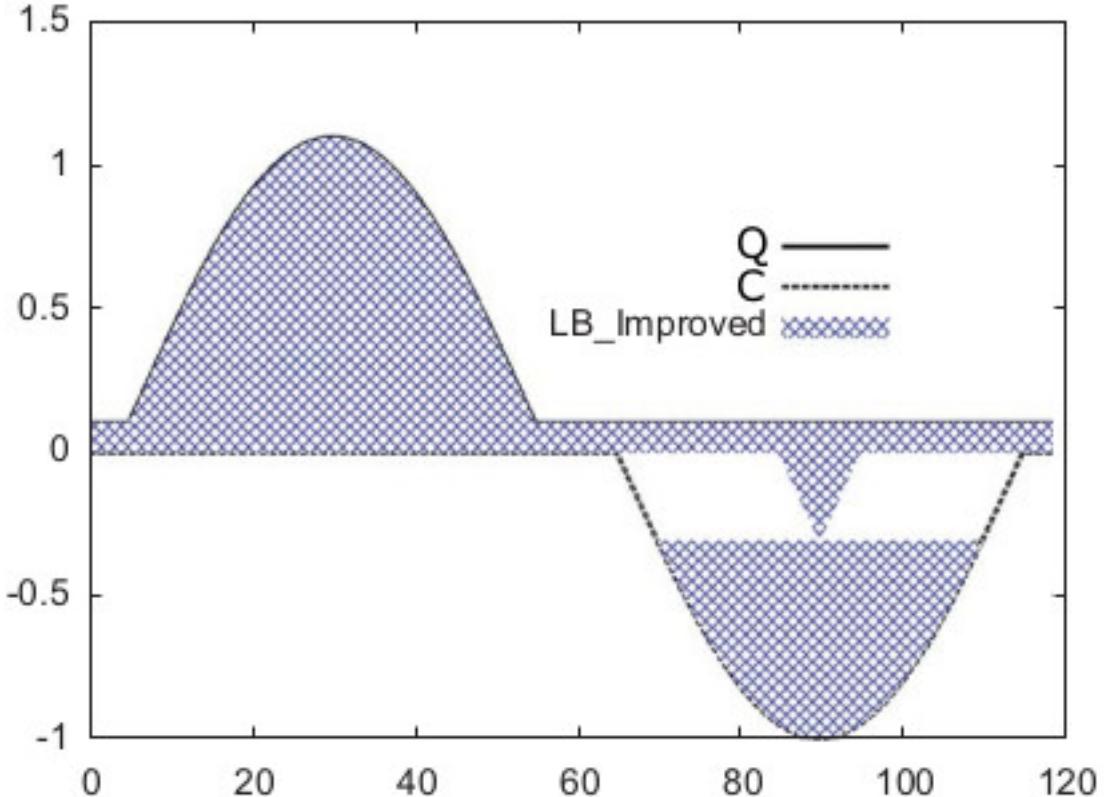


FIGURE 12 – Exemple de LB_Amélioré : l'aire de la région marquée est $\text{LB}_\text{Amélioré}_1(Q, C)$

sequences than DTW, while requiring exactly the same computational cost.

1 Alignements et le cadre DTW Soit \mathcal{X}^N l'ensemble des séries temporelles à temps discret prenant des valeurs dans un espace arbitraire. Un alignement w entre deux séries temporelles $Q = (q_1, \dots, q_n)$ et $C = (c_1, \dots, c_m)$ de longueurs n et m respectivement est une paire de vecteurs intégraux croissants $w_k = (w_{1k}, w_{2k})$ de longueur $K \leq n + m - 1$ tels que $1 = w_{11} \leq \dots \leq w_{1K} = n$ et $1 = w_{21} \leq \dots \leq w_{2K} = m$, avec condition de continuité et de monotonie dans la section **DTW**.

Notez que les alignements sont contraints par trois conditions : frontière, continuité et monotonie. Nous écrivons

$\mathcal{A}(n, m)$ pour l'ensemble de tous les alignements entre deux séries temporelles de longueur n et m . Dans sa forme la plus simple, que nous choisissons tous les coefficients de pondération par étape égaux à 1 (sans normalisation) et norme dans l_p , la distance DTW entre Q et C est définie comme

$$DTW(Q, C) \stackrel{\text{def}}{=} \min_{w \in \mathcal{A}(n, m)} D_{Q, C}(w)$$

où, le coût

$$D_{Q, C}(w) = \sum_{k=1}^K \varphi(Q_{w_{1k}}, C_{w_{2k}}) = \sum_{k=1}^K d_p(Q_{w_{1k}}, C_{w_{2k}})^p \quad (1)$$

Lorsque $\mathcal{X} = \mathbf{R}^d$, φ peut typiquement être défini comme la distance Euclidienne au carré l_2 .

2. Minimum doux de tous les scores d'alignement Le noyau d'Alignement Global (GA) est défini comme le minimum doux exponentié de toutes les distances d'alignement,

$$k_{\text{GA}}(Q, C) \stackrel{\text{def}}{=} \sum_{w \in \mathcal{A}(n, m)} e^{-D_{Q, C}(w)} \quad (2)$$

L'équation (2) peut être réécrite en utilisant la fonction de similarité locale κ induite à partir du φ comme $\kappa \stackrel{\text{def}}{=} e^{-\varphi}$:

$$k_{\text{GA}}(Q, C) \stackrel{\text{def}}{=} \sum_{w \in \mathcal{A}(n, m)} \prod_{k=1}^K \kappa(Q_{w1_k}, C_{w2_k})$$

Cuturi [2007] argue que la similarité décrite par k_{GA} incorpore tout le spectre des coûts $D_{Q, C}(w), w \in \mathcal{A}(n, m)$ et fournit ainsi une statistique plus riche que le minimum de cet ensemble, qui est la

seule quantité considérée par la distance DTW. Ils prouvent également les résultats suivants :

- k_{GA} est définitivement positif si $\kappa/(1 + \kappa)$ est définitivement positif sur \mathcal{X}
- L'effort de calcul requis pour calculer k_{GA} évolue en $O(mn)$, similaire à la distance DTW. Plus précisément, la valeur de k_{GA} est égale à $M_{n, m}$ où les coefficients $M_{i, j}$ sont définis par les valeurs limites $M_{0, 0} = 1, M_{0, j} = M_{i, 0} = 0$ et la récurrence

$$M_{i, j} = \kappa(q_i, c_j)(M_{i-1, j-1} + M_{i, j-1} + M_{i-1, j})$$

Mais il y a des problèmes soulevés par les noyaux GA. Cuturi [2007] conjecture que k_{GA} produira des matrices Gram à dominance diagonale K (la matrice Gram K est la matrice de tous les produits scalaires possibles de vecteurs donnés.) sur la plupart des ensembles de données dans le sens où la somme de la magnitude des entrées hors diagonale de telles matrices Gram est beaucoup plus petite que leur trace. La dominance diagonale des matrices Gram est une propriété indésirable, car elle implique que tous les points dans un ensemble d'apprentissage sont orthogonaux les uns aux autres dans l'espace de caractéristiques correspondant. Cette conjecture a jeté des doutes sur l'applicabilité pratique des noyaux d'alignement global.

M [2011] a montré que la dominance diagonale peut en fait être évitée dans la plupart des cas pratiques.

Laissons κ être modifié pour incorporer un exposant $\lambda > 0$ c'est-à-dire $\kappa \stackrel{\text{def}}{=} e^{-\lambda\varphi}$. Ils ont montré que pour un

λ correctement choisi, les noyaux GA peuvent comparer des séquences tant qu'elles partagent des longueurs similaires, dans le sens où l'une n'est pas plus longue que deux fois la longueur de l'autre.

3. Noyaux d'Alignement Global Triangulaires Comme nous l'avons vu, Itakura (1975) et Sakoe & Chiba (1978) ont proposé d'accélérer le calcul des distances DTW grâce à des contraintes supplémentaires sur les alignements. Des idées similaires peuvent être appliquées aux noyaux GA comme le montre cette section.

Comme nous l'avons vu, l'algorithme DTW peut être accéléré en ne considérant qu'un petit sous-ensemble de tous les alignements comme détaillé par **Rabiner & Juang (1993, §4.7)**. Ces contraintes peuvent être formulées avec des poids $\gamma_{i, j}$ qui modifient la fonction de coût de l'Équation (2) en

$$D_{Q,C}^\gamma(w) = \sum_{k=1}^K \gamma_{w1_k, w2_k} \varphi(Q_{w1_k}, C_{w2_k}) \quad (3)$$

Par exemple, Sakoe & Chiba (1978) définissent la bande

$$\gamma_{i,j} = \begin{cases} 1, & \text{si } |i-j| < r \\ \infty, & \text{si } |i-j| \geq r \end{cases}$$

qui garantit que seuls les alignements w proches de la diagonale sont considérés. Cette augmentation de la vitesse vient au prix d'une approximation, puisque le chemin optimal résultant peut s'avérer être sous-optimal.

Les poids introduits dans l'Équation (3) peuvent être naturellement étendus aux noyaux GA. En effet, tout noyau définitivement positif ω sur \mathbf{N} , plutôt qu'un poids $\gamma_{i,j}$, peut être associé à un noyau κ sur \mathcal{X} pour former des noyaux GA :

Théorème : Soit κ un noyau sur \mathcal{X} et $\omega(i, j) = \left(1 - \frac{|i-j|}{r}\right)_+$ un noyau sur \mathbf{N} . Alors en utilisant $\tau^{-1}(\omega\kappa)$ ($\tau^{-1}(x) = \frac{x}{1-x}$) comme un noyau local, le noyau k_{GA} est définitivement positif et peut être calculé avec $O(T \min(n, m))$ opérations. (pour le détail de la preuve voir M [2011])

Nous définissons donc les noyaux TGA (Triangular GA) comme des noyaux GA obtenus en appariant le noyau triangulaire ω avec n'importe quel noyau local κ suivant la construction donnée dans le Théorème.

Considérant d'abord le noyau gaussien κ_σ , son logarithme :

$$\phi_\sigma(Q_{w1_k}, C_{w2_k}) \stackrel{\text{def}}{=} \frac{1}{2\sigma^2} \|Q_{w1_k} - C_{w2_k}\|^2 + \log \left(2 - e^{\frac{\|Q_{w1_k} - C_{w2_k}\|^2}{2\sigma^2}} \right) \quad (4)$$

est un noyau définitivement négatif qui peut être mis à l'échelle par un facteur pour définir un noyau local $e^{-\lambda\phi_\sigma}$ qui peut être facilement ajusté pour corriger toute dominance diagonale comme suggéré dans la section précédente.

Nous définissons un noyau TGA par

$$TGA(\sigma, r)(Q_{w1_k}, C_{w2_k}) = \tau^{-1}(\omega \frac{1}{2} \kappa_\sigma)(Q_{w1_k}, C_{w2_k}) = \frac{\omega(w1_k, w2_k) \kappa_\sigma(Q_{w1_k}, C_{w2_k})}{2 - \omega(w1_k, w2_k) \kappa_\sigma(Q_{w1_k}, C_{w2_k})}$$

La contrainte triangulaire est similaire aux contraintes de fenêtre qui peuvent être utilisées dans l'algorithme DTW. Lorsque $r \rightarrow \infty$, le noyau TGA converge vers le noyau GA original. Lorsque $r = 1$, le noyau TGA devient un noyau gaussien légèrement modifié qui ne peut comparer que des séries de longueur égale. Si $r > 1$, alors seuls les alignements qui remplissent $-r < w1_k - w2_k < r$ sont considérés.

[M, 2011] a également proposé une stratégie pour estimer la valeur de σ basée sur les séries temporelles elles-mêmes et leurs longueurs, à savoir $c \cdot \text{med}(\|Q - C\|) \cdot \sqrt{\text{med}(\|C\|)}$, où $\text{med}(\cdot)$ est la médiane empirique, c est une constante, et Q et C sont des vecteurs échantillonnés à partir de l'ensemble de données. Cela introduit cependant une certaine aléatoire dans l'algorithme lorsque la valeur de σ n'est pas fournie, il pourrait donc être préférable de l'estimer une fois et de la réutiliser dans les évaluations de fonction ultérieures.

La similarité retournée par le noyau TGA peut être normalisée avec l'Équation 5 afin que ses valeurs soient dans l'intervalle $[0, 1]$. L'algorithme prend en charge les séries multivariées et les séries de longueur différente. La distance résultante est symétrique et satisfait à l'inégalité

triangulaire, bien qu'elle soit plus coûteuse à calculer par rapport à DTW. Nous notons TGAK la similarité retournée par le noyau TGA.

$$\exp \left(\log(TGAK(Q, C, \sigma, r)) - \frac{\log(TGAK(Q, Q, \sigma, r)) + \log(TGAK(C, C, \sigma, r))}{2} \right)$$

2.4.3 DTW Doux

Poursuivant avec l'idée du noyau TGA, c'est-à-dire, de régulariser le DTW en le lissant, [Marco Cuturi, 2017] ont proposé un algorithme uniifié utilisant un minimum doux paramétré comme montré ci-dessous, et ont appelé l'écart résultant un DTW doux, qui est différentiable dans tous ses arguments.

Coûts d'alignement : Nous proposons dans cette section une formulation unifiée pour l'écart DTW original et le noyau d'Alignement Global, qui peuvent tous deux être utilisés pour comparer deux séries temporelles $Q = (q_1, \dots, q_n) \in \mathbf{R}^{p \times n}$ et $C = (c_1, \dots, c_m) \in \mathbf{R}^{p \times m}$. Pour un entier n nous écrivons $[n]$ pour l'ensemble $\{1, \dots, n\}$ d'entiers. Étant donné les longueurs de deux séries n et m , nous écrivons $\mathcal{A}_{n,m} \subset \{0, 1\}^{n \times m}$ pour l'ensemble des matrices d'alignement (binaires) qui remplissent les conditions nécessaires du DTW (dans la section DTW) et le schéma d'étape classique (symetric1 dans la section DTW).

Étant donné la matrice de coût local (LCM) $\Delta(Q, C)_{i,j} := d(q_i, c_j) \in \mathbf{R}^{n \times m}$, le produit intérieur $\langle A, \Delta(Q, C) \rangle$ de cette matrice avec une matrice d'alignement $A \in \mathcal{A}_{n,m}$ donne le score de A . Le DTW et le GAK considèrent les coûts de toutes les matrices d'alignement possibles, mais le font différemment :

$$DTW(Q, C) := \min_{A \in \mathcal{A}_{n,m}} \langle A, \Delta(Q, C) \rangle, k_{GA}^{\gamma} := \sum_{A \in \mathcal{A}_{n,m}} e^{-\langle A, \Delta(Q, C) \rangle / \gamma} \quad (1)$$

Les deux formules dans l'Eq. (1) peuvent être calculées avec un seul algorithme. Considérons l'opérateur généralisé min suivant, avec un paramètre de lissage $\gamma \geq 0$:

$$\min\{a_1, \dots, a_n\} := \begin{cases} \min_{i \leq n} a_i, & \gamma = 0 \\ -\gamma \log \sum_{i=1}^n e^{-a_i / \gamma} & \gamma > 0. \end{cases}$$

Avec cet opérateur, nous pouvons définir le γ -DTW doux :

$$dtw_{\gamma}(Q, C) := \min\{\langle A, \Delta(Q, C) \rangle, A \in \mathcal{A}_{n,m}\}.$$

Le score DTW original est récupéré en réglant γ à 0. Lorsque $\gamma > 0$, nous récupérons $dtw_{\gamma} = -\gamma \log k_{GA}$.

Notez que, pour assurer la stabilité numérique, l'opérateur \min^{γ} doit être calculé en utilisant l'astuce habituelle de stabilisation log-sum-exp.

Malgré la considération de tous les alignements et non juste du meilleur, le DTW doux peut être calculé avec une modification mineure de la récursion de Bellman, dans laquelle toutes les opérations ($\min, +$) sont remplacées par $(+, \times)$. En conséquence, le DTW et le DTW doux ont une complexité en temps quadratique & linéaire en espace par rapport aux longueurs des séquences.

Algorithme1: Récursion avant pour calculer dtw_gamma (Q, C)

et les coûts d'alignement intermédiaires

Entrées : Q , C , lissage gamma ≥ 0 , fonction de distance d

$r[0,0] = 0$; $r[i,0] = r[0,j] = \inf$; i dans $[n]$, j dans $[m]$.

Sortie : $(r[n,m], R)$

```

pour j = 1, ..., m faire
    pour i = 1, ..., n faire
        r [i,j] = d(q i , c j ) + min^gamma{r [i-1,j-1] , r[i-1,j] , r[i,j-1] }
    fin pour
fin pour
    
```

Différentiation du DTW doux Une petite variation de l'entrée x provoque un petit changement dans $dtw_0(Q, C)$ ou $dtw_\gamma(Q, C)$. Lorsqu'on considère dtw_0 , ce changement peut être efficacement suivi uniquement lorsque la matrice d'alignement optimale A^* qui apparaît lors du calcul de $dtw_0(Q, C)$ dans l'Éq. (1) est unique. Comme le minimum sur un ensemble fini de fonctions linéaires de Δ , dtw_0 est donc localement différentiable par rapport à la matrice de coûts Δ , avec le gradient A^* . Pour récupérer le gradient de $dtw_0(Q, C)$ par rapport à Q , il suffit d'appliquer la règle de chaîne, grâce à la différentiabilité de la fonction de coût :

$$\nabla_Q dtw_0(Q, C) = \left(\frac{\partial \Delta(Q, C)^T}{\partial Q} \right) A^*$$

où $\partial \Delta(Q, C)/\partial Q$ est le Jacobien de Δ par rapport à Q , une application linéaire de $\mathbf{R}^{p \times n} \rightarrow \mathbf{R}^{n \times m}$. Lorsque d est la distance euclidienne au carré, le transposé de ce Jacobien appliqué à une matrice $B \in \mathbf{R}^{n \times m}$ est (\cdot étant le produit élément par élément) :

$$(\partial \Delta(Q, C)/\partial Q)^T B = 2((\mathbf{1}_p \mathbf{1}_m^T B^T) \cdot Q - CB^T).$$

Lorsque $\gamma > 0$, le gradient de l'Éq. (1) est obtenu via la règle de chaîne,

$$\nabla_Q dtw_\gamma(Q, C) = \left(\frac{\partial \Delta(Q, C)^T}{\partial Q} \right) \mathbf{E}_\gamma A^* \quad \text{où} \quad \mathbf{E}_\gamma A^* := \frac{1}{k_{GA}^\gamma} \sum_{A \in \mathcal{A}_{n,m}} e^{-\langle A, \Delta(Q, C) \rangle / \gamma} \quad (2)$$

Pour calculer le gradient du DTW doux, nous proposons à la place un algorithme qui reste quadratique (nm) en termes de complexité. La clé pour atteindre cette réduction est d'appliquer la règle de chaîne dans l'ordre inverse de la récursion de Bellman donnée dans l'Algorithme 1, à savoir rétro-propager.

Différencier algorithmiquement $dtw_\gamma(Q, C)$ nécessite d'abord un passage en avant de l'équation de Bellman pour stocker tous les calculs intermédiaires et récupérer $R = [r_{i,j}]$ lors de l'exécution de l'Algorithme 1. La valeur de $dtw_\gamma(Q, C)$ — stockée dans $r_{n,m}$ à la fin de la récursion en avant — est ensuite impactée par un changement dans $r_{i,j}$ exclusivement à travers les termes dans lesquels $r_{i,j}$ joue un rôle, à savoir le triplet de termes $r_{i+1,j}, r_{i,j+1}, r_{i+1,j+1}$. Une application simple de la règle de chaîne donne alors :

$$\frac{\partial r_{n,m}}{\partial r_{i,j}} = \frac{\partial r_{n,m}}{\partial r_{i+1,j}} \frac{\partial r_{i+1,j}}{\partial r_{i,j}} + \frac{\partial r_{n,m}}{\partial r_{i,j+1}} \frac{\partial r_{i,j+1}}{\partial r_{i,j}} + \frac{\partial r_{n,m}}{\partial r_{i+1,j+1}} \frac{\partial r_{i+1,j+1}}{\partial r_{i,j}}$$

dans lequel nous avons défini la notation de l'objet principal d'intérêt de la récursion

arrière : $e_{i,j} := \partial r_{n,m}/\partial r_{i,j}$. La récursion de Bellman évaluée à $(i+1, j)$ comme indiqué à la ligne 9 de l'Algorithme 1 (ici $d(i+1, j)$ est $d(q_{i+1}, c_j)$) donne :

$$r_{i+1,j} = d(i+1, j) + \min\{r_{i,j-1}, r_{i,j}, r_{i+1,j-1}\},$$

qui, lorsqu'il est différentié par rapport à $r_{i,j}$, donne le ratio :

$$\frac{\partial r_{i+1,j}}{\partial r_{i,j}} = e^{-r_{i,j}/\gamma} / (e^{-r_{i,j-1}/\gamma} + e^{-r_{i,j}/\gamma} + e^{-r_{i+1,j-1}/\gamma})$$

Le logarithme de cette dérivée peut être commodément exprimé en utilisant les évaluations de \min^γ calculées dans la boucle en avant :

$$\gamma \log \frac{\partial r_{i+1,j}}{\partial r_{i,j}} = \min\{r_{i,j-1}, r_{i,j}, r_{i+1,j-1}\} - r_{i,j} = r_{i+1,j} - d_{i+1,j} - r_{i,j}.$$

De même, les relations suivantes peuvent également être obtenues :

$$\gamma \log \frac{\partial r_{i,j+1}}{\partial r_{i,j}} = r_{i,j+1} - d_{i,j+1} - r_{i,j}. \gamma \log \frac{\partial r_{i+1,j+1}}{\partial r_{i,j}} = r_{i+1,j+1} - d_{i+1,j+1} - r_{i,j}.$$

Nous avons donc obtenu une récursion arrière pour calculer la matrice entière $E = [e_{i,j}]$, en commençant par $e_{n,m} = \partial r_{n,m}/\partial r_{n,m} = 1$ jusqu'à $e_{1,1}$. Pour obtenir $\nabla_Q dtw_\gamma(Q, C)$, notez que les dérivées par rapport aux entrées de la matrice de coût local Δ peuvent être calculées par $\frac{\partial r_{n,m}}{\partial d_{i,j}} = \frac{\partial r_{n,m}}{\partial r_{i,j}} \frac{\partial r_{i,j}}{\partial d_{i,j}} = e_{i,j}$, et donc nous avons que :

$$\nabla_Q dtw_\gamma(Q, C) = \left(\frac{\partial \Delta(Q, C)^T}{\partial Q} \right) E$$

où E est exactement l'alignement moyen $\mathbf{E}_\gamma A$ dans l'Eq. (2). Ces calculs sont résumés dans l'Algorithme 2, qui, une fois Δ calculé, a une complexité $O(nm)$ en temps et en espace. Puisque \min_γ a un gradient continu $1/\gamma$ -Lipschitz, le gradient de dtw_γ est $2/\gamma$ -Lipschitz continu lorsque d est la distance euclidienne au carré.

Algorithme 2 Récursion arrière pour calculer DeltaQ dtw_gamma (Q, C)

Entrées : Q, C, lissage gamma \$\geq 0, fonction de distance d

(., R) = dtw_gamma(Q, C), Delta = [d(qi , cj)]_{i,j}

Sortie : E

d [i,m+1] = d [n+1,j] = 0, i dans [n], j dans [m]

e [i,m+1] = e [n+1,j] = 0, i dans [n], j dans [m]

r [i,m+1] = r [n+1,j] = -Infini, i dans [n], j dans [m]

d [n+1,m+1] = 0 , e[n+1,m+1] = 1 , r[n+1, m+1] = r[n,m]

pour j = m, ..., 1 faire

pour i = n, ..., 1 faire

a = exp (1/gamma * (r [i+1,j] - r [i,j] - d[i+1,j]))

b = exp (1/gamma * (r [i,j+1] - r [i,j] - d [i,j+1]))

c = exp (1/gamma * (r [i+1,j+1] - r [i,j] - d [i+1,j+1]))

e [i,j] = e[i+1,j] * a + e[i,j+1] * b + e[i+1,j+1] * c

fin pour

fin pour

Moyennage avec la géométrie du DTW doux Ceci peut ensuite être utilisé pour calculer des centroïdes avec une optimisation numérique.

Le calcul (approximatif) de barycentres dtw_γ peut être vu comme une première étape vers la tâche de regroupement de séries temporelles sous l'écart dtw_γ . En effet, on peut naturellement formuler ce problème comme celui de trouver des centroïdes Q_1, \dots, Q_k dans N séries temporelles $\{C_i\}_{i \in [N]}$ qui minimisent l'énergie suivante :

$$\min_{Q_1, \dots, Q_k} \sum_{i=1}^N \frac{1}{m_i} \min_{j \in [k]} dtw_\gamma(Q_j, C_i)$$

Pour résoudre ce problème, on peut recourir à une généralisation directe de l'algorithme de Lloyd (1982) dans lequel chaque étape de centrage et chaque étape d'allocation de clustering est effectuée selon l'écart dtw_γ .

Pour apprendre les centroïdes au sein de chaque groupe, nous pourrions utiliser un algorithme de minimum normal (Newton...) pour cette fonction prototype :

$$\min_{x \in \mathbb{R}^{n,m}} \sum_{i=1}^{N_j} \frac{1}{m_i} dtw_\gamma(x, C_i)$$

2.4.4 PDC

Nous présentons ici une distance basée sur la complexité , regroupement par distribution de permutation [Markus, 2012] .

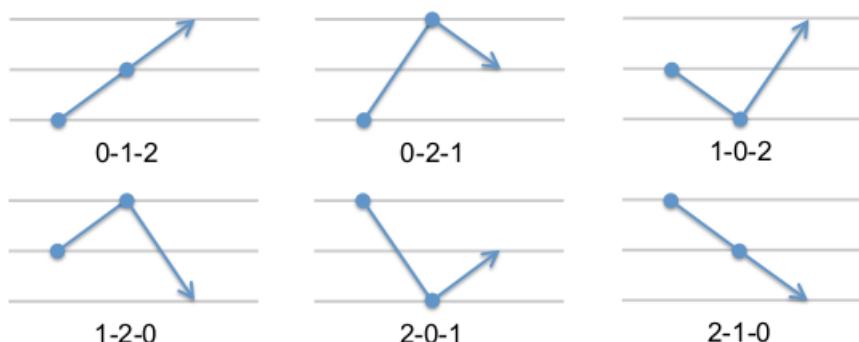


FIGURE 13 – Chaque segment de série temporelle (flèches bleues) est un représentant de l'un des six motifs ordinaux distincts pour une taille de trois. Par exemple, le premier modèle ordinal (0-1-2) englobe tous les modèles pour lesquels la première valeur du segment est la plus petite et la dernière la plus grande, indépendamment de la taille du segment.

Dans ce qui suit, le calcul de la PD (Distribution de Permutation) et la construction d'une mesure de dissimilarité entre les séries temporelles basée sur leur représentation PD sont décrits. En bref, le PD attribue une probabilité à l'occurrence de certains motifs des rangs de valeurs dans une série temporelle. À cette fin, la série temporelle est divisée en sous-séquences d'une longueur fixe, m , qui est également appelée plongement en m -espace. Pour calculer la PD sur des échelles de temps plus grossières, le plongement peut être retardé avec un délai t , de sorte que seul chaque élément t est enregistré lors de la formation des sous-séquences. Pour chaque sous-séquence, les rangs des valeurs sont calculés, par exemple, par produit de tri des valeurs observées. Le rang de distribution de permutation est obtenu

en comptant la fréquence relative des motifs de rang distincts, également appelés motifs ordinaux. Chaque motif de rang possible peut être identifié par une permutation des valeurs de 0 à $m - 1$ et d'où le nom de distribution de permutation. La dépendance aux rangs est une caractéristique distincte du PD car la distribution est déterminée par les valeurs observées relatives les unes aux autres au lieu de leurs valeurs absolues. La figure 13 illustre des motifs ordinaux pour un plongement de $m = 3$.

Étant donnée une série temporelle $X = \{x(i)\}_{i=1}^T$ échantillonnée à des intervalles égaux avec $x(i) \in \mathbf{R}$, le plongement retardé de cette série temporelle dans un espace à m dimensions avec un délai de temps t , est $X' = \{x(i), x(i+t), x(i+2t), \dots, x(i+(m-1)t)\}_{i=1}^{T'}$ avec un total de $T' = T - (m-1)t$ éléments. Le motif ordinal pour un élément $x' \in X'$ peut être obtenu en calculant la permutation des indices de 0 à $(m-1)$ qui place les m valeurs dans un ordre trié. Si deux éléments $x'(i), x'(j)$, $i \neq j$ ont la même valeur, ils conserveront leur ordre original relatif l'un à l'autre. Il existe $m!$ permutations uniques de longueur m , et donc, $m!$ motifs ordinaux distincts.

La distribution de permutation d'une série temporelle est obtenue en comptant les fréquences des motifs ordinaux distincts observés des éléments $x' \in X'$. Soit $\Pi(x)$ la permutation qu'un élément $x \in \mathbf{R}^m$ subit lorsqu'il est trié. La distribution de permutation de X' est

$$p_\pi = \frac{\#\{x' \in X' | \Pi(x') = \pi\}}{T'}$$

Dans le PD, l'ordre temporel des motifs ordinaux est ignoré et la distribution repose uniquement sur la fréquence des motifs uniques dans la série temporelle. La permutation d'ordre $m \geq 2$ est définie comme l'entropie de Shannon de la distribution de probabilité P :

$$H(P) = - \sum_{\pi \in S_m} p_\pi \log p_\pi$$

avec S_m étant l'ensemble de toutes les permutations de m .

La dissimilarité entre deux séries temporelles peut être formalisée comme une dissimilarité de leurs distributions de permutation respectives. La divergence de Kullback-Leibler (KL), également connue sous le nom d'entropie relative de Shannon, est souvent utilisée comme mesure de divergence entre les distributions de probabilité et représente une expansion naturelle de l'entropie comme indice de complexité. Cependant, la divergence KL viole l'inégalité triangulaire et, par conséquent, n'est pas une métrique. Par conséquent, nous employons la distance de Hellinger au carré pour incorporer les distributions de permutation des séries temporelles dans un espace métrique. La distance de Hellinger au carré est :

$$D(P, Q) = \frac{1}{\sqrt{2}} \sqrt{\sum_{i=1}^n (\sqrt{p_i} - \sqrt{q_i})^2}$$

où $P = (p_1, p_2, \dots, p_n)$ et $Q = (q_1, q_2, \dots, q_n)$.

La distance de Hellinger au carré peut être dérivée au moyen d'une approximation de Taylor de la divergence KL. Il s'agit d'une métrique puisqu'elle satisfait à l'inégalité triangulaire, qu'elle est symétrique, non négative et limitée entre zéro et un.

Heuristique de l'entropie Le choix des paramètres m (dimension de plongement) et t (délai temporel) pour le plongement retardé nécessite la prise en compte de deux effets contraires : plus la dimension de plongement est grande, plus le pouvoir de représentation est important, c'est-à-dire que plus de motifs ordinaux distincts peuvent être observés, et ainsi, l'estimation de la fréquence sera de plus en plus fiable. Dans la suite, nous formalisons ces observations dans une heuristique qui guide le choix de l'analyse en dimensionnant si aucune

information préalable n'est disponible. L'heuristique vise à maximiser l'expressivité de la dimension, c'est-à-dire que le PD devrait être aussi uniformément réparti que possible. Cette notion est capturée par l'entropie de permutation. Soit P une distribution de permutation de dimension de plongement m . Soit $\log_0(x) = 0$ si $x = 0$, et $\log(x)$ autrement. L'entropie normalisée de la distribution de permutation est

$$e_P(m) = -\frac{\sum_{\pi \in S_m} p_\pi \log p_\pi}{\log(m!)}$$

L'heuristique choisit la dimension de plongement qui minimise l'entropie normalisée moyenne de l'ensemble des distributions. Pour trouver la dimension de plongement appropriée pour le clustering d'une série de temps, nous calculons l'entropie normalisée moyenne d'un ensemble de distributions. De manière similaire, différents choix de délai temporel t peuvent être sélectionnés. Une illustration d'une recherche en grille sur les paramètres m et t détermine les différences entre deux processus autorégressifs moyens mobiles (ARMA) est donnée à la Figure 14.

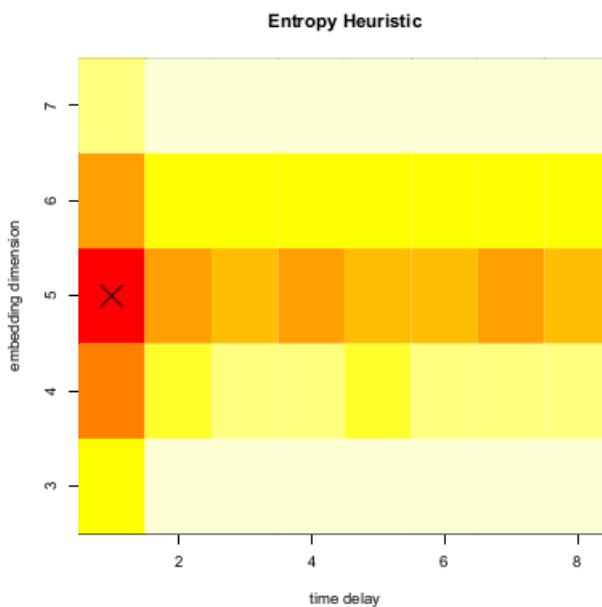


FIGURE 14 – Représentation graphique des estimations de l'entropie moyenne pour des délais et des dimensions d'intégration variables sur un ensemble de données avec des prototypes générés à partir de deux processus ARMA différents.

2.4.5 D'autre distance

Dans notre partie expérimentale, nous choisissons également des méthodes telles que le CORT [Douzal Chouakria A, 2007], l'ACF [Galeano P, 2000], les distances basées sur le périodogramme [Caiado J, 2006] [de Lucas D, 2010], etc. pour enrichir notre analyse du regroupement des séries temporelles. Emprunter les coefficients de corrélation, les coefficients d'autocorrélation ou transformer en analyse spectrale dans le domaine des fréquences. Nous ne les présenterons pas toutes pour tenir compte de l'espace limité dont nous disposons ici.

3 Prototypes des clusters

Il existe quelques méthodes de calcul des prototypes dans la littérature des séries temporelles, mais la plupart de ces publications n'ont pas prouvé l'exactitude de leurs méthodes V. Niennattrakul [2007]. Mais, en général, trois approches peuvent être observées pour définir les prototypes :

1. La séquence médiane de l'ensemble.
2. La séquence moyenne de l'ensemble.
3. Le prototype de recherche locale.

3.1 Nos principales méthodes de recherche

3.1.1 DBA

La distance DTW est très souvent utilisée lors du travail avec des séries temporelles, et donc une fonction de prototypage basée sur le DTW a également été développée dans [François Petitjean a, 2011]. La procédure est appelée moyenne de barycentre DTW (DBA, pour DTW Barycenter Averaging) et est une méthode globale itérative. Cela signifie que l'ordre dans lequel les séries entrent dans la fonction de prototypage n'affecte pas le résultat. Elle est disponible comme une fonction autonome dans dtwclust, simplement nommée DBA.

Techniquement, pour chaque raffinement, c'est-à-dire, pour chaque itération, DBA fonctionne en deux étapes :

- Calcul du DTW entre chaque séquence individuelle et la séquence moyenne temporaire à affiner, afin de trouver des associations entre les coordonnées de la séquence moyenne et les coordonnées de l'ensemble des séquences.
- Mise à jour de chaque coordonnée de la séquence moyenne comme le barycentre des coordonnées qui lui sont associées lors de la première étape.

Soit $Q = \{Q_1, \dots, Q_N\}$ l'ensemble des séquences à moyenner, soit $C = < C_1, \dots, C_T >$ la séquence moyenne à l'itération i et soit $C' = < C'_1, \dots, C'_T >$ la mise à jour de C à l'itération $i + 1$, dont nous voulons trouver les coordonnées. De plus, chaque coordonnée de la séquence moyenne est définie dans un espace vectoriel arbitraire \mathcal{X} (par exemple, habituellement un espace euclidien) : $\forall t \in [1, T], C_t \in \mathcal{X}$.

Nous considérons la fonction assoc, qui lie chaque coordonnée de la séquence moyenne à une ou plusieurs coordonnées des séquences de Q . Cette fonction est calculée lors du calcul du DTW entre C et chaque séquence de Q . La t -ième coordonnée de la séquence moyenne C'_t est alors définie comme $C'_t = \text{barycentre}(\text{assoc}(C_t))$ où $\text{barycentre}\{Q_1, \dots, Q_N\} = \frac{1}{N} \sum_1^N Q_N$

Algorithme5 DBA

```

Entrée : C la séquence moyenne initiale
         Q = {Q_1, ..., Q_N} ensemble de séquence
Sortie : C1
Début
Soit T la longueur de séquence de l'ensemble Q
Soit T1 la longueur de C
Soit assocTab un tableau de taille T1 contenant
dans chaque cellule un ensemble de coordonnées associées à chaque coordonnée de C
Soit m[T, T] une matrice DTW temporaire (coût, chemin)

assocTab <- vide .
pour seq dans Q faire

```

```

m <- DTW(C,seq) 'retourne une liste d'index du DTW'
i <- T1
j <- T
tant que i >=1 et j >=1 faire
    assocTab[i] <- [assocTab[i] , seq_j]
    (i,j) <- seconde (m(i,j))
fin tant que
fin pour
pour i=1 à T faire
    C1[i] = barycentre(assocTab[i])
fin pour

```

fin

Pour l'initialisation, deux facteurs principaux sont à considérer lors de l'amorçage du raffinement itératif : premièrement la longueur de la séquence moyenne de départ, et deuxièmement les valeurs de ses coordonnées.

Si la mise à jour ne modifie pas l'alignement des séquences, alors le théorème de Huygens s'applique ; les barycentres composant la séquence moyenne se rapprocheront des coordonnées de Q. Dans l'autre cas, si l'alignement est modifié, cela signifie que le DTW calcule un meilleur alignement avec une inertie plus petite (qui diminue également dans ce cas). Nous avons donc une garantie de convergence.

Nous détaillerons la complexité temporelle de DBA. Chaque itération du processus itératif est divisée en deux parties :

- Calcul du DTW entre chaque séquence individuelle et la séquence moyenne temporaire (c'est-à-dire, actuelle), pour trouver des associations entre ses coordonnées et celles des séquences. $\Theta(N * T * T1)$
- Mise à jour de la moyenne selon les associations juste calculées. $\Theta(N * T)$

3.1.2 K-shape

[John Paparrizos, 2015] propose K-shape, un nouvel algorithme de clustering basé sur les centroïdes qui peut préserver les formes des séquences de séries temporelles. C'est un algorithme indépendant du domaine, précis et scalable pour le clustering de séries temporelles, avec une mesure de distance qui est invariante à la mise à l'échelle et au décalage.

Mesure de corrélation croisée La corrélation croisée est une mesure statistique avec laquelle nous pouvons déterminer la similarité de deux séquences $Q = (q_1, \dots, q_m)$ et $C = (c_1, \dots, c_m)$, pour simplifier, nous considérons des séquences de longueur égale même si la corrélation croisée peut être calculée sur des séquences de longueurs différentes. Pour atteindre l'invariance de décalage, la corrélation croisée garde C statique et fait glisser Q sur C pour calculer leur produit intérieur pour chaque décalage de Q.

$$CC_w(Q, C) = R_{w-m}(Q, C) \quad w \in \{1, 2, \dots, 2m - 1\} \quad (1)$$

où $R_{w-m}(Q, C)$ est calculé, à son tour, comme :

$$R_k(Q, C) = \begin{cases} \sum_{l=1}^{m-k} q_{l+k} \cdot c_l & k \geq 0 \\ R_{-k}(C, Q) & k < 0 \end{cases} \quad (2)$$

Notre objectif est de calculer la position w à laquelle $CC_w(Q, C)$ est maximisé.

SBD (distance basée sur la forme) Pour concevoir une mesure de distance basée sur la forme, nous utilisons la normalisation du coefficient qui donne des valeurs entre -1 et 1, indépendamment de la normalisation des données. nous dérivons la mesure de distance suivante :

$$SBD(Q, C) = 1 - \max_w \left(\frac{CC_w(Q, C)}{\sqrt{R_0(Q, Q) \cdot R_0(C, C)}} \right)$$

qui prend des valeurs entre 0 et 2, avec 0 indiquant une similarité parfaite pour les séquences de séries temporelles. Jusqu'à présent, nous avons abordé l'invariance de décalage. Pour l'invariance d'échelle, nous transformons chaque séquence Q en $Q_0 = (Q - \mu)/\sigma$, de sorte que sa moyenne μ est zéro et son écart-type σ est un.

Le théorème de convolution affirme que la convolution de deux séries temporelles peut être calculée comme la Transformée de Fourier Discrète Inverse (IDFT) du produit des Transformées de Fourier Discrètes (DFT) individuelles des séries temporelles, où DFT est :

$$F(q_k) = \sum_{r=0}^{m-1} q_r e^{-\frac{2jrk\pi}{m}}, \quad k = 0, \dots, m-1$$

et IDFT est :

$$F^{-1}(q_r) = \frac{1}{m} \sum_{k=0}^{m-1} F(q_k) e^{\frac{2jrk\pi}{m}} \quad r = 0, \dots, m-1$$

où $j = \sqrt{-1}$. L'Équation 1 peut être calculée pour chaque m comme :

$$CC(Q, C) = F^{-1}\{F(Q)F(C)\}$$

En utilisant un algorithme de Transformée de Fourier Rapide (FFT), le temps se réduit à $O(m \log(m))$.

Extraction de la Forme des Séries Temporelles L'extraction de centroïdes significatifs est une tâche difficile qui dépend de manière critique du choix de la mesure de distance. Nous montrons maintenant comment déterminer de tels centroïdes pour le clustering de séries temporelles pour la mesure de distance SBD, afin de capturer les caractéristiques partagées des données sous-jacentes.

nous pouvons exprimer la séquence calculée comme le maximiseur μ_k^* des similarités au carré avec toutes les autres séries temporelles. nous obtenons :

$$\mu_k^* = \operatorname{argmax}_{u_k} \sum_{Q_i \in P_k} \left\{ \frac{\max_w CC_w(Q_i, \mu_k)}{\sqrt{R_0(Q_i, Q_i) \cdot R_0(\mu_k, \mu_k)}} \right\}^2 \quad (3)$$

L'Équation 3 nécessite le calcul d'un décalage optimal pour chaque $Q_i \in P_k$. Comme cette approche est utilisée dans le contexte du clustering itératif, nous utilisons le centroïde précédemment calculé comme référence et alignons toutes les séquences vers cette séquence de référence. C'est un choix raisonnable car le centroïde précédent sera très proche du nouveau centroïde. Pour cet alignement, nous utilisons SBD, qui identifie un décalage optimal pour

chaque $Q_i \in P_k$. Par la suite, comme les séquences sont déjà alignées vers une séquence de référence avant le calcul du centroïde, nous pouvons également omettre le dénominateur de l'Équation 3. En combinant les Équations 1 et 2, nous obtenons :

$$\mu_k^* = \operatorname{argmax}_{u_k} \sum_{Q_i \in P_k} \left\{ \sum_{l=1}^m Q_{il} \cdot \mu_{kl} \right\}^2$$

Pour simplifier, nous exprimons cette équation avec des vecteurs et supposons que les séquences Q_i ont déjà été z-normalisées pour gérer les différences d'amplitude :

$$\mu_k^* = \operatorname{argmax}_{u_k} \sum_{Q_i \in P_k} (Q_i^T \cdot \mu_k)^2 = \operatorname{argmax}_{u_k} \mu_k^T \sum_{Q_i \in P_k} (Q_i \cdot Q_i^T) \cdot \mu_k \quad (4)$$

Dans l'équation précédente, seul μ_k n'est pas z-normalisé. Pour gérer le centrage (c'est-à-dire, la soustraction de la moyenne) de μ_k , nous définissons $\mu_k = \mu_k \cdot K$, où $K = I - \frac{1}{m}O$, I est la matrice identité, et O est une matrice avec tous les uns. De plus, pour que u_k ait une norme unitaire, nous divisons l'Équation 4 par $\mu_k^T \cdot \mu_k$. Finalement, en substituant S pour $\sum_{Q_i \in P_k} (Q_i \cdot Q_i^T)$, nous obtenons :

$$\mu_k^* = \operatorname{argmax}_{\mu_k} \frac{u_k^T \cdot K^T \cdot S \cdot K \cdot \mu_k}{\mu_k^T \cdot \mu_k} = \operatorname{argmax}_{\mu_k} \frac{u_k^T \cdot M \cdot \mu_k}{\mu_k^T \cdot \mu_k} \quad (5)$$

où $M = K^T \cdot S \cdot K$. À travers les transformations ci-dessus, nous avons réduit l'optimisation de l'Équation 5 à l'optimisation de l'Équation 15, qui est un problème bien connu appelé maximisation du Quotient de Rayleigh. Nous pouvons trouver le maximiseur μ_k^* comme le vecteur propre correspondant à la plus grande valeur propre de la matrice réelle symétrique M .

L'Algorithme 2 montre comment nous pouvons extraire la forme la plus représentative des données sous-jacentes en quelques lignes de code.

Algorithme 2 : Extraction de Forme

Entrée : Q , une matrice n-par-m avec des séries temporelles z-normalisées.
 C, un vecteur 1-par-m avec la séquence de référence contre laquelle les séries temporelles de X sont alignées.
 Sortie : $C1$, un vecteur 1-par-m avec le centroïde.
 début

```

Q1 <- [ ]
pour i <- 1 à n faire
    [dist, q] <- SBD(C , Q(i))           // Algorithme 1
    Q1 <- [Q1 ; q]
fin pour
S <- Q1^T * Q1
K <- I - 1/m *0
M <- K^T * S * K
C1 <- Eig(M, 1 )

fin
    
```

Clustering de Séries Temporelles Basé sur la Forme K-Shape est une méthode de clustering partitive basée sur une procédure de raffinement itératif similaire à celle utilisée dans k-means.

K-Shape (voir Algorithme 3) attend en entrée l'ensemble de séries temporelles Q et le nombre de clusters k que nous voulons produire. Initialement, nous attribuons aléatoirement les séries temporelles de Q aux clusters. Ensuite, nous calculons chaque centroïde de cluster en utilisant l'Algorithme 2 (lignes 5-10). Une fois les centroïdes calculés, nous affinons les appartenances des clusters en utilisant la mesure de distance SBD (Algorithme 1) aux lignes 11-17. Nous répétons cette procédure jusqu'à ce que l'algorithme converge ou atteigne le maximum.

Algorithme 3: K-shape

Entrée : Q est une matrice n-par-m contenant n séries temporelles de longueur m qui sont initialement z-normalisées.

k est le nombre de clusters à produire.

Sortie : IDX est un vecteur n-par-1 contenant l'assignation de n séries temporelles à k clusters (initialisé aléatoirement).

C est une matrice k-par-m contenant k centroïdes de longueur m (initialisés comme des vecteurs avec tous zéros).

début

```

iter <- 0
IDX1 <- [ ]
tant que IDX != IDX1 et iter < 100 faire
    IDX1 <- IDX // Étape de raffinement
    pour j <- 1 à k faire
        Q1 <- [ ]
        pour i <- 1 à n faire
            si IDX(i) = j alors
                Q1 <- [Q1 ; Q(i)]
            fin si
        fin pour
        C(j) <- ExtractionForme(Q1,C(j))
    fin pour //étape d'assignation
    pour i <- 1 à n faire
        mindist <- Inf
        pour j <- 1 à k faire
            [dist, q1] <- SBD(C(j), Q(i))
            si dist < mindist alors
                mindist <- dist
                IDX(i) <- j
            fin si
        fin pour
    fin pour
    iter <- iter + 1
fin tant que
fin
    
```

K-Shape évolue linéairement avec le nombre de séries temporelles. Pour comprendre pourquoi, nous analyserons la complexité computationnelle de l'Algorithme 3, où n est le nombre

de séries temporelles, k est le nombre de clusters et m est la longueur des séries temporelles. Dans l'étape d'assignation, K-Shape calcule la dissimilarité de n séries temporelles à k centroïdes en utilisant SBD, qui nécessite $O(m \cdot \log(m))$ temps. Ainsi, la complexité temporelle de cette étape est $O(n \cdot k \cdot m \cdot \log(m))$. Dans l'étape de raffinement, pour chaque cluster, K-Shape calcule la matrice M, qui nécessite $O(m^2)$ temps, et effectue une décomposition en valeurs propres sur M, qui nécessite $O(m^3)$ temps. Ainsi, la complexité de cette étape est $O(\max n \cdot m^2, k \cdot m^3)$. Au total, K-Shape nécessite $O(\max n \cdot k \cdot m \cdot \log(m), n \cdot m^2, k \cdot m^3)$ temps par itération.

4 Algorithme

4.1 Regroupement hiérarchique de séries temporelles

4.2 Partitionnement en groupe

k-Means and k-Medoids, FCM (Fuzzy c-Means) algorithm and Fuzzy c-Medoids algorithm

4.3 Regroupement basé sur un modèle

Généralement, les méthodes basées sur un modèle utilisent soit des approches statistiques, par exemple COBWEB , soit des approches de réseaux neuronaux, par exemple ART ou Self-Organization Map .

4.4 Regroupement basé sur la densité

L'un des algorithmes les plus connus qui fonctionne sur la base du concept de densité est DBSCAN. OPTICS [188] est un autre algorithme basé sur la densité qui la question de la détection de groupes significatives dans des données de densité variable.

4.5 Regroupement basé sur une grille

STING et Wave Cluster sont deux exemples typiques d'algorithmes de regroupement basés sur le concept de grille. mais il n'existe pas de travaux dans la littérature appliquant des approches basées sur les grilles pour le regroupement de séries temporelles.

4.6 Regroupement en plusieurs étapes

4.7 Nos principales méthodes de recherche

4.7.1 TADpole

Le clustering TADPole a été proposé dans [Nurjahan Begum, 2015] et est implémenté dans dtwclust avec la fonction TADPole.

La solution proposée s'inspire de DP, l'algorithme de clustering basé sur la densité récemment proposé par [Saeed, 2011], afin que l'algorithme DP original puisse regrouper un ensemble de données.

Dans l'algorithme TADPole, ils enrichissent le cadre de clustering DP et exploitent les bornes supérieures et inférieures de DTW pour élaguer les calculs de distance inutiles, résultant en une accélération d'au moins un ordre de grandeur. Les seuls paramètres nécessaires sont la distance de coupure (d_c) et, optionnellement, le nombre de clusters (k).

L'algorithme DP L'algorithme DP suppose que les centres des clusters sont entourés de voisins de densité locale inférieure et sont à une distance relativement plus élevée de tout point ayant une densité locale supérieure. Par conséquent, pour chaque point i dans l'ensemble de données, l'algorithme DP calcule deux quantités :

- Densité locale ρ_i .
- Distance par rapport aux points de densité locale supérieure (d_i).

Pour un point de données i , sa densité locale ρ_i peut être calculée par la formule suivante :

$$\rho_i = \sum_j \mathbb{1}_{d_{ij} < d_c}$$

De plus, l'algorithme DP définit également une variable d_i , qui représente la distance du nœud i au nœud voisin j le plus proche dont la densité est supérieure à celle du nœud i . d_i est défini comme l'équation :

$$d_i = \begin{cases} \min d_{ij} & \text{si } \exists j, \rho_j > \rho_i \\ \max d_k & \text{sinon } k \in \text{tous les nœuds}/i \end{cases}$$

Voici les algorithmes pour calculer ρ_i et d_i .

Algorithme : Calcul de la Densité Locale

Entrée : D , matrice de distance de toutes les paires
dc, distance de coupure

Sortie : rho, le vecteur de densité locale pour tous les n points dans l'ensemble de données

début

```

pour i = 1 à n
    rho(i) = compter (D(i, autresObjets) < dc)
fin

```

fin

Algorithme : Distance aux Points de Densité Supérieure

Entrée : D , matrice de distance de toutes les paires
rho, le vecteur de densité locale

Sortie : d , vecteur de distance du Plus Proche Voisin (PPV) des points de densité supérieure

début

```

[liste_triee_d, indiceTri] = trier(rho, 'descend')
# liste_triee_d : tableau des points de densité supérieure
pour j = 2 à n
    d(indiceTri(j)) = PPVDist(liste_triee_d(j))
fin pour
d(indiceTri(1)) = max(d(2:n))

```

fin

Dans la ligne 6, cette liste est triée en ordre décroissant (rho). De la ligne 6 à 8, pour chaque point dans l'ordre trié, les distances du Plus Proche Voisin de leurs points de densité supérieure sont calculées. Pour le cas spécial du point de densité la plus élevée (qui par définition n'a pas de voisin de densité supérieure), cette distance est calculée à la ligne 10.

Étant donné les ρ_i et d_i pour chaque objet i , l'algorithme DP calcule les centres des clusters \mathcal{X} , et effectue les attributions de clusters basées sur ces centres. Les centres des clusters sont sélectionnés en utilisant une heuristique simple : les points avec des valeurs élevées de $(\rho_i \times d_i)$ sont plus susceptibles d'être des centres. Nous donnons l'algorithme de sélection des centres de cluster dans l'Algorithm3.

Algorithme : Sélection des Centres de Cluster

Entrée : d , vecteur de distance PPV des points de densité supérieure
rho, le vecteur de densité locale

```

k, nombre de clusters
Sortie : X, centres des clusters
début
    \mathcal{X} = topK(trier(rho.*d, 'descend'), k)
fin
    
```

Étant donné les valeurs triées de $(\rho_i \times d_i)$ en ordre décroissant, les k premiers éléments sont sélectionnés comme centres des clusters (ligne 7).

L'étape finale de l'algorithme DP est l'attribution de cluster. Chaque élément de données reçoit l'étiquette du cluster de son Plus Proche Voisin (PPV) de la liste des points ayant une densité locale supérieure à la sienne.

```

Algorithme : Algorithme d'Attribution de Cluster
Entrée : X, centres des clusters
        d, vecteur de distance PPV des points de densité supérieure
        indiceTri, indice trié des éléments basé sur rho décroissant
Sortie : C, clusters
début
    pour i = 1 à taille(X)
        C(X(i)) = i //attribuer les étiquettes de cluster pour les centres
    fin pour
    pour j = 1 à n
        si C(indiceTri(j)) == vide //pas encore d'étiquette de cluster
            C(indiceTri(j)) = C(PPV(indiceTri(j)))
        fin si
    fin pour
fin
    
```

Élagage lors du Calcul de la Densité Locale Dans cette étape de l'algorithme TAD-Pole, les entrées sont les matrices complètement calculées de borne inférieure (LB Matrix) et de borne supérieure (UB Matrix). Pour chaque paire d'objets (i, j) , lors du calcul de leurs densités locales, nous élaguons leur calcul de distance (D_{ij}) selon les quatre cas suivants illustrés dans ?? :

- Cas A : Les objets i et j sont identiques

La distance DTW de deux objets identiques, i et j , est égale à leur distance ED. Il s'agit d'une simple recherche dans la matrice de distance de borne supérieure, et ne nécessite aucun calcul de distance DTW réel. Ce cas est logiquement possible mais très rare (15))

- Cas B : $UB_{Matrix}(i, j) < d_c$

Si la distance de borne supérieure entre les objets i et j est inférieure à la distance de coupure (d_c), alors i et j sont définitivement à une distance inférieure à d_c l'un de l'autre (15)). Par conséquent, nous pouvons élaguer le calcul de distance DTW de ces deux objets.

- Cas C : $LB_{Matrix}(i, j) > d_c$

Si la distance de borne inférieure de i et j est supérieure à la distance de coupure, alors ces deux objets ne sont définitivement pas à une distance de d_c l'un de l'autre (15)). Nous pouvons donc élaguer admissiblement leur calcul de distance DTW.

- Cas D : $LB_{Matrix}(i, j) < d_c$ et $UB_{Matrix}(i, j) > d_c$

Dans ce cas, nous ne pouvons pas déterminer si la distance DTW réelle entre i et j est inférieure à d_c . Par conséquent, seulement dans ce cas avons-nous besoin de calculer D_{ij} (15)).

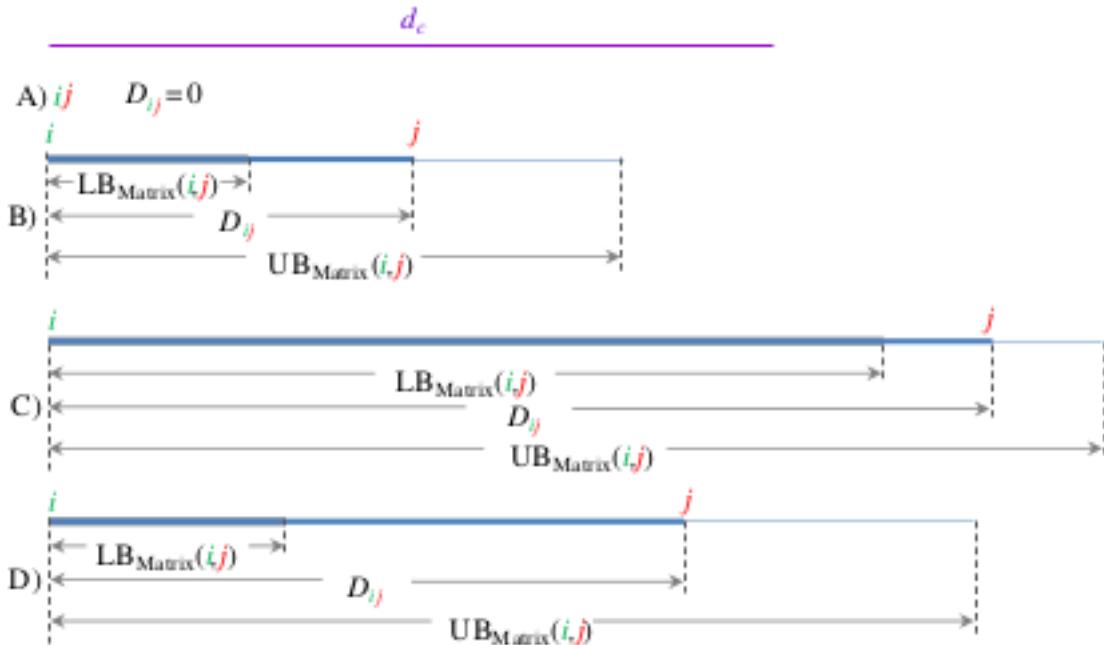


FIGURE 15 – Les quatre cas mutuellement exclusifs et exhaustifs d’élagage du calcul de distance lors du calcul de la densité locale. Notez que la distance de coupure d_c , représentée par la ligne violette en haut, s’applique à tous les quatre cas ci-dessous. Le cas A est difficile à représenter visuellement, car i et j coïncident.

Avec cette intuition à l'esprit, nous spécifions l'algorithme formel d'élagage de distance pendant le calcul de la densité locale dans l'algorithme ci-dessous.

Algorithme : Algorithme d'Élagage pendant le Calcul de la Densité Locale

Entrée : LB, matrice de borne inférieure complètement calculée

UB, matrice de borne supérieure complètement calculée

Data, l'ensemble de données

dc, distance de coupure

Sortie : rho, vecteur de densité locale pour tous les points dans l'ensemble de données

D_Sparse, matrice de distance partiellement remplie

début

D_Sparse = vide

pour i = 1 à taille(Data)

 objectsWithin_dc = vide

 pour j = 1 à taille(Data)

 si i == j continue;

 sinon

 si LB(i,j) == UB (i,j) //cas A

 objectsWithin_dc = [objectsWithin_dc j]

 sinon si UB (i,j) < dc //cas B

 objectsWithin_dc = [objectsWithin_dc j]

 sinon si LB (i,j) > dc // cas C

 continue

 //cas D)

```

        sinon si LB (i,j) < dc et UB (i,j) > dc
            D_Sparse (i,j) = calculerDist(Data(i),Data(j))
            si D_Sparse (i,j) < dc
                objectsWithin_dc = [objectsWithin_dc j]
            fin si
        fin si
    fin pour
    rho(i) = longueur(objectsWithin_dc)
fin pour
fin

```

Élagage lors du Calcul de la Distance NN à partir de la Liste de Densité Supérieure La stratégie d'élagage pour cette étape fonctionne en deux phases. D'abord, pour chaque élément, nous trouvons une borne supérieure de la distance NN à partir de sa liste de densité supérieure. Dans la deuxième phase, nous effectuons l'élagage réel basé sur ces bornes supérieures.

— Phase 1 Calcul de la borne supérieure

Étant donné D_{sparse} et ρ_i pour chaque élément i , nous initialisons la borne supérieure de sa distance NN à partir de sa liste de densité supérieure, ub_i , à ∞ . Pour chaque élément j dans la liste de densité supérieure de i , nous avons soit la distance DTW réelle (D_{ij}) déjà calculée, soit accès à la borne supérieure ($UB(i,j)$) de cette distance. Nous parcourons la liste de densité supérieure de l'élément i , et si l'actuel $ub_i > D_{ij}$ ou $ub_i > UB(i,j)$, nous mettons à jour l'actuel ub_i à D_{ij} (si déjà disponible), ou à $UB(i,j)$, sinon. Par conséquent, nous pouvons garantir que la distance NN à partir de la liste de densité supérieure pour l'élément i ne peut être plus grande que ub_i . Nous donnons une intuition visuelle de ce calcul de borne supérieure dans 16.

Nous donnons l'algorithme de calcul de la borne supérieure pour le calcul de la distance NN à partir d'une liste de densité supérieure.

Algorithme : Calcul de la Borne Supérieure pour le Calcul de la Distance NN à partir de la Liste de Densité Supérieure

Entrée : UB, matrice de borne supérieure complètement calculée

Data, l'ensemble de données

D_Sparse, matrice de distance partiellement remplie

d_liste, liste des points de densités supérieures

Sortie : ub, vecteur de borne supérieure des distances NN à partir des points de densité supérieure

début

```

ub = inf(taille(Data))
pour i = 1 à taille(Data)
    pour j = 1 à taille(d_liste(i))
        objetDensiteSup = d_liste(i)(j)
        si D_Sparse(i, objetDensiteSup) != vide
            si ub_i > D_Sparse(i, objetDensiteSup)
                ub_i = D_Sparse(i, objetDensiteSup)
            fin si
    sinon

```

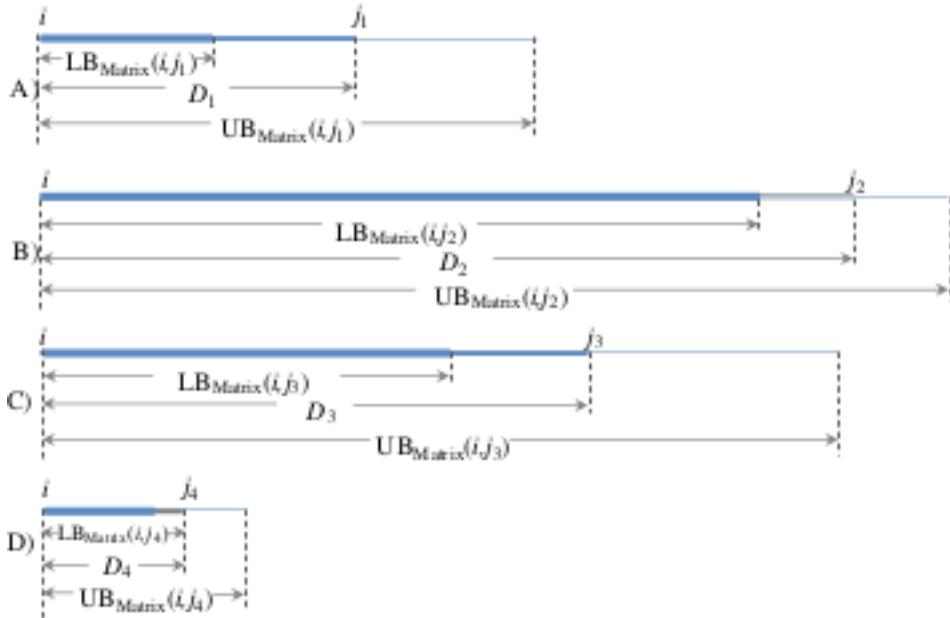


FIGURE 16 – Une illustration de l’élagage de distance pendant le calcul de la distance NN à partir d’une liste de densité supérieure d’un objet. Depuis l’objet i , les éléments dans la liste de densité supérieure sont $j_1 - j_4$. Après la Phase 1, ub_i sera $UB(i, j_4)$. Dans la Phase 2, les calculs de distance de D_{ij_2} et D_{ij_3} sont élagués.

```

        si ub_i > UB(i, objetDensiteSup)
            ub_i = UB(i, objetDensiteSup)
        fin si
    fin si
    fin pour
fin pour
fin

```

À la fin de cette phase de TADPole, nous avons ub , le vecteur de borne supérieure des distances NN à partir des points de densité supérieure, calculé. Nous décrivons maintenant comment exploiter ub pour élaguer les calculs de distance pendant le calcul de la liste de densité supérieure.

— Phase 2 Élagage

Nous donnons l’algorithme d’élagage pendant le calcul des distances NN à partir des listes de densité supérieure de tous les objets.

Algorithme : Algorithme d’Élagage pendant le Calcul des Distances NN à partir des Listes de Densité Supérieure de Tous les Objets

Entrée : LB, matrice de borne inférieure complètement calculée
Data, l’ensemble de données
D_Sparse, matrice de distance partiellement remplie
d_liste, liste des points de densités supérieures
ub, vecteur de borne supérieure des distances NN à partir
des points de densité supérieure

Sortie : d, vecteur de distance NN des points de densité supérieure

```

début
    pour i = 1 à taille(Data)
        temp_d = vide
        pour j = 1 à taille(d_liste(i))
            objetDensiteSup = d_liste(i)(j)
            si LB(i, objetDensiteSup) > ub(i)
                continue //élaguer le calcul de distance
            sinon
                si D_Sparse(i, objetDensiteSup) != vide
                    temp_d = [temp_d D_Sparse(i, objetDensiteSup)]
                sinon // calculer la distance
                    D_Sparse(i, objetDensiteSup) = \\
                        calculerDist(Data(i),Data(objetDensiteSup))
                    temp_d = [temp_d D_Sparse(i, objetDensiteSup)]
                fin si
            fin si
        fin pour
    d(i) = min(temp_d)
fin

```

Après cette phase de TADPole, pour chaque élément i nous avons accès à la distance NN à partir des points ayant une densité locale supérieure (d_i). À ce stade, étant donné ρ_i et d_i pour chaque objet i , l'algorithme TADPole calcule les centres des clusters X en utilisant l'algorithme (Sélection des Centres de Cluster) de DP, et effectue les attributions de clusters basées sur ces centres selon l'algorithme (Algorithme d'Attribution de Cluster) de DP.

5 Évaluation des clusters

La classification est souvent perçue comme une procédure non supervisée, rendant l'évaluation de sa performance quelque peu subjective, selon [Arbelaitz O, 2013]. Une distinction principale entre les techniques de validation réside dans les informations utilisées lors du processus de validation. Certains approches, associées à la validation externe, comparent une partition à une partition de référence, tandis que d'autres, relevant de la validation interne, se basent uniquement sur l'analyse des données partitionnées. La validation externe est principalement pertinente dans un cadre expérimental contrôlé, car la structure véritable des données reste inconnue dans les applications réelles, rendant la partition de référence inaccessible.

Lorsqu'une partition de référence est disponible, en plus d'utiliser un tableau de confusion, la pratique courante consiste à comparer la partition générée par l'algorithme de clustering à cette partition de référence en utilisant divers indices de comparaison des partitions, tels que Rand, Rand ajusté, Jaccard, Fowlkes-Mallows et Variation de l'information.

En l'absence de partition de référence, diverses méthodes permettent de valider une partition, notamment en évaluant la compacité et la séparation des clusters à partir des données partitionnées elles-mêmes. Pour cela, on utilise des indices spécifiques, tels que Dunn, Davies-Bouldin et Calinski-Harabasz.

Bien que certains indices puissent s'adapter aux partitions floues, notre discussion se limitera aux classifications nettes, laissant de côté l'évaluation des classifications floues.

Il est crucial de saisir la nature des mesures prises par un Indice de Validation de Clustering (CVI) pour interpréter adéquatement ses résultats. Cependant, en raison de contraintes d'espace, nous nous concentrerons uniquement sur les indices disponibles dans le package dtwclust.

5.1 Les indices internes

La plupart des indices évaluent la cohésion des groupes (à l'intérieur de groupe ou intra-variance) et la séparation des groupes (entre les groupes ou inter-variance) et les combinent pour calculer une mesure de qualité. La combinaison est effectuée par une division (indices de type ratio) ou une somme (indices de type sommation).

5.1.1 Notation

Définissons un ensemble de données X comme un ensemble de N objets représentés par des vecteurs dans un espace à m dimensions : $X = \{x_1, x_2, \dots, x_N\} \subset \mathbb{R}^m$. Une partition ou une classification dans X est un ensemble de regroupements disjoints qui divise X en K groupes : $C = \{c_1, c_2, \dots, c_K\}$ où $\bigcup_{c_k \in C} c_k = X$, $c_k \cap c_{k'} = \emptyset \forall k \neq k'$. \bar{c}_k désigne le centroïde d'un groupe et, de la même manière, \bar{X} désigne le centroïde de l'ensemble de données.

On suppose des fonctions de distance symétriques,

5.1.2 L'indice de Dunn

Il s'agit d'un indice de type ratio dans lequel la cohésion est estimée par le diamètre maximal des grappes et la séparation par la distance du plus proche voisin.

$$D(C) = \frac{\min_{c_k \in C} \left\{ \min_{c_l \in C \setminus \{c_k\}} \{ \delta(c_k, c_l) \} \right\}}{\max_{c_k \in C} \{ \Delta(c_k) \}},$$

où

$$\delta(c_k, c_l) = \min_{x_i \in c_k} \min_{x_j \in c_l} \{d(x_i, x_j)\},$$

$$\Delta(c_k) = \max_{x_i, x_j \in c_k} \{d(x_i, x_j)\}.$$

Nous voulons que le numérateur soit le plus grand possible et que le dénominateur soit le plus petit possible, de sorte que l'indice de Dunn soit le plus grand possible.

5.1.3 L'indice de Calinski–Harabasz

Il s'agit d'un indice de type ratio où la cohésion est estimée sur la base des distances entre les points d'une groupe et son centroïde. La séparation est basée sur la distance entre les centroïdes et le centroïde global.

$$CH(C) = \frac{N - K}{K - 1} \cdot \frac{\sum_{c_k \in C} |c_k| d(\bar{c}_k, \bar{X})}{\sum_{c_k \in C} \sum_{x_i \in c_k} d(x_i, \bar{c}_k)}.$$

Nous voulons que le numérateur soit le plus grand possible et que le dénominateur soit le plus petit possible, de sorte que l'indice de Calinski–Harabasz soit le plus grand possible.

5.1.4 L'indice de Davies–Bouldin

Il estime la cohésion sur la base de la distance entre les points d'une groupe et son centroïde et la séparation sur la base de la distance entre les centroïdes.

$$DB(C) = \frac{1}{K} \sum_{c_k \in C} \max_{c_l \in C \setminus \{c_k\}} \left\{ \frac{S(c_k) + S(c_l)}{d(\bar{c}_k, \bar{c}_l)} \right\},$$

où

$$S(c_k) = \frac{1}{|c_k|} \sum_{x_i \in c_k} d(x_i, \bar{c}_k).$$

Nous voulons que le numérateur soit le plus petit possible et que le dénominateur soit le plus grand possible , de sorte que l'indice de Davies–Bouldin soit le plus petit possible .

5.1.5 Indice de Davies–Bouldin*

Cette variation de l'indice de Davies-Bouldin se définit comme suit

$$DB^*(C) = \frac{1}{K} \sum_{c_k \in C} \frac{\max_{c_l \in C \setminus \{c_k\}} S(c_k) + S(c_l)}{\min_{c_l \in C \setminus \{c_k\}} d(\bar{c}_k, \bar{c}_l)},$$

Nous voulons que le numérateur soit le plus petit possible et que le dénominateur soit le plus grand possible , de sorte que l'indice de Davies–Bouldin* soit le plus petit possible .

5.1.6 L'indice de Silhouette

Cet indice est un indice normalisé. La cohésion est mesurée sur la base de la distance entre tous les points d'une même groupe et la séparation est basée sur la distance entre les plus proches voisins. Il se définit comme suit

$$Sil(C) = \frac{1}{N} \sum_{c_k \in C} \sum_{x_i \in c_k} \frac{b(x_i, c_k) - a(x_i, c_k)}{\max\{a(x_i, c_k), b(x_i, c_k)\}},$$

où

$$a(x_i, c_k) = \frac{1}{|c_k|} \sum_{x_j \in c_k} d(x_i, x_j),$$

$$b(x_i, c_k) = \min_{c_l \in C \setminus \{c_k\}} \left\{ \frac{1}{|c_l|} \sum_{x_j \in c_l} d(x_i, x_j) \right\}.$$

Nous voulons que b soit le plus grande possible et que a soit le plus petit possible, de sorte que l'indice de Silhouette soit le plus grand possible.

5.1.7 L'indice de Fonction du score

Il s'agit d'un indice de type sommation où la séparation est mesurée sur la base de la distance entre les centroïdes de la groupe et le centroïde global et la cohésion est basée sur la distance entre les points d'une groupe et son centroïde. Il se définit comme suit

$$SF(C) = 1 - \frac{1}{e^{bcd(C)-wcd(C)}},$$

où

$$bcd(C) = \sum_{c_k \in C} \frac{|c_k| d(\bar{c}_k, \bar{X})}{N \times K},$$

$$wcd(C) = \sum_{c_k \in C} \frac{1}{|c_k|} \sum_{x_i \in c_k} d(x_i, \bar{c}_k).$$

Nous voulons que bcd soit le plus grande possible et que wcd soit le plus petit possible, de sorte que l'indice de fonction du score soit le plus grand possible.

5.1.8 L'indice de COP

Bine que cet indice ait été proposé à l'origine pour être utilisé en conjonction avec un algorithme de post-traitement de la hiérarchie des groupes il peut également être utilisé comme un CVI ordinaire. Il s'agit d'un indice de type ratio dans lequel la cohésion est estimée par la distance entre les points d'une groupe et son centroïde, et la séparation est basée sur la distance entre les plus lointains voisins. Sa définition est la suivante :

$$COP(C) = \frac{1}{N} \sum_{c_k \in C} |c_k| \frac{\frac{1}{|c_k|} \sum_{x_i \in c_k} d(x_i, \bar{c}_k)}{\min_{x_i \notin c_k} \max_{x_j \in c_k} d(x_i, x_j)}.$$

Nous voulons que le numérateur soit le plus petit possible et que le dénominateur soit le plus grand possible , de sorte que l'indice de COP soit le plus petit possible .

5.2 Les indices externes

Il n'est pas possible de déterminer a priori quel indice sera le plus efficace ; il convient donc de le tester pour chaque application spécifique. Plusieurs CVIs peuvent être utilisés et

comparés les uns aux autres, éventuellement en utilisant un vote majoritaire pour décider du résultat final, mais il n'y a pas de meilleur indice.

Étant donné un ensemble de n éléments, S , et deux partitions de S à comparer, à savoir X et Y , nous considérons toutes les paires d'éléments dans S et comptons les paires comme suit :

- Vrai positif (TP) : Le nombre de paires d'éléments qui sont dans le même sous-ensemble dans X et dans le même sous-ensemble dans Y .
- Vrai négatif (TN) : Le nombre de paires d'éléments qui sont dans des sous-ensembles différents dans X et dans des sous-ensembles différents dans Y .
- Faux positif (FP) : nombre de paires d'éléments qui se trouvent dans le même sous-ensemble dans X mais dans des sous-ensembles différents dans Y .
- Faux négatif (FN) : Le nombre de paires d'éléments qui sont dans des sous-ensembles différents dans X mais dans le même sous-ensemble dans Y .

5.2.1 L'indice de Rand

L'indice de Rand (RI) mesure la similarité entre deux partitions de clustering, basé sur le nombre de décisions concordantes et discordantes. Les paires d'éléments sont comptées selon qu'elles se trouvent dans les mêmes ou dans différents clusters dans les deux partitions. La formule du RI est :

$$RI = \frac{TP + TN}{TP + FP + FN + TN}$$

où RI varie de 0 à 1, 1 indiquant une correspondance parfaite.

5.2.2 L'indice de Rand ajusté (ARI)

L'ARI ajuste le RI en tenant compte du hasard, avec une formule normalisée :

$$ARI = \frac{RI - \text{Expected}[RI]}{\text{Max}[RI] - \text{Expected}[RI]}$$

5.2.3 L'indice de Jaccard

Considérons deux ensembles A et B , l'indice de Jaccard (J) est défini comme la taille de l'intersection divisée par la taille de l'union de deux ensembles.

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

J varie de 0 à 1, avec 0 indiquant aucune similarité et 1 une identité complète. L'indice de Jaccard est particulièrement utile pour les problèmes binaires et de classification.

5.2.4 L'indice de Similarité (TSclust)

Nous introduisons un indice de similarité fourni avec le paquet TSclust. Il consiste à calculer un indice qui mesure le degré d'accord entre la partition de clusters vraie $G = \{G_1, \dots, G_k\}$ (la “vérité terrain”), supposée connue, et la solution de clustering expérimentale $A = \{A_1, \dots, A_k\}$ obtenue par une méthode de clustering en cours d'évaluation. L'indice de similarité $\text{Sim}(G, A)$ est défini par :

$$\text{Sim}(G, A) = \frac{1}{k} \sum_{i=1}^k \max_{1 \leq j \leq k} \text{Sim}(G_i, A_j),$$

où

$$\text{Sim}(G_i, A_j) = \frac{|G_i \cap A_j|}{|G_i| + |A_j|},$$

avec $\|\cdot\|$ représentant la cardinalité des éléments de l'ensemble.

5.2.5 L'indice de Fowlkes-Mallows (FM)

FM est défini comme la moyenne géométrique de la précision et du rappel, calculée par :

$$FM = \sqrt{\frac{TP}{TP + FP} \cdot \frac{TP}{TP + FN}}$$

5.2.6 La Variation d'Information (VI)

La VI entre deux clusterings C et C' est définie en termes d'information mutuelle $I(C; C')$ et d'entropie $H(C)$ et $H(C')$ de chaque clustering.

$$VI(C, C') = H(C) + H(C') - 2I(C; C')$$

Une VI de 0 indique que les deux clusterings sont identiques en termes de contenu d'information. L'entropie $H(C)$ d'un clustering est calculée comme suit :

$$H(C) = - \sum_{i=1}^k P(i) \log P(i)$$

où $P(i)$ est la probabilité qu'un élément choisi aléatoirement dans l'ensemble de données appartienne au cluster i dans C , et k est le nombre de clusters dans C .

L'information mutuelle $I(C; C')$ entre deux clusterings est définie comme :

$$I(C; C') = \sum_{i=1}^k \sum_{j=1}^{k'} P(i, j) \log \frac{P(i, j)}{P(i)P'(j)}$$

où $P(i, j)$ est la probabilité qu'un élément choisi aléatoirement dans l'ensemble de données appartienne à la fois au cluster i dans C et au cluster j dans C' , $P(i)$ est tel que défini précédemment, et $P'(j)$ est la probabilité qu'un élément choisi aléatoirement appartienne au cluster j dans C' .

Deuxième partie

Partie Pratique

6 R instruction

Cette section est une préparation à l'analyse du problème.

6.1 Contexte

Dans l'écosystème R, cinq bibliothèques se consacrent à la classification des séries temporelles, offrant une variété d'approches allant de méthodes générales à des solutions plus spécialisées. Le package TSclust se démarque avec sa riche gamme de métriques spécialement conçues pour divers aspects des séries temporelles, y compris leur représentation et leur complexité. dtwclust, quant à lui, se spécialise dans l'exploitation des distances DTW (Dynamic Time Warping), s'alignant sur les méthodes centrées sur la forme des séries. LPWC et pdc adoptent également des approches basées sur la forme, le premier en utilisant des distances de corrélation pondérées et le second en se basant sur les distances de distribution de permutation. BNPTSclust, offrant une perspective basée sur le modèle via un modèle linéaire dynamique et un algorithme bayésien, montre une approche distincte mais est limité à des séries temporelles annuelles, mensuelles ou trimestrielles.

Des bibliothèques comme TSrepr et TSdist enrichissent les outils de classification des séries temporelles. TSrepr facilite la réduction du bruit et des coûts de calcul grâce à des techniques de sélection des caractéristiques. TSdist, de son côté, propose des méthodes avancées pour mesurer la distance entre les séries temporelles.

La classification des séries temporelles diffère des clustering traditionnels principalement par la manière d'évaluer la similarité, souvent via la distance. Tant les approches basées sur les caractéristiques que celles basées sur les modèles convergent vers la création de nouvelles mesures de distance ou de similarité.

Le choix judicieux de l'algorithme de clustering est crucial pour prévenir toute influence indésirable des variations de distance sur le processus de regroupement. Nous observons que certains algorithmes reconnus, tels que les k-means, peuvent rencontrer des difficultés dues à la définition inappropriée de la dissimilarité entre les séries temporelles. De même, l'utilisation de l'approche de clustering hiérarchique nécessite une attention particulière, surtout si la méthode de Ward est choisie, en raison de sa dépendance aux distances euclidiennes entre les centroids.

6.2 Préparer les données et Visualisation des données

Notre exploration approfondie des bibliothèques R dédiées à la classification des séries temporelles révèle deux méthodes principales de stockage des données. Dans la première, adoptée par TSclust, pdc, et BNPTSclust, les données sont formatées en dataframes, matrices, ou mts (spécifiquement pour TSclust), avec chaque colonne représentant une série temporelle et chaque ligne un point temporel. Cependant, TSclust unique dans le sens où, dans une matrice, chaque ligne représente une série temporelle. La seconde méthode, utilisée par TSrepr, dtwclust, et LPWC, organise également les données en dataframes et matrices mais inverse la représentation, avec chaque ligne représentant une série temporelle et chaque

colonne un point dans le temps. Cette distinction souligne l'importance de prêter attention au format de stockage des données pour éviter toute confusion.

dtwclust se distingue en ne suivant pas les conventions de TSclust car il prend en charge toutes les distances définies par le package proxy, ce qui lui permet de traiter les données ligne par ligne, contrairement à TSclust.

Les données étudiées proviennent du package R et de l'archive UCR Time Series Classification [Chen et al., 2015], couvrant une vaste gamme d'applications et illustrant la diversité de l'archive avec des exemples dans la santé, la finance, l'environnement, l'industrie, l'astronomie et la reconnaissance de formes.

Notre sélection de sujets de recherche s'appuie sur la longueur des séries temporelles, l'invariance des ensembles de données, la présence de modèles synthétiques, et la complexité des données, pour étudier les nuances des différentes complexités temporelles et invariances.

Finalement, nous examinons sept ensembles de données et discutons de la conversion entre les objets 'mts', 'matrix', et 'data.frame'. La conversion d'un objet mts en matrix se fait simplement avec 'as.matrix()', tandis que la conversion d'une matrix en mts nécessite la fonction 'ts()' et des informations temporelles spécifiques. Cette exploration souligne la flexibilité et les considérations nécessaires lors de la manipulation de séries temporelles dans R.

```
# Conversion de matrix en mts
my_mts <- ts(my_matrix)
```

Pour transformer un objet mts en data.frame, la fonction 'as.data.frame()' est directement utilisable.

Pour convertir un data.frame en un objet mts, une étape préparatoire s'impose. Il est essentiel de vérifier que toutes les colonnes concernées par la conversion sont de type numérique.

```
# Si 'my_df' a une colonne de date ou autre non numérique, assurez-vous de ne pas
# l'inclure
numeric_columns <- my_df[, sapply(my_df, is.numeric)]  
  
# Conversion de data.frame en mts
my_mts_from_df <- ts(as.matrix(numeric_columns))
```

Notons que chaque colonne est une série chronologique pour ces trois types d'objets !

Pour savoir comment transposer un data.frame, nous devons tout d'abord nous assurer que toutes ses colonnes sont numériques, puis transformer le data.frame en une matrice pour la transposition.

```
df_numeric <- data.frame(lapply(df, as.numeric))  
  
# Conversion en matrice et transposition
df_transposed <- t(as.matrix(df))  
  
# Convertissez en data.frame si nécessaire
df_transposed_df <- as.data.frame(df_transposed)
```

6.2.1 synthetic.tseries [Pablo Montero, 2014]

TSclust offre une série de données synthétiques, accessible via `synthetic.tseries`, qui comprend trois échantillons de taille $T=200$ pour six modèles autorégressifs de premier ordre, comme décrit dans le tableau 6. Chaque modèle intègre des ϵ_t gaussiens indépendants et identiquement distribués, avec une moyenne nulle et une variance unitaire, explorant à la fois des dynamiques linéaires et non linéaires. Le modèle M1 est un processus AR(1) avec une autocorrélation modérée. M2 adopte une approche bilinéaire avec une moyenne conditionnelle proche d'une fonction quadratique, manifestant une forte non-linéarité. M3, un modèle autorégressif exponentiel, affiche une complexité non linéaire subtile, flirtant avec la linéarité. M4 introduit une rupture significative avec une non-linéarité marquée à travers un modèle autorégressif à seuil. Les modèles M5 et M6, respectivement un modèle autorégressif non linéaire général et un modèle à transition douce, illustrent des non-linéarités plus légères. Ces choix modélisent un éventail de comportements non linéaires dans la moyenne conditionnelle, offrant un cadre riche pour évaluer l'efficacité des mesures de dissimilarité dans TSclust pour le clustering.

M1 AR	$X_t = 0.6X_{t-1} + \varepsilon_t$
M2 Bilinear	$X_t = (0.3 - 0.2\varepsilon_{t-1})X_{t-1} + 1.0 + \varepsilon_t$
M3 EXPAR	$X_t = (0.9 \exp(-X_{t-1}^2) - 0.6)X_{t-1} + 1.0 + \varepsilon_t$
M4 SETAR	$X_t = (0.3X_{t-1} + 1.0)I(X_{t-1} \geq 0.2) - (0.3X_{t-1} - 1.0)I(X_{t-1} < 0.2) + \varepsilon_t$
M5 NLAR	$X_t = 0.7 X_{t-1} (2 + X_{t-1})^{-1} + \varepsilon_t$
M6 STAR	$X_t = 0.8X_{t-1} - 0.8X_{t-1}(1 + \exp(-10X_{t-1}))^{-1} + \varepsilon_t$

TABLE 6 – Modèles génératifs pour les séries temporelles dans `synthetic.tseries`.

Les premières étapes sont le chargement du package et du jeu de données :

```
data("synthetic.tseries")
class(synthetic.tseries)
[1] "mts"     "ts"      "matrix"
```

Nos données sont formatées en objet "mts".

Voici le code pour visualiser les données :

```
xyplot(synthetic.tseries , scales = list(x = list(relation = "same"),
y = list(relation = "same")), layout = c(3, 6), as.table = TRUE,
# Panels ordered like a table, top to bottom, then left to right
xlab = "Time",
strip = strip.custom(strip.names = TRUE, strip.levels = TRUE))
```

Nous avons identifié un besoin de réduction du bruit dans cet ensemble de données. Il est donc essentiel de sélectionner des mesures de distance stables face à différentes complexités.

6.2.2 Shape complexities [Brandmaier, 2015]

Les signatures de forme sont générées en enroulant le contour d'un objet autour de son centroïde au cours du "temps", produisant une série temporelle qui capture la distance du centre aux points du contour en fonction de l'angle radial ([Gonzalez R, 1992] [Keogh E, 2009]). La longueur NN de cette série temporelle dépend de la résolution angulaire $\Delta\phi =$

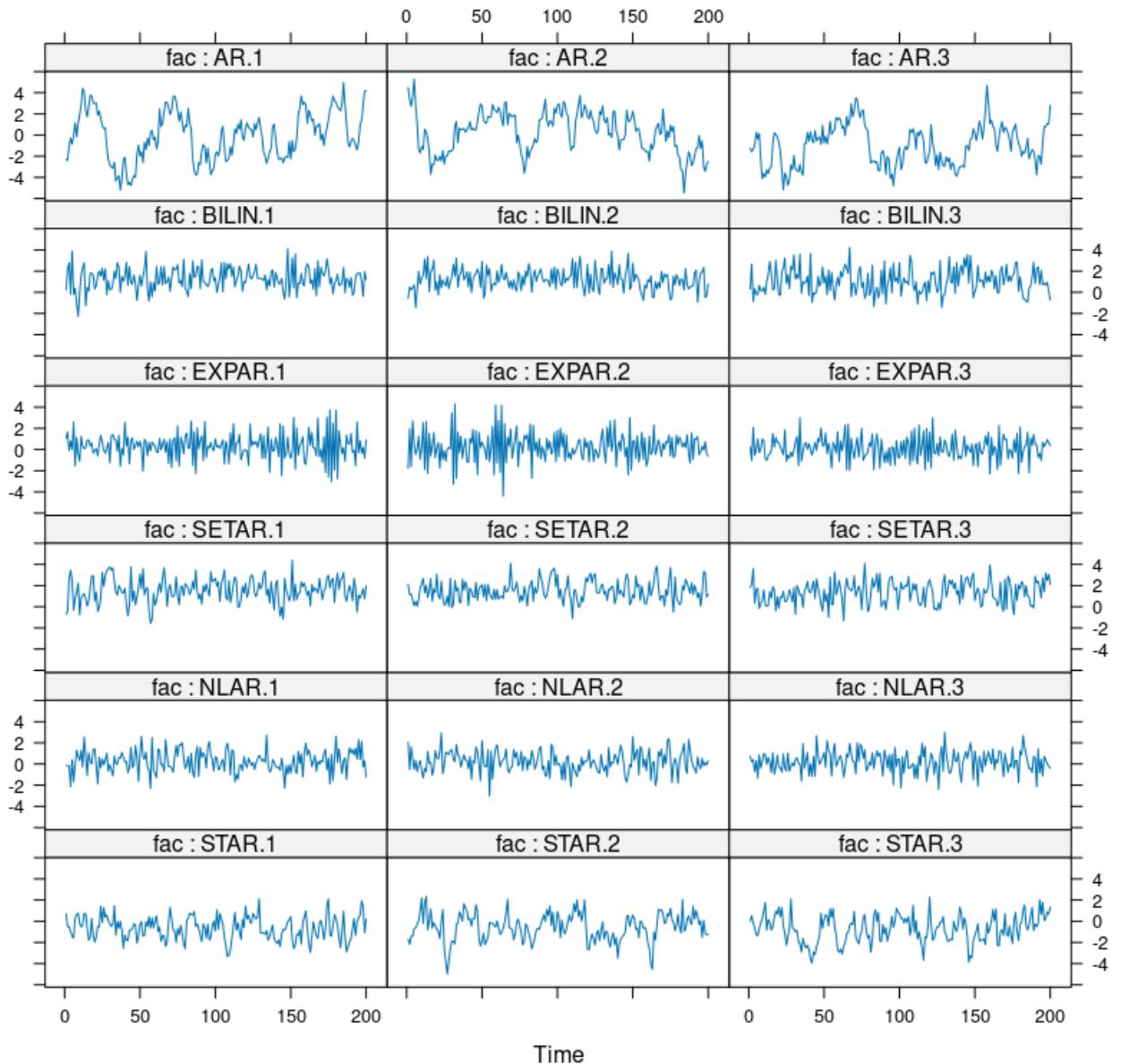


FIGURE 17 – Dix-huit paires de séries temporelles synthétiques

π/N en radians. Le package pdc propose la fonction traceImage pour faciliter ce processus. Pour garantir l'applicabilité, il est crucial que les descriptions d'objets soient invariantes par rapport à la translation, la rotation (phase) et l'échelle.

Nous examinons ces principes à travers un ensemble de données contenant deux formes : une étoile à quatre branches et une étoile à cinq branches, chacune disponible en deux tailles (100% et 150%) et deux orientations (0° et 45°). Ces formes sont converties en séries temporelles avec une résolution angulaire de 3,6° par point temporel. Ci-dessous, le code pour visualiser ces formes.

```
#Importer le jeu des données
data("star.shapes", package="pdc")
library(plotrix)
#Un tracé polaire du paquet plotrix
```

```
oldpar <- polar.plot(star.shapes[, 1])
par(oldpar)
# Convertir en ts
star <- ts(star.shapes)
#plotter les séries.
xyplot(star , scales = list(x = list(relation = "same") , y = list(relation = "free")
layout = c(1,ncol(star)),xlab = "Time" ,strip= FALSE)
```

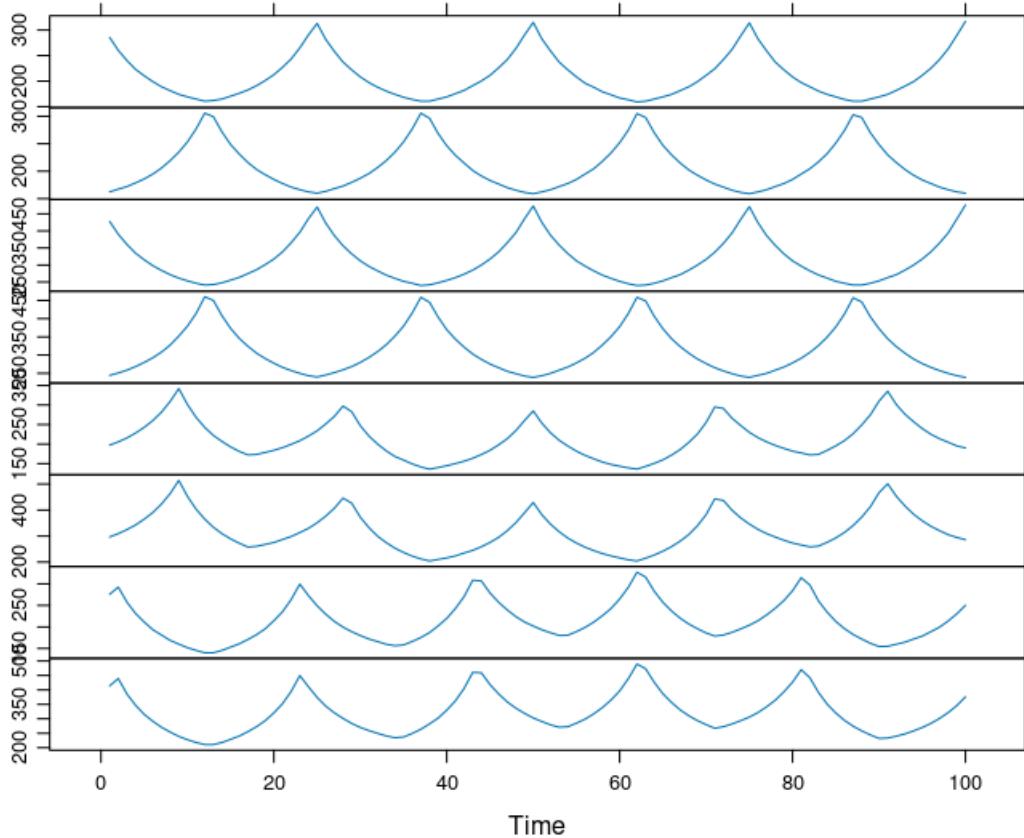


FIGURE 18 – Séries temporelle des étoiles

En examinant les figures 18 et 19, il est notable que toutes présentent des formes analogues, distinguées par des variations de phase entre les groupes, traduisant des débuts différents, ou des variations d'échelle sur l'axe vertical. Les catégories varient également en termes de fréquence, correspondant aux configurations pentagonales et quadrangulaires des étoiles.

6.2.3 paired.tseries [Pablo Montero, 2014]

Nous explorons un ensemble de données composé de 18 paires de séries temporelles issues de divers secteurs, dont la finance, la science, la médecine et l'industrie. Cet ensemble est accessible gratuitement via <http://www.cs.ucr.edu/~eamonn/SIGKDD2004/>, et intégré à TSclust [Pablo Montero, 2014] sous le nom paired.tseries, stocké sous format "mts". Chaque paire provient d'un domaine unique de l'archive des séries temporelles de l'Université de

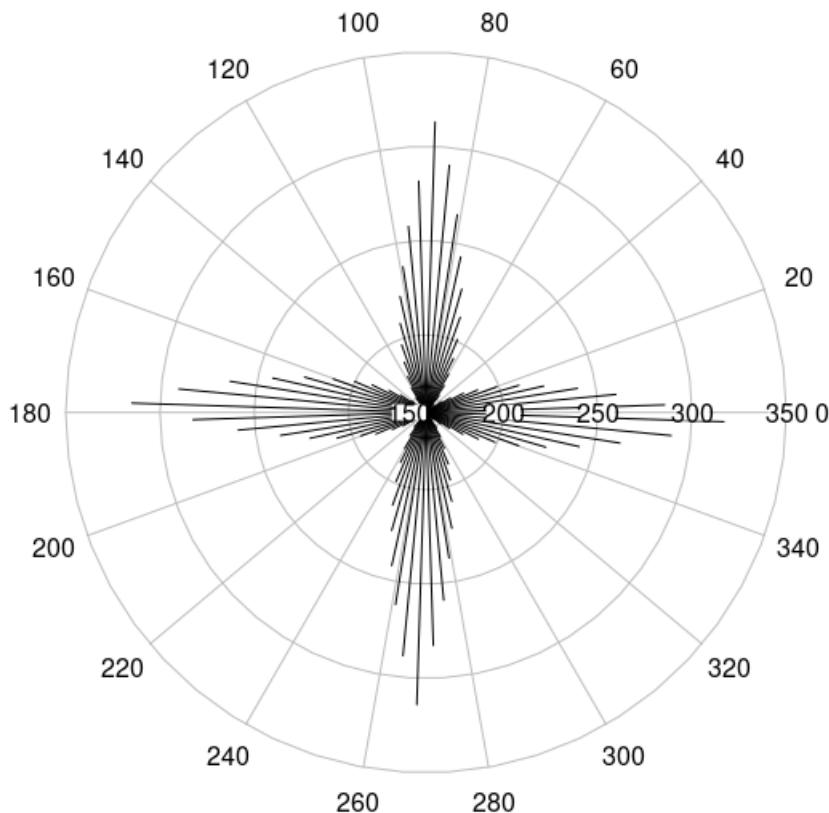


FIGURE 19 – Etoile régulier à quatre points

Californie à Riverside (UCR). Parmi les exemples, nous trouvons des séries thoraciques et abdominales de l'archive Fetal-ECG, ainsi que des séries de demande d'énergie d'avril à juin issues de l'archive Italy-Power-Demand. Les 36 séries temporelles sont illustrées dans la figure 20, révélant des différences marquées entre certaines paires. Voici également le code pour générer ce graphique :

```
xyplot(paired.tseries[,1:36] , scales = list(x = list(relation = "same") ,
y = list(relation = "free")),
layout = c(1,36),xlab = "Time" ,strip= FALSE)
```

Cet ensemble de données présente une complexité notable, nécessitant que la mesure de distance entre les séries temporelles soit robuste face à divers types de transformations. Il est impératif que cette mesure soit invariante à la phase temporelle des séries, ainsi qu'aux transformations linéaires, y compris les décalages temporels, la mise à l'échelle des valeurs et la dérive le long de l'axe des valeurs. De plus, une résilience à l'inadéquation locale, comme les valeurs aberrantes, est requise. Compte tenu de ces multiples exigences d'invariance, nous sommes en présence d'un ensemble de données d'une grande complexité.

6.2.4 Consommation d'électricité des appareils ménagers.[Chen et al., 2015]

L'ensemble de données provient d'ACS-F1, la première édition de la base de données des signatures de consommation d'appareils. Il comprend la consommation énergétique d'ap-

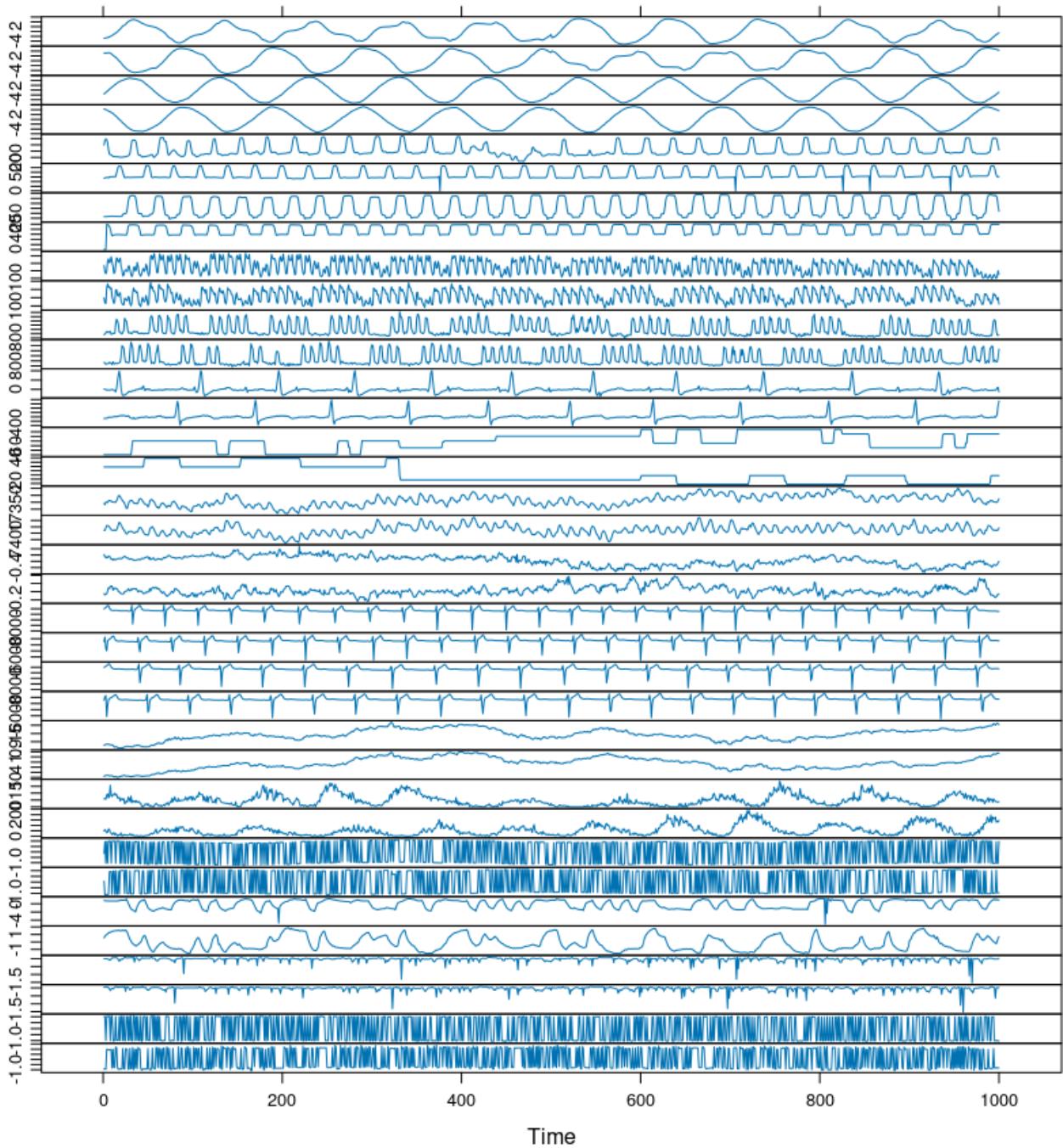


FIGURE 20 – Dix-huit paires de séries temporelles provenant des archives de séries temporelles de l’UCR

pareils électroménagers courants, caractérisée par des périodes prolongées de faible activité entrecoupées de brèves pointes de forte consommation lorsqu'un appareil est en fonction.

Les données se répartissent en 10 catégories d'appareils : chargeurs de téléphones portables, machines à café, postes informatiques (écran inclus), réfrigérateurs et congélateurs, chaînes hi-fi (lecteurs CD), lampes (CFL), chargeurs d'ordinateurs portables, fours à micro-ondes, imprimantes, et téléviseurs (LCD ou LED).

L'ensemble se compose de 100 séries temporelles, chacune s'étendant sur 1460 points

temporels.

Pour des raisons de clarté visuelle, seuls les 300 premiers points temporels sont affichés, la haute fréquence des données rendant difficile l'observation des tendances sur des périodes plus longues. Ci-dessous, le code pour visualiser ces données :

```

setwd("/home/heli/Documents/GM9/Projet/PhaseII/dataset/UCRArchive_2018/ACSF1")
ACSF <- read.table('ACSF1_TRAIN.tsv')
colnames(ACSF) <- c("label",paste("t",1:(ncol(ACSF)-1),sep=""))
library(ggplot2)
library(dplyr)

#regrouper les séries par les classes
selected <- ACSF[,1:300] %>% group_by(label) %>% slice_head(n=1)

library(data.table)
# préparer les données pour le tracé.
data_plot <- melt(data.table(ID = 1:nrow(selected),
                             class = selected$label,
                             selected[,-1]),
                   id.vars = c("ID", "class"),
                   variable.name = "Time",
                   variable.factor = FALSE
)
data_plot[, Time := as.integer(gsub("t", "", Time))]

# tracer les séries
ggplot(data_plot, aes(Time, value, group = ID)) +
  facet_wrap(~class, ncol = 2, scales = "free_y") +
  geom_line(color = "blue", alpha = 0.65) +
  labs(x = "Time", y = "Load (normalised)") +
  theme_bw()

```

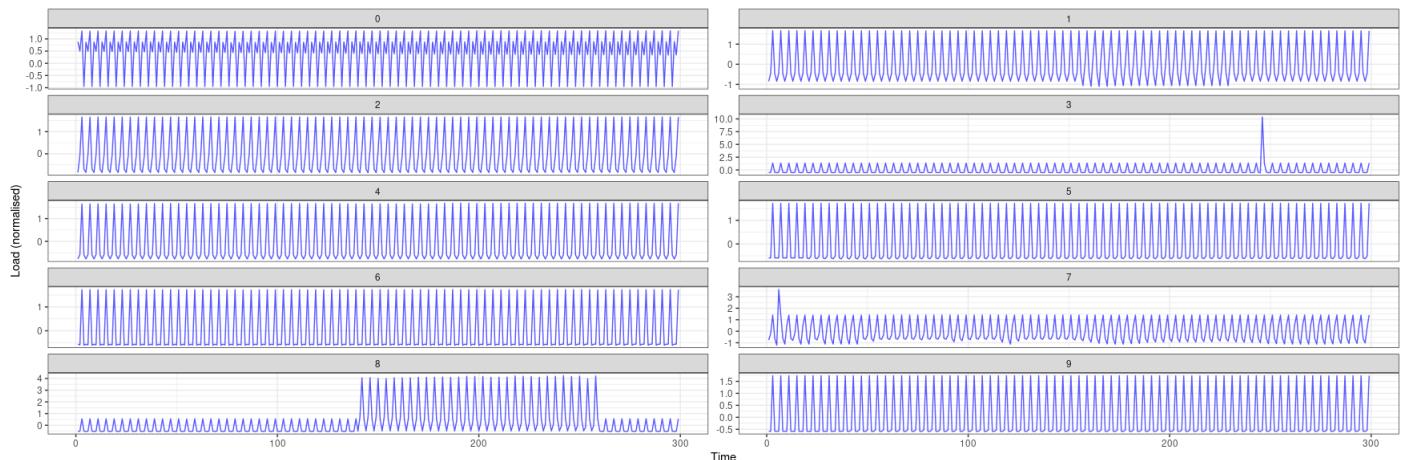


FIGURE 21 – Premier 300 instants temporelles de consommation d'électricité des appareils

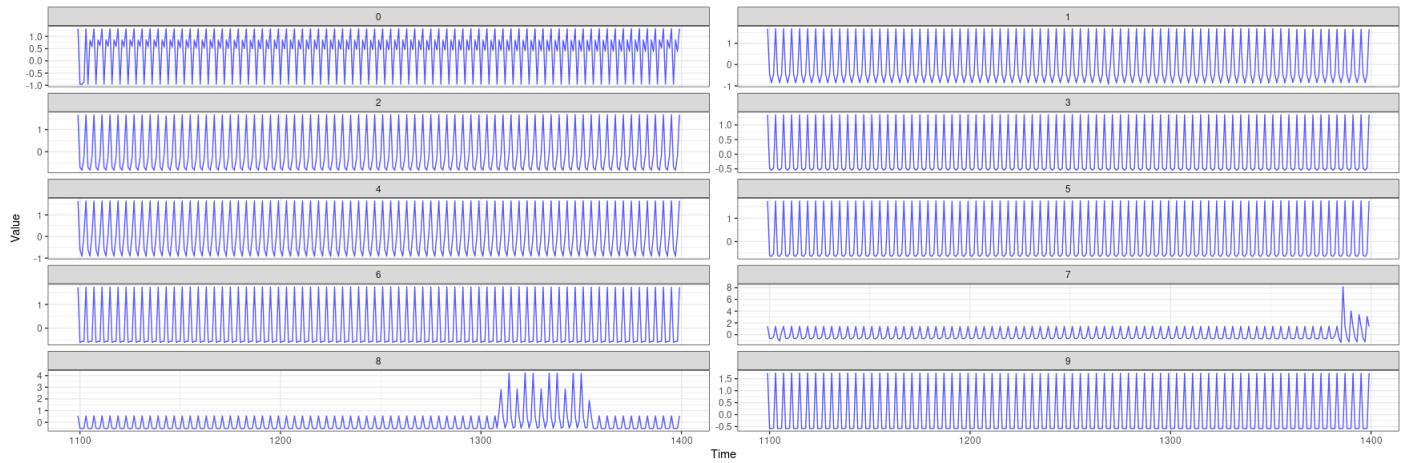


FIGURE 22 – Dernier 300 instants temporelles de consommation d'électricité des appareils

Les figures 21 et 22 montrent que, malgré une fréquence identique pour toutes les séries, l'intensité de leur croissance et leurs modèles de variation diffèrent. La mesure de distance entre les séries temporelles doit donc être invariante à la phase temporelle. Plus crucial encore, chaque catégorie d'appareils ménagers présente des périodes actives distinctes. Il est essentiel d'élaborer une méthode permettant de négliger le nombre et le moment des phases actives pour se concentrer exclusivement sur les modèles d'activité, comme illustré dans ???. Cette considération diffère des cas précédemment examinés.

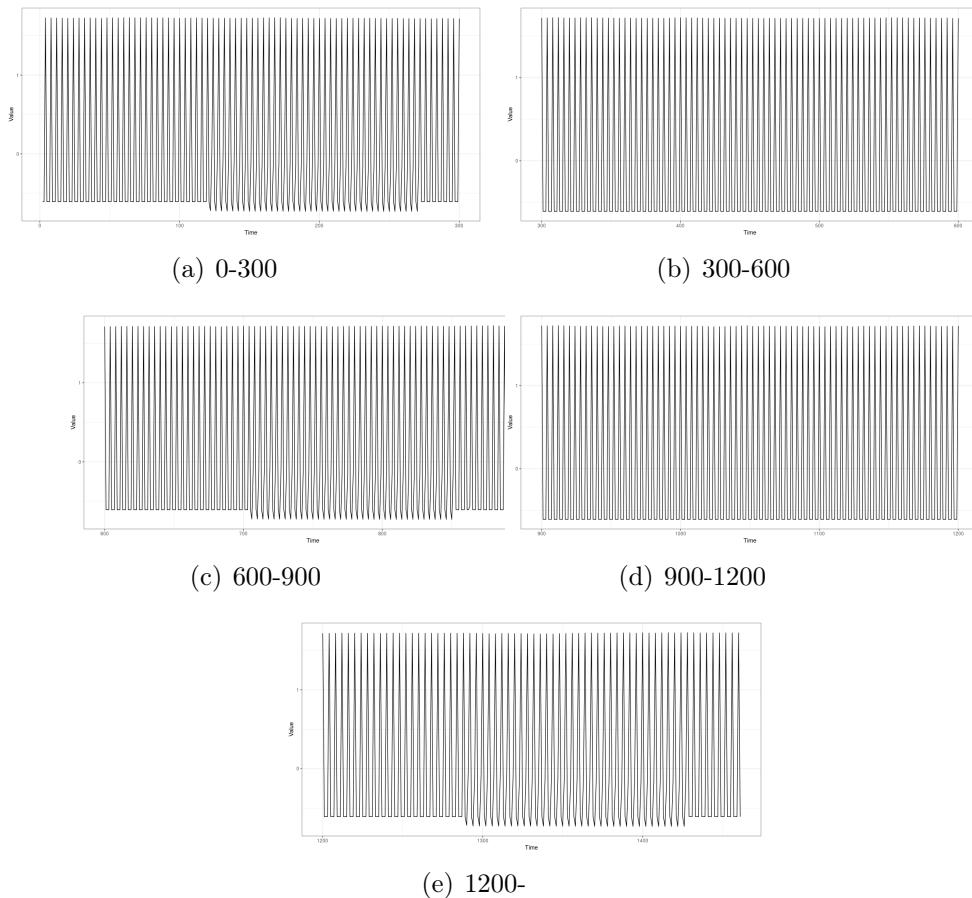


FIGURE 23 – différent intervalle

6.2.5 Consommation d'électricité [Laurinec]

Un ensemble de données contenant les séries chronologiques de la consommation d'électricité de 50 consommateurs pendant 2 semaines. Chaque jour comporte 48 mesures (données semi-horaires). Chaque ligne représente une série temporelle d'un consommateur. Il y a 672 valeurs, soit 14 jours de mesures. Voici le code pour importer le jeu de données et tracer son image :

```
#importer le jeu de données
data("elec_load")
#préparer 8 séries pour tracer
data_plot <- melt(data.table(ID = 1:8,
                               elec_load[1:8,]),
                  id.vars = c("ID"),
                  variable.name = "Time",
                  variable.factor = FALSE
)
#éliminer le "V" dans le nom de colonne
data_plot[, Time := as.integer(gsub("V", "", Time))]
#tracer les séries séparément.
ggplot(data_plot, aes(Time, value, group = ID)) +
  facet_wrap(~ID, ncol = 2, scales = "free_y") +
  geom_line(color = "blue", alpha = 0.65) +
  labs(x = "Time", y = "Value") +
  theme_bw()
```

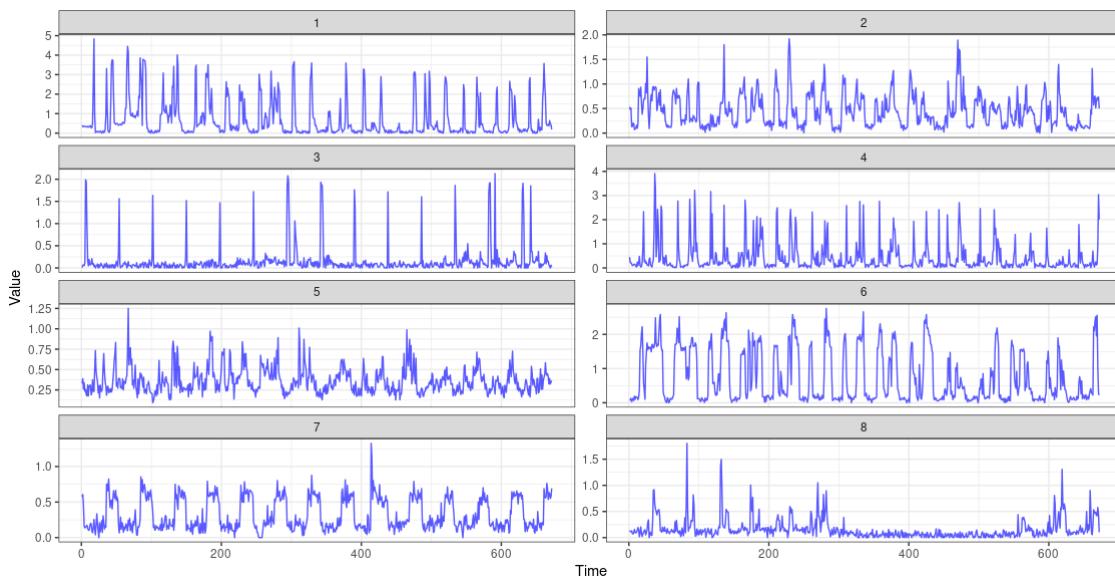


FIGURE 24 – la consommation d'électricité de 8 consommateurs

Nous ne disposons pas de critères définis pour classer ces séries chronologiques de consommation d'électricité. Cependant, nous avons constaté par l'observation que chaque série temporelle a des cycles, et nous aimerais donc regrouper les séries temporelles qui ont un comportement similaire en matière de consommation d'électricité au cours du cycle. Certains sont en effet similaires, voir la figure 24.

6.2.6 interest.rates [Pablo Montero, 2014]

Notre dernier exemple [Pablo Montero, 2014] porte sur les données de taux d'intérêt à long terme incluses dans TSclust et disponibles en chargeant `interest.rates`. Ce jeu de données est constitué de 18 séries temporelles de longueur $T = 215$ représentant les taux d'intérêt mensuels à long terme (obligations à 10 ans), émis en pourcentage par an, pour dix-sept pays et l'Union économique et monétaire (UEM). La période d'observation va de janvier 1995 à novembre 2012 (source OCDE : <http://www.oecd.org/>). Les obligations d'État à long terme sont l'instrument dont le rendement est utilisé comme "taux d'intérêt" représentatif pour chaque zone. Ces indicateurs sont donc un outil essentiel de la politique monétaire et sont pris en compte dans le traitement de variables telles que l'investissement, l'inflation et le chômage. En particulier, l'un des critères de convergence requis pour tous les pays de la zone euro est lié à cet indicateur.

```
class(interest.rates)
[1] "mts" "ts"
```

Le jeu de données est stocké sous la forme de 'mts'.

```
library("lattice")
xyplot(interest.rates , scales = list(x =list(relation = "same"),
y = list(relation ="same")), layout = c(3, 6), as.table = TRUE,
# Panels ordered like a table, top to bottom, then left to right
xlab = "Time",
strip = strip.custom(strip.names = TRUE, strip.levels = TRUE))
```

Nous pouvons voir une image 25 de l'évolution des taux d'intérêt dans le temps pour différents pays.

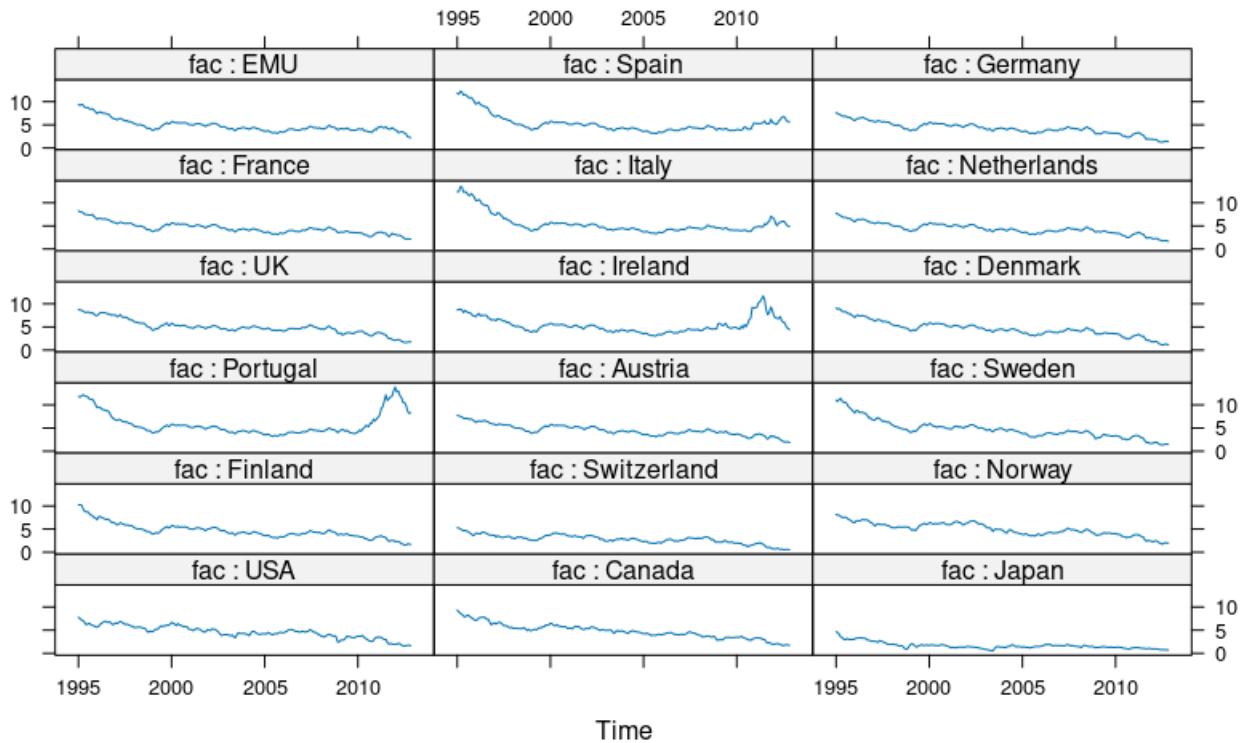


FIGURE 25 – L'évolution des taux d'intérêt dans le temps pour différents pays

Nous examinons maintenant l'augmentation mensuelle en pourcentage des taux d'intérêt dans chaque pays. Plus précisément, nous souhaitons effectuer une analyse en clustering avec les séries transformées en prenant la première différence des logarithmes, c'est-à-dire en utilisant les séries $Y_T^i, i = 1, \dots, 18$, données par $\log X_t^i - \log X_{t-1}^i, t = 2, \dots, T$. La figure se montre dans la figure 26 ici :

6.3 Approches basées sur la forme

Les approches basées sur la forme visent à définir de nouvelles mesures de distance ou de similarité directement applicables aux données brutes, un concept central dans les packages TSclust et dtwclust. Les packages LPWC et pdc proposent également des distances uniques et des algorithmes pour optimiser les paramètres de ces distances, bien que LPWC ait été exclu de notre étude en raison de limitations d'utilisation.

Nous explorons d'abord la méthode offerte par TSclust. Étant donné que TSclust définit uniquement des distances sans spécifier l'algorithme de calcul du centroïde, nous sommes libres d'explorer d'autres méthodes de clustering en dehors des k-means. Le package stats fournit la fonction hclust() pour le clustering hiérarchique, tandis que le package cluster (Maechler, Rousseeuw, Struyf, Hubert, et Hornik 2014) inclut agnes() pour un clustering hiérarchique et diana() ainsi que pam() pour un clustering hiérarchique divisé et un clustering par partitionnement autour des médoïdes, respectivement.

La figure 7 répertorie les distances disponibles dans TSclust, soulignant la variété des mesures adaptées aux analyses basées sur la forme.

Dans TSclust, les distances sont catégorisées selon trois approches principales. Pour celles basées sur la forme, nous avons choisi des mesures emblématiques telles que la distance eu-

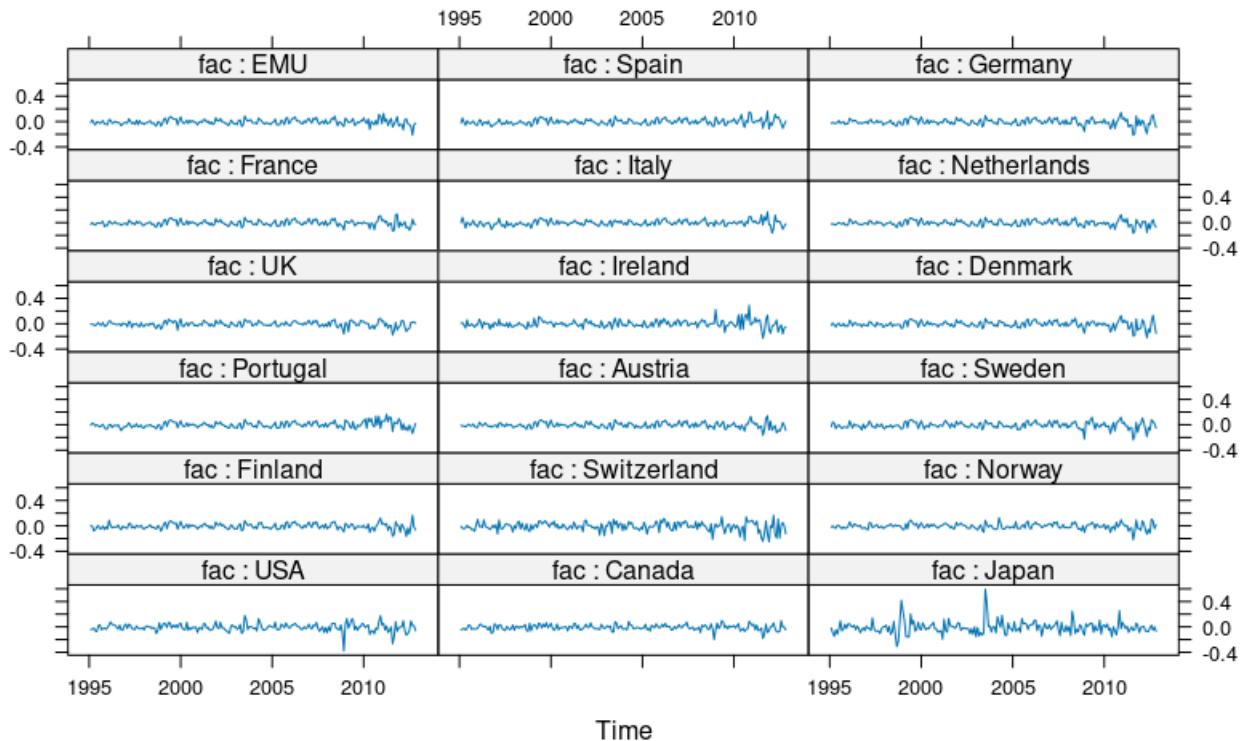


FIGURE 26 – Taux d’intérêt à long terme transformés en prenant la première différence des logarithmes.

clidienne, ACF, CORT, CDM, et PDC, pour leur capacité à capturer l’essence des séries temporelles. Nous écartons les distances basées sur le périodogramme et l’estimation spectrale non paramétrique, car elles présentent des similitudes fonctionnelles avec l’ACF et sont omises ici pour optimiser l’utilisation de l’espace.

Certaines de ces distances transforment les données brutes en un ensemble de caractéristiques décrivant la série temporelle, permettant d’incorporer sa structure temporelle. Cette approche, semblable aux méthodes basées sur les caractéristiques, est néanmoins classée ici comme basée sur la forme. Cela est justifié lorsque le nombre de caractéristiques extraites correspond aux dimensions des données originales, assurant ainsi l’invariance souhaitée.

Regroupement à l'aide des distances TSclust Nous devons d'abord charger le paquet TSclust. Les données pour les opérations suivantes sont le jeu de données synthetic.tseries du paquet TSclust. Ce jeu de données est présenté sous la forme d'un objet mts.

```
library("TSclust")
data("synthetic.tseries")
```

Nous commençons par tester la mesure de dissimilarité ACP , qui calcule la distance euclidienne entre les fonctions d'autocorrélation estimées et peut être évaluée dans TSclust en invoquant diss.ACF. La fonction diss() est fournie pour faciliter l'application de n'importe quelle fonction de dissimilarité de TSclust à une matrice de données de séries temporelles. La fonction diss() prend un jeu de données et une chaîne spécifiant la dissimilarité TSclust à utiliser, et renvoie un objet dist avec toutes les dissimilarités par paire. Les arguments

Mesures de dissimilarité	Fonction dans TSclust
Approches sans modèle	
Sur la base de données brutes	
d_L (Euclidean)	diss.EUCL
d_F	diss.FRECHET
d_{DTW}	diss.DTW
Couvrir à la fois la proximité sur les valeurs et sur le comportement	
d_{CORT}	diss.CORT
Sur la base de corrélations	
d_{COR1}	diss.COR, beta = NULL
d_{COR2}	diss.COR et une valeur spécifiée pour le bêta
Basé sur des autocorrélations simples et partielles	
d_{ACF}	diss.ACF
d_{ACFU}	diss.ACF sans paramètres
d_{ACFG}	diss.ACF, $p \neq 0$
d_{PACF}	diss.PACF
d_{PACFU}	diss.PACF sans paramètres
d_{PACFG}	diss.PACF, $p \neq 0$
Basé sur des periodogrammes	
d_{PER}	diss.PER
d_{NP}	diss.PER, normalize = TRUE
d_{LNP}	diss.PER, normalize = TRUE, logarithm = TRUE
Basé sur des estimateurs spectraux non paramétriques	
$d_{W(LS)}$	diss.SPEC.LLR, method = "LS"
$d_{W(LK)}$	diss.SPEC.LLR, method = "LK"
d_{GLK}	diss.SPEC.GLK
d_{SD}	diss.SPEC.IDS
Basé sur la transformée en ondelettes discrète	
d_{DWT}	diss.DWT
Basé sur une représentation symbolique	
$d_{MINDIST.SAX}$	diss.MINDIST.SAX
Approches basées sur des modèles	
d_{PIC}	diss.AR.PIC
d_{MAH}	diss.AR.MAH
$d_{MAH_{ent}}$	diss.AR.MAH, dependence = TRUE
$d_{LCP_{Cep}}$	diss.AR.LPC.CEPS
Approches fondées sur la complexité	
d_{CID}	diss.CID
d_{PDC}	diss.PDC
d_{CDM}	diss.CDM
d_{NCD}	diss.NCD
Approche basée sur la prédiction	
d_{PRED_h}	diss.PRED

TABLE 7 – Distances dans TSclust

supplémentaires requis par la dissimilarité particulière doivent également être transmis à la fonction diss(). Dans notre exemple, diss() est utilisée comme suit.

```
ACF.dis <- diss(synthetic.tseries, "ACF", p = 0.05)
```

Maintenant, l'objet ACF.dis contient la matrice de dissimilarité calculée en appliquant ACF à chaque paire de séries dans synthetic.tseries et peut être pris comme point de départ pour plusieurs méthodes conventionnelles de regroupement. Par exemple, nous utilisons hclust() du paquet de base stats pour effectuer un algorithme de regroupement hiérarchique. Nous obtenons la solution à six clusters à partir de l'algorithme de hiérarchique en utilisant la fonction cutree() de stats.

```
ACF.hclus <- cutree(hclust(ACF.dis), k = 6)
```

Comme le nombre correct de groupes est connu, une technique de regroupement partiel telle que l'algorithme PAM (Partitioning Around Medoids), très répandu, est également

envisagée. Partitioning Around Medoids (PAM) est également envisagée. Nous utilisons la fonction pam() dans le paquet cluster.

```
library("cluster")
ACF.pamclus <- pam(ACF.dis, k = 6)$clustering
```

Regroupement à l'aide des distances et algo de dtwclust Toutes les distances dans le paquet dtwclust peuvent être composées en cinq méthodes, qui ont un effet significatif sur le maintien de l'invariance du désalignement local et de l'invariance de la phase. L'invariance de phase est toutefois limitée. Comme indiqué ci-dessus, la distance dtw pouvant traiter des ensembles de données de longueur inégale, la sélection d'un centre de groupe parmi ces ensembles devient problématique. Nous spécialisons donc les fonctions de chercher centroïde en fonction des propriétés des différentes distances

Algo	Fonction de centroïde	Distance Utilisée
TADPole		DTW(lb,ub)
K-medoides	PAM	DTW
K-means	DBA	DTW
K-means	Centroïde Soft-DTW	Soft-DTW
K-means	Extraction de Forme	SBD

TABLE 8 – Clustering de séries temporelles implémentées dans `dtwclust`, et leurs mesures de distance correspondantes.

Voici comment les utiliser :

```
#convertir le jeu de données
data <- t(as.matrix(synthetic.tseries))
#importer dtwclust
library(dtwclust)
# Réinterpolation linéaire à la même longueur
data <- reinterpolate(data, new.length = ncol(data))
# z-normalization
data <- zscore(data)
#dtw + DBA
pc_dtw <- tsclust(data, k = 6L, seed = 8L,
distance = "dtw_basic", centroid = "dba",
norm = "L2", window.size = 20L)
#K-shape
pc_ks <- tsclust(data, k = 6L, seed = 8L,
distance = "sbd", centroid = "shape")
#TADPole
pc_tp <- tsclust(data, k = 6L, type = "tadpole", seed = 8L,
control = tadpole_control(dc = 2.5, window.size = 20L))
#dtw + PAM
pc_k <- tsclust(data, k = 6L, seed = 8L,
distance = "dtw_basic", centroid = "pam")
#sdtw + Centroïde Soft
pc_soft <- tsclust(data, k = 6L, seed = 8L,
```

```
distance = "sdtw", centroid = "sdtw_cent")
```

Regroupement à l'aide des distances et algo de pdc Bien que le paquet TSclust permette également d'utiliser la distance PDC, le paquet pdc fournit de meilleurs algorithmes (introduit dans section 3.4.4) pour estimer les paramètres nécessaires à la distance, tels que la taille de la fenêtre.

```
library(pdc)
clust <- pdclust(synthetic.tseries)
```

6.4 Approches basées sur les caractéristiques

TSclust a deux représentations DWT, SAX, et est intégré à la distance.

```
SAX.dis <- diss(synthetic.tseries, "MINDIST.SAX", w = 20, alpha = 5)
SAX.plot(synthetic.tseries, w = 20, alpha = 5)
#DWT Distance Temporairement non supporté
DWT.dis <- diss(synthetic.tseries, "DWT")
```

Ou utilisez la méthode de représentation fournie par TSrepr.

```
data_dft <- repr_matrix(data, func = repr_dft, args = list(coef = 12), normalize = TRUE,
dim(data_dft))

res_km <- kmeans(data_dft, 5, nstart = 10)

# prepare data for plotting
data_plot <- melt(data.table(ID = 1:nrow(data_dft),
                               class = res_km$cluster,
                               data_dft),
                  id.vars = c("ID", "class"),
                  variable.name = "Time",
                  variable.factor = FALSE
                  )

data_plot[, Time := as.integer(gsub("V", "", Time))]

# prepare centroids
centers <- melt(data.table(ID = 1:nrow(res_km$centers),
                            class = 1:nrow(res_km$centers),
                            res_km$centers),
                  id.vars = c("ID", "class"),
                  variable.name = "Time",
                  variable.factor = FALSE
                  )

centers[, Time := as.integer(gsub("V", "", Time))]
```

```
# plot the results
ggplot(data_plot, aes(Time, value, group = ID)) +
  facet_wrap(~class, ncol = 2, scales = "free_y") +
  geom_line(color = "grey10", alpha = 0.65) +
  geom_line(data = centers, aes(Time, value), color = "firebrick1", alpha = 0.80, size = 1) +
  labs(x = "Time", y = "Load (normalised)") +
  theme_bw()
```

6.5 Approches basées sur un modèle

Chez TSclust, nous proposons trois distances basées sur des modèles ARIMA. ARIMA(p, d, q) combine les composantes d'autorégression (AR), de différenciation (I) pour rendre la série stationnaire, et de moyenne mobile (MA). En d'autres termes, nous supposons que chaque séquence provient d'un modèle ARIMA et qu'il faut d'abord estimer les paramètres du modèle correspondant. La distance est ensuite calculée à l'aide de la distance euclidienne ou de la matrice de corrélation.

$$(1 - \phi_1 L - \phi_2 L^2 - \cdots - \phi_p L^p)(1 - L)^d Y_t = (1 + \theta_1 L + \theta_2 L^2 + \cdots + \theta_q L^q) \varepsilon_t$$

Y_t représente la valeur de la série au temps t après d différenciations. L est l'opérateur de décalage, où $LY_t = Y_{t-1}$. Les termes $\phi_1, \phi_2, \dots, \phi_p$ sont les paramètres de l'autorégression (AR) qui mesurent l'impact des p valeurs passées sur la valeur actuelle. Les termes $\theta_1, \theta_2, \dots, \theta_q$ sont les paramètres de la moyenne mobile (MA) qui mesurent l'impact des q erreurs de prédiction passées sur l'erreur actuelle. ε_t est l'erreur de prédiction au temps t .

Aussi, BNPTSclust applique un modèle linéaire dynamique couplé à un algorithme bayésien non paramétrique, sous hypothèse de quelque probabilité a priori, utilisant un méthode de Monte-Carole de Chaîne de Markov Gibbs sampling pour estimer les cluster . Soit $y_i = \{y_{it} : t = 1, \dots, T\}$, $i = 1, \dots, n$, une série temporelle et $y = \{y_1, \dots, y_n\}$ représente l'échantillon de séries temporelles pris en considération. Chaque observation est modélisée comme si elle suivait un modèle linéaire dynamique de la forme :

$$y_{it} = F'_{it} \theta_{it} + \varepsilon_{it} \quad (1)$$

$$\theta_{it} = \rho \theta_{i,t-1} + \nu_{it} \quad (2)$$

où $\varepsilon_{it} \sim \mathcal{N}(0, \sigma_{\varepsilon i}^2)$ et $\nu_{it} \sim \mathcal{N}(0, \sigma_{\theta i}^2)$.

Cependant, nous n'aborderons pas ses aspects théoriques et pratiques en raison de contraintes de temps et d'espace.

6.6 Evaluation

Pour un jeu de données dont nous connaissons la classification, nous expérimentons en deux étapes : la première consiste à voir si les distances que nous avons choisies sont adaptées à ce jeu de données. La seconde consiste à déterminer le nombre de groupes dans lesquels il est le plus logique de diviser l'ensemble de données. Pour les ensembles de données dont nous ne connaissons pas la classification, nous utilisons notre expérience pour choisir la distance, après quoi nous passons à la deuxième étape.

Première étape : 1 Voisin le plus proche pour la matrice de distance Évalue une matrice de distance de regroupement dans le cadre d'un schéma d'apprentissage supervisé : validation croisée du plus proche voisin. Cela permet d'obtenir une estimation approximative de la pertinence d'une fonction de distance pour la discrimination entre les classes si la vérité de base est connue.

```
library("pdc")
library("TSclust")
library("dtwclust")
true_cluster <- rep(1:6, each = 3)
loo1nn(clust,true_cluster)
```

Deuxième étape : Les indices internes et externe pour clustering évaluation Mesure de la distance inter-groupe et de la distance intra-groupe à l'aide des indices de section précédente.

```
library("NBclust")
library("clusterSim")
cluster.evaluation(true_cluster, ACF.hclus)
cvi(ACF.hclus,true_cluster, type ="external")
C <- index.S(ACF.dist,ACF.hclus)
```

7 Problème Consommation d'électricité

Il y a 50 séries temporelles d'une longueur de 672, en particulier des séries temporelles d'une longueur de 2 semaines. Il est évident que la dimensionnalité est trop élevée et que la malédiction de la dimensionnalité peut se produire. C'est pourquoi nous devons réduire la dimensionnalité d'une manière ou d'une autre. L'une des meilleures approches consiste à utiliser des représentations de séries temporelles afin de réduire la dimensionnalité, le bruit et de mettre l'accent sur les principales caractéristiques des séries temporelles.

Nous avons constaté, sur la base du contexte de l'ensemble de données et des images 24 présentées par les données, que la consommation d'électricité des clients est basée sur un cycle d'un jour et 48 points temporels.

Nous étudions la consommation d'électricité des clients cycle par cycle et classons les clients ayant une consommation similaire.

Utilisons l'une des méthodes de représentation basées sur un modèle de base, le **profil saisonnier moyen**. Une remarque très importante ici, la normalisation des séries temporelles est une procédure nécessaire avant tout regroupement ou classification de séries temporelles. Cela est dû au fait que nous voulons extraire des courbes typiques de consommation et ne pas faire de regroupements basés sur un montant de consommation.

Nous commençons par la méthode kmeans et hierarchique(complete) pour les mesures et choisissons le nombre de groupes en fonction de les Indices de dunn , de silhouette et de Davies-Bouldin.

Après expérimentation, nous avons constaté que les méthodes kmeans et hiérarchique nous permettaient de diviser l'ensemble de données en deux groupes selon les indices de dunn et silhouette. Les deux méthodes ayant les mêmes catégories de regroupement, nous ne donnons

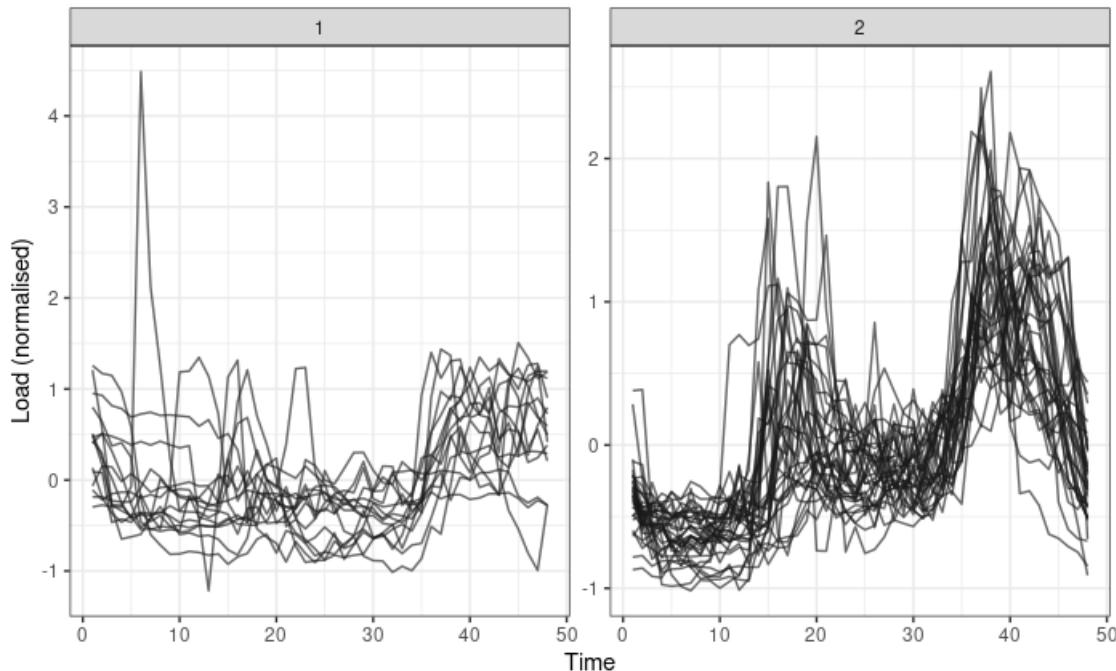


FIGURE 27 – Profil saisonnier moyen + Kmeans + Euclidean + Silhouette

ici que les résultats de la classification des kmeans avec distance euclidean pour l'indice de silhouette.

Avec l'image 27, nous pouvons clairement distinguer que, selon la nature de la méthode représentative "Profil saisonnier moyen", nous pouvons classer la consommation d'électricité des différents ménages en deux catégories : ceux dont la consommation journalière fluctue beaucoup en milieu de journée et le soir, et ceux dont la consommation journalière est très moyenne, avec une tendance à augmenter lentement le soir.

Pour l'indice de Davies-Bouldin, je ne recommande que la méthode des kmeans car l'indice de Davies-Bouldin calcule les distances intergroupes à l'aide de la centroïde, dont le concept est proche de celui de la méthode des kmeans.

Si nous utilisons l'indice de Davies-Bouldin pour nous guider dans la catégorisation du nombre de groupes, nous constatons que le kmeans donne neuf groupes 28, avec des séries temporelles plus étroitement ajustées dans chaque groupe, et nous constatons que les courbes de consommation d'électricité sont également nettement différentes pour les différents groupes. Si nous voulons catégoriser les modèles de consommation d'électricité en détail, je recommande personnellement l'indice de Davies-Bouldin pour une catégorisation plus détaillée.

Nous pouvons utiliser la distance CORT si nous nous intéressons non seulement à leur approximation en termes de magnitude, mais aussi à leur corrélation en termes d'augmentation ou de diminution dans le temps, qui est définie par la formule suivante.

$$CORT(X_T, Y_T) = \frac{\sum_{t=1}^{T-1} (X_{t+1} - X_t)(Y_{t+1} - Y_t)}{\sqrt{\sum_{t=1}^{T-1} (X_{t+1} - X_t)^2} \sqrt{\sum_{t=1}^{T-1} (Y_{t+1} - Y_t)^2}}.$$

$$d_{CORT}(X_T, Y_T) = \phi_k [CORT(X_T, Y_T)] \cdot d(X_T, Y_T),$$

où X_T, Y_T séries temporelles avec dimension T , $\phi_k(\cdot)$ est une fonction d'accord adaptative

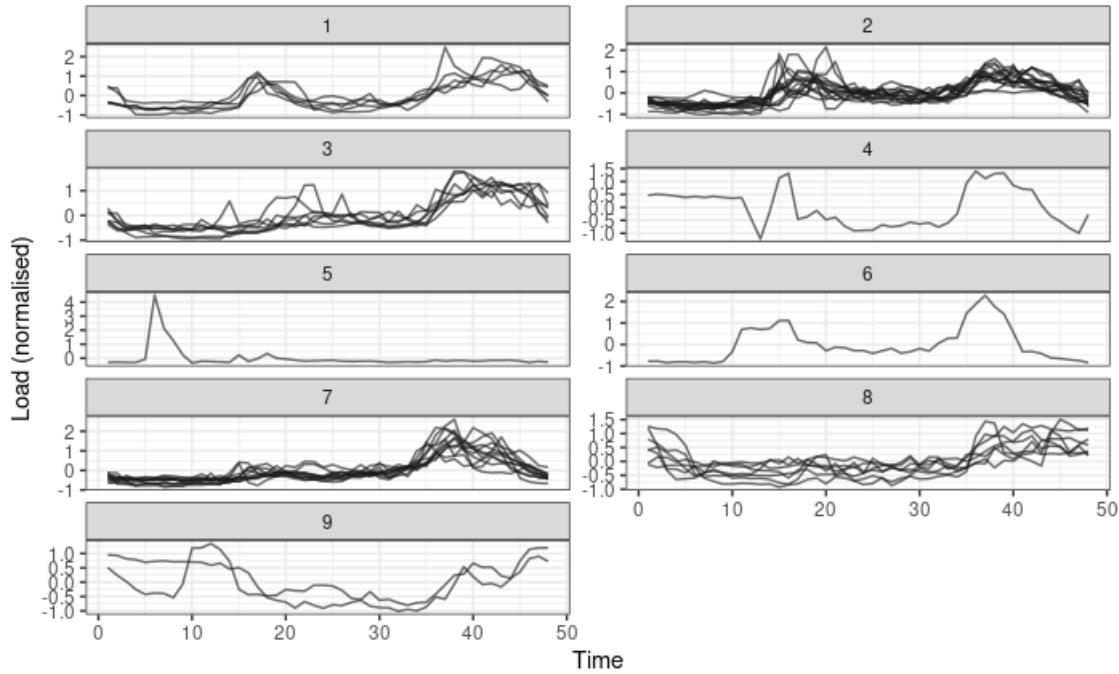


FIGURE 28 – Profile sainsonnier moyen + Kmeans + Euclidean + DB

permettant de moduler automatiquement une distance conventionnelle de données brutes $d(X_T, Y_T)$ en fonction de la corrélation temporelle.

Nous avons également suivi la configuration originale, où la représentation est Profil sainsonnier moyen , avec la méthode kmeans, la distance est CORT, et le nombre de groupes catégoriels est déterminé sur la base de l'indice de silhouette et de l'indice de DB, respectivement. Les résultats sont dans les figure 29 , 30

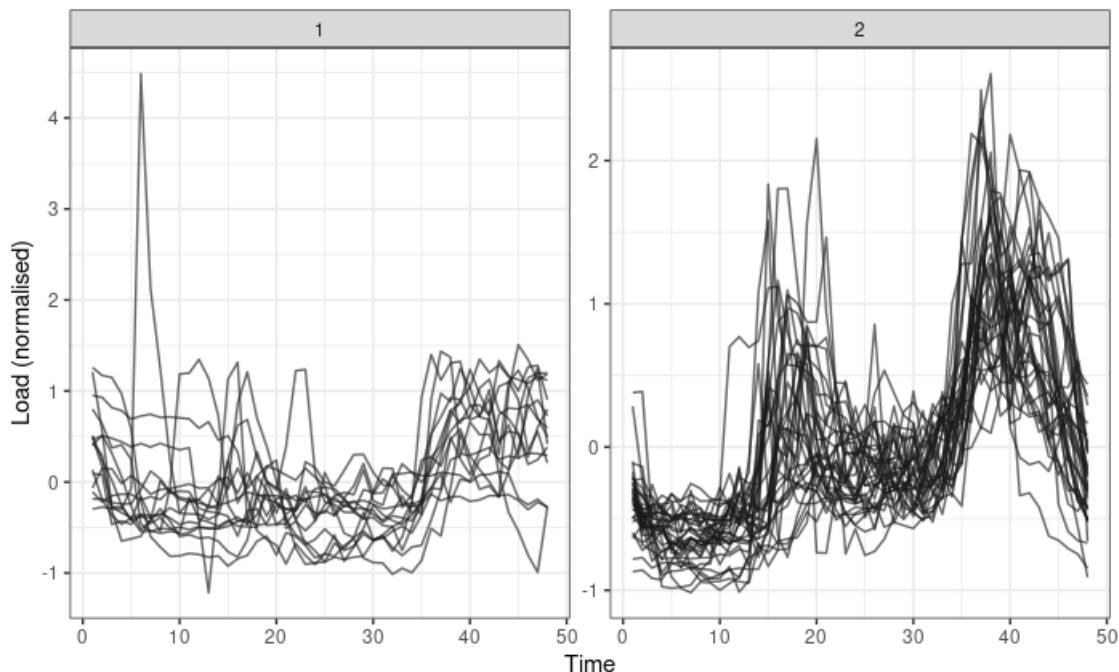


FIGURE 29 – Profile sainsonnier moyen + Kmeans + CORT + Silhouette

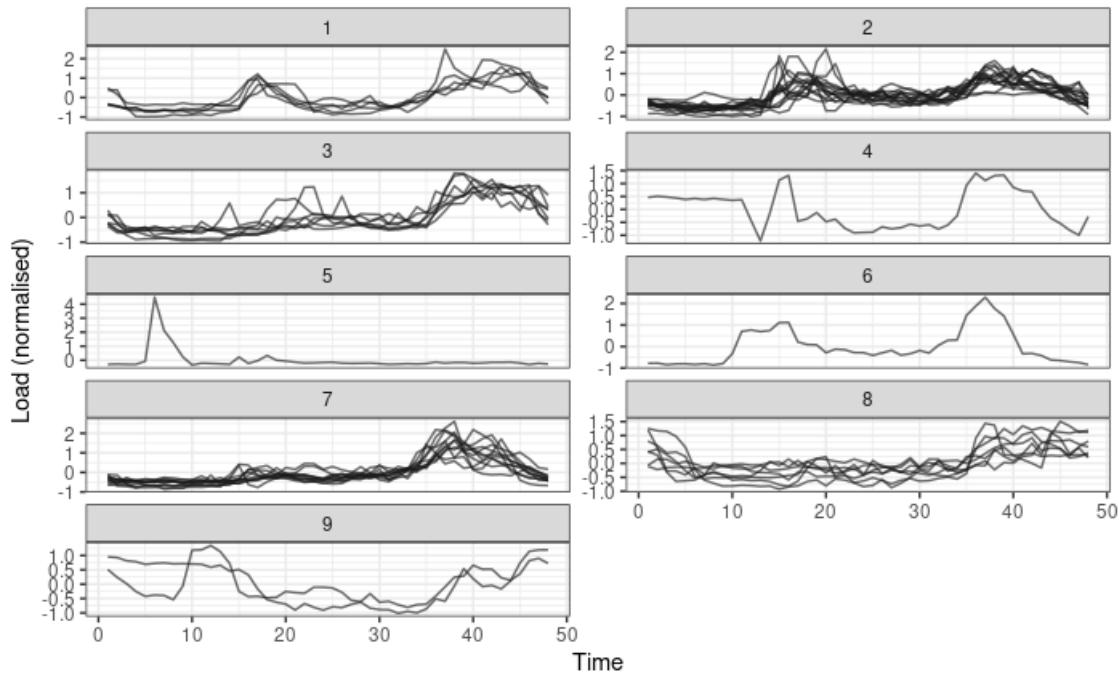


FIGURE 30 – Profile sainsonnier moyen + Kmeans + CORT + DB

Nous n'avons constaté aucun changement dans les résultats du regroupement par rapport à la distance euclidienne.

Nous utilisons maintenant la représentation feaclip pour extraire les caractéristiques de la consommation d'électricité de chaque jour et regrouper la consommation sur une période de deux semaines. Le regroupement est déterminé par l'indice de dunn à l'aide de la méthode des kmeans et de la distance euclidienne .

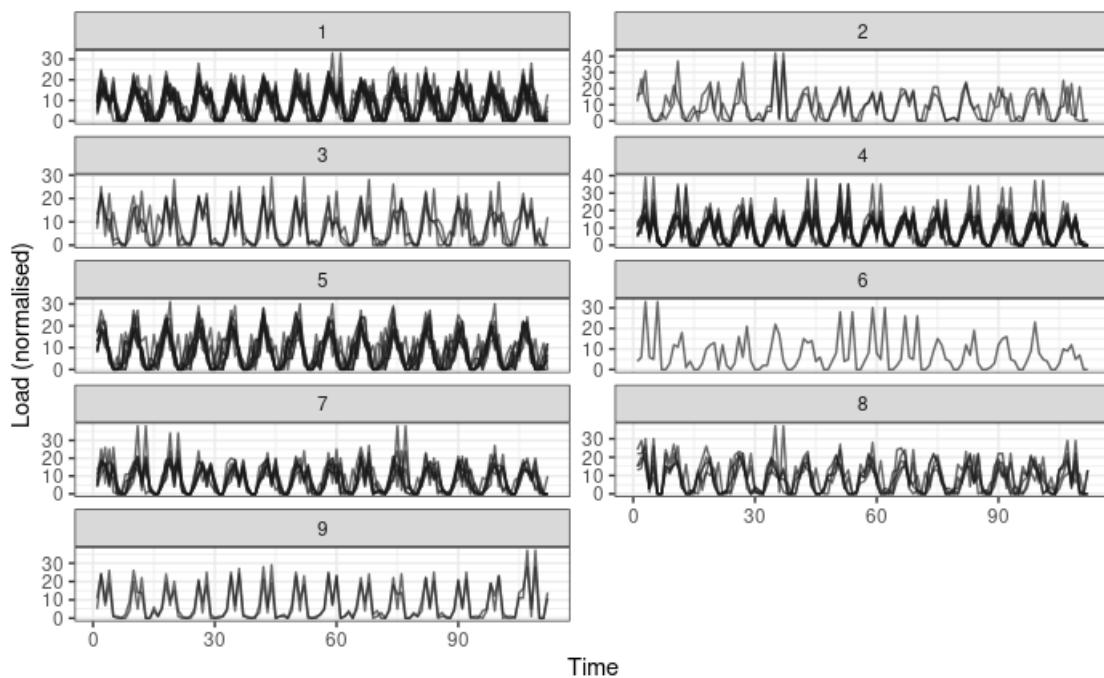


FIGURE 31 – Feaclip + Kmeans + Euclidean + DUNN

L'image 31 montre que la consommation d'électricité est divisée en 9 groupes et que chaque groupe a une consommation d'électricité légèrement différente, mais la série chronologique de chaque groupe a la forme d'un cycle, ce qui prouve que le modèle de consommation d'électricité de chaque ménage est à peu près le même chaque jour.

8 Problème Les signatures de forme des objets

Sur la base de nos analyses précédentes de l'ensemble de données sur les étoiles, nous avons constaté que nos séries temporelles sur les formes formées par les pentagones et les quadrangles étaient extrêmement similaires sur les images. Nous avons donc voulu explorer ce qui pouvait être fait pour élargir la distance entre ces deux ensembles. Nous avons d'abord choisi les distances PD, ACF et COR pour tester leur comportement en cas de répétitions régulières de motifs, de désalignements sur la ligne de temps et de déflation ou inflation numérique. Cela signifie que l'invariance de phase et l'invariance d'échelle global sont garanties.

Si nous devons nous assurer de l'invariance d'échelle globale, nous pouvons effectuer la z-normalisation des données, mais ces trois méthodes ne sont pas sensibles à l'échelle globale.

Tout d'abord, une brève introduction à la définition de la distance pour le cor, l'acf et le pacf est donnée.

Distance basé sur corrélation

Un premier et simple critère de dissimilarité consiste à considérer le facteur de corrélation de Pearson entre X_T et Y_T donné par

$$COR(X_T, Y_T) = \frac{\sum_{t=1}^{T-1} (X_t - \bar{X}_T)(Y_t - \bar{Y}_T)}{\sqrt{\sum_{t=1}^{T-1} (X_t - \bar{X}_T)^2} \sqrt{\sum_{t=1}^{T-1} (Y_t - \bar{Y}_T)^2}},$$

avec \bar{X}_T et \bar{Y}_T les valeurs moyennes des réalisations en série X_T et Y_T respectivement. Golay, Kollias, Stoll, Meier, Valavanis et Boesiger (2005) construisent un algorithme k-moyens flou utilisant les deux distances basées sur la corrélation croisée suivantes :

$$d_{COR.1}(X_T, Y_T) = \sqrt{2(1 - COR(X_T, Y_T))},$$

et

$$d_{COR.2}(X_T, Y_T) = \sqrt{\left(\frac{1 - COR(X_T, Y_T)}{1 + COR(X_T, Y_T)}\right)^\beta}, \quad \text{avec } \beta \geq 0.$$

Distances basées sur l'autocorrélation

Dans l'analyse des séries chronologiques, il est courant de tracer l'ACF et le PACF en fonction des décalages temporels. Ces tracés sont appelés corrélogrammes et permettent de visualiser les modèles variables dans le temps par le biais de la dépendance sérielle. Les corrélogrammes sont très utiles pour découvrir les cycles, la saisonnalité ou d'autres modèles de séries temporelles, ainsi que pour déterminer un modèle de série temporelle approprié pour les données observées.

Soit $\hat{\rho}_{X_T} = (\rho_{1,X_T}, \dots, \rho_{L,X_T})^T$ et $\hat{\rho}_{Y_T} = (\rho_{1,Y_T}, \dots, \rho_{L,Y_T})^T$ les vecteurs d'autocorrélation estimés de X_T et Y_T respectivement, pour un certain L tel que $\hat{\rho}_{i,X_T} \approx 0$ et $\hat{\rho}_{i,Y_T} \approx 0$ pour $i > L$. Galeano et Peña (2000) définissent une distance entre X_T et Y_T comme suit.

$$d_{ACF}(X_T, Y_T) = \sqrt{(\hat{\rho}_{X_T} - \hat{\rho}_{Y_T})^T \Omega (\hat{\rho}_{X_T} - \hat{\rho}_{Y_T})},$$

où Ω est une matrice de poids.

Des distances analogues peuvent être construites en considérant les fonctions d'autocorrélation partielle (PACF) au lieu des ACF. Ci-après, les notations d_{PACF_U} et d_{PACF_G} seront utilisées pour désigner la distance euclidienne entre les coefficients d'autocorrélation partielle estimés avec des poids uniformes et avec des poids géométriques décroissant avec le décalage, respectivement.

Puisque nous possédons déjà les étiquettes du regroupement, nous expérimentons d'abord son 1 Voisin le plus proche pour la matrice de distance (pour l'ACF et le PACF, des poids uniforme est utilisée, et $d_{COR,1}$ est utilisée)

Nous observons que la précision de lo01nn peut être de 100% pour toutes les distances. Une précision élevée est nécessaire pour garantir que les résultats de la classification sont conformes aux attentes.

Examinons à nouveau ses regroupements. Nous utilisons des regroupement hiérarchique avec "complete linkage", sur quatre distances.

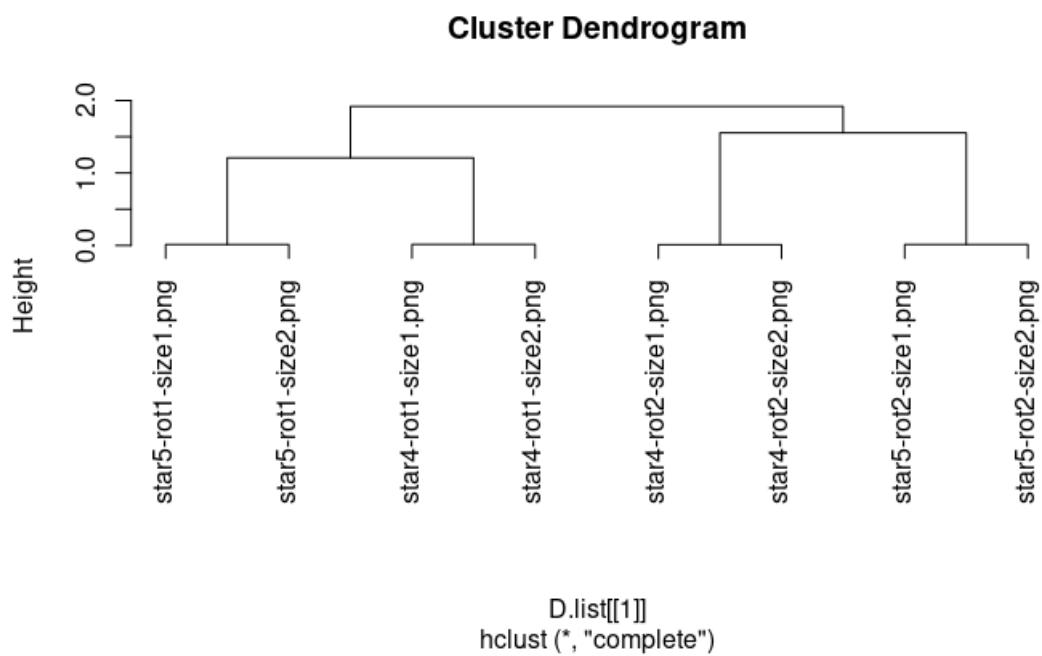


FIGURE 32 – hiéraarchie(complete) de CORT

Pour le résultat de la classification de la distance CORT 32, nous constatons qu'il ne peut pas jouer un rôle dans le maintien de l'invariance de phase, et la conclusion correspondante peut être tirée de son équation de définition de la distance.

Les résultats du regroupement pour les deux distances acf 33 et pacf 34 sont bons et préservent efficacement l'invariance de phase et l'invariance d'échelle globale. D'après le dendrogramme hiérarchique, PACF est moins impressionné par le nombre de répétitions de motifs que ACF. Comme PACF prend en compte les effets conditionnelles de tous les facteurs à des intervalles décalage sur les coefficients de corrélation, il est plus complet et permet de compenser les effets d'un nombre réduit de répétitions de motifs.

Enfin, Pour le clustering de PD, Nous avons fixé un délai minimum plus important de 5 afin d'accroître la robustesse face aux erreurs de discréétisation lors de la recherche du délai

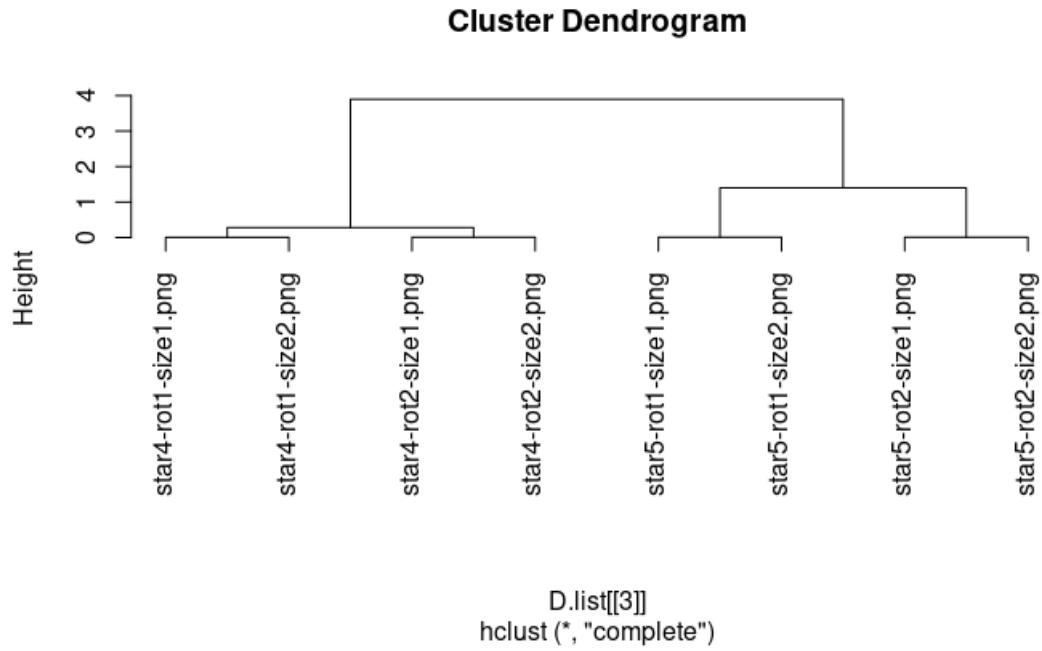


FIGURE 33 – hiéraarchie(complete) de ACF

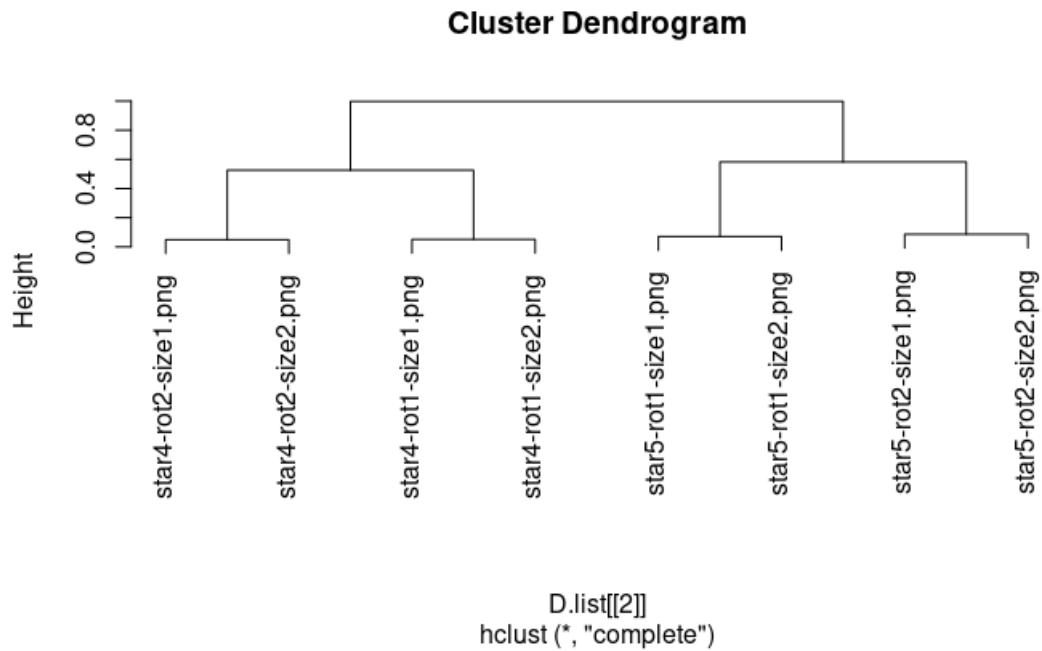


FIGURE 34 – hiéraarchie(complete) de PACF

optimal. La dimension d'intégration a été automatiquement choisie comme étant $m = 4$ et le délai comme étant $t = 9$.

Une étoile à quatre points et une étoile à cinq points, chacune dans quatre transformations différentes d'échelle et de rotation, sont correctement classées dans deux catégories différentes. Comme l'ACF, il est également influencé par le nombre de répétitions du motif.

Nous choisissons maintenant la classification fournie par dtwclust. Les distances sont les distances introduites sdb, dtw, softdtw. Nous choisissons la classification hiérarchique, mais

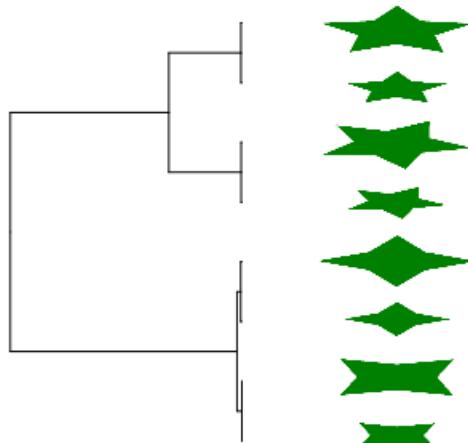


FIGURE 35 – hiérarchie(complete) de PD

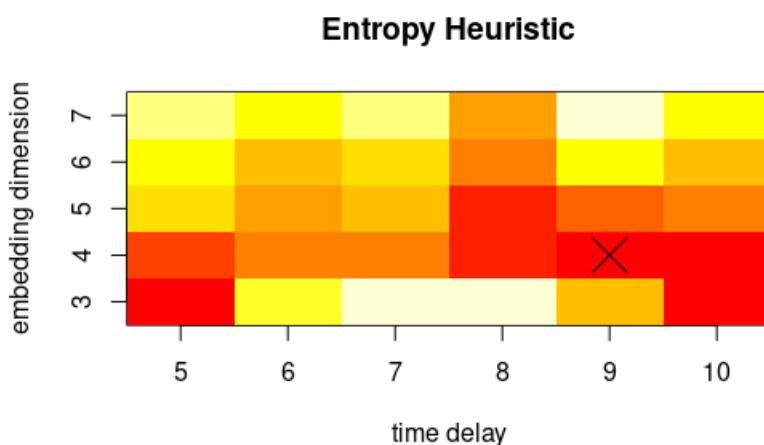


FIGURE 36 – Entropie de configuration de PD

le dtwclust ne peut fournir que "average linkage". les autres "liens" n'ont pas été créés bien qu'il soit dans le manuel.

Tout d'abord, sdtw et dtw, et SBD sont sensibles à la magnitude numérique, nous normalisons donc les données. Ensuite, nous constatons qu'ils ont tous deux une précision de 100% pour loolnn, ce qui n'a pas beaucoup de sens ici car nous normalisons les données de manière à ce qu'il y ait deux données identiques.

Nous constatons que les résultats de classification de sbd 39 sont très bons. Le même reçoit également le fléau de moins de répétitions de motifs.

Les résultats de la classification de dtw 37 et sdtw 38, en revanche, ne sont pas satisfaisants, bien que la distance dtw et ses variantes s'adaptent aux désalignements locaux et maintiennent en conséquence l'invariance de phase dans une certaine mesure. Mais il ne détecte pas bien les dissimilarités dans le cas de deux catégories aux formes similaires.

Pour illustrer davantage l'utilité potentielle de ces distances, un autre ensemble de données comprenant des objets de trois classes différentes a été créé : verres, bouteilles et poissons. Les fichiers ont été sélectionnés sur <http://openclipart.org/>. Nous utilisons les méthodes suivantes, PACF + hiérarchie(complete), SBD+hier, PD + hiérarchi(complete),

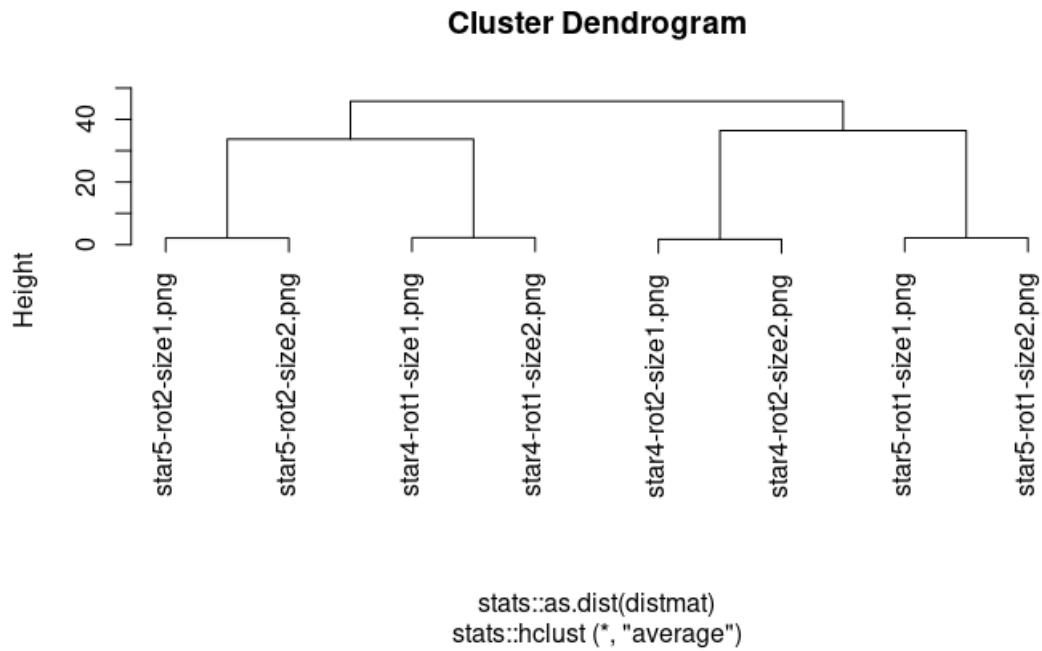


FIGURE 37 – hiérachie(average) de dtw

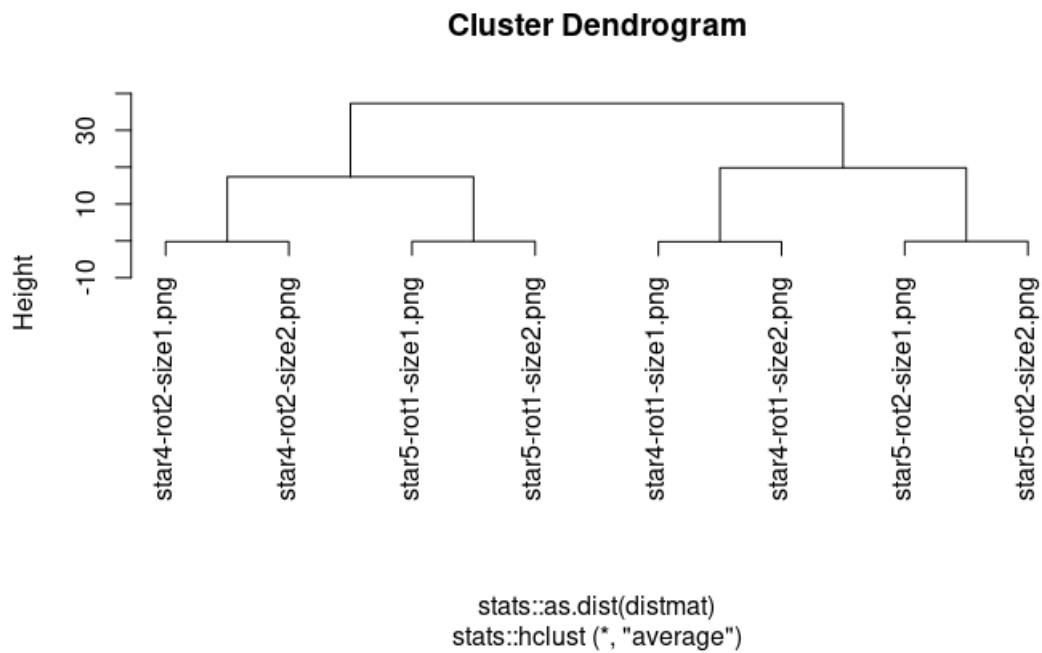


FIGURE 38 – hiérachie(average) de sdtw

dtw(symmetric1, windows= 20L) + hier, softdtw + hier, TADPOLE.

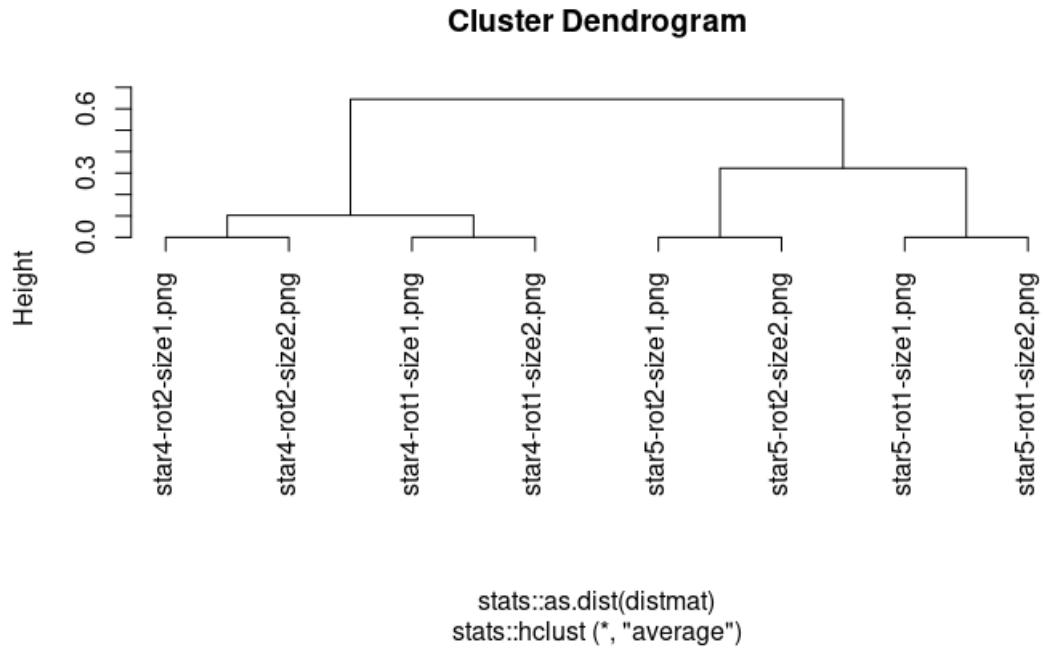


FIGURE 39 – hiéraichi(average) de sbd

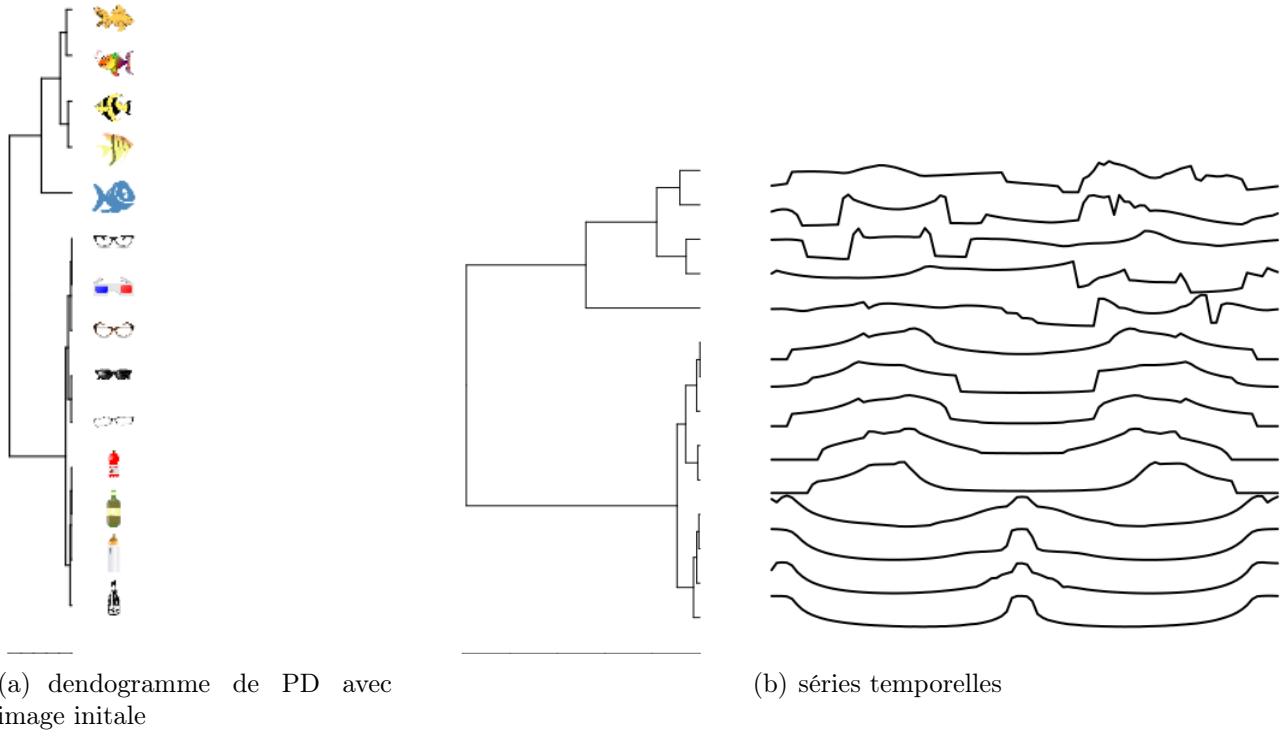


FIGURE 40 – PD + hiéarichi(complete)

Nous avons d'abord expérimenté la méthode de clustering hiérarchique de PD distance. pdc nous a fourni un dendrogramme hiérarchique des cas de classification des graphes initiaux (bouteille, verres, poisson). Les images des séries temporelles correspondantes nous sont également fournies 40. En ce qui concerne le graphes original, nous voyons qu'il y a

différentes directions, différentes sortes de poissons et une série de bouteilles et de verres aux contours similaires. Du point de vue des images des séries temporelles, nos images temporelles des poissons ne sont pas très similaires, mais pour les bouteilles et les verres, les séries temporelles résultantes sont similaires d'un groupe à l'autre. Pour que les bouteilles et les verres puissent mieux se regrouper, il faut que la distance maintienne une certaine invariance du désalignement local. Pour mieux classer les poissons, nous nous concentrerons sur l'amélioration de l'analyse de similarité des caractéristiques locales (indépendamment du moment de l'apparition).

Il n'est pas exagéré de dire que ce sont les caractéristiques offertes par pd distance. Nous pouvons voir que tous les poissons, les verres et les bouteilles sont bien regroupés 40.

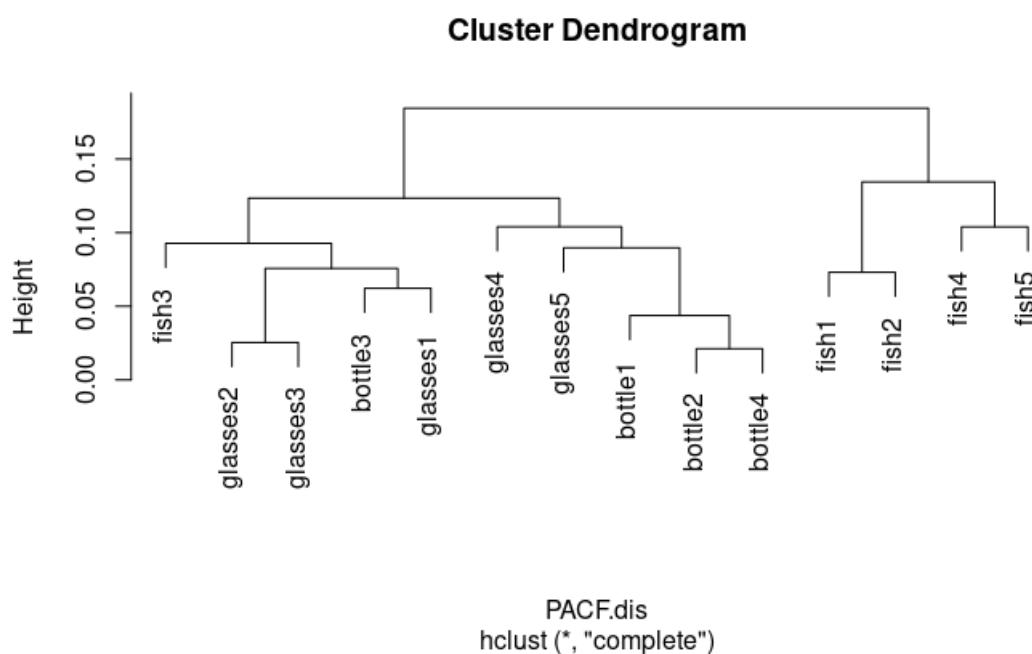


FIGURE 41 – hiérarchie(complete) de PACF des données des images complexes

Nous examinons maintenant les résultats du regroupement pour les distances pacf 41 et acf 42, et nous constatons qu'elles ne regroupent plus correctement aucun type de forme, qu'elles n'assurent pas l'invariance du désalignement local et qu'elles n'extraient pas la similarité des caractéristiques locales.

On fait de normalisation des données pour les suivantes, en maintenant l'invariance d'échelle globale.

Nous constatons que, quelle que soit la classification de ces distances en sdtw 44, dtw 43 et sbd 45, les bouteilles et les verres sont les premiers à se regrouper, ce qui prouve qu'ils assurent l'invariance du désalignement local. En examinant de plus près les images des séries temporelles, nous constatons que les images des poissons numéro trois et quatre, respectivement, présentent dans une certaine mesure la tendance des séries temporelles du graphique des lunettes, de sorte que dans le cas de la classification de dtw, nous constatons qu'ils sont incorrectement classés dans la classe des lunettes. sdtw peut avoir un avantage absolu affaibli pour l'invariance du désalignement local en raison de la prise en compte de tous les cas d'alignement dans dtw. Avec sdtw, nous pouvons voir qu'il obtient un meilleur résultat de classification qu'avec dtw, et que seul le poisson numéro trois est mal classé dans la classe des lunettes.

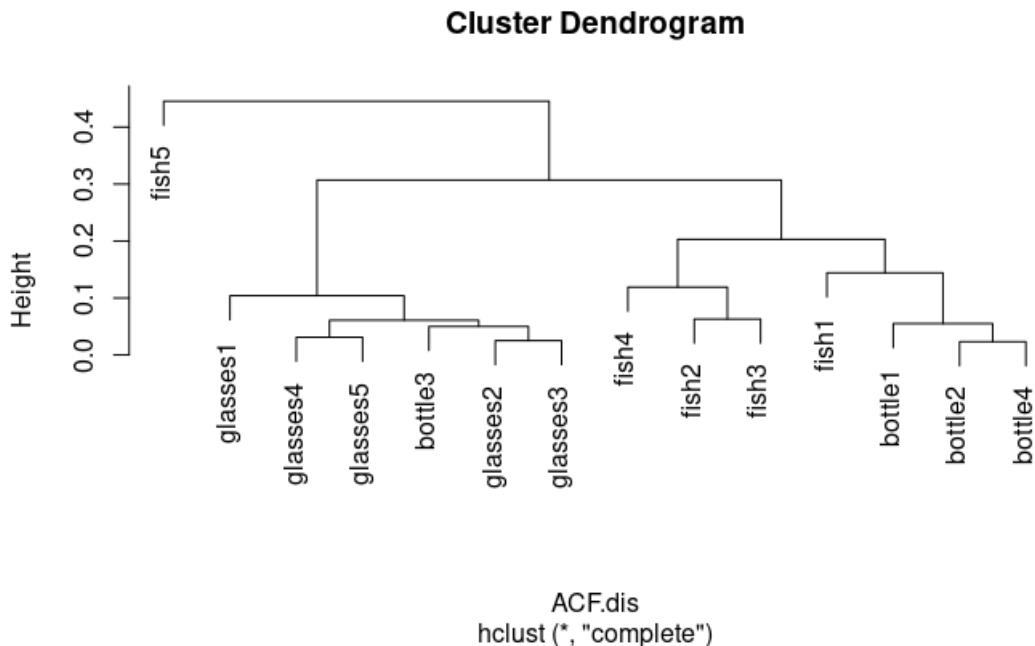


FIGURE 42 – hiéraichie(complete) de ACF des données des images complexes

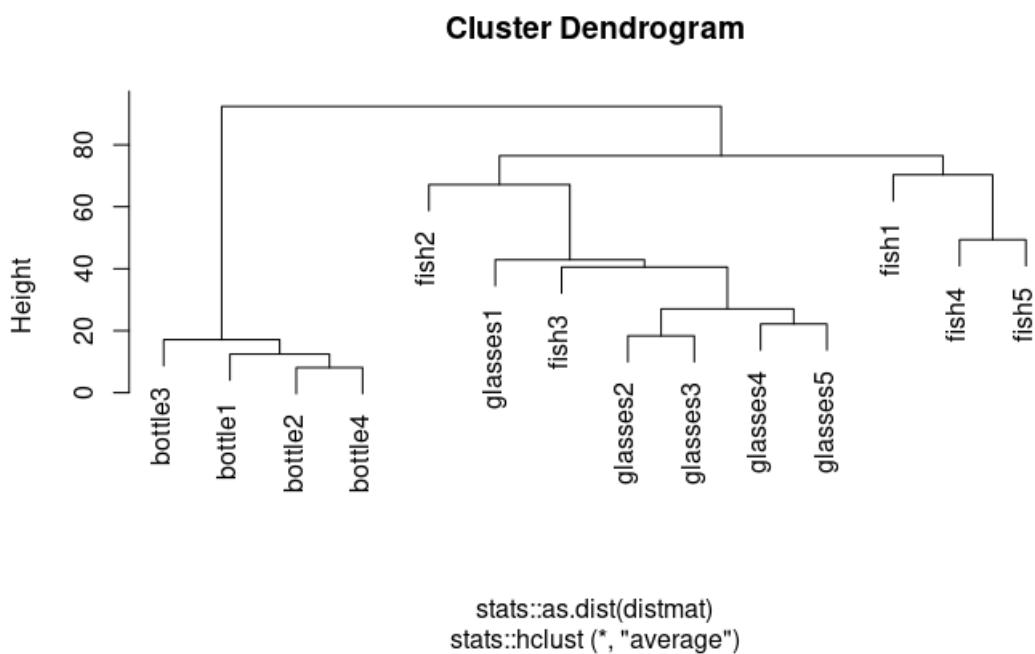


FIGURE 43 – hiéraarchie(average) de DTW des données des images complexes

Nous comparons sbd et dtw et constatons que dtw est plus intuitif à interpréter, et que sbd sélectionne des coefficients de corrélation que nous avons parfois du mal à comprendre directement à partir de l'image. Nous constatons également que, bien que sbd maintienne l'invariance du désalignement local, il est encore fortement influencé par la corrélation.

Nous constatons que les résultats de tadpole 9 sont très mauvais, ce qui prouve que les distances intergroupes avec la formation dtw ne sont pas très grandes, de sorte qu'il nous est difficile de résoudre le problème correspondant avec un algorithme de type densité comme

Cluster Dendrogram

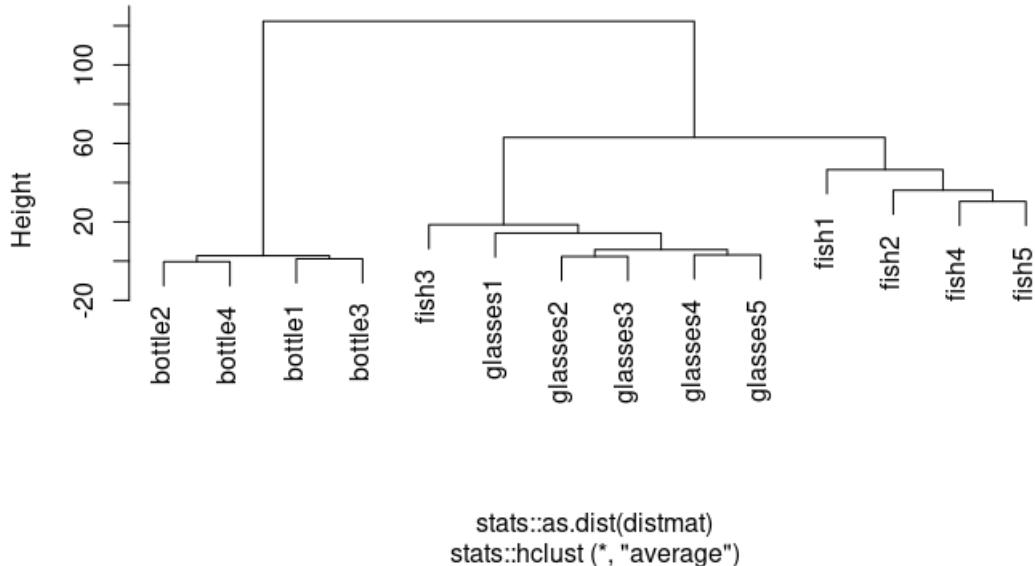


FIGURE 44 – hiéraarchie(average) de SDTW des données des images complexes

Cluster Dendrogram

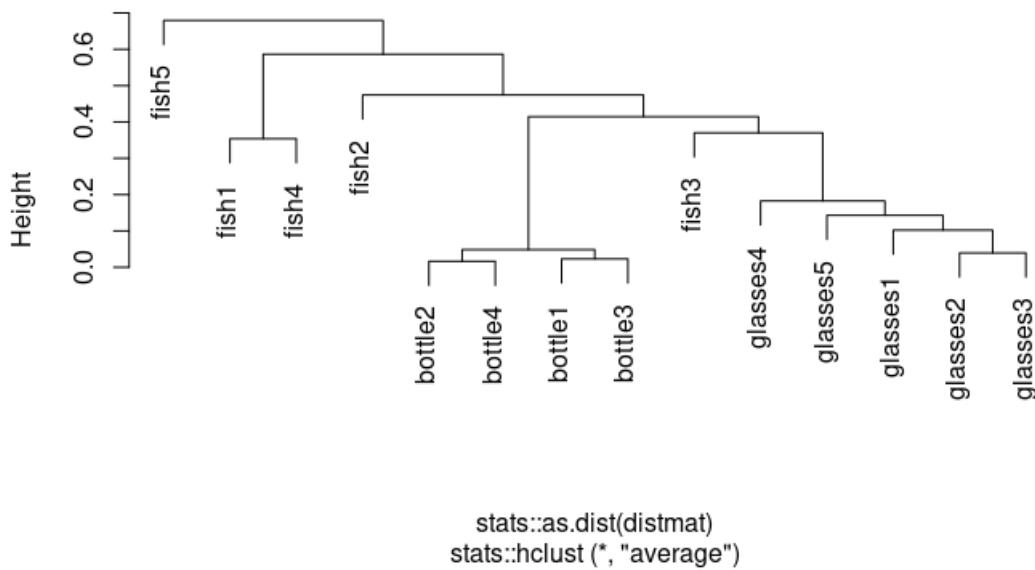


FIGURE 45 – hiéraarchie(average) de SBD des données des images complexes

tadpole.

On étudie également si, étant donné que PD fournit un bon regroupement, l'utilisation de la distance dans le cadre d'une compression sans perte peut également donner lieu à une situation de regroupement similaire. Nous commençons par transformer les données en chaînes de caractères à l'aide de la méthode sax, une étape qui permet également d'agréger les informations provenant des segments locaux. Ensuite, chaque série temporelle est compressée à l'aide d'une méthode de type Huffman pour l'extraction des caractéristiques ce qui, à

	Tadpole
fish1	3
fish2	3
fish3	3
fish4	3
fish5	3
bottle1	1
bottle2	2
bottle3	3
bottle4	2
glasses1	3
glasses2	3
glasses3	3
glasses4	3
glasses5	3

TABLE 9 – TADPole des données des images complexes

l'instar de la méthode PD, constitue également une forme de résumé de la distribution des caractéristiques locales.

Nous utilisons la distance CDM, définie comme suit.

La dissimilarité basée sur la compression CDM est calculée :

$$d(x, y) = \frac{C(xy)}{(C(x) + C(y))}$$

où $C(x)$ et $C(y)$ sont les tailles en octets des séries compressées x et y , et $C(xy)$ est la taille en octets de la série x et y concaténées. L'algorithme utilisé pour la compression des séries est choisi avec type peut être "gzip", "bzip2" ou "xz",

Nous avons constaté qu'en utilisant l'PAA pour réduire la longueur de la série temporelle de 100 à 50, et lorsque le nombre d'alphabets SAX est fixé à 8, nous pouvons obtenir un résultat de regroupement très précis 46 en utilisant la distance CDM, mais lorsque le nombre de caractères SAX et la longueur de la réduction de l'PAA varient, nous avons constaté que le regroupement n'est pas toujours précis. Par conséquent, si nous pouvons proposer un algorithme pour prédire les paramètres sélectionnés tels que la distance pd en utilisant l'entropie, il n'y a pas encore d'algorithme définitif proposé.

9 Problème 18 paires de séries temporelles couvrant différents domaines

Comme dans nos analyses précédentes de ce jeu de données paired.tseries, il est composé de séries temporelles provenant de différents domaines, dix-huit paires de trente-six séries temporelles au total. Bien que nous ayons analysé dans notre analyse précédente qu'il est nécessaire de conserver de nombreux invariants pour réduire la distance à l'intérieur du groupe. Toutes les séries temporelles que j'ai rencontrées sont évaluées en fonction de leurs caractéristiques locales, qui sont classées en deux catégories selon qu'elles sont affectées par le temps ou non. La première catégorie est celle où l'importance du moment doit être prise en compte, comme dans l'analyse des cours boursiers et de la consommation d'électricité.

Cluster Dendrogram

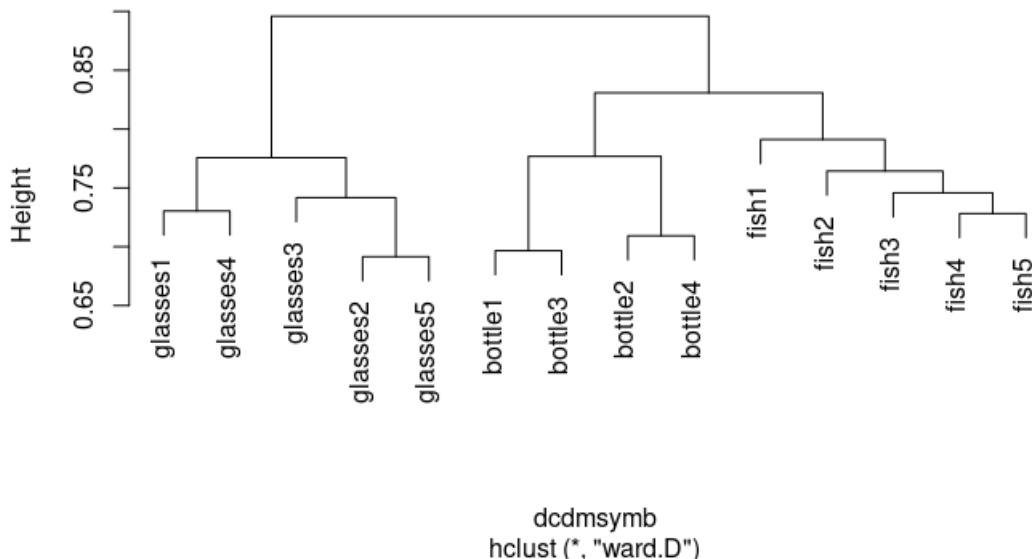


FIGURE 46 – hiéraarchie(average) de SAX + CDM des données des images complexes

Dans l'autre catégorie, l'importance du temps n'a aucune incidence sur l'analyse des caractéristiques locales, comme le regroupement de séries temporelles présentant des formes et des motifs similaires. La situation dans laquelle nous nous trouvons actuellement relève clairement de la deuxième catégorie, où les séries temporelles provenant de différents domaines et présentant des caractéristiques locales différentes n'ont pas besoin d'être prises en compte en même temps que les moments. D'une manière générale, nous pouvons donc analyser le problème du regroupement des séries temporelles à l'avenir de deux manières : premièrement, pour voir si les caractéristiques locales doivent être intégrées pour prendre en compte le temps. Deuxièmement, nous pouvons analyser si la similarité de ces caractéristiques locales dépend fortement de certains invariances, et si c'est le cas, nous pouvons choisir de définir différentes distances pour différents invariances. Nous disposons maintenant de pd comme distance as qui peut analyser de manière sensible la plupart des caractéristiques locales, qui ne dépendent pas des moments et qui atteignent tous les invariances dans une certaine mesure. Sa limite est qu'il n'est pas capable d'analyser l'étendue de la croissance numérique, une situation qui est rare dans le regroupement de séries temporelles.

Nous avons d'abord choisi la métrique loolnn pour voir l'effet de nos distances sur les distances intergroupes.

CORT	EUCL	DTW	PACF	ACF	PDC
30.6	27.8	41.7	69.4	69.4	72.2

TABLE 10 – précision des matrice de distance par validé croisé de paried.tseris

Nous obtenons que dans ce cas, les précisions validés croisées des matrices de dissimilarité sont toutes inférieures à la PDC 10.

Les dendrogrammes hiérarchiques sont difficiles à observer lorsque leur nombre est élevé. Nous choisissons alors la mesure Q. la mesure Q ad hoc compte le nombre de paires correctement retrouvées au niveau de regroupement le plus bas divisé par le nombre total de paires.

Ainsi, la mesure évalue la structure connue du niveau de regroupement le plus bas sans tenir compte de la structure inconnue de niveau supérieur. DTW est moins applicable lorsque la

EUCL	DTW	PACF	PDC (m=5,t=1)	SAX(w=343,alpha =27)CDM	PDC (m=5,t=8)
27.8	33.3	61.1	72.2	1	1

TABLE 11 – Q-mesure de hiérarchi sur paried.tseris

longueur de la série temporelle dépasse 1000 en raison du temps de calcul important requis.

Nous avons constaté que 61% des PACF ont réalisé une fusion au sein du groupe, tandis que 72 % des distances PD avec des paramètres prédis par l'entropie ont réalisé une fusion au sein du groupe. Nous continuons à explorer les distances de regroupement PD et SAX+CDM et nous constatons que lorsque PAA choisit 343, et que le nombre de listes alphabétiques de SAX atteint 27, sa mesure Q peut atteindre 1 ; pour PD qui choisit m=5 et t=8, sa Q-mesure peut également atteindre 1.

10 une collection de séries temporelles synthétiques

L'objectif de cette section est de nous donner une idée des modèles probabilistes des séries temporelles et des distances qu'ils génèrent en conséquence. Pour ces six modèles autorégressifs du premier ordre énumérés dans le tableau 6.

Commençons par quelques distances liées à la modélisation probabiliste des séries temporelles. Tout d'abord, il y a nos vieux amis PACF et ACF, qui sont des distances utilisées pour estimer le nombre de paramètres dans les modèles probabilistes AR et MA, respectivement. La deuxième est la distance basée sur les périodogrammes comme IP, LNP qui, selon le Théorème de Wold, est liée au coefficient d'autocorrélation de l'ACF. Le périodogramme d'une série est un estimateur de la fonction de densité spectrale. L'analyse spectrale est l'adaptation de l'analyse de Fourier pour traiter des fonctions temporelles stochastiques plutôt que déterministes, plus détaile voir ??.

Les mesures de dissimilarité basées sur des modèles supposent que les modèles sous-jacents sont générés à partir de structures paramétriques spécifiques. La principale approche dans la littérature consiste à supposer que les processus générateurs de X_T et Y_T suivent des modèles ARIMA inversibles. Dans ce cas, l'idée est d'ajuster un modèle ARIMA à chaque série et de mesurer ensuite la dissimilarité entre les modèles ajustés. La première étape consiste à estimer la structure et les paramètres des modèles ARIMA. La structure est soit supposée donnée, soit estimée automatiquement à l'aide, par exemple, le critère d'information d'Akaike (AIC) ou le critère d'information bayésien de Schwartz (BIC). Les valeurs des paramètres sont généralement ajustées à l'aide d'estimateurs des moindres carrés généralisés. Certaines des mesures de dissimilarité les plus pertinentes dérivées de la littérature sous l'hypothèse de modèles ARIMA sous-jacents sont fournies par TSclust, comme AR.PIC , AR.LPC.CEPS...

Nous procédons comme suit. Les séries contenues dans synthetic.tseries sont soumises à un regroupement en utilisant différentes mesures de dissimilarité et plusieurs algorithmes de regroupement. En supposant que le regroupement est régi par la similarité entre les modèles sous-jacents, la "vraie" solution de regroupement est donnée par les six regroupements impliquant les trois séries issues du même modèle générateur. Ensuite, les solutions expérimentales de regroupement sont comparées à la véritable solution de regroupement en utilisant la fonction de similarité de regroupement cluster.evaluation() incluse dans TSclust.

Rappelons que la sortie de cluster.evaluation() se situe entre 0 et 1 et admet une interprétation simple : plus elle est proche de 1, plus la concordance entre les vraies partitions et les partitions expérimentales est élevée.

PACF	ACF	IP	AR.PIC
0.72	0.74	0.78	0.56

TABLE 12 – fonction de similarité de hiérarchi sur synthetic.tseris

Comme le montrent les graphiques, la situation du regroupement AR.PIC n'est pas très bonne, car PIC prend en compte des modèles AR, mais certains d'entre eux sont des modèles non linéaires. Notre IP, cependant, obtient un bon regroupement pour l'estimation spectrale des processus stochastiques, et le PACF et l'ACF ont également un regroupement relativement bon.

11 les taux d'intérêt mensuels à long terme

Nous avons mentionné précédemment deux aspects de l'analyse des séries chronologiques pour déterminer si le problème doit être important pour chaque moment, et les caractéristiques sont également basées sur le temps. À ce stade, nous n'avons pas d'invariance de phase, d'invariance de désalignement local, d'invariance d'échelle globale, et l'invariance de négligence locale n'est pas nécessaire car chaque moment est important. Il est également possible d'avoir besoin de l'invariance d'amplitude et de décalage, nous pouvons encore utiliser la normalisation pour résoudre ce problème..

Pour ce type de problème de regroupement de séries temporelles, qui est également la base des problèmes de regroupement de séries temporelles, nous recommandons de choisir certaines distances qui sont corrélées avec le modèle probabiliste de la série temporelle, c'est-à-dire, comme dans le problème précédent, l'ACF, le PACF , les distances basé sur les périodogramme, et les distances basé sur modèle.

En utilisant les séries $Y_T^i, i = 1, \dots, 18$, données par $\log X_t^i - \log X_{t-1}^i, t = 2, \dots, T$, En fait, Y_T mesure approximativement les variations en pourcentage des taux d'intérêt d'un mois à l'autre. Et cette nouvelle analyse en groups vise donc à regrouper les pays dont les taux d'intérêt augmentent de façon similaire d'un mois à l'autre.

L'objectif du regroupement est maintenant d'identifier des structures de dépendance similaires cachées derrière les profils ondulés et bruyants observés dans la figure 26. Notez que la non-stationnarité de la série originale a été supprimée en prenant les différences, et que les mesures de dissimilarité construites sous l'hypothèse de stationnarité peuvent donc être utilisées. À titre d'illustration, nous effectuons un regroupement hiérarchique avec un lien complet et utilisons plusieurs mesures de dissimilarité mises en œuvre dans TSclust, à savoir les dissimilarités basées sur les fonctions d'autocorrélation simples, le logarithme des périodogrammes normalisés et la métrique AR introduite par Piccolo. Les coefficients moyens de Silhouette ont été examinés pour plusieurs solutions de k-cluster et il a été observé qu'un nombre de $k = 5$ groupes produit une solution raisonnablement compacte.

Des résultats 13 similaires sont obtenus avec les trois dissimilarités. Le Japon, la Suisse et le Portugal sont des objets isolés. Le Japon montre clairement la plus grande variabilité dans les changements relatifs de leurs taux d'intérêt. La Suisse présente également une variabilité importante et une large séquence de mois avec une croissance négative substantielle. Le Portugal est plus proche du reste des pays, bien qu'il ne présente que très peu de mois

	ACF	LNP	PIC
EMU	1	1	1
Spain	1	1	1
Germany	2	2	1
France	2	1	2
Italy	1	1	1
Netherlands	2	2	1
UK	2	2	3
Ireland	1	3	1
Denmark	2	2	2
Portugal	3	3	4
Austria	2	1	2
Sweden	2	2	1
Finland	2	2	2
Switzerland	4	4	4
Norway	2	2	1
USA	2	2	3
Canada	2	2	3
Japan	5	5	5

TABLE 13 – Solutions à 5 clusters pour les trois mesures de dissimilarité.

de forte décroissance. En ce qui concerne les autres pays, il est complexe de trouver des différences substantielles, bien qu'il vaille la peine de souligner que quatre ensembles de pays sont situés ensemble avec les trois critères de dissimilarité considérés, à savoir Espagne, Italie, UEM, Allemagne, Pays-Bas, Suède, Norvège, Finlande, Danemark et États-Unis, Canada, Royaume-Uni.

Conclusion

Le projet final sur le regroupement des séries temporelles s'achève ici. Je résumerai ce projet par les trois aspects suivants : les gains, les lacunes et les perspectives.

Le projet a été très gratifiant. Une grande expérience a été acquise dans chaque processus de regroupement de séries temporelles. La première étape a consisté à analyser le problème sous les trois angles suivants. La première consiste à examiner la longueur de chaque série temporelle, si la représentation est choisie pour réduire sa longueur, et à examiner la saisonnalité de la série temporelle, la périodicité, si les caractéristiques sont extraites sur la base d'un certain modèle, s'il est nécessaire de réduire le bruit, etc. Deuxièmement, dans le problème du regroupement, notez si les caractéristiques (généralement locales) de la série temporelle sont corrélées avec les moments d'occurrence. Si c'est le cas, chaque moment est important et nous utilisons des distances qui sont corrélées avec le modèle probabiliste de la série temporelle, comme Basé sur les autocorrélations simples et partielles, Basé sur les périodogrammes, Basé sur les estimateurs spectraux non paramétriques Si ce n'est pas le cas, nous analysons le dernier point pour voir quel type d'invariance doit être maintenu au sein du groupe et, sur la base de l'invariance nécessaire, la distance correspondante est finalement choisie.

La deuxième étape consiste à proposer une méthode de regroupement, qui correspond à trois méthodes pour chacune des trois perspectives ci-dessus, une méthode basée sur les caractéristiques, une méthode basée sur le modèle et une méthode basée sur la forme. Cependant, nous constatons que ces trois méthodes empruntent parfois les unes aux autres et que les limites entre elles ne sont pas très claires. Toutes les méthodes peuvent être divisées en cinq parties, la représentation et la distance, qui sont la clé d'un bon ou d'un mauvais regroupement. Pour les différents types d'algorithme, nous avons également les trois parties suivantes : l'extraction du centre du groupe, la sélection de l'algorithme de type de regroupement et enfin l'évaluation du regroupement. Dans chaque partie, nous apprenons le plus possible sur les méthodes existantes et nous nous concentrons sur les différentes parties correspondant à DTW et à ses variantes.

Plus que théorique, ce projet m'a permis de me familiariser avec l'analyse des problèmes de clustering temporel en R. J'ai appris à stocker et à présenter les séries temporelles, ainsi que les cinq paquets conçus pour le regroupement des séries temporelles. L'accent est mis sur l'étude et l'application des packages pdc, TSclust, et dtwclust ainsi que du package de représentation des séries temporelles TSrepr. Apprenez également à utiliser des packages tels que NbClust, clusteSim, clue, etc. pour analyser les résultats du clustering. Dans l'ensemble, il a été possible de trouver la méthode de clustering idéale dans R pour différents problèmes de clustering de séries temporelles.

Au final, en lisant et en comprenant plus de trente documents, j'ai pu inconsciemment consolider et élargir ma base de connaissances.

La conduite de ce projet a également présenté de nombreuses lacunes. Tout d'abord, il n'y avait pas de plan limité pour la conduite du projet. En termes de contenu, on a essayé de tout couvrir sans faire de choix rationnel. Trop de recherches ont été effectuées sur le dtw et ses variantes, qui ont un faible champ d'application, et pas assez de détails et d'approfondissement ont été donnés aux modèles probabilistes de séries temporelles, qui ont un large champ d'application. En termes de temps, il n'y a pas eu de planification détaillée du moment où une série de tâches devait être accomplie, ce qui a conduit à une concentration excessive du temps de travail, accompagnée de retards importants. Le deuxième point est que trop de logiciels de traduction ont été utilisés et que le vocabulaire français utilisé n'a

pas été bien examiné.

Enfin, j'espère avoir l'occasion d'analyser d'autres problèmes de regroupement de séries temporelles à l'avenir. J'espère également avoir l'occasion d'en apprendre davantage sur l'analyse des séries temporelles, l'apprentissage supervisé des séries temporelles, la détection des anomalies dans les séries temporelles, etc.

Références

- M. Bicego A. Panuccio and V. Murino. A hidden markov model based approach to sequential data clustering. *Structural, Syntactic, and Statistical Pattern Recognition, T. Caelli, A. Amin, R. Duin, R. De, and M. Kamel, Eds.*, 2002.
- E. Keogh S. Lonardi G. Janacek A.A.J. Bagnall, C. "Ann" Ratanamahatana. A bit level representation for time series data mining with shape based similarity. *Data Min. Knowl. Discov.* 13(1), 2006.
- G. Janacek A.J. Bagnall. Clustering time series with clipped data,. *Mach. Learn.* 58 (2), 2005.
- Muguerza J Pérez JM Perona I Arbelaitz O, Gurrutxaga I. An extensive comparative study of cluster validity indices. *Pattern Recognition*, 2013.
- E. Bingham. Random projection in dimensionality reduction : applications to image and text data,. *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2001.
- Andreas M. Brandmaier. pdc : An r package for complexity-based clustering of time series. *Journal of Statistical Software*, 2015.
- Y. Manolopoulos C. Faloutsos, M. Ranganathan. Fast subsequence matching in time-series databases. *ACM SIGMOD Rec.* 23 (2), 1994.
- A.J. Bagnall S. Lonardi C. Ratanamahatana, E. Keogh. A novel bit level time series representation with implications for similarity search and clustering. *Proceedings of 9th Pacific-Asian International Conference on Knowledge Discovery and Data Mining*, 2005.
- Y. Cai and R. Ng. Indexing spatio-temporal trajectories with chebyshev polynomials,. *Proceedings of 2004 ACM SIGMOD International*, 2004.
- Peña D Caiado J, Crato N. A periodogram-based metric for time series classification. *Computational Statistics Data Analysis*, 2006.
- Yanping Chen, Eamonn Keogh, Bing Hu, Nurjahan Begum, Anthony Bagnall, Abdullah Mueen, and Gustavo Batista. The ucr time series classification archive, July 2015. www.cs.ucr.edu/~eamonn/time_series_data/.
- Vert J.-P. Birkenes Ø. and Matsui T. Cuturi, M. A kernel for time series based on global alignments. *In Proceedings of ICASSP, volume II, pp. 413 – 416*, 2007.
- I. Essa T. Starner D. Minnen, C.L. Isbell. Discovering multivariate motifs using subsequence density estimation and greedy mixture learning. *Proc. Natl. Conf. Artif. Intell.* 22(1), 2007.
- M. Essa C. Isbell D. Minnen, T. Starner. Discovering characteristic actions from on-body sensor data. *Proceedings of 10th IEEE International Symposium on Wearable Computers*, 2006.
- Casado de Lucas D. Classification techniques for time series and functional data. *Ph.D. thesis, Universidad Carlos III de Madrid*, 2010.

Nagabhushan PN Douzal Chouakria A. Adaptive dissimilarity index for measuring time series proximity. *Advances in Data Analysis and Classification*,, 2007.

R. Valkanov E. Ghysels, P. Santa-Clara. Predicting volatility : getting the most out of return data sampled at different frequencies. *J. Econom* 131 (1–2), 2006.

K. Chakrabarti S. Mehrotra E. Keogh, M. Pazzani. A simple dimensionality reduction technique for fast similarity search in large time series databases. *Knowl. Inf. Syst.* 1805(1), 2000.

M. Pazzani S. Mehrotra E. Keogh, K. Chakrabarti. Locally adaptive dimensionality reduction for indexing large time series databases. *ACM SIGMOD Rec* 27 (2), 2001.

Chotirat Ann Ratanamahatana Eamonn Keogh. Exact indexing of dynamic time warping. 2005.

Michael Pazzani Sharad Mehrotra Eamonn Keogh, Kaushik Chakrabarti. Dimensionality reduction for fast similarity search in large time series databases. *KAIS Long paper submitted*, 2000.

C. Faloutsos F. Korn, H.V. Jagadish. Efficiently supporting ad hoc queries in large datasets of time sequences. *ACM SIGMOD Record* 26, 1997.

A. Gatt J. Hunter S.Sripada Y. Freer C. Sykes F. Portet, E. Reiter. Automatic generation of textual summaries from neonatal intensive care data,. *Artif. Intell.* 173 (7), 2009.

c Alain Ketterlin a b Pierre Gancarski a b Francis Petitjean a, b. A global averaging method for dynamic time warping, with applications to clustering. *Pattern Recognition* 44 (2011) 678–693, 2011.

K. Kawagoe G. Duan, Y. Suzuki. Grid representation of time series data for similarity search. *The institute of Electronic, Information, and Communication Engineer*, 2006.

O. M. Tatar G. E. Batista, E. J. Keogh and V. M. de Souza. Cid : An efficient complexity-invariant distance for time series. *Data Mining and Knowledge Discovery*, pages 1–36, 2013.

Peña D Galeano P. Multivariate analysis in vector time series. *Resenhas do Instituto de Matemática e Estatística da Universidade de São Paulo*, 2000.

Woods R Gonzalez R. Addison-wesley, reading, mas-sachusetts. *Digital Image Processing*, 1992.

L. Wei S. Lonardi J. Lin, E. Keogh. Experiencing sax : a novel symbolic representation of time series”. *Data Min. Knowl. Discov.* 15 (2), 2007.

S. Lonardi J. Lin, E. Keogh and B. Chiu. A symbolic representation of time series, with implications for streaming algorithms. *Proceedings of 8th ACM SIGMOD Workshop on Research Issues Data Mining and Knowledge Discovery – DMKD '03*, 2003.

Luis Gravano John Paparrizos. k-shape : Efficient and accurate clustering of time series. *SIGMOD '15, May 31–June 4, 2015, Melbourne, Victoria, Australia.*, 2015.

A.W. Fu K. Chan. Efficient time series matching by wavelets. *Proceedings of 1999 15th International Conference on Data Engineering, vol. 15, no. 3,,* 1999.

V. Puttagunta K. Kalpakis, D. Gada. Distance measures for effective clustering of arima time-series. *Proceedings 2001 IEEE International Conference on Data Mining,* 2001.

E. Keogh. Hot sax : efficiently finding the most unusual time series subsequence. *Proceedings of Fifth IEEE International Conference on Data Mining ICDM05,* 2005.

Xi X Vlachos M Lee SH Protopapas P Keogh E, Wei L. Supporting exact indexing of arbitrarily rotated shapes and periodic time series under euclidean and warping distance measures. *The VLDB Journal,* 2009.

Kruskall and Liberman. The symmetric time-warping problem : From continuous to discrete.

Peter Laurinec. *TSrepr : Time series representations in a use case.* TSrepr 1.1.0. URL : <https://petolau.github.io/package>.

Peter Laurinec and Mária Lucká. Comparison of representations of time series for clustering smart meter data. *Proceedings of the World Congress on Engineering and Computer Science,* 2016.

Daniel Lemire. Faster retrieval with a two-pass dynamic-time- warping lower bound. 2009.

Lonardi S Chiu B Lin J, Keogh E. A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, DMKD'03, pp. 2–11. ACM, New York,* 2003.

Wei L Lonardi S Lin J, Keogh E. Experiencing sax : A novel symbolic representation of time series. *Data Mining and Knowledge Discovery, 15(2), 107–144,* 2007.

Cuturi M. Fast global alignment kernels. In *Proceedings of the 28th international conference on machine learning (ICML-11), pp. 929–936,* 2011.

D. Piccolo M. Corduas. Time series clustering and classification by the autoregressive metric. *Comput. Stat. Data Anal. 52(4),* 2008.

S. Mallat. A wavelet tour of signal processing. *Academic Press, San Diego, deuxième édition,* 1999.

Mathieu Blondel 2 Marco Cuturi. Soft-dtw : a differentiable loss function for time-series. *Proceedings of the 34 International Conference on Machine Learning, Sydney, Australia, PMLR 70, 2017.,* 2017.

Brandmaier Andreas Markus. *Permutation Distribution Clustering and Structural Equation Model Trees.* PhD thesis, Dissertation, Saarland University, Saarbrücken, 2012.

E. Keogh S. Lonardi N. Kumar, N. Lolla. Time-series bitmaps : a practical visualization tool for working with large time series databases. *SIAM 2005 Data Min,* 2005.

Jun Wang 1 Eamonn Keogh Nurjahan Begum, Liudmila Ulanova. Accelerating dynamic time warping clustering with a novel admissible pruning strategy. *KDD '15, August 11 - 14, 2015, Sydney, NSW, Australia,* 2015.

- José A. Vilar Pablo Montero. Tsclust : An r package for time series clustering. *Journal of Statistical Software*, 2014.
- X. Lian Y. Liu Q. Chen, L. Chen. Indexable pla for efficient similarity search. *Proceedings of the 33rd International Conference on Very large Data Bases*, 2007.
- A. Swami R. Agrawal, C. Faloutsos. Efficient similarity search in sequence databases. *Found. Data Organ. Algorithms* 46, 1993.
- et al. Saeed, M. A public-access intensive care unit database. critical care medicine, 39(5), 952, 2011.) called tadpole(time-series anytime dp). *Multiparameter Intelligent Monitoring in Intensive Care II (MIMIC-II)*, 2011.
- Teh Ying Wah Saeed Aghabozorgi, Ali Seyed Shirkhorshidi. Time-series clustering – a decade review. *Information Systems*, 2015.
- HIROAKI SAKOE and SEIBI CHIBA. Dynamic programming algorithm optimization for spoken word recognition. 1978.
- C. Ratanamahatana V. Niennattrakul. Inaccuracies of shape aver- aging method using dynamic time warping for time series data,. *Comput. Sci. 2007 (2007) 513–520.*, 2007.
- Sri Harsha Dumpala KNRKRAJU Alluri Suryakanth Gangashetty Anil Kumar Vuppala. Analysis of constraints on segmental dtw for the task of query-by-example spoken term detection.
- T. Amagasa S. Uemura Y. Morinaka, M. Yoshikawa. The l-index : an indexing structure for efficient subsequence matching in time sequence databases,. *Proceedings of 5th PacificAisa Conference on Knowledge Discovery and Data Mining*, 2001.
- H. Shatkay S.B. Zdonik. Approximate queries and representations for large data sequences. *Proceedings of the Twelfth International Conference on Data Engineering*, 1996.
- Hui Zhang and Tu Bao Ho. Unsupervised feature extraction for time series clustering using orthogonal wavelet transform. *Informatica* 30 305–319, 2006.