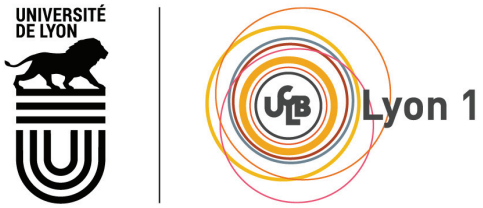
Détection d'anomalies dans les séries temporelles : application aux masses de données sur les pneumatiques

Seif-Eddine Benkabou



N° d'ordre NNT : 2018LYSE1046

THÈSE DE DOCTORAT DE L'UNIVERSITÉ DE LYON
opérée au sein de
l'Université Claude Bernard Lyon 1

École Doctorale ED512
Informatique et Mathématiques (InfoMaths)

Spécialité de doctorat : Informatique

Soutenue publiquement le 21/03/2018, par :
Seif-Eddine BENKABOU

Détection d'anomalies dans les séries temporelles : Application aux masses de données sur les pneumatiques.

Devant le jury composé de :

Mme. Salima Benbernou, Professeur, Université Paris 5	Présidente
M. Younès Bennani, Professeur, Université Paris 13	Rapporteur
M. Yann Guermeur, DR CNRS, LORIA-Nancy	Rapporteur
Mme. Lydia Boudjeloud-Assala, MCF-HDR, Université de Lorraine	Examinatrice
M. Marc Sebban, Professeur, Université de Saint-Etienne	Examinateur
M. Khalid Benabdeslem, MCF-HDR, Université Lyon1	Directeur de thèse
M. Bruno Canitia, Responsable R&D, LOMG	Invité

UNIVERSITE CLAUDE BERNARD - LYON 1

Président de l'Université	**M. le Professeur Frédéric FLEURY**
Président du Conseil Académique	M. le Professeur Hamda BEN HADID
Vice-président du Conseil d'Administration	M. le Professeur Didier REVEL
Vice-président du Conseil Formation et Vie Universitaire	M. le Professeur Philippe CHEVALIER
Vice-président de la Commission Recherche	M. Fabrice VALLÉE
Directrice Générale des Services	Mme Dominique MARCHAND

COMPOSANTES SANTE

Faculté de Médecine Lyon Est – Claude Bernard	Directeur : M. le Professeur G.RODE
Faculté de Médecine et de Maïeutique Lyon Sud – Charles Mérieux	Directeur : Mme la Professeure C. BURILLON
Faculté d'Odontologie	Directeur : M. le Professeur D. BOURGEOIS
Institut des Sciences Pharmaceutiques et Biologiques	Directeur : Mme la Professeure C. VINCIGUERRA
Institut des Sciences et Techniques de la Réadaptation	Directeur : M. X. PERROT
Département de formation et Centre de Recherche en Biologie Humaine	Directeur : Mme la Professeure A-M. SCHOTT

COMPOSANTES ET DEPARTEMENTS DE SCIENCES ET TECHNOLOGIE

Faculté des Sciences et Technologies	Directeur : M. F. DE MARCHI
Département Biologie	Directeur : M. le Professeur F. THEVENARD
Département Chimie Biochimie	Directeur : Mme C. FELIX
Département GEP	Directeur : M. Hassan HAMMOURI
Département Informatique	Directeur : M. le Professeur S. AKKOUCHE
Département Mathématiques	Directeur : M. le Professeur G. TOMANOV
Département Mécanique	Directeur : M. le Professeur H. BEN HADID
Département Physique	Directeur : M. le Professeur J-C PLENET
UFR Sciences et Techniques des Activités Physiques et Sportives	Directeur : M. Y.VANPOULLE
Observatoire des Sciences de l'Univers de Lyon	Directeur : M. B. GUIDERDONI
Polytech Lyon	Directeur : M. le Professeur E.PERRIN
Ecole Supérieure de Chimie Physique Electronique	Directeur : M. G. PIGNAULT
Institut Universitaire de Technologie de Lyon 1	Directeur : M. le Professeur C. VITON
Ecole Supérieure du Professorat et de l'Education	Directeur : M. le Professeur A. MOUGNIOTTE
Institut de Science Financière et d'Assurances	Directeur : M. N. LEBOISNE

Avant-propos

Remerciements

Mes premières pensées vont à mon directeur de thèse, Monsieur Khalid Benabdeslem, Maître de conférence-HDR à l'Université de Claude Bernard Lyon et Monsieur Bruno Canitia, Responsable R&D au sein de l'entreprise LIZEO Group.

Khalid, tu as toujours été présent depuis la fin du Master. Tu m'as tant appris, aussi humainement que scientifiquement. Je t'en serai toujours reconnaissant et je ne saurais assez te remercier.

Bruno, ta bienveillance ainsi que ton esprit scientifique ont largement contribué au bon déroulement de mes recherches au sein de l'entreprise.

Monsieur Yann Guermeur, Directeur de recherche au CNRS et Monsieur Younes Bennani, Professeur à l'Université Sorbonne-Paris-Cité m'ont fait l'immense honneur d'accepter de rapporter ma thèse. Je ne saurais assez les remercier.

J'exprime ma gratitude à Monsieur Marc Sebban, Professeur à l'Université de Saint-Étienne, Madame Salima Benbernou, Professeure à l'université Paris 5 et et Madame Lydia Boudjeloud Assala , Maître de Conférence-HDR à l'Université de Lorraine, de m'avoir honoré par leur présence dans le jury.

Mes plus profonds remerciements vont à mon Père, Monsieur Benkabou Belabess, et à ma Mère, Madame Benkabou Nacera. Vous m'avez toujours aidé, soutenu et encouragé. Vous avez su me donner toutes les chances pour réussir. Que vous trouverez, dans la réalisation de ce travail, l'aboutissement de vos efforts ainsi que l'expression de ma plus chaleureuse gratitude.

Et pour terminer, je remercie ma famille et en particulier mes deux surs Ikram et Aicha pour m'avoir soutenu dans mes efforts. Je remercie également mon épouse, Sarah, pour son amour, son écoute et surtout son soutien infaillible qui m'a été essentiel durant ces années de thèse.

À ma fille Zeineb Isra

Résumé

La détection d'anomalies est une tâche cruciale qui a suscité l'intérêt de plusieurs travaux de recherche dans les communautés d'apprentissage automatique et fouille de données. La complexité de cette tâche dépend de la nature des données, de la disponibilité de leur étiquetage et du cadre applicatif dans lequel elles s'inscrivent.

Dans le cadre de cette thèse, nous nous intéressons à cette problématique pour les données complexes et particulièrement pour les séries temporelles uni et multi-variées. Le terme "anomalie" peut désigner une observation qui s'écarte des autres observations au point d'éveiller des soupçons. De façon plus générale, la problématique sous-jacente (aussi appelée détection de nouveautés ou détection de valeurs aberrantes) vise à identifier, dans un ensemble de données, celles qui diffèrent significativement des autres, qui ne se conforment pas à un "comportement attendu" (à définir ou à apprendre automatiquement), et qui indiquent un processus de génération différent. Les motifs "anormaux" ainsi détectés se traduisent souvent par de l'information critique.

Nous nous focalisons plus précisément sur deux aspects particuliers de la détection d'anomalies à partir de séries temporelles dans un mode non-supervisé. Le premier est global et consiste à identifier des séries relativement anormales par rapport une base entière. Le second est dit contextuel et vise à détecter localement, les points anormaux par rapport à la structure de la série étudiée. Pour ce faire, nous proposons des approches d'optimisation à base de clustering pondéré et de déformation temporelle pour la détection globale ; et des mécanismes à base de modélisation matricielle pour la détection contextuelle.

Enfin, nous présentons une série d'études empiriques sur des données publiques pour valider les approches proposées et les comparer avec d'autres approches connues dans la littérature. De plus, une validation expérimentale est fournie sur un problème réel, concernant la détection de séries de prix aberrants sur les pneumatiques, pour répondre aux besoins exprimés par le partenaire industriel de cette thèse.

Mots clés— détection d'anomalies, séries temporelles, DTW, clustering, optimisation.

Abstract

Anomaly detection is a crucial task that has attracted the interest of several research studies in machine learning and data mining communities. The complexity of this task depends on the nature of the data, the availability of their labeling and the application framework on which they depend.

As part of this thesis, we address this problem for complex data and particularly for uni and multivariate time series. The term "anomaly" can refer to an observation that deviates from other observations so as to arouse suspicion that it was generated by a different generation process. More generally, the underlying problem (also called novelty detection or outlier detection) aims to identify, in a set of data, those which differ significantly from others, which do not conform to an "expected behavior" (which could be defined or learned), and which indicate a different mechanism. The "abnormal" patterns thus detected often result in critical information.

We focus specifically on two particular aspects of anomaly detection from time series in an unsupervised fashion. The first is global and consists in detecting abnormal time series compared to an entire database, whereas the second one is called contextual and aims to detect locally, the abnormal points with respect to the global structure of the relevant time series. To this end, we propose an optimization approaches based on weighted clustering and the warping time for global detection ; and matrix-based modeling for the contextual detection.

Finally, we present several empirical studies on public data to validate the proposed approaches and compare them with other known approaches in the literature. In addition, an experimental validation is provided on a real problem, concerning the detection of outlier price time series on the tyre data, to meet the needs expressed by, LIZEO, the industrial partner of this thesis.

Keywords— anomaly detection, time series, DTW, clustering, optimization.

Table des matières

Table des figures

Liste des tableaux

Liste des Algorithmes

1 Introduction Générale

1.1 Contexte et motivations

La détection d'anomalies est une tâche primordiale en apprentissage automatique et fouille de données. En effet, sa mise en œuvre permet d'améliorer considérablement les modèles sous-jacents et ainsi comprendre le comportement des données associées. En pratique, le but de cette tâche est d'apprendre, à partir d'un ensemble de données, un modèle reflétant au mieux la "normalité" pour détecter comme anormale toute autre donnée s'écartant significativement de ce dit modèle [Aggarwal13].

L'apprentissage du modèle de la normalité peut être effectué dans plusieurs paradigmes qui dépendent fortement de la disponibilité des labels. Dans le cas où chaque observation est associée à un label (*normal ou anormal*), la détection d'anomalies est dite *supervisée* (e.g., *rare category detection*). Dans ce mode, le but est d'apprendre un modèle capable de séparer au mieux les données normales des données anormales. Dans le cas où uniquement les labels de la majorité sont disponibles, la détection est dite *semi-supervisée* (e.g., *novelty detection*). L'idée consiste à apprendre un modèle pour mieux comprendre et cerner la normalité des données. Le troisième paradigme est celui du *non-supervisé* (e.g., *outlier detection*) dans lequel aucune information n'est disponible a priori. La démarche consiste à apprendre un modèle de normalité en se basant uniquement sur des hypothèses de proximité ou de densité. Par exemple, une donnée est jugée anormale et est supposée différente du reste, si elle est éloignée de son voisinage le plus proche où si elle se trouve dans des régions à faible densité dans l'espace de description.

Dans la littérature, la détection d'anomalies a été largement étudiée par les communautés des statistiques, apprentissage automatique et data mining [Chandola09]. Cette tâche a donc suscité plusieurs travaux de recherche selon la nature des données, la disponibilité des labels sur la normalité et les domaines d'application qui sont divers. Dans cette thèse, nous nous focaliserons sur cette problématique de détection pour les données complexes de nature temporelle et plus particulièrement pour les séries temporelles, uni et multi variées.

Du point de vu applicatif, cette thèse s'inscrit dans le cadre d'une convention CIFRE entre le laboratoire LIRIS et la société Lizeo Online Media Group, dénommée par la suite LIZEO[1]. En effet, l'entreprise s'inscrit dans une vision globale d'analyse de ses données avec des techniques de data mining et d'apprentissage automatique. Les données des pneumatiques sont décrites autour de deux dimensions importantes qui intéressent principalement les clients : la performance et le prix. Plus particulièrement, un sous-ensemble de performances est représenté par la notion d'étiquetage qui décrit les pneumatiques par des représentations vectorielles à travers des caractéristiques (variables) mixtes (qualitatives et quantitatives). Le prix pour chaque pneumatique est par contre représenté par une série temporelle définie sur plusieurs mois et issue d'un ensemble de sites de commerce électronique. Face à ces deux dimensions, une incertitude demeure à la fois sur l'étiquetage, car il est réalisé par les manufacturiers de pneumatique eux-mêmes, et sur les prix, car ils sont récupérés sur les sites web qui les commercialisent. De plus, contrairement à l'étiquetage, il n'existe pas de référence pour les prix. Cette incertitude est donc portée par des étiquetages et/ou des prix dits aberrants.

1.2 Contributions

Pour répondre à la problématique évoquée ci-dessus, nous nous intéressons dans cette thèse à deux facettes de la détection d'anomalies dans les séries temporelles en mode non-supervisé, à savoir la détection globale et la détection locale (e.g., contextuelle). Nous proposons quatre approches pour la première et une approche pour la seconde.

Concernant la détection globale dont l'objectif est de détecter des séries anormales par rapport à un ensemble de séries, notre première approche (*DetecS*), dite à deux niveaux séquentiels, consiste à effectuer un clustering spectral sur les séries temporelles et d'utiliser à l'issue de cette étape une fonction de score spécifique pour détecter

1. Cette thèse s'est déroulée selon une "Convention Industrielle de Formation par la Recherche" (CIFRE) proposée par l'Agence Nationale de la Recherche Technique (ANRT). Ce mode de financement consiste en un partenariat entre une entreprise et une université. Dans le cas de la présente thèse CIFRE, les partenaires ont été d'une part, la société LIZEO GROUP et d'autre part, l'université Claude Bernard Lyon 1 (UCBL).
Site web : http ://www.lizeo-online-media-group.com

les séries anormales. Contrairement à l'approche séquentielle, dans les trois autres approches, dites *embedded*, le clustering est intrinsèquement lié à la détection d'anomalies à travers un mécanisme de pondération. Deux visions sont développées à cet effet, (1) une vision globale (DOTS et ℓ_2-DAT) où chaque série se voit attribuer un score d'anormalité par rapport à l'ensemble des clusters et (2) une deuxième vision à caractère local (L2GAD) où chaque série se voit attribuer un score d'anormalité par cluster. Ces approches permettent d'évaluer la contribution de chaque série au modèle du clustering.

Concernant la détection contextuelle des observations anormales au sein d'une série temporelle, nous proposons une approche basée sur la reconstruction des observations de la série via ses observations les plus représentatives. La détection d'anomalies se fait donc en analysant les résidus de la reconstruction et l'aspect temporel des observations est pris en considération et est modélisé par une loi probabiliste.

Enfin, étant donnée la nature massive des données de l'entreprise LIZEO, les approches proposées dans cette thèse ont été implémentées et déployées dans un environnement *Big Data* aboutissant à un outil de détection robuste permettant le passage à l'échelle.

1.3 Organisation du manuscrit

Le reste de la thèse est structuré comme suit :

- Dans le Chapitre 2, nous introduisons la problématique de la détection d'anomalies dans les séries temporelles. Nous nous focalisons plus précisément sur deux familles d'approches, celles basées sur la proximité ; et celles basées sur la modélisation paramétrique. Tout d'abord, nous définissons la notion de similarité entre les séries, et les différentes mesures sous-jacentes. Ensuite, nous présentons quelques techniques de transformation. Enfin, nous concluons ce chapitre par une liste des différents domaines d'applications de la détection d'anomalies dans les données temporelles au sens large.

- Le Chapitre 3 est consacré à nos contributions à la détection non-supervisée des séries temporelles anormales par rapport à un ensemble de séries (e.g., détection globale). Il fait l'objet de quatre approches différentes. La première approche (*DetecS*), dite à deux niveaux séquentiels, consiste à développer un clustering spectral sur les séries temporelles et calculer ainsi une fonction de score spécifique pour détecter les séries anormales. Ensuite, nous reformulons, à travers les trois autres approches dites (*embedded*), la tâche de détection comme un clustering pondéré où la détection et le clustering sont effectués de manière simultanée. La différence entre ces trois approches réside principalement dans le mécanisme de pondération ainsi qu'au niveau de la détection, globale (DOTS et

ℓ_2-DAT) ou locale (L2GAD). Enfin, nous présentons les résultats obtenus via les expérimentions que nous avons menées pour valider et évaluer nos approches vis-à-vis des approches de l'état de l'art.

- Dans le Chapitre 4, nous nous intéressons à la détection des observations anormales au sein d'une série temporelle, uni ou multi variée soit elle (e.g., détection contextuelle). Nous proposons une approche non-supervisée (appelée LADOP) où la dépendance temporelle, est modélisée par la loi de Poisson. Les observations sont reconstruites via une combinaison linéaire de certaines observations de référence, et la détection est faite en analysant les résidus. Nous étudions ensuite, la convergence et la complexité de l'algorithme proposé, et nous présentons des résultats préliminaires obtenus sur un jeu de données publiques afin de confirmer la faisabilité de notre approche.

- Le Chapitre 5 est dédié à la présentation du framework de détection d'anomalies que nous avons développé pour notre partenaire industriel LIZEO. Cette détection concerne particulièrement les prix aberrants sur les pneumatiques. En effet, le but consiste à détecter pour un pneu mis en vente par plusieurs sites web marchands, les séries temporelles de prix qui sont anormales par rapport à l'ensemble des autres prix. Nous proposons un POC (*Proof of Concept*) de détection, composé de deux modules, a) une chaine de pré-traitement, de nettoyage et de transformation des données permettant de les rendre exploitables et b) un module de détection comprenant l'ensemble des approches que nous avons décrites dans les chapitres précédents. Étant donnée la nature massive des données, le framework proposé a été implémenté et déployé dans un environnement distribué (Spark/Scala) de telle sorte qu'il puisse passer à l'échelle. Les résultats des expérimentations sont pertinents et ont permis de mettre en évidence certaines défaillances dans le système de collecte des données de l'entreprise.

- Nous concluons ce rapport dans le Chapitre 6 avec un bilan sur les contributions proposées et les résultats obtenus via les différentes expérimentations menées durant cette thèse. Nous discutons aussi des perspectives à court et à moyen termes et qui concernent notamment l'apprentissage de métrique, l'intégration des feedbacks des experts et l'apprentissage de la dépendance temporelle pour la détection contextuelle.

2

Détection d'anomalies dans les séries temporelles

▷ Dans ce chapitre, nous introduisons la problématique de la détection d'anomalies dans les séries temporelles. Nous nous intéressons plus précisément à deux familles d'approches : celles basées sur la proximité et la similarité des séries temporelles ; et celles basées sur la modélisation paramétrique. Tout d'abord, nous définissons la notion de similarité entre les séries ; et les différentes mesures sous-jacentes. Ensuite, nous présentons les différentes techniques à base de transformation. Enfin, nous concluons par une liste des différents domaines d'applications de la détection d'anomalies dans les données temporelles au sens large. ◁

Plan du chapitre

2.1 Introduction

La détection d'anomalies consiste à mettre en évidence des données ayant un comportant différent par rapport à la majorité des données [Aggarwal13]. En général, ces données *particulières* sont minoritaires et leur détection peut se faire suivant les trois paradigmes connus en apprentissage automatique. En effet, la détection peut être effectuée en mode *supervisé* dans le cas où l'on dispose d'une information sur la normalité des données. Autrement dit, toute donnée dans la base d'apprentissage peut être étiquetée comme étant normale ou anormale. Dans le mode *semi-supervisé*, la base d'apprentissage n'est supposée contenir que des données normales. Quant à la détection en mode *non-supervisé*, aucune information sur la normalité des données n'est disponible, a priori [Chandola09]. La détection d'anomalie a donc suscité plusieurs travaux de recherche selon la nature des données, la disponibilité des labels sur la normalité et les domaines d'application qui sont divers (cf. section 2.5). Dans cette thèse, nous nous focaliserons sur la tâche de détection dans les données temporelles et plus particulièrement sur les séries temporelles uni et multi variées.

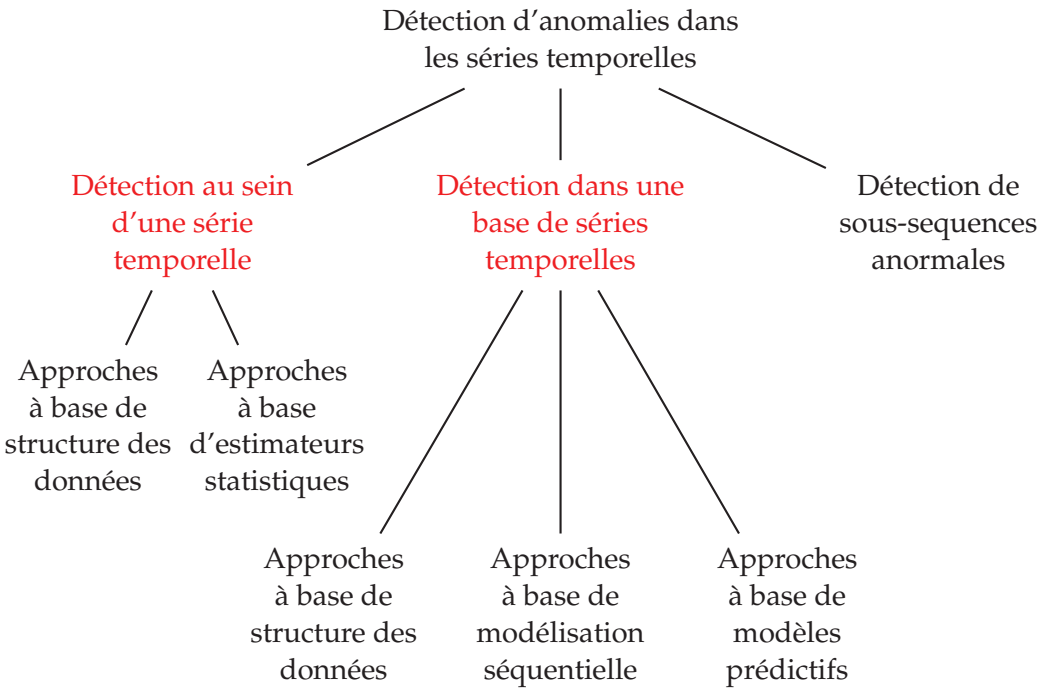


FIGURE 2.1 – Différents problèmes de détection d'anomalies dans les séries temporelles.

Une série temporelle est par définition une suite d'observations représentant leur évolution au cours du temps. Ces observations peuvent être des valeurs uniques ou des vecteurs numériques. Dans le premier cas, il s'agit d'une série temporelle *uni-variée* (e.g., une évolution de températures). Dans le deuxième cas, la série temporelle est dite

multi-variée (e.g., une suite de différentes métriques d'un serveur dans un centre de calcul).

La détection d'anomalies dans les séries temporelles peut être étudiée sous plusieurs angles comme le montre la Figure 2.1 :

- **Détection contextuelle (*détection locale*) :** cette tâche vise à détecter des observations anormales au sein d'une série temporelle, qu'elle soit uni-variée ou multi-variée. Ces observations peuvent être normales dans un contexte temporel et anormales dans un autre comme indiqué dans la Figure 2.2. Cette Figure montre un exemple d'anomalie contextuelle dans une série de temporelle de température mensuelle. Une température basse est tout à fait normale en hiver (l'instant t_1), en revanche elle ne l'est plus en plein été (instant t_2).

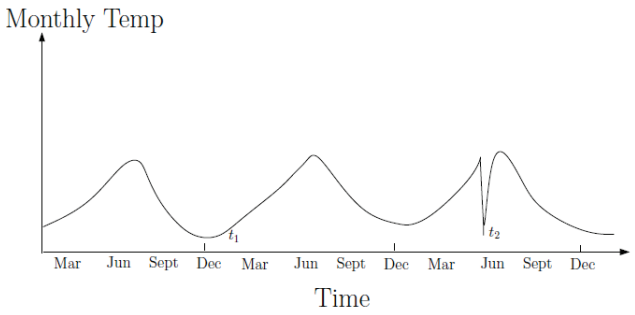


FIGURE 2.2 – Une anomalie contextuelle en t_2 dans une série de température mensuelle [Chandola09].

- **Détection des séries anormales (*détection globale*)** : détecter des séries temporelles anormales revient à savoir si une série est relativement anormale par rapport à une base de séries disponibles. La figure 2.3 montre quelques séries anormales (en rouge) par rapport à une majorité de séries normales (en vert). En effet, dans la littérature [Gupta14], cette base de séries est souvent supposée ne contenir que des séries normales, autrement dit la détection se fait en mode *semi-supervisé*. Or il est souvent difficile d'obtenir ce genre de bases. Dans la plus part des domaines d'application, la détection doit se faire plutôt dans un mode *non-supervisé* à cause de l'absence des connaissances a priori. Cela rend la tâche particulièrement plus difficile. En effet, l'anormalité de certaines séries par rapport à une majorité pourrait être liée à plusieurs raisons :

  1. Les séries normales sont générées par un seul modèle génératif, alors que les séries anormales sont générées par un autre modèle.

  2. Les observations de toutes les séries (normales ou anormales) sont générées par le même modèle. Par contre, une minorité des observations de certaines

séries ont été générées par un autre modèle génératif, ce qui rend ces mêmes séries globalement anormales.

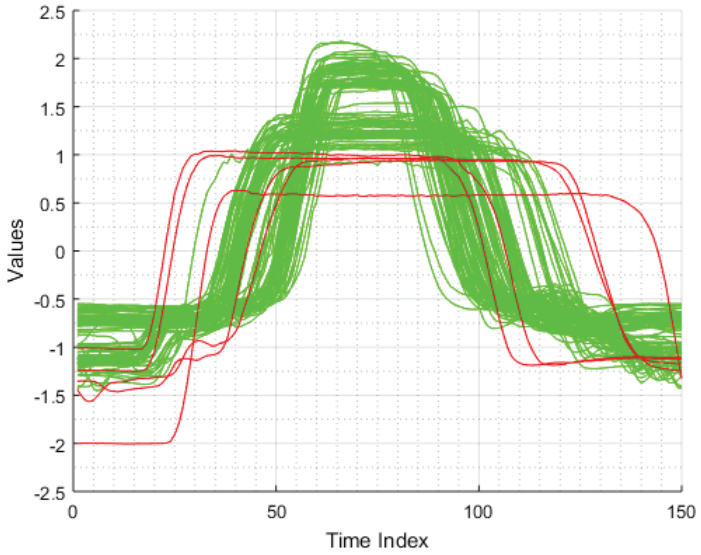


FIGURE 2.3 – Des séries anormales (en rouge) par rapport à un ensemble de séries normales.

— **Détection de sous-séries anormales** : détecter des sous-séries anormales au sein d'une série temporelle, revient à trouver la partie de la série qui s'écarte le plus par rapport à l'ensemble des sous-séries. La figure 2.4 montre un exemple d'une

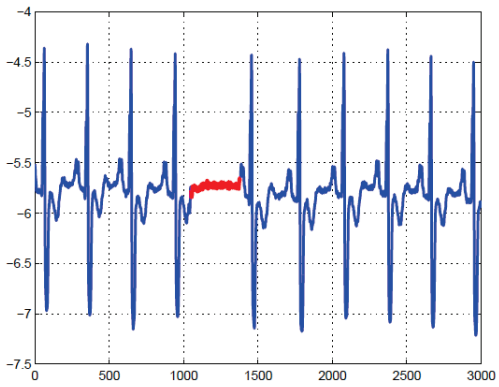


FIGURE 2.4 – Une sous série anormale (en rouge) au sein d'une série temporelle [Chandola09].

série temporelle contenant une sous-série anormale. En effet, les observations constituant cette sous-série, ne sont pas forcément anormales si elles étaient analysées individuellement. En revanche, leur présence durant une période longue, fait que cette sous-série est anormale. La taille des sous-séries étant fixée par l'utilisateur, ce type de détection est équivalent à la détection contextuelle si cette taille est unitaire.

Toutefois, la détection d'anomalies dans les séries temporelles posent plusieurs problèmes liés à la fois, à la nature temporelle des données et au principe même de la détection. Comme nous l'avons indiqué ci-dessus, une anomalie peut être une observation au sein d'une série temporelle, une sous-série au sein d'une série temporelle ou même la série elle-même dans sa globalité. Par ailleurs, plusieurs techniques de détection se basent sur la proximité et exploitent la similarité entre les séries temporelles, et le fait que dans la majorité des domaines d'applications les séries ne soient pas régulièrement échantillonnées, alignées ou de même taille, pourrait rendre difficile le calcul des similarités. De plus, le choix de la mesure de similarité n'est pas toujours évident et dépend fortement de la nature des séries. En effet, dans le cas de la détection contextuelle, faire la différence entre une observation bruitée et une observation anormale est une tâche assez compliquée. Dans le cas de la détection globale, plusieurs approches supposent un alignement temporel parfait entre les séries temporelles, alors que ce n'est pas souvent le cas.

Dans la suite, nous nous intéressons aux différentes approches qui relèvent de la détection globale. A noter que ces approches s'adaptent facilement pour les deux autres cas de détection (la contextuelle et celle des sous-séries anormales). En effet, plusieurs approches ont été proposées pour la détection globale. Nous présentons particulièrement deux familles. La première utilise la notion de similarité, et considère comme anormale toute série s'écartant de la majorité. La deuxième famille, consiste à modéliser la normalité par un modèle paramétrique, et considère comme anormale toute série temporelle ayant une faible probabilité d'être générée par le modèle appris.

Afin de surmonter les problèmes liés à la détection dans les séries temporelles, et notamment ceux liés à la nature temporelle des données, certaines approches utilisent des mesures de similarité spécifiques alors que d'autres sont parfois précédées par une étape de transformation. Cette étape est essentielle, notamment pour les approches paramétriques qui se basent sur la modélisation séquentielle. Elle permet de transformer les séries en séquences de symboles, de projeter les séries dans un espace différent (e.g., espace temps vers l'espace fréquentiel).

2.2 Détection globale de séries anormales

Dans cette section, nous présentons une liste non exhaustive des différentes approches qui ont été proposées pour la détection globale des séries temporelles anor-

males. La plupart de ces approches nécessitent une base d'apprentissage ne contenant que des séries temporelles normales. Leur objectif commun est de détecter si une série temporelle de test est anormale par rapport à cette base d'apprentissage. Ce processus de détection s'inscrit dans le cadre *semi-supervisé* et se fait généralement en plusieurs étapes :

1. Modéliser la normalité à partir de la base d'apprentissage. La différence entre les approches se distingue principalement dans cette étape.

2. Calculer le score d'anormalité de chaque observation de la série temporelle de test.

3. Agréger les scores d'anormalité de toutes les observations afin d'obtenir un score d'anormalité global pour la série de test (par exemple en moyennant les scores de toutes les observations ou en prenant uniquement les scores des observations les plus anormales).

4. Déclarer la série temporelle de test comme étant anormale si son score dépasse un certain seuil donné par l'utilisateur.

Dans la suite nous détaillons le mécanisme de ces étapes à travers plusieurs approches de détection, notamment celles basées sur la notion de similarité et celles basées sur la modélisation paramétrique.

2.2.1 Approches basées sur la similarité

Plusieurs techniques existent dans cette famille d'approches. Certaines sont basées sur le principe de fenêtrage (découpage en sous-série) et d'autres sur la structure des données (k-plus proches voisins et clustering).

Approches à base de fenêtrage

Pour ces approches, les séries sont segmentées en plusieurs fenêtres de taille fixe (nous les appelons des sous-séries) afin de localiser la cause de l'anormalité d'une série dans l'une (ou dans plusieurs) de ses sous-séries. Ces approches partent du principe que l'anormalité d'une série temporelle pourrait se justifier par la présence d'une ou de plusieurs sous-séries anormales parmi l'ensemble de ses sous-séries [Gao02, Cabrera01, Endler98, Ghosh99a, Ghosh99b]. Les approches à base de fenêtrage tentent donc d'extraire des sous-séries de taille fixe (disant m) à partir des séries temporelles normales appartenant à une base d'apprentissage. Le score d'anormalité est ainsi calculé comme étant l'agrégation des scores de ses sous-séries comme le montre La Figure2.5 :

Cette technique peut se formaliser de la façon suivante :

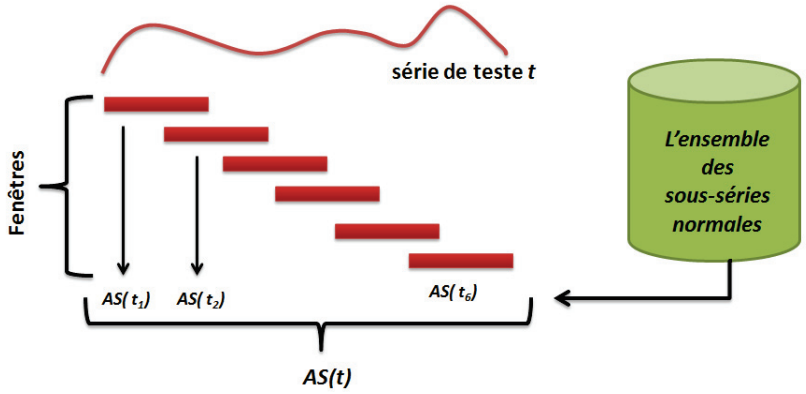


FIGURE 2.5 – Exemple d'agrégation du score d'anormalité pour une série temporelle de test [Gupta14].

Étant donné une base d'apprentissage de N séries normales, $T = \{T_1, T_2, ...T_N\}$, et une base de test $T'_{test} = \{T'_1, \ldots, T'_z\}$. La détection procède comme suit :

1. Extraire p sous-séries de chaque série temporelle $T_i : t_{i1}, \ldots, t_{ip}$, où p peut être calculé comme $|T_i| + m - 1$ dans le cas d'une extraction par fenêtrage glissant avec un pas unitaire.

2. Pour chaque série temporelle de test T'_i, extraire les $(|T'_i| + m - 1)$ sous-séries, $t'_{i1}, t'_{i2}, \ldots, t'_{ip}$.

3. Calculer le score d'anormalité de chaque sous-série de la série de test $AS(t'_{ij})$ en fonction de sa similarité avec les sous-séries de la base de séries normales.

4. Agréger les scores d'anormalité de toutes les sous-séries d'une série de test pour obtenir un score global.

Les approches utilisant cette technique diffèrent généralement dans la manière d'obtenir les sous-séries et de calculer le score d'anormalité pour chaque sous-série. Le score d'une sous-série de test peut être calculé en fonction de sa distance avec son $k^{\text{ème}}$ plus proche voisin dans la base d'apprentissage. D'autre techniques utilisent des modèles spécifiques, comme le OC-SVM [Schölkopf99], qui est une adaptation du SVM au problème de la classification à une seule classe. L'idée consiste à modéliser l'ensemble des données appartenant à la même et unique classe. La normalité des données est donc représentée par ce modèle. Ainsi, chaque sous-série de

test est classée comme normal ou anormale par le modèle appris. Toutefois, d'autres techniques utilisent des algorithmes de détection, initialement prévus pour des données vectorielles, pour calculer le score des sous-séries de la série temporelle de test [Lane97a, Hofmeyr98, Lane97b, Lane99].

Ces techniques peuvent être appliquées aux différents types de détection (contextuelle, globale ou détection des sous-séries). En effet, du moment où la série est segmentée en plusieurs sous-séries, on peut facilement détecter si elle est anormale ou pas. Cependant, ces approches sont assez sensibles au choix de la taille des segments et de la manière dont ils sont extraits. Aussi, leur complexité temporelle est assez importante car chaque sous-série de test est comparée avec l'ensemble de toutes les sous-séries de la base d'apprentissage. Cette complexité est alors de l'ordre $O((nL)^2)$, où L est la taille moyenne de toutes les séries et n est le nombre de toutes les séries temporelles (apprentissage et test).

Approches basées sur la structure des données

Cette famille d'approches se base essentiellement sur la notion de la similarité entre les séries temporelles de la base d'apprentissage et celle de test [Lane97a, Budalakoti06, Budalakoti09]. Elles utilisent des mesures de similarité adéquates pour calculer la proximité entre les séries temporelles de test et les séries d'apprentissage afin d'en déduire le score d'anormalité [Chandola08, Sequeira02]. En effet, ces techniques font l'hypothèse que les séries anormales sont assez différentes de celles qui sont normales au point de les qualifier par des mesures de similarité.

Le score d'anormalité d'une série temporelle de test par rapport à une base de séries normales est calculé selon deux méthodologies différentes :

1. La première se base sur le clustering. Une fois la mesure de similarité choisie, les séries temporelles sont réparties en plusieurs groupes via un processus de clustering de telle sorte que la similarité intra-cluster soit maximale et que la similarité inter-cluster soit minimale.

   La répartition des séries temporelles de la base se fait généralement via des algorithmes de clustering comme le $k-$means [Nairac99, Rebbapragada09], l'algorithme EM [Dempster77] (*espérance-maximisation*) dans [Pan09], le clustering dynamique dans [Sequeira02], le $k-$medoïde dans [Budalakoti06, Budalakoti09], le clustering hiérarchique dans [Portnoy01] ou le clustering des séries dans l'espace propre [Gupta13].

   Toutefois cette répartition peut se faire aussi via d'autres algorithmes d'apprentissage automatique comme le OC-SVM [Eskin02, Evangelista, Ma03a, Szymanski04] ou les cartes auto-organisatrices [González03].

   Une fois les séries temporelles réparties dans des groupes, chaque groupe se voit attribuer une série représentante que nous désignons par le mot medoïde tout au long de ce manuscrit. Le medoïde d'un groupe est tout simplement la série temporelle la plus représentative de l'ensemble des séries de ce groupe

[Budalakoti06, Budalakoti09]. Ainsi, le score d'anormalité d'une série test est calculé en fonction de sa distance avec le medoïde le plus proche. La différence entre ces techniques réside principalement dans le choix de la mesure de similarité ainsi que dans la façon d'extraire des groupes à partir de la base d'apprentissage des séries normales.

2. La deuxième méthodologie repose sur le principe du k-plus proches voisins. En effet, le score d'une série de test est calculé comme étant la distance de cette série par rapport à la $k^{\text{ème}}$ plus proche série voisine dans la base d'apprentissage.

Ces techniques permettent de détecter l'anormalité d'une série temporelle de test par rapport à une base de séries normales. Cependant, elles ne permettent pas de localiser directement la cause de cette anormalité comme pourraient le faire des techniques à base de fenêtrage. Néanmoins, un simple post-traitement peut être utilisé pour détecter la cause. L'efficacité de ces approches dépend fortement de la mesure de similarité. Toutefois, l'un des problèmes majeurs de ce type d'approches est leur aspect semi-supervisé. En effet, elles nécessitent une base de séries temporelles normales, alors que c'est rarement le cas dans la majorité des domaines d'application.

2.2.2 Approches basées sur les modèles paramétriques

Les approches dans cette catégorie se basent sur la modélisation statistique des séries normales pour détecter les séries anormales. Certaines se basent sur des modèles prédictifs, d'autres sur la modélisation séquentielle des séries en passant pas une phase de transformation.

Approches basées sur les modèles prédictifs

Dans cette famille de techniques, on suppose que les séries normales sont générées par un modèle, alors que les séries anormales sont générées par un autre modèle différent. L'idée consiste à apprendre les paramètres de ce modèle, qui est supposé avoir généré les séries normales, et d'estimer ensuite la probabilité qu'une série temporelle de test soit anormale ou normale en se basant sur le modèle appris. Ces techniques procèdent en quatre étapes :

1. Apprendre un modèle à partir d'une base de séries temporelles normales où les m observations de chaque série (*l'historique*) sont utilisées pour prédire la $(m+1)^{\text{ème}}$ observation.

2. Prédire pour une série temporelle de test, l'observation de chaque instant t via le modèle appris. L'erreur de prédiction de chaque observation est calculée comme étant la différence entre la valeur réellement observée et la valeur prédite.

3. Calculer le score d'anormalité de chaque observation comme étant l'erreur de prédiction.

4. Agréger les scores d'anormalité de toutes les observations pour obtenir un score d'anormalité global pour la série de test.

La différence entre ces techniques réside notamment dans le choix du modèle à apprendre. Nous pouvons les catégoriser de la façon suivante :

Les modèles de régression

Ces techniques utilisent des méthodes connues comme la régression linéaire, le processus gaussien ou les machines à vecteurs supports pour la régression [Basu07, Hill10, Ma03b]. Les sous-séries de taille m, qui sont extraites de toutes les séries temporelles normales sont utilisées comme données d'apprentissage pour apprendre les modèles. Le jeu de données d'apprentissage constitué de sous-séries est alors donné par, $D = \{X(t), y(t), t = m, \ldots, n-1\}$, avec $X(t) = [x(t-m+1), \ldots, x(t)]$ et $y(t) = x(t+1)$.

Ensuite, une fonction de régression linéaire Eq 2.1 est construite avec un vecteur de poids $w \in R^m$ et une fonction de mapping $\phi(X(t))$:

$$y = w^\top \phi(X(t)) + b \tag{2.1}$$

La différence entre les méthodes réside principalement dans la manière d'apprendre cette fonction (l'apprentissage des coefficients w) :

— **Régression Linéaire :**
Pour une régression linéaire simple, l'apprentissage des coefficients se fait en minimisant la somme des carrés des résidus $(y(i) - x(i+1))$. La fonction identité est utilisée comme fonction de mapping $\phi(X(t)) = X(t)$. Ce qui revient à résoudre le problème d'optimisation suivant :

$$\min_w \frac{1}{n} \sum_i (y_i - w^\top X(i))^2 \tag{2.2}$$

— **Machines à vecteurs supports pour la régression :** Ces modèles se basent sur la fonction de perte $\epsilon - insensitive$ [Ma03b]. La solution de l'équation 2.1 est obtenue via l'optimisation quadratique du problème suivant :

$$\min_w \frac{1}{2}\|w\|^2 + C\sum_i (\xi_i + \xi_i^*)$$

$$\text{sous les contraintes : } \begin{cases} y_i - (w^\top \phi(X(t)) + b) \leq \xi_i + \epsilon \\ (w^\top \phi(X(t)) + b) - y_i \leq \xi_i^* + \epsilon \\ \xi_i, \xi_i^* \geq 0 \end{cases} \tag{2.3}$$

Les estimateurs statistiques des séries temporelles

Ces techniques sont basées sur des modèles classiques de représentation des séries temporelles telles que les moyennes mobiles, l'auto-régression, ARMA (*autoregressive moving average*) ou l'ARIMA (*autoregressive integrated moving average*). Ces modèles prennent en entrée la série temporelle entière ainsi que la taille de l'historique m. La différence de ces modèles réside principalement dans leur façon de faire la prédiction :

— **Moving Average (MA) :**
Dans ce modèle, la série temporelle est considérée comme un processus à moyenne mobile d'historique m. Ainsi, la valeur à instant t dépend linéairement des valeur $x(t-k)$, avec $0 \leq k \leq m$:

$$y(t) = \sum_{i=0}^{m} b_i x(t-i) + \epsilon(t) \tag{2.4}$$

avec $b_1, \ldots, b_m$ qui représentent les coefficients du modèles et $\epsilon(t)$ représente le bruit blanc à l'instant t. Par exemple, si chaque valeur $x(t)$ est égale à la moyenne des m valeurs qui la précèdent, les coefficients du modèle sont $b_i = \frac{1}{m}$ et $\epsilon(t) = 0$.

— **Modèle d'auto-régression (AR) :**
Le modèle AR représente la série temporelle d'une manière récursive. La prédiction faite à l'instant t dépend des prédictions faites sur les m observations précédentes :

$$y(t) = \sum_{i=1}^{m} a_i y(t-i) + \epsilon(t) \tag{2.5}$$

avec $a_1, \ldots, a_m$ qui représentent les coefficients auto-régressifs du modèle et $\epsilon(t)$ qui représente le bruit blanc à l'instant t.

— **Le modèle ARMA :**
Ce modèle est une combinaison des deux précédents modèles. En effet, il calcule la prédiction en utilisant un filtre non-récursif et un autre récursif.

$$y(t) = \overbrace{\sum_{i=1}^{m} a_i y(t-i)}^{AR} + \overbrace{\sum_{i=0}^{m} b_i x(t-i)}^{MA} + \epsilon(t) \tag{2.6}$$

Les techniques décrites ci-dessus se basent sur l'historique de chaque série normale pour ajuster les modèles. La taille de cet historique est hypothétique et doit être choisie avec prudence. Si cette taille est assez grande, la dimensionnalité (nombre de variables utilisées) des données d'apprentissage devient importante, ce qui pourrait avoir un effet sur la complexité temporelle. Si la taille de l'historique est très petite, une perte d'information est possible (e.g., des séries normales périodiques avec un long cycle). Cependant, toutes ces approches se basent sur l'existence d'un modèle génératif pour détecter les séries anormales, ce qui constitue une hypothèse assez forte.

Approches basées sur la modélisation séquentielle des séries

Ces approches se basent sur les chaines de Markov cachées (HMM) [Yang03, Sun06]. L'hypothèse faite par ces approches, suppose qu'une série temporelle normale n'est en réalité qu'une observation indirecte d'une série temporelle cachée dont le processus est Markovien. Ainsi, les séries temporelles sont supposées être modélisables via un HMM alors que les séries anormales ne le sont pas [Chandola08, Florez-Larrahondo05, Gao02, Qiao02, Zhang03].

Soit une base de séries temporelles normales, on peut apprendre un modèle HMM(Θ), qui décrit le comportement normal de l'ensemble des séries via ses différents paramètres telles que la matrice de transition entre les états, la matrice des observations cachées et la distribution initiale des états. Le procédé des techniques basées sur les chaines de Markov cachées est le suivant :

1. Les séries temporelles normales sont transformées en séquences de symboles via les différentes méthodes de transformations discutées dans la section 2.4.

2. Étant donnée une base d'apprentissage composée de séquences normales, $S = \{S_1, \ldots, S_n\}$, déterminer le modèle Θ du HMM qui explique au mieux la normalité de cette base. L'algorithme de Baum-Welch [Baum70] qui est une généralisation de EM peut être utilisé pour estimer les différents paramètres du modèle.

3. Pour toute série de la base de test, calculer sa transformation en séquence, S'_i, puis calculer sa probabilité $P(S'_i|\Theta)$ via le modèle global appris. Si cette probabilité est faible par rapport à un certain seuil, la séquence S'_i est considérée comme étant anormale.

La possibilité d'interpréter ces modèles, représente un grand avantage pour ces techniques. Ainsi, la cause d'une anomalie peut se justifier facilement en analysant le modèle graphique probabiliste. En revanche, l'inconvénient de ces techniques est que les HMM ne permettent pas le passage à l'échelle notamment avec les grandes masses de données d'aujourd'hui. L'apprentissage des modèles, le choix des paramètres ainsi que leur initialisation requièrent un temps considérable. De plus, le fait que ces techniques reposent sur l'hypothèse forte de l'existence d'un processus caché de nature markovienne, qui génère les séries temporelles normales, peut réduire drastiquement leur performance si cette hypothèse n'est pas vérifiée.

2.3 Les mesures de similarité

Dans cette section, nous présentons les différentes mesures de distance, généralement proposées, pour mesurer la similarité entre les séries temporelles. Selon la catégorisation proposée par [Esling12], ces mesures peuvent être basées sur trois différentes catégories : la forme (*shape-based*), l'édition (*edit-based*) et les caractéristiques extraites des séries temporelles (*features-based*).

2.3.1 Mesures basées sur la forme des séries

Dans cette catégorie, les séries sont comparées en utilisant directement leurs valeurs originales ainsi que leur formes via des techniques différentes :

Distances type L_p

Les distances L_p sont issues des différentes normes L_p. Elles sont rigides et ne mesurent la distance qu'entre des séries de même taille. En revanche, elles ont été utilisées massivement dans les différentes tâches de l'apprentissage automatique à cause à leur simplicité. Étant données deux séries temporelles de même taille $X = \{x_0, x_1, ...x_{N-1}\}$ et $Y = \{y_0, y_1, ...y_{N-1}\}$, les différentes distances L_p sont reportées dans la Table 2.1.

Distance	p	Formule
Manhattan	$p = 1$	$\sum_{i=1}^{n} \lvert x_i - y_i \rvert$
Minkowski	$1 < p < \infty$	$(\sum_{i=1}^{n} \lvert x_i - y_i \rvert^p)^{1/p}$
Euclidean	$p = 2$	$\sqrt{\sum_{i=1}^{n} (x_i - y_i)^2)}$
Infinite norm	$p = \infty$	$\max_{i=1,...,n} \lvert x_i - y_i \rvert$

TABLE 2.1 – Les différentes distances L_p.

Short Time Series distance

La *Short Time Series distance* (STS) a été proposé par [Möller-Levet03] pour mesurer la distance entre les séries temporelles qui sont échantillonnées d'une façon irrégulière. Elle est définie par :

$$STS(X, Y) = \sqrt{\sum_{k=0}^{n-1} \left(\frac{y_{k+1} - y_k}{t_{k+1} - t_k} - \frac{x_{k+1} - x_k}{t'_{k+1} - t'_k} \right)} \tag{2.7}$$

où t, t' représentent les indices temporels de X et Y, respectivement. Les séries doivent avoir la même taille N. Leurs indices peuvent commencer et s'arrêter dans

des contextes temporels différents, par contre leur incrémentation doit être identique :

$$t_{k+1} - t_k = t'_{k+1} - t'_k, \forall k = 0, \ldots, n-1 \tag{2.8}$$

Distance Dissim

Cette distance a été proposée par [Frentzos07] dans le but de mesurer la similarité entre deux séries temporelles n'ayant pas forcément la même taille et n'ayant pas été échantillonnées de la même façon. Cela veut dire que les séries sont représentées par un ensemble fini d'indices temporels, mais ces indices peuvent être différents pour chaque série à condition qu'ils commencent et s'arrêtent au même contexte temporel.

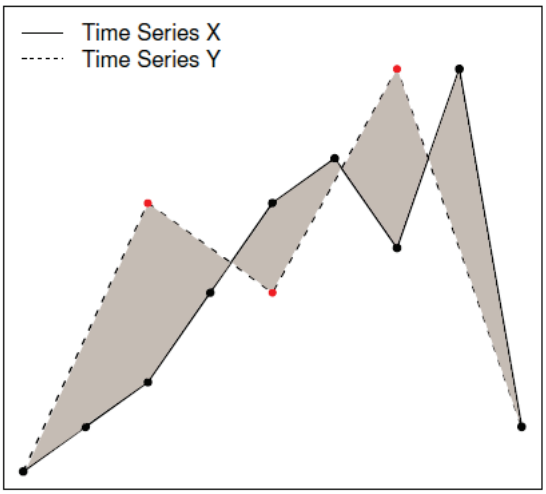


FIGURE 2.6 – Exemple de la distance *Dissim* entre deux séries temporelles échantillonnées différemment, [Mori16].

De plus, une représentation continue est requise pour chaque série entre deux indices successifs, de telle sorte qu'il y ait une linéarité entre les séries, comme le montre la figure 2.6. En d'autres termes, calculer la *dissim distance*, revient à calculer la somme des intégrales de toutes les régions engendrées par l'échantillonnage. Cette distance est définie donc par :

$$Dissim(X, Y) = \sum_{i=0}^{K-1} \int_{t_i}^{t_{i+1}} D_{X,Y}(t)dt \tag{2.9}$$

où, $T = t_0, \ldots, t_{K-1}$ représente l'ensemble global des indices temporels, regroupant ainsi les indices des deux séries X et$Y$. La distance euclidienne entre les deux séries à un instant t est représentée par $D_{X,Y}(t)$. Toutefois, pour simplifier le calcul d'intégrale

dans l'équation (2.9), une approximation de cette distance, à base de trapèzes, a été proposée par les mêmes auteurs. Ce qui donne la formule suivante :

$$Dissim\_approx(X,Y) = \sum_{i=0}^{n-1} \left(D_{X,Y}(t_i) + D_{X,Y}(t_{i+1})\right) \times (t_{i+1} - t_i) \tag{2.10}$$

DTW : Dynamic Time Warping

Afin de surmonter les inconvénients liés aux distances rigides telles que les distances L_p, de nombreux mesures de similarité ont été spécialement conçues pour des séries temporelles. Parmi ces mesures, la plus populaire est certainement la DTW (*Dynamic Time Warping*) [Vintsyuk68].

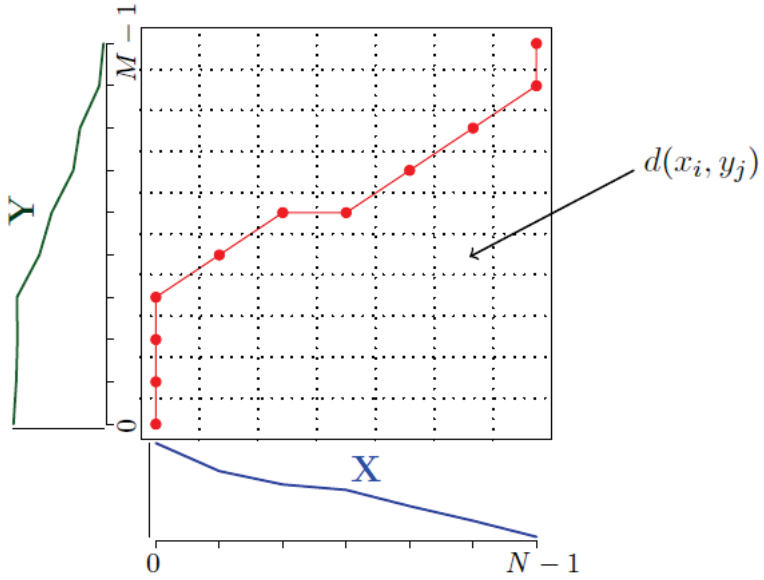


FIGURE 2.7 – *DTW* entre deux séries X et Y de différentes tailles, [Mori16].

Cette mesure est connue pour être robuste par rapport aux différentes transformations telles que la déformation locale et le décalage dans le temps. De plus, elle peut mesurer la similarité entre des séries de tailles différentes. Comme le montre la Figure 2.7, l'objectif de cette distance est de trouver un alignement optimal entre deux séries $X = \{x_0, x_1, ...x_{N-1}\}$ et $Y = \{y_0, y_1, ...y_{M-1}\}$ en cherchant le chemin minimal dans une matrice de distance D qui défini un appariement entre elles. Chaque case dans cette matrice est définie comme étant la distance euclidienne entre une paire de points (x_i, y_j). Le problème d'optimisation est formulé avec trois contraintes. La condition des bornes force le chemin à commencer par la position $D(0,0)$ et à s'arrêter dans la position $D(N-1, M-1)$. La condition de la continuité limite la taille du pas, en forçant le

chemin à continuer en choisissant l'une des cellules adjacentes. Enfin, la condition de la monotonie interdit au chemin de reculer dans les positions de la matrice. Ainsi, le problème d'optimisation peut être formulé de la façon suivante :

$$DTW(X,Y) = \begin{cases} 0, & \text{si}(M-1) = (N-1) = 0 \\ \infty, & \text{si } (M-1) = 0 \text{ ou } (N-1) = 0 \\ \Omega, & \text{sinon.} \end{cases} \tag{2.11}$$

où

$$\Omega = d(x_0, y_0) + \min \begin{cases} DTW(tail(X), tail(Y)), \\ DTW(X, tail(Y)), \\ DTW(tail(X), Y) \end{cases}$$

A note que $d(.,.)$ est la distance euclidienne et $tail(X)$ une fonction qui renvoie une nouvelle série $\{x_1, ...x_{N-1}\}$ qui prend tous les éléments X sauf son premier élément x_0. En utilisant la programmation dynamique, ce problème d'optimisation peut être résolu par l'algorithme 1.

Algorithme 1 : DTW

Entrées : $X = \{x_0, \ldots, x_{n-1}\}$, $Y = \{y_0, \ldots, y_{m-1}\}$
Résultat : Distance entre X et Y

Initialisation : $DTW = matrice[N+1, M+1]$
Pour $i \leftarrow 1$ **à** n
  $DTW(i, 0) = \infty$
fin_pour
Pour $j \leftarrow 1$ **à** m
  $DTW(0, j) = \infty$
fin_pour
Pour $i \leftarrow 1$ **à** n
  Pour $j \leftarrow 1$ **à** m
    $$DTW(i,i) = d(x_i, y_j) + \min \begin{cases} DTW(i-1, j) \\ DTW(i, j-1) \\ DTW(i-1, j-1) \end{cases}$$
  fin_pour
fin_pour
retourner $DTW[n, m]$

De plus, il est à noter qu'il est assez fréquent de rajouter une contrainte temporelle supplémentaire comme celle de Sakoe-Chiba [Sakoe78] afin de réduire le nombre de déplacements verticaux ou horizontaux que le chemin peut suivre consécutivement. En effet, cette contrainte consiste à mettre un ruban symétrique autour de la diagonale et à forcer le chemin à y rester comme le montre la Figure 2.8. Cet ajustement permet en

effet d'éviter d'apparier des points qui sont assez distants par rapport à l'axe temporel, et de réduire la complexité de l'algorithme.

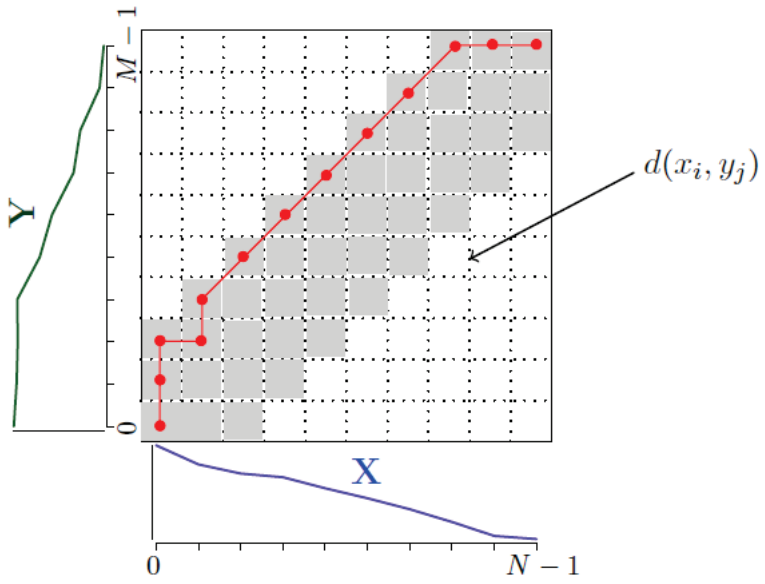


FIGURE 2.8 – *DTW* avec une restriction dans l'espace de recherche, [Mori16]

.

L'un des inconvénients de la DTW est sans doute sa complexité quadratique qui pose problème notamment dans le cas des séries temporelles de grande taille. Afin de surmonter ce problème, plusieurs bornes inférieures de cette distance ont été proposées. Il s'agit des approximations de la DTW qui renvoient toujours une valeur inférieure à la distance originale. En effet, elles sont moins gourmandes en temps par rapport à la DTW. L'une des bornes la plus utilisée est celle proposée par [Keogh05b]. Elle consiste à calculer l'enveloppe (minimale et maximale) d'une des séries à comparer et de calculer une distance en fonction de ces deux enveloppes.

Étant données la taille de la largeur de la marge de Sakoe-Chiba (r), et deux séries X et Y, la première étape consiste à calculer les enveloppe minimale et maximale d'une des séries, par exemple X :

$$U_i = max(X_{i-r}, X_{i+r}) \tag{2.12}$$
$$L_i = min(X_{i-r}, X_{i+r}) \tag{2.13}$$

La distance, notée *LB_Keogh*, est calculée comme étant la distance euclidienne entre les points de la série Y qui ne sont pas contenus dans les deux enveloppes et le point le plus proche de l'enveloppe correspondante.

$$LB\_Keogh(X,Y) = \sum_{i=0}^{N-1} \begin{cases} (Y_i - U_i)^2, & \text{if } Y_i > U_i \\ (Y_i - L_i)^2, & \text{if } Y_i < L_i \\ 0, & \text{sinon.} \end{cases} \tag{2.14}$$

Cependant, cette borne inférieure n'est valable que pour des séries temporelles ayant la même taille.

2.3.2 Mesures à base d'édition

Les mesures basées sur l'édition ont été initialement conçues pour calculer la similarité entre deux séquences symboliques (composées de string). Ces mesures sont utilisées dans plusieurs applications de détection d'anomalies dans les séquences [Budalakoti06, Budalakoti09, Chandola08]. Elles sont basées sur l'idée de compter le nombre minimum d'opérations d'édition (suppression, insertion, remplacement) nécessaire pour transformer une séquence à une autre.

Quant à l'utilisation aux séries temporelles, ces mesures ne sont pas directement applicables du fait qu'il soit assez difficile de trouver un appariement exact pour des valeurs réelles, ce qui n'est pas le cas pour les string dans les séquences. Néanmoins, quelques adaptations ont été proposées [Chen05, Chen04]. Autrement, les séries de tailles différentes peuvent être comparées par ces mesures du moment où des opérations d'insertion et de suppression sont utilisées.

Edit distance pour les séquences réelles (*EDR*)

Afin d'adapter la notion d'édition pour des valeurs réelles, la distance entre les points des deux séries temporelles est comprise entre 0 et 1 dans [Chen05]. Ainsi, deux points x_i et y_j sont considérés comme étant égaux si leur distance est inférieure à un certain seuil ϵ, défini par l'utilisateur. Autrement, ils seront considérés comme points différents et la distance entre eux revient à 1.

Cette mesure permet de faire des sauts dans l'alignement, ce qui implique que les points ne sont pas forcément tous appariés. En revanche, une pénalité par saut (taille du saut) peut être rajoutée à la distance finale.

$$EDR(X,Y) = \begin{cases} N & \text{si } M-1=0 \\ M & \text{si } N-1=0 \\ \Pi & \text{sinon.} \end{cases} \tag{2.15}$$

où

$$\Pi = d(x_0,y_0) + \min \begin{cases} EDR(tail(X), tail(Y)) + d_{edr}(x_0,y_0,\epsilon), \\ EDR(tail(X), Y) + 1, \\ EDR(X, tail(Y)) + 1. \end{cases}$$

avec d_{edr} qui représente la distance entre deux points de la série et qui prend 0 ou 1 :

$$d_{erd}(x_i, y_i, \epsilon) = \begin{cases} 0 & \text{if } |x_i - y_i| \leq \epsilon \\ 1 & \text{sinon} \end{cases}$$

Le processus de calcul de cette mesure est formulé d'une façon récursive dans Eq (2.15) et peut être résolu par la programmation dynamique comme l'algorithme de la DTW.

Longest Common Subsequence distance *(LCSS)*

La mesure de similarité *Longest Common Subsequence distance (LCSS)* a été proposé par [Vlachos02]. Comparer deux séries temporelles en utilisant cette mesure, revient à trouver la plus longue sous-série commune entre ses deux séries tout en autorisant des sauts. Tout comme la *EDR*, la distance entre deux points est réduite à 0 ou 1 par rapport à un certain seuil ϵ. Le processus de calcul de cette mesure peut être formulé par la récursivité suivante :

$$LCSS(X, Y) = \begin{cases} 0 & \text{si } M - 1 = 0 \text{ ou } N - 1 = 0 \\ LCSS(tail(X), tail(Y)) + 1 & \text{si } |x_0 - y_0| \leq \epsilon \\ \Psi & \text{sinon.} \end{cases} \tag{2.16}$$

avec $\Psi = \max(LCSS(tail(X), Y), LCSS(X, tail(Y)))$

Mesure d'édition avec une pénalité réelle *(ERP)*

La troisième adaptation du concept d'édition pour les séries temporelles qui a été proposée par [Chen04], peut être vue comme une combinaison entre la DTW et la mesure EDR. En effet, tout comme cette dernière, cette mesure autorise les sauts mais elle les pénalise différemment. La *Edit distance with Real Penalty (ERP)* pénalise les sauts par l'ajout de la distance euclidienne entre les points non appariés et une constante g définie par l'utilisateur. Le calcul de cette mesure peut être formulé de la façon suivante :

$$ERP(X, Y) = \begin{cases} \sum_{i=0}^{N-1} |y_i - g| & \text{si } M - 1 = 0 \\ \sum_{i=0}^{N-1} |x_i - g| & \text{si } N - 1 = 0 \\ \kappa & \text{sinon.} \end{cases} \tag{2.17}$$

$$\text{avec } \kappa = \min \begin{cases} ERP(tail(X), tail(Y)) + d(x_0, y_0) \\ ERP(tail(X), Y) + d(x_0, g) \\ ERP(X, tail(Y)) + d(y_0, g). \end{cases}$$

2.3.3 Mesures basées sur la transformation

Les mesures de similarité dans cette catégorie consiste à extraire des caractéristiques (*features*) à partir des séries temporelles, et de comparer ces séries en se basant sur leurs caractéristiques au lieu de d'utiliser directement leurs valeurs originales.

Les distances basées sur la corrélation de *Pearson*

La corrélation de *Pearson* entre deux séries de même taille $X = \{x_0, x_1, ...x_{N-1}\}$ et $Y = \{y_0, y_1, ...y_{N-1}\}$ est définie par l'équation suivante :

$$PC(X,Y) = \frac{\sum_{i=0}^{n-1}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{(x_i - \bar{x})^2}\sqrt{(y_i - \bar{y})^2}} \tag{2.18}$$

avec $\bar{x}$ et $\bar{y}$ les moyennes des séries X et Y respectivement. En se basant sur cette caractéristique (corrélation), deux mesures ont été proposées par [Golay98] :

$$d_{PC_1} = (\frac{1 - PC}{1 + PC})^{\beta} \tag{2.19}$$

$$d_{PC_2} = 2(1 - PC) \tag{2.20}$$

avec β un paramètre positif défini par l'utilisateur. Plus la corrélation est forte entre les deux séries, plus la valeur retournée par la mesure est proche de zéro.

Les distances basées sur la corrélation croisée

La mesure proposée par [Warren Liao05] est basée sur la corrélation-croisée. Cette corrélation avec un pas de décalage k est définie par la formule suivante :

$$CC_k(X,Y) = \frac{\sum_{i=0}^{n-1-k}(x_i - \bar{x})(y_{i+k} - \bar{y})}{\sqrt{(x_i - \bar{x})^2}\sqrt{(y_{i+k} - \bar{y})^2}} \tag{2.21}$$

avec $\bar{x}$ et $\bar{y}$ les moyennes des séries X et Y respectivement. En se basant sur la corrélation croisée dans Eq (2.21), les mêmes auteurs ont proposé la mesure suivante :

$$d_{CC}(X,Y) = \sqrt{\frac{(1 - CC_0(X,Y))}{\sum_{k=1}^{max} CC_k(X,Y)}} \tag{2.22}$$

La distance basée sur les intervalles temporels

Une mesure, appelée *TQuest*, a été proposée par [Aßfalg06]. Elle est basée sur les intervalles temporels. Dans un premier temps, chaque série est représentée sous forme d'un ensemble d'intervalles temporels tels que :

— Toutes les valeurs de la série dans un intervalle donné, soient strictement supérieures à un seuil τ défini par l'utilisateur .
— Chaque intervalle doit être le plus large possible.

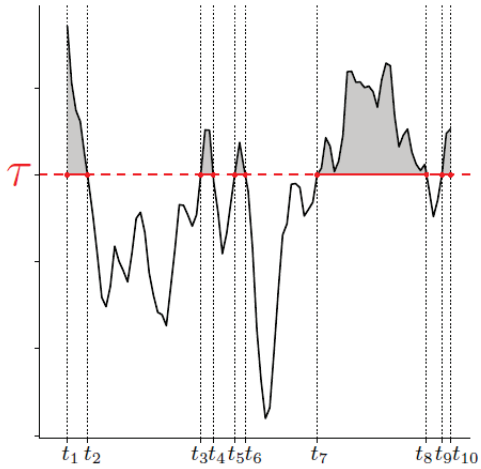


FIGURE 2.9 – Segmentation de la série pour le calcul de *TQuest* [Mori16].

Par exemple, dans la Figure 2.9, la série temporelle X est représentée par un ensemble de 5 intervalles temporels : $S_X = \{(t_1, t_2), (t_3, t_4), (t_5, t_6), (t_7, t_8), (t_9, t_10)\}$. Chaque intervalle étant caractérisé par ses indices de départ et d'arrêt (I_d, I_f), la similarité entre deux intervalles (I, I'), revient à comparer leurs indices par l'équation suivante :

$$d_{int}(I, I') = \sqrt{(I_d - I'_d)^2 + (I_f - I'_f)^2} \tag{2.23}$$

Intuitivement, deux intervalles sont égaux si leurs indices démarrent et s'arrêtent au même contexte temporel. Une fois les séries temporelles X et Y représentées par leurs ensembles d'intervalles S_X et S_Y respectivement, la distance *TQuest* est obtenue par la formule suivante :

$$TQuest(X, Y) = \frac{1}{|S_X|}\left[\sum_{s \in S_X} \min_{t \in S_Y} d_{int}(s, t)\right] + \frac{1}{|S_Y|}\left[\sum_{s \in S_Y} \min_{t \in S_X} d_{int}(t, s)\right] \tag{2.24}$$

En effet, l'idée de cette mesure est d'apparier chaque intervalle de la série X avec l'intervalle temporel le plus proche, en terme de distance d'intervalles Eq (2.23), de la série Y et vice versa.

Comparaison et choix des mesures

Dans cette partie, nous dressons un bilan sur l'ensemble des mesures de similarité décrites ci-dessus. Dans la Table 2.2, nous comparons les mesures selon quatre critères différents :

- L'alignement temporel,
- Le calcul de distance entre des séries de différentes tailles,
- La complexité temporelle des mesures,
- Le nombre minimum des paramètres à fournir pour la mesure.

MESURES	ALIGNEMENT	#TAILLE	COMPLEXITÉ	PARAMÈTRES
L_p	✗	✗	$O(n)$	0
DTW	✓	✓	$O(n^2)$	1
LB_KEOGH(DTW)	✓	✗	$O(n)$	1
DISSIM	✓	✓	$O(n^2)$	0
EDR	✓	✓	$O(n^2)$	2
LCSS	✓	✓	$O(n)$	2
ERP	✓	✓	$O(n^2)$	2
CORRÉLATION	✗	✗	$O(n \log n)$	0
TQUEST	✓	✓	$O(n^2 \log n)$	1

TABLE 2.2 – Comparaison entre mesures de similarité.

Pour nos algorithmes de détection globale, que nous allons présenter dans le chapitre suivant, et qui se basent principalement sur la notion de similarité, nous avons choisi d'utiliser la DTW pour les séries de tailles différentes et la LB_Keogh(DTW) pour les séries de même taille. Ce choix a été motivé notamment par la qualité de l'alignement temporel que fournit la DTW et la robustesse de son utilisation.

2.4 Techniques de transformation des séries temporelles

Plusieurs problèmes peuvent impacter l'analyse et le traitement des séries temporelles. Ils sont liés à leur grande dimensionnalité (nombre d'observations) et au bruit. Pour remédier à ces problèmes, une étape de transformation peut être une solution tangible à la détection. Dans certains cas, les anomalies ne peuvent être détectées que si les séries ont été transformées et analysées dans un espace différent de celui de l'origine. Par ailleurs, la complexité temporelle de certaines approches de détection à base de similarité (e.g., *voisins les plus proches, clustering*) pourrait être considérablement réduite par la transformation des séries (e.g., la réduction de dimensionnalité).

Un autre point important concerne la normalisation qui constitue une étape essentielle de pré-traitement. Par exemple, dans le cas des séries temporelles multi-variées, les approches de détection supposent que les séries soient définies sur la même échelle alors que ce n'est pas souvent le cas, d'où l'intérêt de la normalisation pour que toutes les séries contribuent équitablement au calcul de la similarité. Dans la sous-section suivante, nous présentons quelques méthodes de transformation utilisées dans la littérature.

2.4.1 Méthodes à base d'agrégation

Les méthodes basées sur l'agrégation réduisent la taille de la série temporelle en remplaçant ses segments par des agrégations, comme la moyenne ou la médiane. Ces méthodes permettent à la fois de réduire la dimensionnalité de la série temporelle, de la rendre plus régulière et d'estomper le bruit ainsi que les valeurs manquantes. Par contre, ces méthodes pourraient engendrer une perte d'information importante de la série, en rendant ainsi la détection plus difficile.

Parmi les méthodes les plus connues est la PAA (*Piecewise Aggregate Approximation*) qui a été proposée par [Lin03]. Dans cette méthode, la dimensionnalité d'une série temporelle de taille n est réduite à une dimension inférieure m en divisant la série en m segments de même taille. L'agrégation des valeurs de chaque segment est ensuite calculée et stockée dans un vecteur de taille m, appelé vecteur des *coefficients de PAA*. La PAA de la série est représentée par ce vecteur (Figure 2.10).

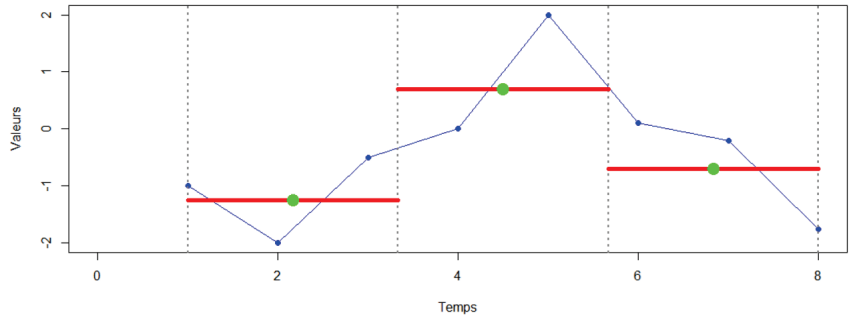


FIGURE 2.10 – Représentation des valeurs de la série par les 3 coefficients de la PAA (en vert).

Plus formellement, une série temporelle $t = t_1, \ldots, t_n$ de taille n est transformée en un vecteur $z = (z_1, \ldots, z_m)$ de taille m. Chaque élément z_i peut être calculé par l'équation suivante :

$$z_i = \frac{m}{n} \sum_{j=\frac{n}{m}\times(i-1)+1}^{\frac{n}{m}\times i} t_i \tag{2.25}$$

Enfin, si m est proche de n, la transformation sera assez similaire à la série originale. Sinon, elle sera beaucoup plus courte et ceci pourrait impliquer une perte d'information.

2.4.2 Méthodes à base de discrétisation

Les méthodes basées sur la discrétisation transforment la série temporelle en une séquence de symboles discrets. Cette transformation est souvent motivée par la possibilité de réutiliser les algorithmes classiques de détection d'anomalies dans les séquences ; et l'utilisation des mesures de similarité spécifiquement adaptées pour les séquences symboliques.La discrétisation peut donc être effectuée en trois étapes :

1. Représenter la répartition des valeurs de la série par un histogramme (plusieurs classes).
2. Assigner à chaque classe un symbole.
3. Transformer la série en remplaçant chacune de ses valeurs par le symbole de sa classe.

En effet, les techniques de discrétisation varient principalement sur la manière dont les valeurs de la séries sont représentées (e.g., le nombre de classes, la largeur des classes et le choix des symboles). La méthode SAX (*Symbolic Aggregate approXimation*) est sans doute l'une des méthodes de discrétisation la plus connue [Lin03]. Elle permet de combiner la réduction de la dimensionnalité en utilisant la PAA, décrite ci-dessus, et la représentation symbolique basée sur la discrétisation des valeurs par une loi gaussienne (cf. la Figure 2.11).

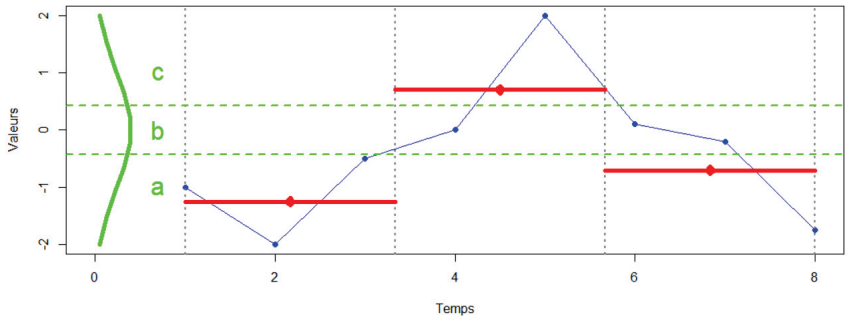


FIGURE 2.11 – Représentation symbolique d'une série de 8 valeurs par la séquence *aca* en utilisant 3 symboles ($\alpha = 3$) et trois segments.

De plus, elle permet d'obtenir une borne inférieure des distances de type L_p [Keogh07]. Tout d'abord, la série doit être normalisée, ensuite elle est divisée en m segments de même taille. Chaque segment m_i est représenté par une agrégation de ses valeurs, ce qui revient à appliquer la PAA. Ensuite, les coefficients du PAA sont traduits en symboles grâce à un paramètre α fixé par l'utilisateur. Ce paramètre permet de découper la distribution des valeurs, supposée normale, en α régions équiprobables. Chaque segment est alors représenté par le symbole de la région à laquelle il

appartient. Plusieurs variantes de cette approche ont été proposées notamment dans [Malinowski13] où les auteurs proposent une représentation prenant en considération les tendances de la série (via des régressions locales) contrairement à SAX où uniquement une simple agrégation (e.g., moyenne, medians,...) est prise en compte.

2.5 Applications

Plusieurs domaines d'application ont fait appel à la détection d'anomalies dans les séries temporelles. Nous présentons dans cette section quelques applications sur des données environnementales, industrielles, biologiques, astronomiques, de surveillance et économiques.

2.5.1 Données environnementales

Les données environnementales ont fait l'objet de plusieurs études autour de la détection d'anomalies. Dans [Hill10, Hill09], les auteurs détectent les erreurs de mesures de la vitesse du vent en utilisant des réseaux bayésiens dynamiques. Les données sont issues du corpus *WATERS Network Corpus*. Dans [Angiulli07], les auteurs détectent des anomalies environnementales en analysant des données de la pluie, de la température de la surface de la mer, et l'humidité. Les séries temporelles de précipitation sont issues de *Pacif Marine Environmental Laboratory of the U.S. National Oceanic*. Des anomalies spatio-temporelles ont été détectées par [Birant06] en analysant la hauteur des vagues de la mer noire, la mer de Marmara, la mer Égée et la mer méditerranéenne. Les anomalies représentent des zones avec des vagues de hauteur importante dans un contexte temporel particulier par rapport à des zones voisines. Dans [Yuxiang05], les auteurs explorent la partie sud de la Chine en analysant une série temporelle multi-variée de 1992 jusqu'à 2002 contenant des informations sur le pourcentage de couverture nuageuse, la vapeur, la pression, les précipitations ainsi que l'amplitude thermique. Leur analyse permet de mettre en évidence des zones ayant un comportement anormal par rapport aux zones voisines (e.g., détection de zones de sécheresse ou d'inondation).

2.5.2 Données issues de capteurs industriels

En utilisant des séries temporelles issues de l'enregistreur des données de vols (*flight data recorder*), les auteurs dans [Basu07] détectent des changements bruts dans l'altitude ainsi que dans l'angle d'inclinaison latérale de l'avion. La détection se fait via des modèles des prédictions simples. Si l'observation s'écarte de la médiane des observations voisines, elle est détectée comme étant anormale. En utilisant un algorithme de clustering de type k-moyennes, des anomalies ont été détectées dans les réacteurs d'avion en analysant les données de pression et vibration [Nairac99]. Les auteurs dans [Bu07] détectent des sous-séries anormales à partir des données de consommation d'énergie

en utilisant la transformée de Haar qui est une ondelette représentée par une fonction constante par morceaux.

2.5.3 Données astronomiques

La détection d'anomalies dans les données d'astronomie a fait l'objet de plusieurs études. Par exemple, dans [Keogh05a], les auteurs détectent des sous-séquences anormales dans les données de la télémétrie spatiale. Les auteurs dans [Yankov08] détectent des sous-séries anormales à partir des données de lumière des étoiles.

2.5.4 Données biologiques

En analysant les données d'électrocardiogramme, les auteurs dans [Keogh05a], détectent des sous-séries anormales. Dans le domaines de l'anthropologie, de la zoologie et de la médecine, la forme de certains objets et espèces était modélisée sous forme de séries temporelles [Wei06]. Ces mêmes auteurs utilisent des techniques de détection pour mettre en évidence des papillons significativement différents par rapport à un ensemble de papillons similaires. Dans un tout autre contexte et à partir d'un jeu de données de globules rouges, des techniques ont été utilisées pour détecter des dacryocytes qui sont des globules rouges en forme de larmes. En effet, leur existence est souvent synonyme d'une anomalie grave dans le sang [Wei06]. En microbiologie, un jeu de données sur des champignons a été utilisé pour détecter les cellules (les spores) qui germent un tube germinatif. Enfin, des animaux sauvages menacés d'extinction ont été équipés de capteurs afin de détecter des anomalies dans leur comportement migratoire.

2.5.5 Réseaux informatiques

Plusieurs techniques de détection d'anomalies dans les données temporelles ont été spécialement conçues pour détecter des intrusions dans les réseaux informatiques [Sequeira02, Hofmeyr98, Lane97b, Lane99, Angiulli07, Warrender99]. Les auteurs dans [Lakhina04] utilisent des séries temporelles multi-variées mesurant le nombre de bits, de paquets, le niveau des adresses IP ainsi que d'autres métriques, pour découvrir des anomalies qui correspondent à des attaques de type DOS (*denial of service*) et DDOS (*distributed denial of service*). Des caractéristiques spécifiques aux données ont été rajoutées à l'ensemble d'apprentissage comme la durée de communication, le type de protocole utilisé, le nombre d'octets transmis par une connexion TCP. En se basant sur ces données, les auteurs dans [Portnoy01] détectent des anomalies en utilisant un clustering hiérarchique. Dans [Endler98], des modèles markoviens ont été appris à partir des données d'audit du système d'exploitation Solaris de MIT, pour détecter des scénarios d'intrusion tels que les tentatives répétitives pour deviner un mot de passe ou l'accès distant à une ressource sans en avoir les droits. Aussi, les commandes Unix

des utilisateurs ont été utilisées pour détecter les séquences d'intrusion en utilisant des SVMs [Szymanski04]. Des modèles auto-régressifs ont été utilisés dans [Jiang06] pour apprendre la dépendance entre les différents points d'intensités de flux dans le trafic du réseau.

2.5.6 Données économiques

Plusieurs données économiques avec un aspect temporel ont été étudiées pour détecter tout type d'anomalies. Par exemple, les auteurs dans [Gupta12a] mettent en évidence des pays avec un comportement anormal dans les différents composants de leurs PIBs (e.g., consommation, investissement, dépenses et exportations) à travers le temps en utilisant des méthodes spécifiques pour la détection de communautés. Les mêmes auteurs ont détecté, à partir d'un jeu de données de Budget, des états américains ayant un comportement anormal par rapport à la distribution de la dépense dans différents domaines. Dans le domaine de la politique, des données de vote du congrès américain ont été analysées par [Otey06] pour découvrir qu'un homme politique républicain avait voté différemment que son parti pour plusieurs projets de loi.

2.6 Conclusion

Dans ce chapitre, nous avons introduit le problème de la détection d'anomalies dans les séries temporelles. Nous avons présenté deux familles d'approches qui ont été proposées pour traiter ce problème. La première se base sur la similarité et la proximité entre les séries temporelles. La deuxième se base sur la modélisation de la normalité par des modèles prédictifs ou par modélisation séquentielle. Certaines approches nécessitent des mesures de similarité spécifiques pour pouvoir détecter les dites anomalies. Pour cela, nous avons abordé les différentes mesures de similarité proposées et nous les avons comparées par rapport à plusieurs critères notamment en terme de complexité et de robustesse. D'autres approches passent par une étape de transformation pour modéliser la normalité des séries. Aussi, nous avons discuté brièvement l'idée et l'intérêt de cette étape, ainsi que quelques méthodes connues comme PAA et SAX. Enfin, nous avons montré quelques exemples d'application de la détection d'anomalies dans les données temporelles au sens large.

L'inconvénient majeur des approches étudiées dans ce chapitre, est l'aspect *semi-supervisé*. En effet, toutes les approches nécessitent une base d'apprentissage de séries temporelles normales. Or il est assez difficile d'obtenir de telle bases de séries. Dans le chapitre suivant, nous présenterons nos différentes contributions qui permettent de détecter des anomalies dans un mode complètement *non-supervisé*.

3

Détection globale non-supervisée : Approches à base de pondération

▷ *Dans ce chapitre, nous nous intéressons à la détection non-supervisée des séries temporelles anormales par rapport à un ensemble de séries (e.g., détection globale). Ainsi, nous présentons les quatre approches que nous avons proposées. La première approche (DetecS), dite à deux niveaux séquentiels, consiste à effectuer un clustering spectral sur les séries temporelles et à calculer ainsi une fonction de score spécifique pour détecter les séries anormales. Ensuite, nous reformulons, à travers les trois autres approches dites (embedded), la tâche de détection comme un clustering pondéré où la détection et le clustering se font simultanément. La différence entre ces trois approches résident principalement dans le mécanisme de pondération ainsi qu'au niveau de la détection, globale (DOTS et ℓ_2-DAt) ou locale (L2GAD). Enfin, nous présentons les résultats obtenus via les expérimentions extensives que nous avons menées pour valider et évaluer nos approches par rapport aux approches de l'état de l'art.*

◁

Plan du chapitre

3.1 Introduction

Comme mentionné dans le chapitre précédent, les approches de détection de séries temporelles anormales, nécessitent une base de séries normales, a priori. Par exemple, certaines approches basées sur la similarité, font un clustering sur une base de séries normales et calculent le score d'anormalité d'une série de test en fonction de sa distance par rapport à au médoïde le plus proche. Le problème avec ce genre d'approches, est la nécessité d'une base d'apprentissage, souvent difficile à obtenir dans la plus part des applications réelles. Toutefois, les approches basées sur le clustering peuvent être aussi appliquées dans le cas non-supervisé. Cependant, elles s'avèrent moins efficaces du fait que les séries anormales peuvent influencer le processus du clustering et par conséquent, rendre la détection biaisée.

Afin de surmonter ces problèmes, nous proposons dans ce chapitre quatre approches de détection dans un mode totalement *non-supervisé*, qui nécessite aucune connaissance a priori sur la normalité. Dans la section 3.2, nous détaillons la première approche (appelée DetectS), dite à deux niveaux séquentiels, et qui consiste à effectuer un clustering spectral sur la base des séries, suivi d'une fonction de score spécifique pour détecter les séries anormales. Dans les sections 3.3.1,3.3.2 et 3.3.3, nous décrivons les trois autres approches, de type *embedded*, qui , contrairement à la première, le clustering est intrinsèquement lié à la détection. Pour ce faire, nous proposons des algorithmes à base de pondération de séries temporelles. Deux visions sont développées à cet effet, (1) une vision globale (DOTS et ℓ_2-DAT) où chaque série se voit attribuer un score d'anormalité par rapport à l'ensemble des clusters et (2) une deuxième vision locale (L2GAD) où chaque série se voit attribuer un score d'anormalité par cluster. Ces approches permettent d'évaluer la contribution de chaque série par rapport à la structure de la base.

3.2 Détection séquentielle à base de clustering spectral (*DetectS*)

Dans un cadre non-supervisé, nous considérons $T = \{t_1, t_2, ..., t_n\}$ une base de n séries temporelles de tailles différentes. L'hypothèse de départ est de supposer que les séries anormales sont celles qui s'écartent significativement de la majorité des autres séries, et devraient donc maximiser l'inertie de leur cluster. Pour détecter des séries anormales, nous supposons que les séries normales sont largement majoritaires dans l'ensemble T, et qu'il y ait plusieurs profils de normalité. L'idée consiste donc à trouver ces différents profils qui peuvent exister dans la base, et de déclarer une série temporelle comme étant anormale, si elle diffère significativement par rapport aux séries temporelle de son profil.

Nous modélisons donc la base des séries T sous forme d'un graphe $G = (V, E)$ non-orienté où chaque série t_i est représentée par un sommet V_i et le poids de chaque

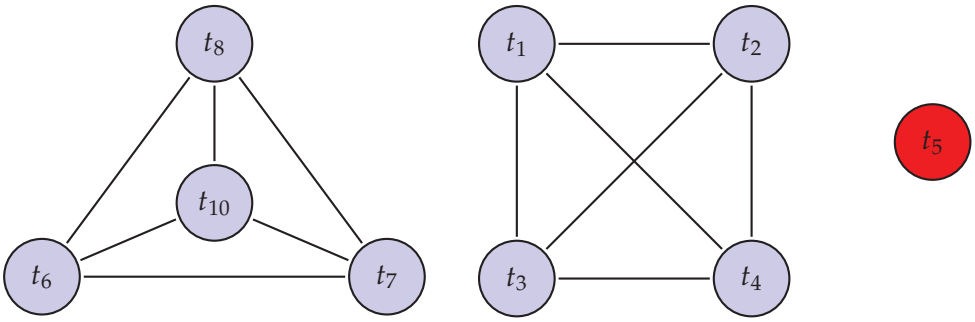


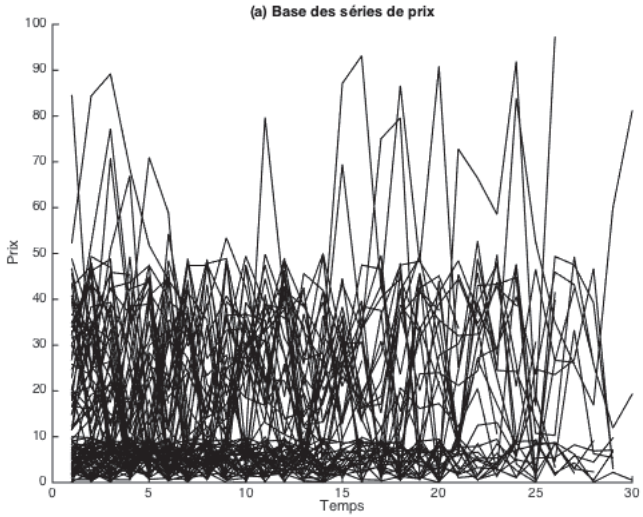
FIGURE 3.1 – Représentation d'une base de séries sous forme d'un graphe. Exemple de deux clusters avec une série anormale en rouge t_5.

arrête E_{ij} représente la similarité entre t_i et t_j (cf. Figure 3.1). L'idée consiste à trouver une partition de ce graphe de telle sorte que les arrêtes inter-profils E_{inter} aient un poids faible (*dissimilitude*) et que les arrêtes intra-profils E_{intra} aient un poids plus élevé (*similitude*). Une fois le graphe partitionné, chaque série se voit attribuer un score d'anormalité en fonction de son "éloignement" par rapport à toutes les autres séries de son profil. Nous proposons tout d'abord, de valuer chaque arrête E_{ij} par un poids en fonction d'un alignement temporel calculé par la DTW (*Dynamic Time Wrapping*) [Vintsyuk68] entre chaque paire de séries t_i et t_j. Nous obtenons donc une matrice de similarité S pour le graphe G, telle que $S_{ij} = \frac{1}{1+DTW(t_i,t_j)}$. Une fois la matrice de similarité S obtenue, la matrice d'affinité M est calculée via un noyau gaussien. La matrice Laplacienne normalisée L est ensuite déduite à partir de S et M. Le graphe G est partitionné par un clustering spectral [Ng01] en faisant un $k-$means sur les k premiers vecteurs propres de la matrice laplacienne L.

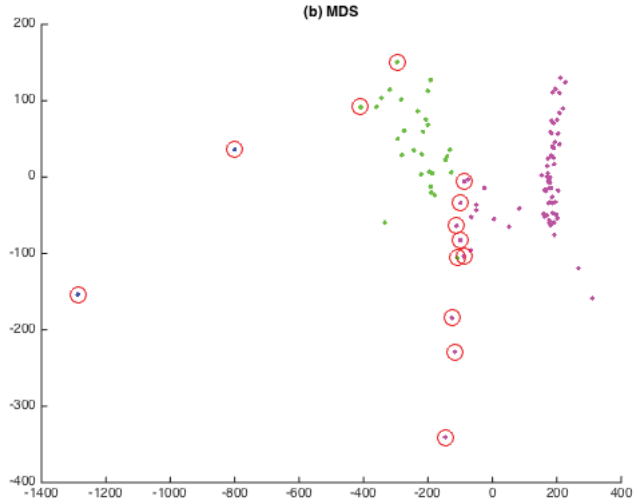
Nous obtenons ensuite les k clusters, $\{c_1, c_2, ..c_k\}$, qui correspondent aux k profils de normalité supposés être présents dans la base T des séries temporelles. A l'issue de l'étape du clustering, pour chaque série t_i, un score d'anormalité local $A(t_i, c^*)$ est calculé par rapport à chaque profil c^* : $A(t_i, c^*) = \frac{\sum_{t_j \in c^*} DTW(t_i,t_j)}{\sum_{t_l \in c^*} \sum_{t_z \in c^*} DTW(t_l,t_z)}$. En effet, le score $A(t_i, c^*)$ représente le ratio entre l'inertie de toutes les séries temporelles du profil c^* par rapport à la série t_i et l'inertie totale du profil c^*. Plus le score $A(t_i, c^*)$ est important, plus la série t_i est éloignée du profil c^*. Finalement, le score d'anormalité global de chaque série est calculé comme étant la moyenne de tous ses scores locaux $A(t_i, c^*)$. L'ensemble de ces développements définit notre approche de détection que nous formulons dans l'algorithme 2.

3.2.1 Validation expérimentale

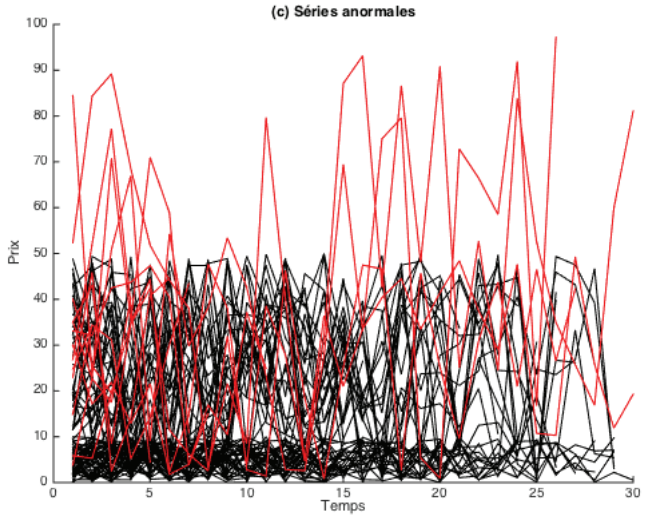
Pour valider notre approche, nous l'avons testée sur un échantillon de la base des séries temporelles des prix de pneumatiques de l'entreprise. Pour un pneu donné,



(a) Séries temporelles de prix d'un pneu durant un mois.



(b) Série anormales détectées. Visualisation via le Multi-Dimensional Scaling.



(c) Séries anormales visibles en couleur rouge.

FIGURE 3.2 – Validation expérimentale du DetectS.

Algorithme 2 : DetectS

Entrées : Une base T de n séries temporelles de tailles différentes $T = \{t_1, \ldots, t_n\}$, k : nombre de profils de normalité
Résultat : Les score d'anormalité de chaque série t_i de la base T

1 Construire la matrice de similarités, $S_{ij} = \frac{1}{1+DTW(t_i,t_j)}$.
2 Construire la matrice d'affinité M avec un noyau gaussien : $M_{ij} = \exp(\frac{-S_{ij}^2}{2\sigma^2})$; $M_{ii} = 0$.
3 Construire la matrice Laplacienne normalisée $L = D^{-1/2}MD^{-1/2}$ avec $D_{ij} = \sum_{j=1}^{N} S_{ij}$.
4 Construire la matrice $U \in \mathbb{R}^{n \times k}$ formée à partir des k plus grands vecteurs propres de L.
5 Normaliser U : $\hat{U}_{ij} = \frac{U_{ij}}{\sqrt{\sum_j U_{ij}^2}}$.
6 Construire k profils : $C = \{c_1, c_2, ..., c_k\}$ par un *k-means* appliqué sur les n lignes de $\hat{U}$.
7 **Pour** $c_i \leftarrow C$
8 | **Pour** $j = 1 : |c_i|$
9 | | $A(t_j, c_i) = \frac{\sum_{t_h \in c_i} DTW(t_j,t_h)}{\sum_{t_l \in c_i} \sum_{t_z \in c_i} DTW(t_l,t_z)}$
10 | **fin_pour**
11 **fin_pour**

l'ensemble des séries de prix est représenté par $T = \{t_1, t_2, t_3, ..., t_n\}$ où chaque $t_i = \{p_1, p_2, ...p_{di}\}$ représente la série de prix de vente de ce pneu par un site marchand durant une période donnée, $di \in \mathbb{N}$. Les séries t_i n'ont pas forcément toutes la même taille di. La base en question représente 100 séries temporelles de prix d'un pneu donné pendant un mois (Figure 5.8(a)).

Sur 3 profils de séries différents, l'approche a permis de détecter des séries anormales pour chacun de ces profils (Figure 5.8 (b)). Ces détections (entourées en rouge dans la figure) ont été jugées cohérentes et ont donc été validées par nos experts métier. En effet, les séries temporelles détectées présentent des tendances significativement différentes par rapport à la base étudiée (Figure 5.8 (c)), et correspondent à titre d'exemple à des erreurs de matching (e.g., matcher une série temporelle t_i à un pneu *Michelin* alors qu'elle décrit plutôt les prix d'un pneu *Pirelli*).

3.3 Détection globale par clustering pondéré

Dans cette section, nous présentons nos trois approches de détection globale qui sont basées sur le clustering pondéré. Contrairement, à l'approche précédente, nous

reformulons la détection comme un clustering pondéré où le clustering est intrinsèquement lié à la détection. La différence entre ces trois approches réside principalement dans le mécanisme de pondération ainsi qu'au niveau de la détection, globale (DOTS et ℓ_2-DAt) ou locale (L2GAD).

3.3.1 Approche à base d'entropie (*DOTS*)

Dans cette seconde approche, l'hypothèse de départ est de supposer que les séries anormales sont celles qui s'écartent significativement de la majorité des autres séries, et devraient donc maximiser l'inertie de leur cluster. A l'issue de cette approche, ces séries auront des poids faibles comparés à ceux des séries supposées normales.

En effet, nous considérons que le poids d'une série temporelle dans un cluster représente la probabilité de sa contribution dans la formation dudit cluster et dans le calcul de son prototype (*médoïde*). Partant du principe que les séries au sein du même cluster partagent une certaine similitude, elles devraient contribuer équitablement à la formation de ce cluster. En d'autres termes, elles devraient partager plus ou moins la même probabilité de contribution, que nous proposons de représenter dans cette approche par l'entropie des poids. En effet, l'entropie de Shannon permet de mesurer la quantité de désordre dans un système (e.g., contribution dans la formation d'un cluster). Par exemple, l'entropie d'une variable aléatoire T prenant ses valeurs dans l'ensemble $\{t_1, t_2, \ldots, t_n\}$ avec une fonction de masse $p(t_i) = w_i$ est définie par :

$$H(T) = -\sum_{t} p(t) \log p(t) = -\sum_{t} w \log w \tag{3.1}$$

Ainsi, les séries anormales devraient maximiser cette mesure à cause de leurs faibles poids par rapport à la majorité des séries dans le même cluster. L'idée consiste alors à associer à chaque série t_i un poids w_i dans une nouvelle fonction objective de clustering P. Le but étant d'assigner des poids faibles à des séries minimisant à la fois l'inertie de leurs clusters et l'entropie de l'ensemble de tous les poids de la base T [Benkabou18, Benkabou16a]. Nous cherchons donc à minimiser P :

$$\min_{G,M,W} P(G,M,W) = \overbrace{\sum_{l=1}^{k}\sum_{i=1}^{n} g_{il} w_i \mathbf{dtw}(t_i, m_l)}^{within-clusters} + \lambda \overbrace{\sum_{i=1}^{n} w_i \log(w_i)}^{negative-entropy} \tag{3.2}$$

$$\text{sous les contraintes suivantes :} \begin{cases} \sum_{l=1}^{k} g_{il} = 1; & 1 \leq i \leq n \\ g_{il} \in \{0,1\}; & 1 \leq i \leq n \\ \sum_{i=1}^{n} w_i = 1 \end{cases}$$

où

— k est le nombre de clusters.
— G est une matrice de partition de taille $n \times k$. g_{il} est une valeur binaire. g_{il}=1 indique que la série temporelle t_i est assignée au cluster l.
— $M = \{m_1, m_2, ..., m_k\}$ est un ensemble contenant les indices des k séries temporelles représentantes (médoïdes) des k clusters, respectivement.
— $W = \{w_1, w_2, ..., w_n\}$ est l'ensemble des poids des séries.
— Le terme de régularisation λ est utilisé pour contrôler la dispersion des poids.
— $\mathbf{dtw}(t_i, t_j)$ est une fonction qui permet de calculer la distance entre deux séries temporelles t_i et t_j par la DTW décrite dans la section 2.3.

Le premier terme dans (3.2) représente la somme de la distorsion des intra-clusters, alors que le deuxième terme désigne l'entropie négative des poids W. La minimisation de P dans (3.2) sous contraintes forme une classe de problèmes d'optimisation non linéaire avec contraintes dont les solutions sont inconnues. En effet, il est difficile d'optimiser trois variables simultanément. Ainsi, nous adoptons une optimisation alternative pour résoudre ce problème, ce qui fonctionne bien pour un certain nombre de problèmes d'optimisation pratiques [Gu11]. Pour ce faire, nous devons minimiser la fonction objective P avec des variables inconnues G, M et W en résolvant d'une manière itérative les trois sous-problèmes de minimisation suivants :

— **Problème** P_1 : Étant donné M et W, calculer G (mise à jour de la matrice de partitions).
— **Problème** P_2 : Étant donné G et W, calculer M (mise à jour des des médoïdes).
— **Problème** P_3 : Étant donné G et M, calculer W (mise à jour des poids).

Le Problème P_1 est résolu par l'équation 3.3 et cela consiste à affecter chaque série temporelle t_i à un seul cluster en comparant la distance entre t_i et les différents médoïdes m_l :

$$g_{il} = \begin{cases} 1 \text{ if } \mathbf{dtw}(t_i, m_l) \leq \mathbf{dtw}(t_i, m_v); 1 \leq v \leq k \\ \\ 0 \text{ } \textit{Otherwise} \end{cases} \tag{3.3}$$

Le Problème P_2 est résolu par le calcul des k médoïdes des clusters via l'équation 3.4.

Soit $I_l = \sum_j w_j \mathbf{dtw}(t_i, t_j)$, l'inertie pondérée de toutes les séries temporelles t_j du cluster l par rapport à la série t_i. Calculer le médoïdes m_l pour le cluster l revient à trouver la série t_i ayant l'inertie pondérée minimale :

$$m_l = \underset{t_j}{\operatorname{argmin}} \sum_{i|i\neq j,\ g_{il}=g_{jl}=1} w_j \mathbf{dtw}(t_i, t_j). \tag{3.4}$$

Le calcul du médoïde dans (3.4) est influencé par les poids des séries temporelles du même cluster. En faisant ainsi, les séries anormales n'ont pas d'influence sur le calcul

du médoïde à cause de leur poids très faibles.

Enfin, la solution du **Problème** P_3 est donnée par le théorème suivant :

Théorème 1. *En fixant G et M, W est mis à jour comme suit :*

$$w_i = \frac{\mathrm{e}^{\frac{-D_i}{\lambda}}}{\sum_{t=1}^{n} \mathrm{e}^{\frac{-D_t}{\lambda}}} \tag{3.5}$$

Où

$$D_i = \sum_{l=1}^{k} g_{il} \mathbf{dtw}(t_i, m_l). \tag{3.6}$$

Démonstration. Notre équation (3.2) peut être réécrite de la façon suivante :

$$P(G, M, W) = \sum_{i=1}^{n} w_i \overbrace{\sum_{l=1}^{k} g_{il} \mathbf{dtw}(t_i, m_l)}^{D_i} + \lambda \sum_{i=1}^{n} w_i \log(w_i).$$

$$= \sum_{i=1}^{n} w_i D_i + \lambda \sum_{i=1}^{n} w_i \log(w_i). \tag{3.7}$$

où D_i est une constante par rapport à G et M. On minimise la fonction par les multiplicateurs de Lagrange. Soit μ le multiplicateur et $\Phi(W, \mu)$ le Lagrangien :

$$\Phi(W, \mu) = \sum_{i=1}^{n} w_i D_i + \lambda \sum_{i=1}^{n} w_i \log(w_i) + \mu \left(\sum_{i=1}^{n} w_i - 1 \right). \tag{3.8}$$

Pour minimiser $\Phi(W, \mu)$, le gradient de Φ par rapport à W et μ doit être égale à zéro. Ainsi,

$$\frac{\partial \Phi(W, \mu)}{\partial w} = D_i + \lambda (\log(w_i) + 1) + \mu = 0 \tag{3.9}$$

$$\frac{\partial \Phi(W, \mu)}{\partial \mu} = \sum_{i=1}^{n} w_i - 1 = 0. \tag{3.10}$$

De l'équation (3.8) on obtient

$$w_i = \mathrm{e}^{\frac{-D_i - \mu - \lambda}{\lambda}} = \mathrm{e}^{\frac{-D_i - \lambda}{\lambda}} \mathrm{e}^{\frac{-\mu}{\lambda}} \tag{3.11}$$

En substituant l'équation (3.11) dans l'équation (3.9), on obtient une nouvelle formulation des contraintes des poids en fonction de μ :

$$\sum_{i=1}^{n} \mathrm{e}^{\frac{-D_i-\lambda}{\lambda}} \mathrm{e}^{\frac{-\mu}{\lambda}} = 1 \tag{3.12}$$

De l'équation (3.12), on obtient :

$$\mathrm{e}^{\frac{-\mu}{\lambda}} = \frac{1}{\sum_{i=1}^{n} \mathrm{e}^{\frac{-D_i-\lambda}{\lambda}}} \tag{3.13}$$

Finalement, en substituant l'équation (3.13) dans l'équation (3.11), on obtient la formulation finale des poids en fonction des D_i :

$$w_i = \frac{\mathrm{e}^{\frac{-D_i}{\lambda}}}{\sum_{t=1}^{n} \mathrm{e}^{\frac{-D_i}{\lambda}}} \tag{3.14}$$

□

Subséquemment, nous pouvons résumer tous les développements mathématiques ci-dessus dans l'Algorithme 3. Nous l'appelons **DOTS** (pour *Detection of Outlier Time Series*).

Algorithme 3 : DOTS

Entrées : une base de séries temporelles $T = \{t_1, t_2, ..., t_n\}$, les paramètres λ et k.

Résultat : les séries temporelles t_i de la base T triées selon leur score d'anormalité

1 Choisir aléatoirement k séries temporelles de T comme médoïdes $\{m_1, m_2, ..., m_k\}$
2 Générer aléatoirement les poids $[w_1, w_2, ..., w_n]$ tel que $\sum_{i=1}^{n} w_i = 1$
3 **Répéter**
4 | Calculer G, la matrice des partitions par l'équation (3.3)
5 | Mettre à jour la liste des médoïdes M par l'équation (3.4)
6 | Mettre à jour les poids W par l'équation (3.5)
7 **Jusqu'à** *Convergence par rapport à W*;
8 Tries les séries t_i dans un ordre ascendant selon leur score d'anormalité.

Analyse algorithmique

Dans l'algorithme 3, on commence par choisir aléatoirement k séries temporelles dans la base totale T et on les considère comme représentants (médoïdes) $M^0 =$

$\{m_1, m_2, ..., m_k\}$. Aussi, on génère aléatoirement un poids pour chaque série t_i dans la base $W^0 = [w_1^0, w_2^0, ..., w_n^0]$ de telle sorte que $\sum_{i=1}^{n} w_i = 1$.

On initialise le nombre d'itération h à zéro, et on commence à résoudre les trois sous-problèmes, à savoir P_1, P_2 et P_3 l'un après l'autre de manière itérative jusqu'à la convergence. Premièrement, nous fixons les deux variables M et W pour résoudre P_1 comme suit : Soit $\hat{M} = M^h$ et $\hat{W} = W^h$, on minimise $P(G, \hat{M}, \hat{W})$ par l'équation (3.3) pour obtenir G^{h+1} (*à chaque itération la séries t_i est affectée à son cluster le plus proche*).

Ensuite, on fixe G et W pour résoudre P_2 comme suit : Soit $\hat{G} = G^h$ et $\hat{W} = W^h$, on minimise $P(\hat{G}, M, \hat{W})$ par l'équation (3.4) pour obtenir M^{h+1} (*le médoïde de chaque cluster m_k est remis à jour en minimisant la somme des distances par rapport aux autres séries du même cluster*).

Finalement, on fixe G et M pour résoudre P_3 et calculer les poids des séries : Soit $\hat{G} = G^h$ et $\hat{M} = M^h$, on minimise $P(\hat{G}, \hat{M}, W)$ par l'équation (3.5) pour obtenir W^{h+1}. Si $W^{h+1} = W^h$ alors l'algorithme converge, sinon on incrémente ($h = h + 1$) et on résout à nouveau le problèmes P_1, P_2 et P_3 jusqu'à ce que les poids ne soient plus modifiés.

Convergence

L'algorithme DOTS converge vers un optimum local dans un nombre fini d'itérations.
On note qu'il n'y a qu'un nombre fini de partitions possibles G, et on peut montrer que chaque partition G n'apparaît pas plus d'une fois dans le processus de clustering. Supposons que $G^{h_1} = G^{h_2}$ où $h_1 \neq h_2$. Etant donné G^h, on peut calculer notre minimiseur M^h par l'équation (3.4). Pour G^{h_1} et G^{h_2}, on obtient les minimiseurs M^{h_1} et M^{h_2}, respectivement. il est clair que $M^{h_1} = M^{h_2}$ puisque $G^{h_1} = G^{h_2}$. En utilisant G^{h_1} et M^{h_1}, G^{h_2} and M^{h_2}, on peut calculer le minimiseur W^{h_1} et W^{h_2}, respectivement, par l'équation (3.5). Encore une fois, il est clair que $W^{h_1} = W^{h_2}$. Ainsi, on obtient :

$$P(G^{h_1}, M^{h_1}, W^{h_1}) = P(G^{h_2}, M^{h_2}, W^{h_2}).$$

On note que la séquence $P(G, M, W)$ générée par le DOTS est strictement décroissante. Ceci dit, l'algorithme converge en un nombre fini d'itérations lorsque les poids W se stabilisent.

Complexité

La complexité temporelle du DOTS est de l'ordre de $O((n(n-1)/2)L^2) + O(n \times max(hk, \log n))$

Calculer la distance entre deux séries temporelles t_i et t_j en utilisant la DTW, nécessite en général $O(max(|t_i|, |t_j|)^2)$. Soit L la longueur de la série temporelle la plus longue dans la base T. Ainsi, pour la calculer la distance entre chaque paire de séries, on a besoin de $O((n(n-1)/2)L^2)$ opérations. Pour mettre à jour les clusters, on a besoin de nk opérations. La complexité de cette étape est donc $O(nk)$. De même pour la mise à jour des k médoïdes, on a besoin de $O(nk)$ opérations. Étant données G et M,

on a besoin de uniquement $O(2n)$ opérations pour calculer l'ensemble des poids W. La dernière étape consiste à trier les séries en fonction de leur poids et cela nécessite $n \log n$ opérations. Par conséquent, la complexité temporelle du DOTS est de l'ordre de $O((n(n-1)/2)L^2) + O(n \times max(hk, \log n))$ où h est le nombre total d'itérations.

Régularisation

Le paramètre de la régularisation λ est utilisé pour contrôler la dispersion des poids W

Si $\lambda > 0$, d'après l'équation (3.5), w_i est inversement proportionnel à D_i dans l'équation (3.6). Plus le D_i est grand, plus le w_i est petit, et plus la série temporelle t_i est considérée comme anormale. Cependant, si $\lambda < 0$, le w_i est proportionnel à D_i. Plus le D_i est grand, plus le w_i est aussi important, et plus les séries anormales vont avoir un poids important ce qui va à l'encontre de notre idée de pondération. Par conséquent, λ ne peut pas être inférieur à zéro.

3.3.1.1 Validation sur le jeu de données "*Gun-Problem*" data set

Pour évaluer l'efficacité de notre méthode, nous avons mené une expérience sur le jeu de données *Gun-Problem* [Ratanamahatana04] issu du domaine de la vidéo surveillance. Ce jeu de données est fréquemment utilisé par la communauté pour valider les différentes techniques proposées pour l'analyse des séries temporelles en général [Chen15]. Il comporte deux classes, chacune est représentée par 100 séries. Les séries ont été obtenues en utilisant un sujet féminin et un autre masculin en une seule session. Les deux classes sont :

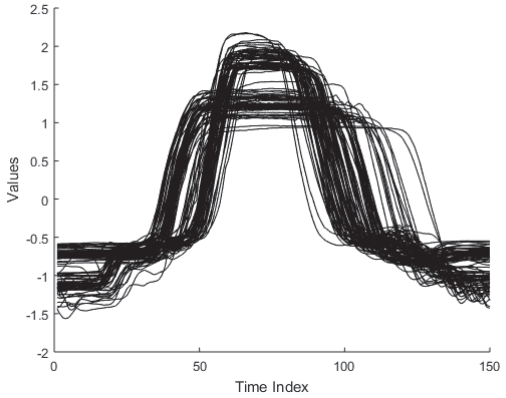
— **la classe Gun-Draw :**
Les sujets ont leur mains sur les côtés et le pistolet dans l'étui. Ils tirent à hauteur de la hanche après avoir retiré le pistolet de l'étui et l'avoir pointé sur la cible pendant une seconde. Une fois le tir effectué, ils remettent le pistolet dans l'étui et les mains sur les côtés. La Figure 3.3(a) illustre quelques extraits de la vidéo.

— **la classe Point :** Dans cette classe, les sujets ont aussi leurs mains à leurs côtés sauf qu'ils n'ont pas de pistolet. Ils pointent la cible pendant une seconde, uniquement par leur index pour simuler un tir. Puis ils remettent leurs mains à les côtés. La Figure 3.3(c) illustre quelques extraits de la vidéo.

Pour les deux classes, la trajectoire du centroïde de la main droite (abscisses et ordonnées) a été enregistrée. Cependant, nous nous intéressons qu'à l'axe des abscisses pour des raisons de simplicité. Les Figures 3.3(b) et 3.3(d) montrent les séries temporelles de cet axe pour les deux classes *Gun-Draw* et *Point*, respectivement.

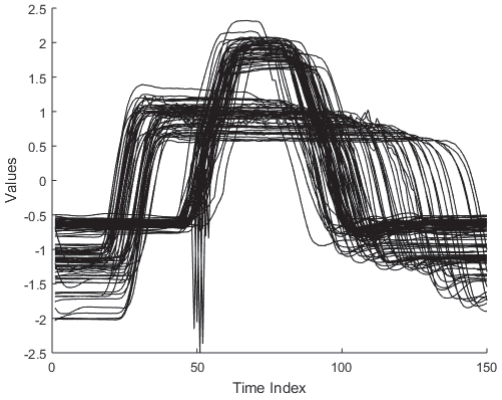
Pour valider le bon fonctionnement du DOTS, nous avons construit un nouveau jeu de données en prenant toutes les séries de la première classe, à savoir *Gun-Draw*, et en les considérant comme normales. Ensuite, nous avons rajouté à la base 5 séries

(a) La classe *Gun-Draw*



(b) Séries temporelles de la classe *Gun-Draw*

(c) La classe *Point*

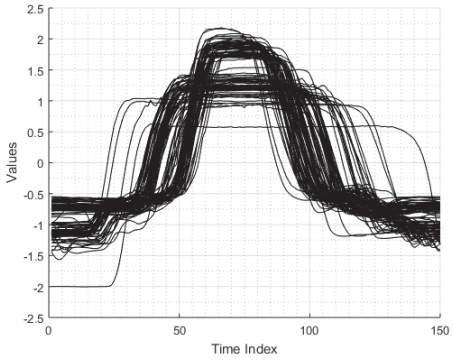


(d) Séries temporelles de la classe *Point*

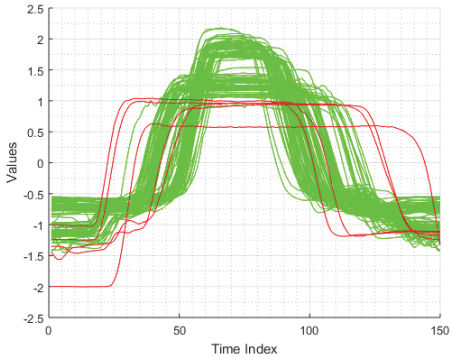
FIGURE 3.3 – Le jeu de données *Gun* [Ratanamahatana04]

temporelles de la deuxième classe, *Point*, tout en les considérant comme anormales. Le but étant de savoir si l'algorithme DOTS est capable de détecter les 5 séries de la classe *Point* comme anormales par rapport à la majorité des séries qui sont issues de la première classe *Gun-Draw*. Après application, nous avons obtenu les résultats présentés dans la Figure 3.4

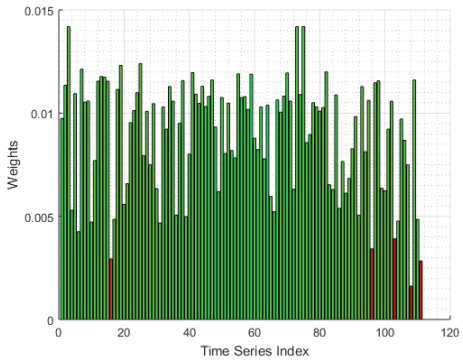
En détail, la Figure 3.4(a) montre les séries temporelles du nouveau jeu de données (100 de la classe *Gun-Draw* et 5 de la classe *Point*). La Figure 3.4(b) distingue les séries normales de celles détectées comme étant anormales avec les couleurs verte et rouge, respectivement. En effet, les séries en rouge correspondent exactement aux 5 séries de la classe *Point*, ce qui montre que notre approche a correctement détecté les séries anormales parmi la majorité des séries de la classe *Gun-Draw*. Nous présentons la distribution des poids dans la Figure 3.4(c). A noter qu'une série est considérée comme



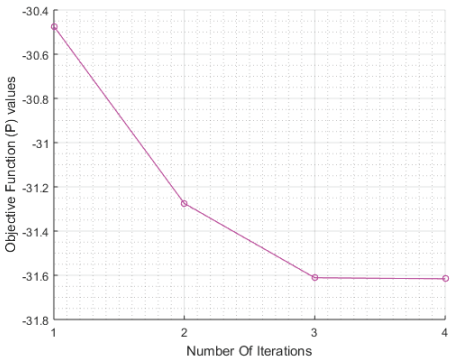
(a) Séries temporelles de la base T



(b) Series anormales (*en rouge*), detectées par DOTS



(c) Distribution des poids des séries



(d) Convergence

FIGURE 3.4 – Validation du DOTS sur le jeu de données *Gun*.

anormale si son poids est très faible. Les barres en rouge dans l'histogramme représentent les poids des séries anormales et ceux en vert représentent les poids des séries normales. On peut remarquer aussi, que les poids les plus élevés sont attribués équitablement aux médoïdes, censés définir naturellement les différents profils de normalité dans le jeu de données. Finalement, la convergence de l'algorithme est reportée dans la Figure 3.4(d), où on peut voir que l'algorithme DOTS converge après un nombre fini d'itérations. L'axe horizontal représente le nombre d'itérations, et l'axe vertical représente les différentes valeurs de la fonction objective durant le processus de l'optimisation. La courbe est strictement décroissante, ce qui assure au moins une convergence vers un optimum local.

3.3.2 Approche par pénalisation Ridge (ℓ_2-DAT)

Tout comme l'approche précédente, nous considérons que le poids d'une série temporelle dans un cluster représente la probabilité de sa contribution dans la formation dudit cluster, et dans le calcul de son médoïde. Cependant, nous nous basons dans cette approche sur un mécanisme de pondération différent de celui utilisé par l'algorithme DOTS. En effet, nous utilisons la pénalisation *Ridge* pour la pondération et le calcul des poids des séries temporelles.

Cette pénalisation par la norme ℓ_2 est souvent utilisée dans des problèmes de régression [Hastie01], comme dans le cas de la régression linéaire multiple où l'utilisation de cette pénalité permet de réduire la variance des coefficients des variables, en affectant des poids importants aux variables pertinentes et poids assez faibles aux variables marginales. Dans le cas de la régression linéaire multiple, nous avons $X = \{x_1, x_2, ..., x_n\} \in \mathbb{R}^{n \times p}$, qui représente la matrice des n observations dans l'espace R^P et $Y = \{y_1, y_2, ..., y_n\} \in \mathbb{R}^n$ qui représente le vecteur cible. L'objectif de la régression via la pénalisation Ridge est de trouver le vecteur de coefficients $W \in \mathbb{R}^p$ minimisant l'erreur quadratique $\|Y - XW\|_2^2$ tout en ayant une complexité moins importante par rapport à la complexité de l'estimateur des moindres carrés classique. Trouver W revient à résoudre le problème d'optimisation suivant :

$$W_{ridge} = \underset{W}{\operatorname{argmin}} \overbrace{\|Y - XW\|_2^2}^{Loss} + \lambda \overbrace{\|W\|_2}^{Penalty} \tag{3.15}$$

Nous utilisons donc cette technique pour pénaliser le vecteur W des poids des séries temporelles dans le but d'écarter les séries anormales en leur assignant un poids très faible. L'idée de cette approche est d'utiliser la pénalisation par la norme ℓ_2 dans la fonction objectif Q pour réduire la complexité des poids et par conséquent associer des poids très faibles à des séries temporelles maximisant l'inertie de leur clusters [Benkabou17b]. Nous cherchons donc à minimiser Q :

$$\min_{G,M,W} Q(G, M, W) = \overbrace{\sum_{i=1}^{n} w_i \sum_{l=1}^{k} g_{il} dtw(t_i, m_l)}^{within-clusters} + \lambda \overbrace{\|W\|_2}^{penalty} \tag{3.16}$$

sous les contraintes suivantes : $\begin{cases} \sum_{l=1}^{k} g_{il} = 1; 1 \leq i \leq n \\ g_{il} \in \{0, 1\}; 1 \leq i \leq n; 1 \leq l \leq k \\ \sum_{i=1}^{n} w_i = 1 \end{cases}$

Où

— k est le nombre de clusters.
— G est une matrice de partition de taille $n \times k$. g_{il} est une valeur binaire. g_{il}=1 indique que la série temporelle t_i est assignée au cluster l.

— $M = \{m_1, m_2, ..., m_k\}$ est un ensemble contenant les indices des k séries temporelles représentantes (médoïdes) des k clusters, respectivement.
— $W = \{w_1, w_2, ..., w_n\}$ est l'ensemble des poids des séries.
— Le terme de régularisation λ est utilisé pour contrôler la complexité de la norme L_2 des poids W.

Le premier terme dans (3.16) représente la somme des intra-clusters tandis que le deuxième représente la partie concernant la régularisation ℓ_2 des poids W.
Tout comme la fonction objectif P dans (3.2), la minimisation de Q dans (3.16) forme une classe de problèmes d'optimisation non-linéaire sous contraintes dont les solutions sont inconnues. Ainsi, nous adoptons la même optimisation alternative pour résoudre Q, ce qui revient à minimiser la fonction objectif en résolvant d'une manière itérative les trois sous-problèmes de minimisation suivants :

— **Problème** Q_1 : Étant donné M et W, calculer G (mise à jour de la matrice de partitions).
— **Problème** Q_2 : Étant donné G et W, calculer M (mise à jour des des médoïdes).
— **Problème** Q_3 : Étant donné G et M, calculer W (mise à jour des poids).

La mise à jour de la matrice de partition G dans le **Problème** Q_1 se fait de la manière suivante :

$$g_{il} = \begin{cases} 1 \text{ if } dtw(t_i, m_l) \leq dtw(t_i, m_v); 1 \leq v \leq k \\ 0 \text{ } Otherwise \end{cases} \tag{3.17}$$

Le calcul des médoïdes dans le **Problème** Q_2 est effectué par l'équation ci-dessous :

$$m_l = \min_{t_j} \sum_{i|i \neq j,\ g_{il}=g_{jl}=1} w_j dtw(t_i, t_j). \tag{3.18}$$

La solution du **Problème** Q_3, consistant à mettre à jour les poids W , est donnée par le théorème suivant :

Théorème 2. *En fixant G et M, W est mis à jour comme suit :*

$$w_i = \frac{2\lambda - n f(t_i) + \sum_{s=1}^{n} f(t_s)}{2\lambda n} \tag{3.19}$$

Où

$$f(t_i) = \sum_{k=1}^{l} g_{il} dtw(t_i, m_l)$$

Démonstration. Nous minimisons la fonction par les multiplicateurs de Lagrange. Soit μ le multiplicateur et $\Theta(W, \mu)$ le Lagrangien :

$$\Theta(W,\mu) = \sum_{i=1}^{n} w_i \sum_{l=1}^{k} g_{il} dtw(t_i, m_l) + \lambda \|W\|_2^2 - \mu\left(\sum_{i=1}^{n} w_i - 1\right) \tag{3.20}$$

Pour minimiser $\Theta(W,\mu)$, le gradient de Θ par rapport à W et μ doit être égale à zéro. Ainsi, nous avons :

$$\frac{\partial\Theta(W,\mu)}{\partial w_i} = 2\lambda w_i - \mu + \sum_{k=1}^{l} g_{il} dtw(t_i, m_l) = 0 \tag{3.21}$$

et

$$\frac{\partial\Theta(W,\mu)}{\partial \mu} = \sum_{i=1}^{n} w_i - 1 = 0 \tag{3.22}$$

De l'équation(3.21), nous obtenons :

$$w_i = \frac{\mu - \sum_{k=1}^{l} g_{il} dtw(t_i, m_l)}{2\lambda} \tag{3.23}$$

En substituant l'équation (3.23) dans l'équation (3.22), nous avons :

$$\sum_{i=1}^{n}\left(\frac{\mu - \sum_{k=1}^{l} g_{il} dtw(t_i, m_l)}{2\lambda}\right) = 1 \tag{3.24}$$

De l'équation (3.24) nous obtenons :

$$\frac{\mu}{2\lambda} = \frac{1 + \sum_{i=1}^{n}\left(\frac{\sum_{k=1}^{l} g_{il} dtw(t_i, m_l)}{2\lambda}\right)}{n} \tag{3.25}$$

En substituant l'équation (3.25) dans l'équation (3.23) :

$$w_i = \frac{1 + \sum_{s=1}^{n}\left(\frac{\sum_{k=1}^{l} g_{sl} dtw(t_s, m_l)}{2\lambda}\right)}{n} - \frac{\sum_{k=1}^{l} g_{il} dtw(t_i, m_l)}{2\lambda} \tag{3.26}$$

De l'équation (3.26) nous obtenons :

$$w_i = \frac{2\lambda - n\sum_{k=1}^{l} g_{il} dtw(t_i, m_l) + \sum_{s=1}^{n}\sum_{k=1}^{l} g_{sl} dtw(t_s, m_l)}{2\lambda n} \tag{3.27}$$

Finalement, nous obtenons la formulation finale des poids :

$$w_i = \frac{2\lambda - n f(t_i) + \sum_{s=1}^{n} f(t_s)}{2\lambda n} \tag{3.28}$$

ou

$$f(t_i) = \sum_{k=1}^{l} g_{il} dtw(t_i, m_l)$$

□

Pour finir, nous résumons tous les développements mathématiques décrits ci-dessus dans l'Algorithme 4. Nous l'appelons ℓ_2**-DAT** (pour ℓ_2 *Detection of Anomalous Time Series*).

Algorithme 4 : ℓ_2**-DAT**

Entrées : une base de séries temporelles $T = \{t_1, t_2, ..., t_n\}$, les paramètres λ et k.

Résultat : les séries temporelles t_i de la base T triées selon leur score d'anormalité

1 Choisir aléatoirement k séries temporelles de T comme médoïdes $\{m_1, m_2, ..., m_k\}$

2 Générer aléatoirement les poids $[w_1, w_2, ..., w_n]$ tel que $\sum_{i=1}^{n} w_i = 1$

3 **Répéter**

4 Mettre à jour G par $g_{il} = \begin{cases} 1, & \text{si } dtw(t_i, m_l) \leq dtw(t_i, m_v) \\ 0, & \text{sinon.} \end{cases}$

5 Mettre à jour la liste des médoïdes M par $m_l = \min_{t_j} \sum_{i|i \neq j,\ g_{il}=g_{jl}=1} w_j dtw(t_i, t_j)$.

6 Calculer les poids de chaque série temporelle $w_i = \frac{2\lambda - n f(t_i) + \sum_{s=1}^{n} f(t_s)}{2\lambda n}$

7 **Jusqu'à** *Convergence par rapport aux poids w*;

8 Tries les séries t_i dans un ordre ascendant selon leur score d'anormalité w_i.

Analyse algorithmique

La convergence et la complexité de l'algorithme ℓ_2-DAT sont identiques à celles du DOTS. L'algorithme converge donc vers un optimum local dans un nombre fini d'itérations, et sa complexité temporelle est de l'ordre de :

$$O((n(n-1)/2)L^2) + O(n \times max(hk, \log n))$$

.

3.3.2.1 Distribution des paramètres

Nous discutons dans cette partie de la distribution des poids W et du comportement par rapport aux différentes valeurs que peut prendre le paramètre de régularisation λ. En outre, nous donnons une analyse théorique sur la façon de choisir la valeur appropriée de ce paramètre, de telle sorte qu'un *seuil de détection* sera appris automatiquement à partir de la distribution des poids.

Analyse de la distribution des poids

Tout d'abord, pour mieux comprendre le comportement de la distribution des poids par rapport aux différentes valeurs de λ, nous avons créé un jeu de données synthétique de 100 séries temporelle ayant la même longueur 30 et regroupés en trois groupes. Chaque cluster contient 5 séries anormales. Pour avoir une idée sur le comportement de la distribution des poids par rapport au λ, nous avons lancé ℓ_2-DAT sur le jeu de données synthétique avec des valeurs différentes allant de 0 jusqu'à 100 avec un pas incrémenté de 0.01.

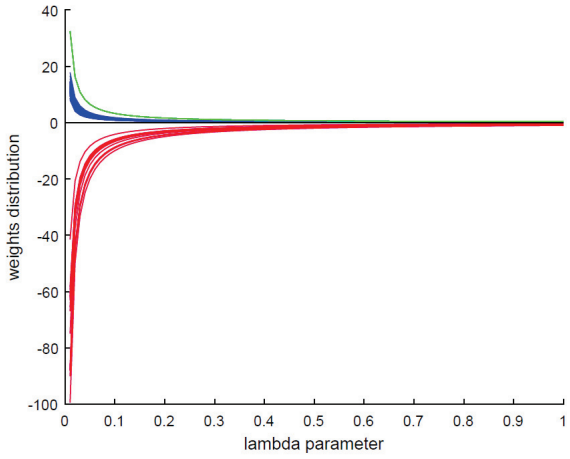


FIGURE 3.5 – Comportement des poids par rapport aux différentes valeurs de λ

La Figure 3.5 montre la variation des poids de chaque série temporelle par rapport aux différentes valeurs λ. Quand cette valeur est plus proche de zéro, on peut voir que certains poids sont négatifs (courbes rouges). En effet, ces poids correspondent à ceux des séries temporelles jugées anormales par l'algorithme. Les courbes vertes représentent la variation dans les poids des séries temporelles représentatives (médoïdes). En effet, il y a trois courbes vertes similaires qui correspondent aux poids des trois séries représentatives dans le jeu de données. Enfin, les courbes bleues représentent la variation des poids des séries normales.

Par conséquent, la Figure 3.5 montre que le paramètre λ a un impact important sur la variation des poids de chaque série temporelle. Ainsi, on peut remarquer que plus la valeur de λ augmente, plus tous les poids semblent converger vers une distribution uniforme, $\forall t_i \in T$, nous devrons avoir $w_i \approx \frac{1}{n}$.

Analyse de la régularisation λ

Avec une hypothèse faible sur la distribution des poids, nous estimons le poids de chaque type de série (médoïde, normale, anormale). Aussi, nous montrons comment

choisir la valeur de λ la plus adéquate pour avoir une distribution appropriée des poids et un seuil de détection automatique.

Nous avons, $T = \{t_1, t_2, ..., t_n\}$, l'ensemble de nos séries temporelles. Soit $T_{normal} = \{t_1, t_2, ..., t_Z\}$ le sous-ensemble de T qui représente l'ensemble des séries normales et $T_{abnormal} = \{t_1, t_2, ..., t_B\}$ le sous-ensemble de T qui représente l'ensemble des séries anormales de telle sorte que $N = Z + B$. Puisque les séries anormales sont supposées être très peu nombreuses dans l'ensemble T, nous avons $B < Z$.

Soit $\mathbb{E}_{t_s \in T}[f] = \frac{1}{n} \sum_{s=1}^{n} f(t_s)$ la valeur que l'on attend en moyenne de la fonction f que nous avons définie dans l'équation (3.19).

Nous supposons que, $\forall t_i \in T_{normal}, f(t_i) < \mathbb{E}_{t_s \in T}[f]$ et, $\forall t_i \in T_{abnormal}, f(t_i) > \mathbb{E}_{t_s \in T}[f]$.

— **Les poids des séries normales**
$\forall t_i \in T_{normal}$, nous avons : $\sum_{s=1}^{n} f(t_s) > n f(t_i)$, alors selon l'équation (3.19), nous devrions avoir :
$$w_i = \frac{1}{n} + \frac{C_{t_i}}{2\lambda n}, \text{ où } C_{t_i} = -n f(t_i) + \sum_{s=1}^{n} f(t_s) > 0$$

— **Les poids des séries anormales**
$\forall t_i \in T_{abnormal}$, nous avons : $\sum_{s=1}^{n} f(t_s) < n f(t_i)$, alors selon l'équation (3.19) nous devrions avoir :
$$w_i = \frac{1}{n} + \frac{H_{t_i}}{2\lambda n}, \text{ où } H_{t_i} = -n f(t_i) + \sum_{s=1}^{n} f(t_s) < 0$$

— **Les poids des séries représentatives**
$\forall t_i \in T$ et si t_i est une série médoïde, nous avons $f(t_i) = 0$ et alors selon l'équation (3.19), nous obtenons :
$$w_i = \frac{1}{n} + \frac{\sum_{s=1}^{n} f(t_s)}{2\lambda n}$$

Comme nous pouvons le voir ci-dessus, la distribution des poids dépend fortement du paramètre de régularisation λ. En effet, si $\lambda = 0$ ou $\lambda \to +\infty$, la solution finale est inutile car les poids calculés sont soit uniformément répartis ($\approx \frac{1}{n}$) ou fortement rétrécis vers zéro, respectivement. Dans les deux cas, aucune information utile ne peut être mise à l'évidence pour dire ce qui est normal et ce qui ne l'est pas. Donc ce paramètre devrait être soigneusement choisi pour avoir une distribution des poids appropriée avec laquelle un seuil de détection pourrait être automatiquement appris.

Dans ce qui suit, nous discutons de comment choisir λ pour avoir la propriété suivante dans la distribution des poids : *une série temporelle est anormale si son poids est négatif*. Pour avoir une telle distribution, le paramètre λ doit vérifier les contraintes suivantes :

Soit $\mathbb{E}[C]_{t_i \in T_{normal}} = \frac{1}{n} \sum_{i=1}^{Z} C_{t_i}$, et $\mathbb{E}[H]_{t_i \in T_{abnormal}} = \dfrac{1}{n} \displaystyle\sum_{i=1}^{B} H_{t_i}$ les valeurs que l'on attend en moyenne des variables H et C respectivement.

$$\begin{cases} \frac{\sum_{i=1}^{n} f(t_i)}{2\lambda n} > 0 \Rightarrow \lambda > 0 \\ \frac{1}{n} - \frac{\mathbb{E}[H]}{2\lambda n} < 0 \Rightarrow \lambda < \frac{\mathbb{E}[H]}{2} \\ \frac{1}{n} + \frac{\mathbb{E}[C]}{2\lambda n} > 0 \Rightarrow \lambda > \frac{-\mathbb{E}[C]}{2} \end{cases}$$

Par conséquent, pour vérifier les contraintes ci-dessus, λ doit être supérieur à 0 et plus petit que $\mathbb{E}[H]$.

3.3.3 Approche par pondération locale (*L2GAD*)

Contrairement aux deux approches précédentes, nous utilisons dans cette approche une pondération locale pour chaque série temporelle par rapport à son cluster. En effet, chaque série se voit attribuer autant de poids que de clusters. Plus le poids d'une série est faible par rapport à un cluster, plus sa probabilité d'être localement anormale dans ce cluster est importante. Un score global d'anormalité pour chaque série est obtenu en agrégeant les différents scores de la série par rapport aux différents clusters.

L'idée de notre approche que nous appelons L2GAD (*Local to global anomaly detection*), est d'intégrer une pondération locale des séries temporelles dans la fonction objective Φ afin de pouvoir traiter les cas où les clusters n'ont pas la même densité [Benkabou17b]. Nous cherchons donc à minimiser Φ :

$$\min_{A,M,W} \Phi(G, M, W) = \sum_{l=1}^{k} \sum_{i=1}^{n} g_{il} w_{il}^{\alpha} \mathbf{dtw}(t_i, m_l) \tag{3.29}$$

sous les contraintes suivantes : $\begin{cases} \sum_{l=1}^{k} g_{il} = 1; 1 \leq i \leq n \\ g_{il} \in \{0,1\}; 1 \leq i \leq n; 1 \leq l \leq k \\ \sum_{l=1}^{k} \sum_{i=1}^{n} w_{il} = 1; 0 \leq w_i \leq 1 \end{cases}$

où

— k est le nombre de clusters.
— G est une matrice de partition de taille $n \times k$. g_{il} est une valeur binaire. g_{il}=1 indique que la série temporelle t_i est assignée au cluster l.
— $M = \{m_1, m_2, ..., m_k\}$ est un ensemble contenant les indices des k séries temporelles représentantes (médoïdes) des k clusters, respectivement.
— W est une matrice de pondération de taille $n \times k$ où chaque série temporelle est représentée par autant de poids que de clusters. Par exemple, les poids de la série t_i sont représentés par le vecteur ligne de la matrice W, qu'on notera $W(t_i) = [w_{i1}, w_{i2}, \ldots, w_{ik}]$.
— α est un paramètre utilisé pour la pondération, et qui permet de contrôler la distribution des poids. Si $\alpha = 0$, le processus de pondération est ignoré par la

fonction objective Φ dans (3.29), et la détection ne peut avoir lieu. Si $\alpha = 1$, le w disparaîtrait à cause des dérivées nécessaires pour la résolution du problème à travers le Lagrangien (3.34).

Tout comme les deux précédentes fonctions objectifs P et Q dans (3.2) et (3.16) respectivement, cette nouvelle fonction Φ forme aussi une classe de problème d'optimisation non linéaire avec des contraintes, dont les solutions sont inconnues et comme il est toujours difficile d'optimiser plusieurs variables simultanément, nous adoptons une optimisation alternative pour résoudre ce problème. Ainsi, la minimisation de Φ revient à optimiser les trois sous-problèmes de minimisation suivants :

— **Problème** Φ_1 : Minimisation de Φ en fixant M et W pour calculer la matrice de partition G. L'optimisation mène à :

$$g_{il} = \begin{cases} 1 \text{ if } \mathbf{dtw}(t_i, m_l) \leq \mathbf{dtw}(t_i, m_v); 1 \leq v \leq k \\ 0 \text{ } \textit{Otherwise} \end{cases} \tag{3.30}$$

— **Problème** Φ_2 : Minimisation de Φ en fixant G et W pour trouver les k médoïdes. Rappelons qu'une série médoïde est la série la plus représentative au sein d'un cluster donné.

$$m_l = \min_{t_j} \sum_{i|i\neq j,\ g_{il}=g_{jl}=1} w_{il}^{\alpha} \mathbf{dtw}(t_i, t_j). \tag{3.31}$$

— **Problème** Φ_3 : Minimisation de Φ en fixant G et M pour calculer la matrice de poids W. La solution du **Problème** Φ_3 est donnée par le théorème suivant :

Théorème 3.

$$w_{il} = \left(\sum_{l=1}^{k} \sum_{s=1}^{n} \left(\frac{g_{il}\mathbf{dtw}(t_i, m_l)}{g_{sl}\mathbf{dtw}(t_s, m_l)} \right)^{\frac{1}{\alpha-1}} \right)^{-1} \tag{3.32}$$

A l'issue du processus de l'optimisation décrit ci-dessus, nous calculons le score d'anormalité globale s_i pour chaque série temporelle t_i comme étant le produit scalaire du vecteur de poids $W(t_i) = [w_{i1}, w_{i2}, \ldots, w_{ik}]$ et le vecteur des distances correspondant $D(t_i) = [\mathbf{dtw}(t_i, m_1), \mathbf{dtw}(t_i, m_2), \ldots, \mathbf{dtw}(t_i, m_k)]$

$$s_i = \langle W(t_i), D(t_i) \rangle = [w_{i1}, \ldots, w_{ik}][dtw(t_i, m_1), \ldots, dtw(t_i, m_k)]^{\top} \tag{3.33}$$

Démonstration. Nous minimisons la fonction par les multiplicateurs de Lagrange. Soit μ le multiplicateur et $\Theta(W, \mu)$ le Lagrangien.

$$\Theta(W, \mu) = \sum_{l=1}^{k} \sum_{i=1}^{n} w_{il}^{\alpha} g_{il} \mathbf{dtw}(t_i, m_l) - \mu \left(\sum_{l=1}^{k} \sum_{i=1}^{n} w_{il} - 1 \right) \tag{3.34}$$

Pour minimiser $\Theta(W, \mu)$, le gradient de Θ par rapport à W et μ doit être égale à zéro. Ainsi, nous avons :

$$\frac{\partial \Theta(W, \mu)}{\partial w_{il}} = (\alpha) w_{il}^{(\alpha-1)} \left(g_{il} \mathbf{dtw}(t_i, m_l)\right) - \mu = 0; 1 \leq i \leq n \tag{3.35}$$

et

$$\frac{\partial \Theta(W, \mu)}{\partial \mu} = \sum_{l=1}^{k} \sum_{i=1}^{n} w_{il} - 1 = 0 \tag{3.36}$$

De l'équation (3.35), nous obtenons :

$$w_{il} = \left(\frac{\mu}{(\alpha) g_{il} \mathbf{dtw}(t_i, m_l)} \right)^{\frac{1}{1-\alpha}} \tag{3.37}$$

En substituant l'équation (3.37) dans l'équation (3.36), nous avons :

$$\sum_{l=1}^{k} \sum_{i=1}^{n} \left(\frac{\mu}{(\alpha) g_{il} \mathbf{dtw}(t_i, m_l)} \right)^{\frac{1}{1-\alpha}} = 1 \tag{3.38}$$

De l'équation (3.38), nous avons :

$$\mu^{\frac{1}{1-\alpha}} = \left(\sum_{l=1}^{k} \sum_{i=1}^{n} \left(\frac{1}{(\alpha) g_{il} \mathbf{dtw}(t_i, m_l)} \right)^{\frac{1}{1-\alpha}} \right)^{-1} \tag{3.39}$$

En substituant l'équation (3.39) dans (3.37) :

$$w_{il} = \frac{\left(\sum_{l=1}^{k} \sum_{s=1}^{n} \left(g_{sl} \alpha \mathbf{dtw}(t_s, m_l)^{-1} \right)^{\frac{1}{1-\alpha}} \right)^{-1}}{\left(g_{il} \alpha \mathbf{dtw}(t_i, m_l) \right)^{\frac{1}{1-\alpha}}} \tag{3.40}$$

Finalement, de l'équation (3.40) nous obtenons la formulation finale des poids pour le L2GAD :

$$w_{il} = \left(\sum_{l=1}^{k} \sum_{s=1}^{n} \left(\frac{a_{il} \mathbf{dtw}(t_i, m_l)}{a_{sl} \mathbf{dtw}(t_s, m_l)} \right)^{\frac{1}{\alpha-1}} \right)^{-1} \tag{3.41}$$

□ □

Subséquemment, nous pouvons résumer tous les développements mathématiques ci-dessus dans l'Algorithme 5.

Algorithme 5 : L2GAD

Entrées : une base de séries temporelles $T = \{t_1, t_2, ..., t_n\}$, les paramètres α et k.

Résultat : les séries temporelles t_i de la base T triées selon leur score global d'anormalité s_i

Choisir aléatoirement k séries temporelles de T comme médoïdes $\{m_1, m_2, ..., m_k\}$
Générer aléatoirement la matrice de poids $W \in R^{n \times k}$ tel que $\sum_{l=1}^{k} \sum_{i=1}^{n} w_{il} = 1$
Répéter
    Calculer G, la matrice de partitions par l'équation (3.30)
    Mettre à jour la liste des médoïdes M par l'équation (3.31)
    Mettre à jour la matrice des poids W par l'équation (3.32)
Jusqu'à *Convergence par rapport à W*;
Calculer le score global d'anormalité s_i de chaque série via l'équation (3.33)
Trier les séries temporelles selon leurs scores d'anormalité.

Analyse algorithmique

Tout comme les deux approches précédentes, l'algorithme L2GAD converge vers un optimum local dans un nombre fini d'itérations. Cependant, la complexité temporelle de L2GAD est légèrement plus importante que celles du DOTS et ℓ-DAT. En effet, contrairement à ces deux approches, la pondération est locale et le nombre d'opérations nécessaires pour calculer tous les poids est de l'ordre de $O(n \times 2k)$ au lieu de $O(2n)$.

3.4 Résultats expérimentaux

Dans cette section, nous représentons les résultats obtenus à l'issue des expérimentations que nous avons menées pour la validation des différentes approches proposées dans ce chapitre pour la détection non-supervisée des séries anormales. Dans les sections suivantes, nous détaillons le protocole expérimental mis en place ainsi que les tests statistiques utilisés pour la significativité des résultats. Nous décrivons aussi les jeux de données utilisés et les différents travaux auxquelles nous nous sommes comparés.

3.4.1 Jeux de données et comparaisons

Pour comparer les algorithmes de détection, nous avons besoin de jeux de données dédiés pour cette tâche. Or, il n' y a pas de jeu de données publiques où les séries temporelles anormales sont connues. Pour surmonter ce problème, nous avons utilisé des jeux de données initialement dédiées pour des tâches de classification ou de clustering,

disponibles dans *UCR Time Series Classification Archive* [Chen15]. Dans le but d'avoir des jeux de données sur lesquelles nous pouvons tester les algorithmes de la détection, nous avons suivi la démarche proposée par [Chandola08]. L'idée consiste à créer un jeu de données labellisé (normal, anormal) à partir du jeu de données initial. Pour chaque base, nous sélectionnons la classe majoritaire et nous la considérons comme ensemble de série normales. Ensuite, nous prenons aléatoirement quelques séries des autres classes et nous les considérons comme des séries anormales. Les caractéristiques des jeux de données sont décrites dans la Table 3.1.

JEUX DE DONNÉES	# SÉRIES (n)	# CLASSES (k)	TAILLE (L)
ADIAC	781	37	176
BEEF	60	5	470
CAR	120	4	577
CRICKETX	780	12	300
CRICKETY	780	12	300
CRICKETZ	780	12	300
DISTALPHALANXOAG	539	3	80
DISTALPHALANXTW	539	6	80
FACEALL	560	14	131
FACEFOUR	112	4	350
FACESUCR	2250	14	131
FISH	350	7	463
INSECTWINGBEATSOUND	2200	11	256
LARGEKITCHENAPPLIANCES	750	3	720
MEDICALIMAGES	1141	10	99
MIDDLEPHALANXOAG	554	3	80
NONIF-ECGTHORAX1	3765	42	750
NONIF-ECGTHORAX2	3765	42	750
OLIVEOIL	60	4	570
OSULEAF	442	6	427
PROXIMALPHALANXTW	605	6	80
SMALLKITCHENAPPLIANCES	750	3	720
SWEDISHLEAF	1125	15	128
TWOLEADECG	1162	2	82
UWAVEGESTURELIBRARYX	4478	8	315
UWAVEGESTURELIBRARYY	4478	8	315
UWAVEGESTURELIBRARYZ	4478	8	315
UWAVEGESTURELIBRARYALL	4478	8	945
WORDSSYNONYMS	905	25	270
WORMSTWOCLASS	258	2	900

TABLE 3.1 – Les caractéristiques des jeux de données utilisés dans les expérimentations

Pour évaluer la performance de nos approches, nous les avons comparées à six algorithmes différents. Les deux premiers sont des approches séquentielles basées sur le

clustering, tandis que les deux suivants sont basés sur les séparateurs à vaste de marge (SVM) [Scholkopf01]. Les deux derniers sont basées sur l'estimation de la densité.

— **DTW+KM** : Cette approche est basée sur la similarité et la proximité. Elle utilise l'algorithme $k-$médoïde pour effectuer un clustering en se basant sur la mesure de similarité DTW. Elle assigne un score d'anormalité à chaque série temporelle en fonction de son éloignement du médoïde de son cluster [Budalakoti09].

— **DTW+HC** : Cette approche est basée sur le clustering hiérarchique et la mesure de similarité DTW. Elle assigne un score d'anormalité à chaque série temporelle en fonction de son éloignement du médoïde de son cluster [Portnoy01].

— **DL-OCSVM** : Cette approche est basée sur deux étapes alternées : la première consiste à apprendre un dictionnaire afin de modéliser l'aspect normal des séries temporelles, tandis que la deuxième consiste à classifier les séries via une version modifiée du OC-SVM [Schölkopf99] pour détecter les séries anormales. en prenant en compte l'écartement de chaque série temporelle du modèle trouvé dans la première étape. [Marco15].

— **FD-OCSVM** : Cette approche analyse les séries temporelles dans le domaine fréquentiel via la transformée de Fourier. Une fois les séries transformées, la détection est faite via le OC-SVM [Schölkopf99].

— **DTW+LOF** : Le local outlier factor est basé sur l'estimation locale de la densité, où cette localité est paramétrée via le k plus proches voisins. La densité locale d'une série est obtenue via ses distances avec ses k plus proches voisins [Breunig00]. En comparant la densité locale d'une série par rapport aux densités locales de ses voisins, il est possible d'identifier des régions de densité similaire, et aussi des séries ayant une faible densité par rapport à celles de leur voisins. Ces séries temporelles sont considérées comme anormales.

— **Parzen-Window** : Cette approche est également basée sur l'estimation de la densité. Elle repose sur l'hypothèse de l'existence d'un modèle probabiliste génératif pour les séries temporelles observées. L'idée de l'approche consiste à estimer la fonction de densité de telle sorte que le modèle appris puisse décrire en mieux les séries observées. Le modèle appris est ensuite utilisé pour détecter les séries anormales en se basant sur la vraisemblance [Yeung02].

3.4.2 Protocole expérimental

Bien que la détection non-supervisée n'utilise pas les labels, ces derniers nous sont nécessaires pour l'évaluation et la comparaison avec les autres approches de l'état de

l'art. Toutefois, au lieu de simplement comparer les algorithmes en terme d'efficacité ou de précision/rappel, l'ordre selon lequel les anomalies sont détectées doit aussi être pris en compte. Dans la classification supervisée, une instance mal classée est certainement une erreur. Ceci est en effet différent dans le cas de la détection non-supervisé. Par exemple, si une grande base de séries temporelles contient uniquement 10 séries anormales, et qu'un algorithme de détection les mis dans son top 15 des séries anormales, le résultat reste très bon même si ce n'est pas parfait. Une bonne stratégie d'évaluation des algorithmes de détection non-supervisée, consiste alors à comparer l'ordre des détections de chaque algorithme et d'appliquer itérativement un seuil (selon lequel la série est considérée ou non comme anormale) du premier rang jusqu'au dernier [Goldstein16, Fawcett06]. Cette stratégie permet de calculer le taux des vrais positifs et des faux positifs pour chaque seuil, ce qui constitue en soi une courbe ROC [1]. Ainsi, l'aire sous la courbe (AUC), qui représente l'intégrale de la courbe ROC, peut être utilisée pour mesurer la performance de la détection. Chaque algorithme est lancé 50 fois sur chaque jeu de données, et sa performance moyenne ainsi que son écart type en terme d'AUC sont calculés pour la comparaison. Pour les algorithmes de clustering, le nombre de clusters k utilisé est celui indiqué dans la table 3.1.

Pour déterminer l'algorithme le plus performant parmi les autres algorithmes à partir des résultats expérimentaux, nous suivons la méthodologie proposée par [Demsar06] et qui consiste à utiliser des tests statistiques pour la significativité des résultats.

L'objectif étant de déterminer s'il y a assez d'évidence pour rejeter l'hypothèse nulle, qui suggère que tous les algorithmes ont la même performance. Tout d'abord, nous utilisons le test de Friedman qui est un test non-paramétrique [Friedman37]. Ce test est basé sur le classement de plusieurs algorithmes par rapport à plusieurs jeux de données. En prenant comme mesure d'évaluation l'AUC, le classement est effectué dans un ordre croissant, de l'algorithme le plus performant à l'algorithme le moins performant. Soit r_i^j le classement du $j^{\text{ème}}$ algorithme sur le $i^{\text{ème}}$jeu de données. Nous utilisons θ pour noter le nombre de jeu de données (ici $\theta = 30$), et δ pour désigner le nombre d'algorithmes (nous nous comparons avec 7 algorithmes ce qui fait que le nombre total des algorithmes testés dans l'expérimentation est égal à 10, $\delta = 10$). Le test de Freidman compare le classement moyen des algorithmes, $R_j = \frac{1}{\theta}\sum_j r_i^j$. Sous l'hypothèse nulle suggérant que les algorithmes ont la même performance (leurs classements devraient être égaux), la statistique de Friedman χ_F^2 dans l'équation (3.42) est distribuée selon une distribution *chi-square* avec $\delta - 1$ degrés de liberté.

$$\chi_F^2 = \frac{12 \times \theta}{\delta(\delta+1)}\left[\sum_{j=1}^{\delta} R_j^2 - \frac{\delta(\delta+1)^2}{4}\right] \tag{3.42}$$

1. ROC : Receiver Operator Characteristic

Une fois cette statistique calculée, elle est comparée à la valeur critique pour un certain seuil de significativité (dans nos expérimentations le seuil α est fixé à 0.05) pour rejeter ou accepter l'hypothèse nulle. Si cette hypothèse est rejetée par le test de Friedman, ce qui veut dire qu'il y a une différence significative entre les performances des algorithmes, nous utilisons un deuxième test statistique pour avoir une idée sur cette différence dans les performances. Dans notre cas, nous utilisons le test Nemenyi afin de comparer deux à deux les différents algorithmes. Ce test est basé sur le calcul d'une statistique q sur la différence entre les classements moyens R_j des algorithmes utilisés. Les performances de deux algorithmes comparés via ce test, sont significativement différentes si la différence entre leur classement moyen est supérieur ou égale à une distance dite critique (CD pour *critical distance*) définie par :

$$CD = q_\alpha \sqrt{\frac{\delta(\delta+1)}{6\theta}} = 3.102 \times \sqrt{\frac{10 \times 9}{6 \times 30}} = 2.1934 \tag{3.43}$$

où les valeurs critiques q_α sont basées sur les rangs statistiques standardisés divisés par $\sqrt{2}$, comme indiqué dans la Table 3.2.

# Algorithmes	1	2	3	4	5	6	7	8	9
$q_{0.05}$	1.96	2.343	2.569	2.728	2.85	2.949	3.031	3.102	3.164

TABLE 3.2 – Les valeurs critiques pour le test de Nemenyi

Comme nous pouvons le voir, la distance critique (CD) dépend uniquement du nombre de jeux de données, du nombre d'algorithmes à comparer ainsi que du niveau de significativité choisi.

3.4.3 Résultats

Dans cette section, nous reportons les résultats obtenus à l'issue des expérimentations que nous avons faites pour évaluer et comparer nos approches (DOTS, ℓ_2-DAT, L2GAD, et DetectS) par rapport aux différentes approches proposées dans l'état de l'art. Les expérimentations concernent la performance de détection de chaque approche ainsi que la complexité temporelle et le passage à l'échelle.

Qualité de la détection

Les résultats concernant la qualité et la performance de la détection sont présentés à la fois visuellement sous forme des courbes ROC dans les Figures (3.7, 3.8, 3.9 et 3.10) et numériquement (AUC) dans la Table 3.3. En effet, nos approches qui sont basées sur le clustering pondéré, à savoir DOTS, ℓ_2DAT et L2GAD sont performantes comparées autres approches sur l'ensemble des bases des séries temporelles à l'exception de certaines bases comme 'Cricket Y', 'Non-InvasiveThorax1', 'Non-InvasiveThorax2' et 'TwoLeadECG' où le DL-OCSVM est plus performant. C'est le cas aussi de la base

'WormTwoClass' où notre approche séquentielle qui est basée sur le clustering spectral, donne des résultats relativement bons par rapport à celles basées sur le clustering pondéré. Pour le reste des bases, nos trois approches les plus performantes arrivent en premier suivies soit par, DL-OCSVM,LOF, DTW+KM ou Parzen-Window.

Comme nous l'avons déjà mentionné dans le protocole expérimental, nous utilisons les résultats présentés dans la Table 3.3 pour dresser des tests statistiques selon la méthodologie proposée par [Demsar06] afin de valider significativement les résultats. Pour cela, nous présentons le classement moyen de chaque approche par base dans la dernière ligne de la Table 3.3 (Classement Moyen). En effet, l'hypothèse nulle selon la quelle toutes les approches ont la même performance est rejetée par le test de Friedman, $\chi_F^2 = 143.38$ où la valeur-p$= 2.0669\text{e}^{-26} < 0.05$. Puisque l'hypothèse nulle est rejetée, ce qui signifie que les performances des approches sont significativement différentes, nous utilisons le test post-hoc de Nemenyi afin d'avoir une idée plus précise sur cette différence et d'en tirer un classement plus pertinent.

Les résultats obtenus par le test Nemenyi sont présentés sous forme d'un diagramme dans la Figure 3.6(a). Quand la différence entre le classement moyen de deux approches est plus petite que la distance critique (CD), une ligne rouge est tracée dans le diagramme entre les deux approches pour montrer que la différence entre leurs performances n'est pas significative. Dans le Figure 3.6(a), nous pouvons voir que notre approche basée sur l'entropie (DOTS) est classée première sans qu'elle soit liée à aucune autre approche, ce qui signifie qu'elle est significativement plus performante en détection que les autres. Le constat est pareil pour nos deux autres approches basées sur le clustering pondéré ℓ_2-DAT, et L2GAD dans les Figures 3.6(b) et 3.6(c) respectivement. La Figure 3.6(d) montre que l'approche basée sur la pondération locale, L2GAD, s'est classée première y compris devant le DOTS et ℓ_2-DAT. Ce qui montre que l'approche locale permet d'améliorer, ne serait ce que légèrement la performance de nos approches de pondération globale, en exploitant notamment la densité de chaque cluster. Globalement, la Figure 3.6(d) montre que nos approches basées sur la pondération n'ont pas de différence significative entre elles, en revanche elle se distinguent nettement par rapport aux autres. Ceci montre que l'idée de la pondération améliore grandement les performances de détection.

Jeu de données	DOTS	ℓ_2-DAT	L2GAD	K-Medoid	DEtectS	HC	DL-OCSVM	FD-OCSVM	LOF	ParzenW
Adiac	**100**±0	97.701±1.498e-14	**100**±0	97.701±1.498e-14	86.552±15.317	64.368±1.498e-14	57.126±21.531	98.851±1.498e-14	96.552±0	56.322±0
Beef	**100**±0	**100**±0	**100**±0	29.167±0	43.333±29.866	29.167±0	65.417±7.3624	33.333±7.4898e-15	95.833±1.498e-14	**100**±0
Car	97±1.1249	97.25±0.40254	**97.5±0**	92.375±8.6959	91.25±9.6625	73.75±0	82.5±0	90±0	94.167±1.498e-14	30±3.7449e-15
CricketX	93.077±1.7498	93.846±0.79581	**94.769±0.84678**	77.209±11.26	74.297±10.334	80.33±1.498e-14	90.747±1.4558	88.791±1.498e-14	92.088±1.498e-14	49.67±7.4898e-15
CricketY	90.967±3.0303	**93.209±1.0228**	92.989±0.83175	81.209±11.198	80.033±7.333	86.813±0	92.703±0.66984	50.549±7.4898e-15	81.978±0	43.516±7.4898e-15
CricketZ	92.143±3.949	93.648±1.3792	**94.747±1.0474**	78.813±14.187	82.571±4.7058	57.363±1.498e-14	86.396±1.0267	77.033±1.498e-14	90.44±1.498e-14	68.462±0
DistalPha-OAG	79.472±0.93195	**80.458±0.54412**	80.318±0.14192	76.415±2.5528	80.28±0	77.212±0	79.341±0.054839	22.003±0	65.316±0	71.716±1.498e-14
DistalPhalanxTW	**84.751±1.059**	82.502±1.1112	84.22±0.89883	80.806±2.9253	78.464±2.3816	64.087±1.498e-14	79.098±0.11488	80.445±1.498e-14	54.042±7.4898e-15	64.193±0
FaceAll	94.847±1.7606	95.169±1.5815	**95.998±1.3834**	90.669±4.599	87.611±3.7271	73.853±1.498e-14	89.151±0.21248	84.77±0	86.234±0	76.42±0
FaceFour	95.221±1.2132	96.471±1.691	**97.059±1.2498**	69.191±19.066	64.632±17.206	51.838±0	81.029±0.46504	80.147±1.498e-14	91.176±1.498e-14	67.647±1.498e-14
FacesUCR	90.376±1.4982	90.153±1.9121	91.774±1.075	85.933±3.6722	87.12±2.902	71.435±0	84.809±0.088461	62.459±1.498e-14	83.292±0	76.545±1.498e-14
fish	77.333±1.3147	78.7±0.89512	**78.833±0.61363**	67±7.4037	71.133±4.3637	75.667±1.498e-14	77±0	44.333±7.4898e-15	70.667±1.498e-14	36±0
InsectWingbeatSound	98.988±0.31507	98.51±0.59808	**98.99±0.31154**	90.961±5.0875	85.982±4.0634	73.298±0	97.064±0.10636	80.583±1.498e-14	77.345±0	85.321±0
LargeKitchenApp	66.895±2.1226	66.568±1.2005	**66.994±1.1705**	50.902±7.1047	56.986±2.5075	46.908±7.4898e-15	38.949±1.9271	63.677±7.4898e-15	35.192±7.4898e-15	65.2±1.498e-14
MedicalImages	**62.427±1.1478**	61.799±0.6612	62.137±0.41196	57.57±1.6628	57.64±1.4902	58.889±7.4898e-15	55.845±0.12209	52.565±0	61.049±1.498e-14	61.016±7.4898e-15
MiddlePha-OAG	**87.469±0.20893**	85.592±1.0764	86.761±1.0616	85.5±1.5437	84.928±0	76.74±0	70.155±0.0098455	23.12±3.7449e-15	46.721±0	58.668±7.4898e-15
NonIF-ECGThorax1	87.148±4.7709	**91.092±0.53521**	89.439±3.3637	71.388±4.2219	34.918±5.3909	16.327±3.7449e-15	97.959±1.498e-14	96.327±1.498e-14	97.041±0	80.816±1.498e-14
NonIF-ECGThorax2	88.658±3.6764	94.066±3.6743	92.464±2.4362	65.684±10.117	28.429±4.5598	16.327±3.7449e-15	98.265±1.498e-14	91.224±1.498e-14	**98.163±1.498e-14**	70.918±1.498e-14
OliveOil	98.533±0.68853	98.533±0.68853	**98.8±0.68853**	71.533±19.993	82.467±16.439	36±0	29±21.348	64±0	94±0	90±0
OSULeaf	99.351±0.23332	99.134±0.17657	**99.361±0.11704**	93.134±6.564	96.753±2.1489	78.041±0	92.711±0.90926	99.072±1.498e-14	98.247±1.498e-14	75.979±0
ProximalPha-TW	**79.758±1.3797**	77.556±1.176	78.081±1.2315	74.175±3.3492	75.303±2.5771	62.904±1.498e-14	68.766±0.31904	75.404±1.498e-14	47.413±0	75.366±0
SmallKitchenApp	**64.234±1.3594**	63.451±1.1072	62.663±0.95289	51.18±4.6309	37.548±0.82036	43.954±0	35.889±2.1975	35.708±0	41.892±7.4898e-15	63.585±7.4898e-15
SwedishLeaf	94.933±3.0421	89.95±1.2045	90.85±1.7541	75.25±10.258	71.733±5.3548	19±0	88.167±1.498e-14	87.833±1.498e-14	88±1.498e-14	85±0
TwoLeadECG	54.915±0.052055	63.424±7.4898e-15	63.424±7.4898e-15	61.489±1.6654	60.129±1.498e-14	58.634±1.498e-14	77.432±1.498e-14	70.303±0	**75.081±1.498e-14**	71.461±1.498e-14
uWaveGestureX	**91.115±0.32692**	90.331±0.29354	90.77±0.32156	87.136±3.7526	89.443±0.5705	59.908±7.4898e-15	75.089±0.75138	90.025±1.498e-14	55.367±7.4898e-15	76.482±1.498e-14
uWaveGestureY	**89.624±1.1147**	80.647±5.1265	87.852±4.5302	79.256±8.4905	74.834±6.6355	57.361±7.4898e-15	57.087±0.80485	55.949±7.4898e-15	49.076±7.4898e-15	46.501±0
uWaveGestureZ	**92.168±1.0759**	90.071±3.2942	91.414±1.5319	84.511±6.1967	88.698±3.632	65.124±0	71.347±0.93341	81.862±1.498e-14	56.013±7.4898e-15	78.053±1.498e-14
uWaveGestureAll	**92.932±0.70851**	90.036±1.8268	91.943±0.70637	87.557±2.5564	84.603±3.1975	71.051±1.498e-14	85.951±0.17345	88.531±0	42.166±7.4898e-15	72.533±1.498e-14
WordsSynonyms	**84.43±1.5079**	83.382±0.95809	84.179±0.94298	73.502±3.8872	76.713±3.7785	50.202±0	78.483±0.34358	72.524±0	81.917±0	80.571±0
WormsTwoClass	55.362±1.3281	55.166±1.3371	55.468±1.3313	53.329±3.3217	**55.906±0**	50.515±0	49.7±0.12442	46.98±7.4898e-15	51.32±7.4898e-15	53.333±7.4898e-15
Classement Moyen	**2.55**	**2.83**	1.98	**6.36**	**6.40**	**8.35**	**6.06**	**6.90**	**6.40**	**7.15**

TABLE 3.3 – Les performances en termes d'AUC des différentes approches de détection sur 30 jeux de données.

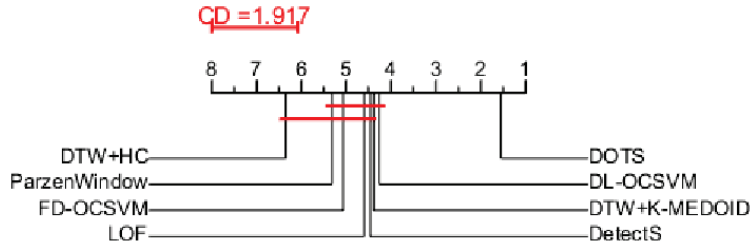
3.4.3.1 Complexité temporelle

Nous nous intéressons dans cette partie à la complexité temporelle de chaque approche et sa capacité de passer à l'échelle. Nos approches sont comparées par rapport aux différences approches en terme de temps de calcul nécessaire pour la détection. La Table 3.4 contient le temps d'exécution moyen de chaque approche (en seconde) par rapport à tous les jeux de données, dont la dernière ligne contient le classement moyen de chaque approche par rapport à chaque jeu de données. En se basant sur ces résultats et sur le classement moyen des approches, nous utilisons encore une fois des tests statistiques pour savoir s'il y a une différence significative dans le temps d'exécution de chaque approche. En effet, l'hypothèse nulle selon la quelle toutes les approches ont le même temps d'exécution, ayant été rejetée par le test de Firedman, $\chi_F^2 = 250.81$ où la p-valeur=$6.7142e^{-49} < 0.05$, nous utilisons le test post-hoc de Nemenyi pour une analyse plus détaillée.

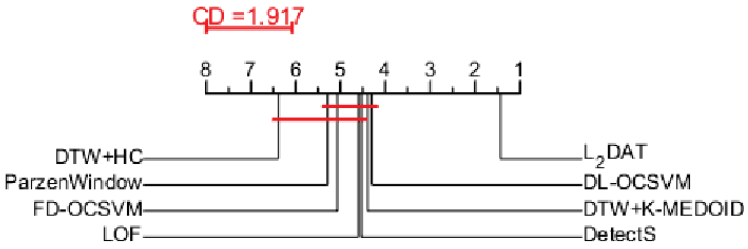
Tout d'abord, les résultats dans la Table 3.4 montrent une différence importante dans les temps d'exécution des approches, notamment pour les grandes bases. Les résultats du test Nemenyi sont présentés dans la Figure 3.11. Plus le classement moyen d'une approche est élevé, moins elle est gourmande en complexité temporelle.

Comme nous pouvons le voir dans la Figure 3.11(d), les approches basées sur le clustering, notamment DTW+KM, ℓ_2-DAT, DTW+HC, et DOTS, sont de loin les approches les plus rapides. A noter aussi, que la complexité de notre approche basée sur la pondération locale ,L2GAD, est plus importante que la complexité de nos approches basées sur la pondération globale, à savoir ℓ_2-DAT et DOTS. Toutefois, la complexité de nos trois approches ci-dessus est relativement plus importante que celle du DTW+KM, et cela s'explique d'ailleurs par le fait de l'étape supplémentaire qui est nécessaire dans le processus de pondération. A noter que la complexité du ℓ_2-DAT n'est pas significativement différente que celle du DTW+KM.

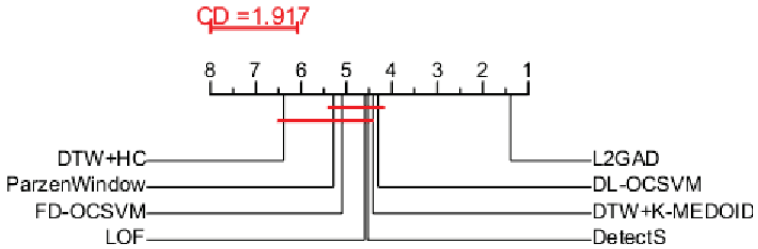
De l'autre côté du classement, notre approche séquentielle DetectS, possède une complexité plus importante que nos trois autres approches *embedded* et cela peut s'expliquer notamment par la décomposition spectrale utilisée dans cette approche. Les approches basées sur l'estimation de la densité (LOF, et ParzenWindow) consomment encore plus de temps et cela à cause du calcul du voisinage qui est assez gourmand en temps. Enfin, les approches basées sur le OC-SVM (DL-OCSVM, DF-OCSVM) consomment beaucoup plus de temps par rapport aux autres, et le DL-OCSVM nécessite un temps de calcul très important et cela est dû à l'apprentissage du dictionnaire pour chaque itération.



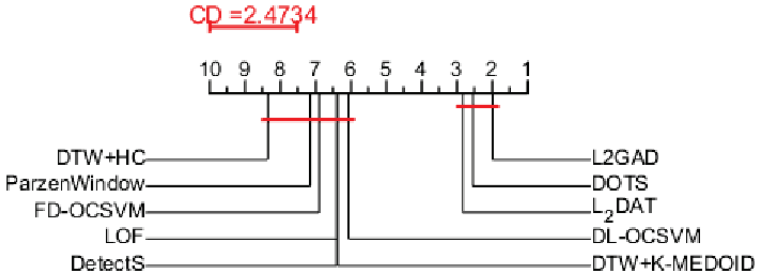
(a) Le classement moyen du DOTS.



(b) Le classement moyen du ℓ_2-DAT.

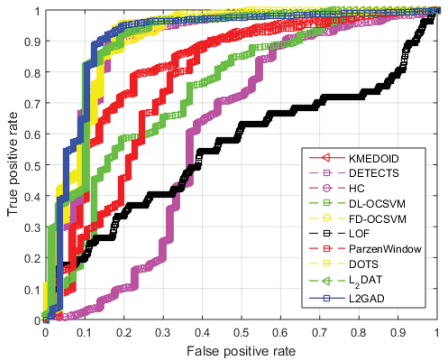


(c) Le classement moyen du L2GAD.

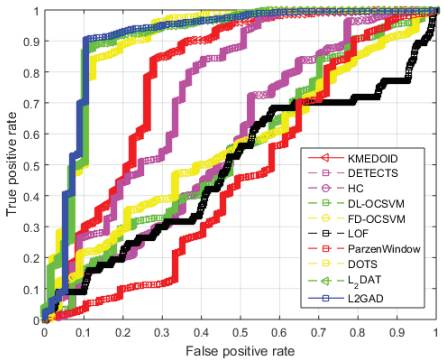


(d) Le classement moyen global.

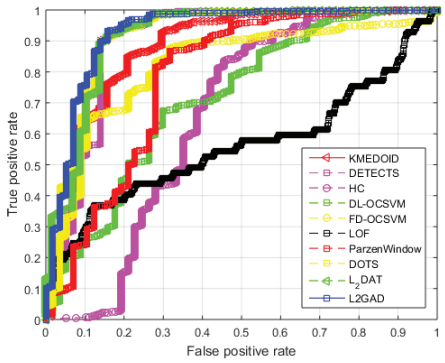
FIGURE 3.6 – Les différents diagrammes de classement moyen des approches de détection en terme d'AUC. Les sous-Figures, 3.6(a), 3.6(b) et 3.6(c), présentent le classement moyen de chacune de nos approches de pondération , prises individuellement, par rapport aux autres approches. La sous-Figure 3.6(d) dresse un classement global où nous pouvons voir que nos approches de pondération sont nettement plus performantes que les autres approches. A noter que la distance critique (CD) dans les trois premières figures est égale à 1.917 alors que dans la dernière figure elle est égale à 2.4734. En effet, sa valeur dépend du nombre d'approches mises en comparaison comme indiqué dans l'équation (3.43).



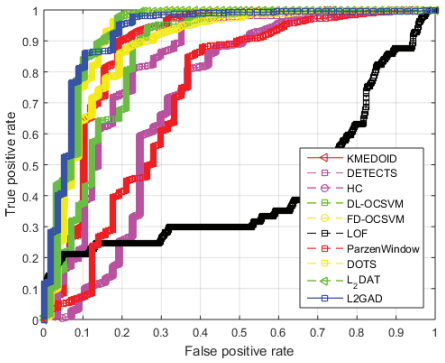
(a) uWaveGestureX



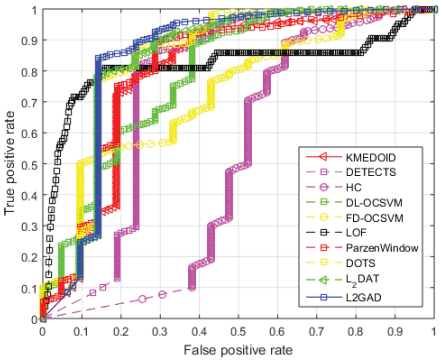
(b) uWaveGestureY



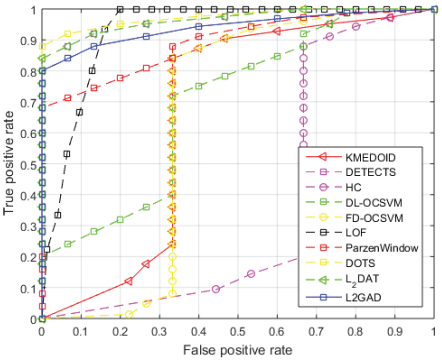
(c) uWaveGestureZ



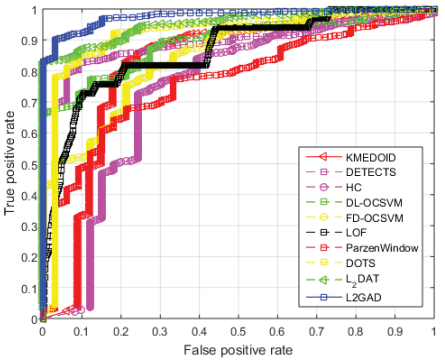
(d) uWave-G-All



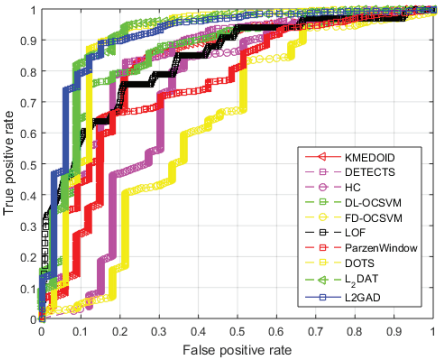
(e) Words



(f) OliveOil

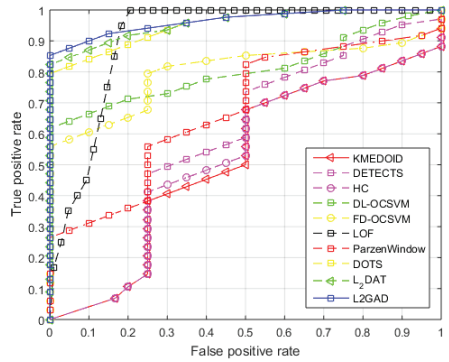


(g) FaceAll

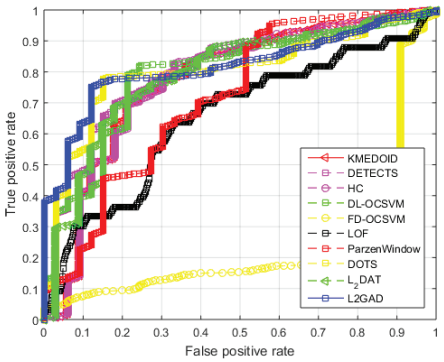


(h) FacesUCR

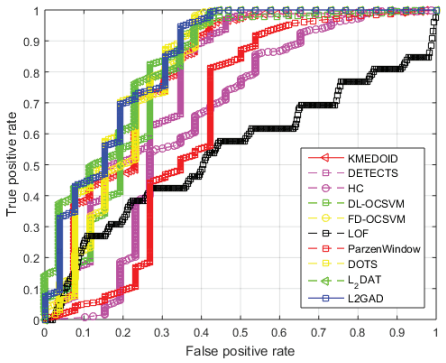
FIGURE 3.7 – Les courbes ROC des différentes approches sur les 30 jeux de données de l'expérimentation.



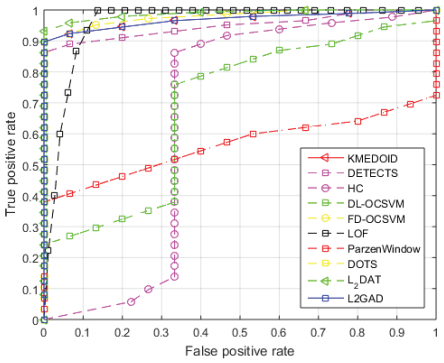
(a) FaceFour



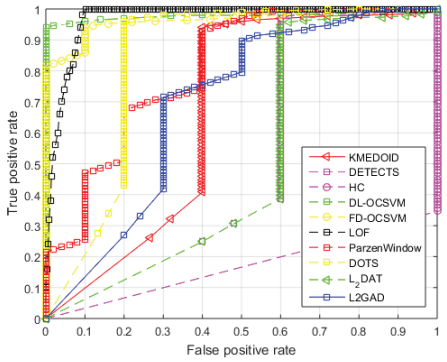
(b) DistalPAG



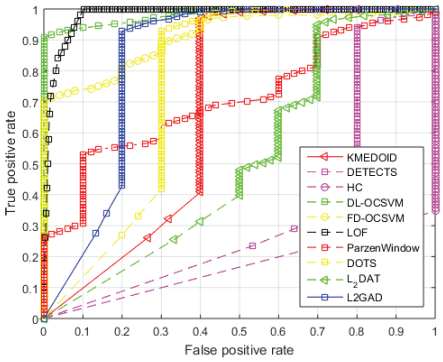
(c) DistalTW



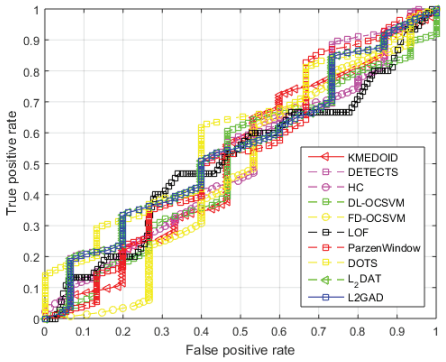
(d) Adiac



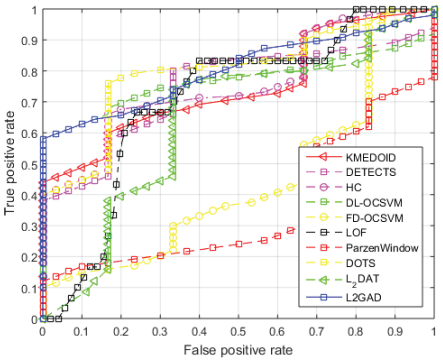
(e) Non-Inv-Thorax1



(f) Non-Inv-Thorax2

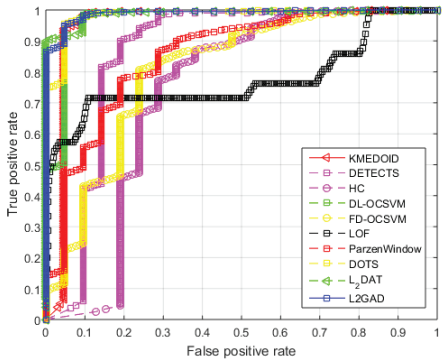


(g) Worm2Class

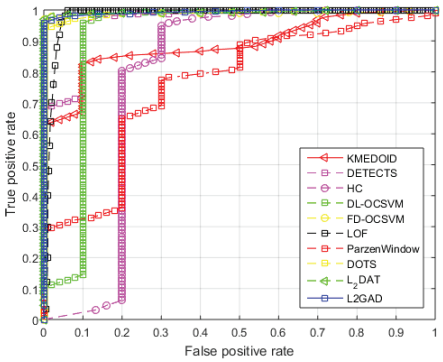


(h) FISH

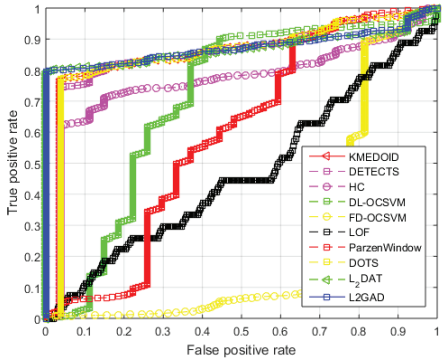
FIGURE 3.8 – Les courbes ROC des différentes approches sur les 30 jeux de données de l'expérimentation (la suite-1).



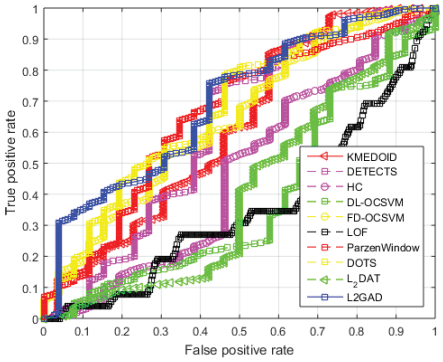
(a) InsectWing



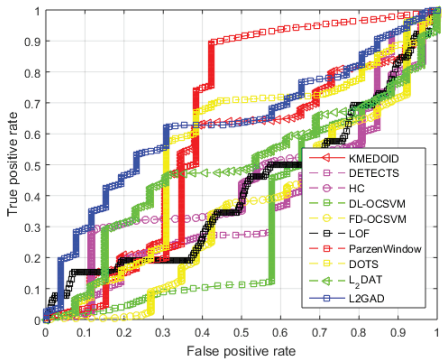
(b) OSULeaf



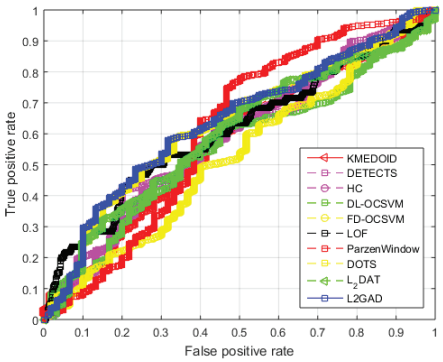
(c) MiddlePOAG



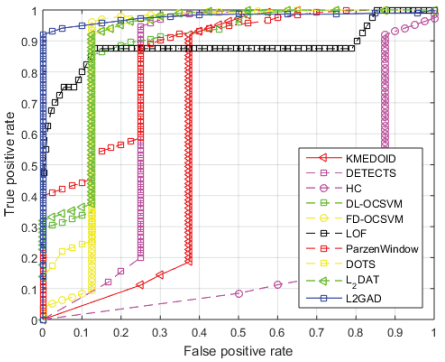
(d) LargeKApp



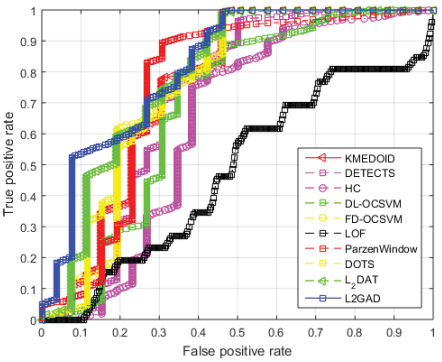
(e) SmallKApp



(f) Medicalresultats/images

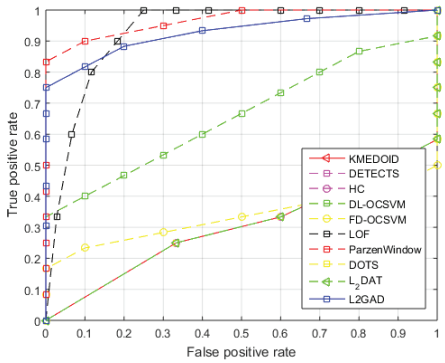


(g) SwedishLeaf

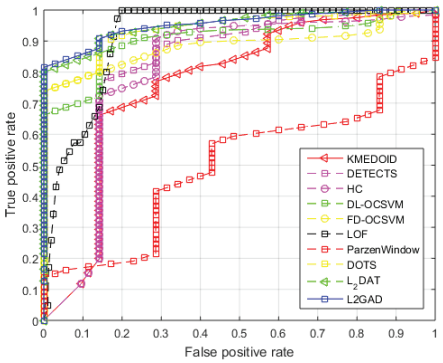


(h) ProximaTW

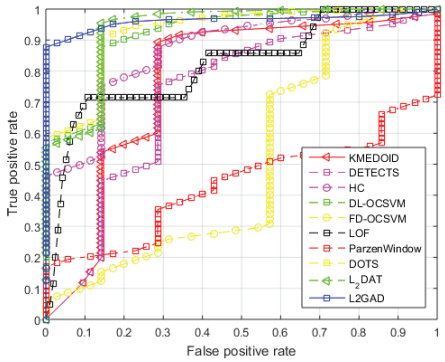
FIGURE 3.9 – Les courbes ROC des différentes approches sur les 30 jeux de données de l'expérimentation (la suite-2).



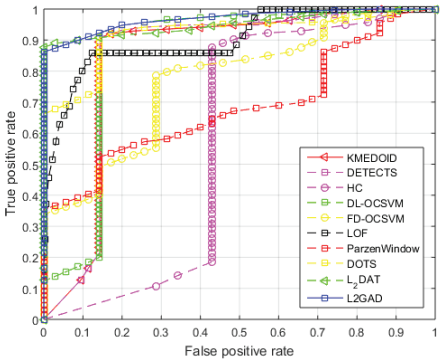
(a) Beef



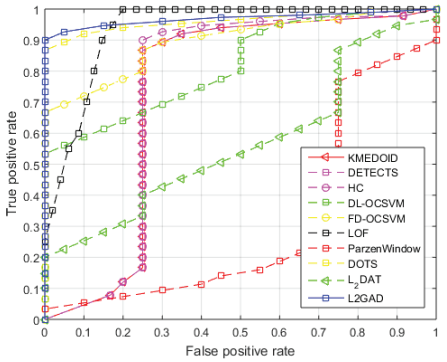
(b) CricketX



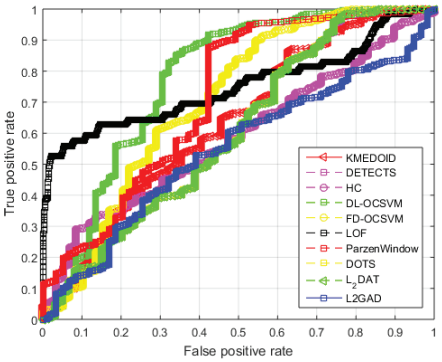
(c) CricketY



(d) CricketZ



(e) Car



(f) TwoLeadECG

FIGURE 3.10 – Les courbes ROC des différentes approches sur les 30 jeux de données de l'expérimentation(la suite-3).

Jeux de données	DOTS	ℓ_2DAT	L2GAD	DTW+K-Medoid	DetectS	DTW+HC	DL-OCSVM	FD-OCSVM	LOF	ParzenW
Adiac	0.00050062±0.00016705	0.00039953±0.00012499	0.020237±0.0060104	0.00035027±0.00013308	0.0096091±0.0017175	0.0021679±0.00067999	2.2396±0.64437	0.011741±0.0019174	0.063634±0.0082224	0.0062482±0.00095831
Beef	0.00017337±5.0906e-05	0.00016161±3.7512e-05	0.012767±0.002499	0.00022326±9.152e-05	0.0080688±0.0013818	0.0024237±0.00044846	0.56905±0.11113	0.0089444±0.0017658	0.069403±0.008294	0.0048387±0.00079985
Car	0.00045891±0.00018639	0.00044036±0.00010818	0.017215±0.0039243	0.00036244±0.00011834	0.012801±0.0011322	0.0023751±0.00037393	0.65049±0.061345	0.02093±0.0046005	0.21844±0.012245	0.018484±0.0014089
CricketX	0.001602±0.00068776	0.00099223±0.00019498	0.022473±0.0061346	0.00091995±0.00032421	0.026823±0.0024219	0.0027395±0.00032815	0.87489±0.18185	0.025587±0.0062888	0.25079±0.014431	0.030919±0.0032435
CricketY	0.0013493±0.00027616	0.0010148±0.00045998	0.024837±0.0055448	0.00085624±0.00038982	0.034042±0.0091872	0.0030736±0.00067389	0.98792±0.16567	0.025354±0.0037117	0.24967±0.021147	0.028776±0.0038353
CricketZ	0.0012778±0.00025361	0.00084136±0.00021132	0.02032±0.004552	0.00077608±0.00029664	0.02997±0.0041402	0.0034058±0.00066428	0.89746±0.19235	0.025574±0.0090697	0.25325±0.029613	0.026447±0.0026703
DistalPha-OAG	0.01649±0.0050292	0.0062658±0.0014098	0.034663±0.012949	0.0050611±0.001139	0.02905±0.010776	0.0055062±0.00063299	6.1631±0.61608	0.14745±0.013951	0.51958±0.04688	0.1±0.0067389
DistalPhalanxTW	0.010963±0.0024441	0.0040757±0.00077362	0.031535±0.008273	0.0027327±0.00084641	0.032853±0.006981	0.0052205±0.00062425	4.9966±1.3474	0.099077±0.0095563	0.38926±0.014919	0.071045±0.0039609
FaceAll	0.018365±0.0025748	0.0061798±0.0012868	0.031945±0.0084123	0.0041408±0.00196	0.091451±0.063368	0.0057064±0.0010564	1.9708±0.17964	0.15887±0.013881	0.68401±0.069107	0.12063±0.012457
FaceFour	0.00045242±7.8011e-05	0.00040327±0.0001822	0.016977±0.0035575	0.00037582±0.00012666	0.015306±0.0043538	0.0034231±0.00059194	0.65999±0.12132	0.013613±0.0012781	0.13862±0.016926	0.01294±0.0025302
FacesUCR	0.020032±0.0050292	0.0066748±0.0009752	0.036293±0.0088259	0.0042306±0.0007424	0.065543±0.0063038	0.0058007±0.00037934	2.0425±0.078322	0.16778±0.017951	0.6807±0.037541	0.12247±0.010282
FISH	0.0008403±0.0001551	0.00055073±0.00018652	0.020283±0.0063609	0.00054285±0.00018407	0.018316±0.0016677	0.0036155±0.00046571	0.77332±0.055549	0.027527±0.0059951	0.28663±0.038179	0.027497±0.0047192
InsectWingbeatSound	0.0088977±0.0019635	0.0031763±0.00094409	0.029144±0.0058407	0.0025044±0.00083486	0.03677±0.0041179	0.0048569±0.00065352	1.6386±0.23842	0.086175±0.0055092	0.62022±0.052907	0.0888±0.014062
LargeKitchenApp	0.0096638±0.0012989	0.003864±0.00067532	0.025817±0.0044414	0.0027845±0.00057909	0.059939±0.0073822	0.0053334±0.00081097	4.3137±0.23931	0.13851±0.012095	2.3975±0.25381	0.76444±0.075564
MedicalImages	0.076235±0.017291	0.022745±0.0062135	0.056468±0.012641	0.010793±0.0055439	0.10688±0.021879	0.012197±0.0029933	3.5022±0.20916	0.35851±0.026685	1.1366±0.069277	0.26233±0.016719
MiddlePha-OAG	0.013545±0.0022193	0.0056275±0.0020347	0.031603±0.0075765	0.0043609±0.0012157	0.028342±0.0083517	0.0058386±0.0007766	5.8796±1.5302	0.13544±0.016426	0.47746±0.060357	0.092819±0.010304
NonIF-ECGThorax1	0.002919±0.00066982	0.0024753±0.0010391	0.024503±0.0049712	0.0020005±0.00093439	0.076777±0.0085517	0.0048723±0.00096297	1.5137±0.068028	0.06009±0.0068159	0.93492±0.083842	0.14922±0.011629
NonIF-ECGThorax2	0.0033624±0.00062231	0.0021102±0.00066216	0.022789±0.0042659	0.0018943±0.00053326	0.072994±0.013103	0.0049527±0.0010525	1.5259±0.10708	0.056745±0.0035002	0.94118±0.057215	0.15429±0.014641
OliveOil	0.00029122±9.0234e-05	0.00032549±0.00017148	0.015071±0.0031304	0.00033333±7.37e-05	0.013451±0.001736	0.0038022±0.00053167	2.1651±0.65157	0.014437±0.0023637	0.17437±0.040199	0.012572±0.0015236
OSULeaf	0.0021772±0.00051883	0.0012496±0.00058346	0.01936±0.0040583	0.00085254±0.00032597	0.024452±0.0036022	0.0040662±0.00053292	1.3334±0.28918	0.0467±0.0030269	0.52391±0.061212	0.045472±0.0052686
ProximalPha-TW	0.011199±0.0043723	0.0039863±0.00066285	0.032643±0.0053099	0.0035279±0.0024161	0.028552±0.0034875	0.0054091±0.00056337	1.7283±0.079183	0.11118±0.0095766	0.41532±0.041984	0.074997±0.0090211
SmallKitchenApp	0.0087523±0.001825	0.0031543±0.00082069	0.021088±0.0030881	0.0024703±0.00058091	0.056349±0.0075467	0.0050073±0.00026469	4.255±0.15358	0.13584±0.013934	2.3362±0.1993	0.7509±0.079656
SwedishLeaf	0.0015529±0.00035598	0.00094641±0.00030031	0.016337±0.0034174	0.00082036±0.00021504	0.02211±0.0013884	0.004154±0.00044476	3.3792±0.33825	0.023206±0.001966	0.13272±0.0070211	0.016803±0.00095744
TwoLeadECG	0.054584±0.016169	0.01939±0.0035579	0.050714±0.010892	0.014603±0.005513	0.060853±0.0076882	0.012045±0.00077033	10.559±0.17756	0.29932±0.022434	0.92129±0.014953	0.18806±0.01647
uWaveGestureX	0.046208±0.007241	0.012779±0.0023994	0.041855±0.008942	0.0093215±0.0035368	0.090811±0.0091756	0.011536±0.00072152	5.4388±0.13765	0.33058±0.024271	3.1147±0.1705	1.4995±0.054404
uWaveGestureY	0.049044±0.021973	0.014346±0.0015376	0.034909±0.0064883	0.01049±0.0027734	0.090162±0.012056	0.012375±0.0014665	4.198±1.0142	0.33635±0.034623	3.1666±0.19969	1.5577±0.069285
uWaveGestureZ	0.053019±0.01445	0.014055±0.0034844	0.042932±0.010936	0.010934±0.0036237	0.081705±0.0090035	0.011315±0.00068698	4.7985±0.82401	0.34612±0.035719	3.1499±0.20501	1.5679±0.10826
UWaveGestureAll	0.042591±0.0064256	0.01327±0.0030494	0.036083±0.0073127	0.0097512±0.001631	0.094121±0.010456	0.012481±0.00069158	10.264±1.2136	0.46505±0.071946	8.5107±0.64777	4.4417±0.29049
WordsSynonyms	0.0071561±0.0021079	0.0029152±0.00079023	0.023369±0.005166	0.0026222±0.00090588	0.071146±0.015613	0.005783±0.00073081	1.8156±0.21363	0.081662±0.0041649	0.65027±0.052381	0.085103±0.0044348
WormsTwoClass	0.0041239±0.0014436	0.0018861±0.00059754	0.02594±0.0078127	0.0015626±0.00060413	0.015395±0.0015568	0.0050916±0.00061186	2.9166±0.53258	0.090837±0.0080629	1.5657±0.069584	0.3351±0.024295
Classement Moyen	7.36	8.73	5.60	9.83	4.90	7.86	1.0	3.8	2	3.9

TABLE 3.4 – Le temps d'exécution des différentes approches sur les 30 jeux de données de l'expérimentaiton. Les résultats montrent notamment une grande différence concernant les grands jeu de données. Ces expérimentations ont été faites sur un poste de travail individuel (i7-4980HQ CPU (2.80GHz) et 16.0 GB de RAM).

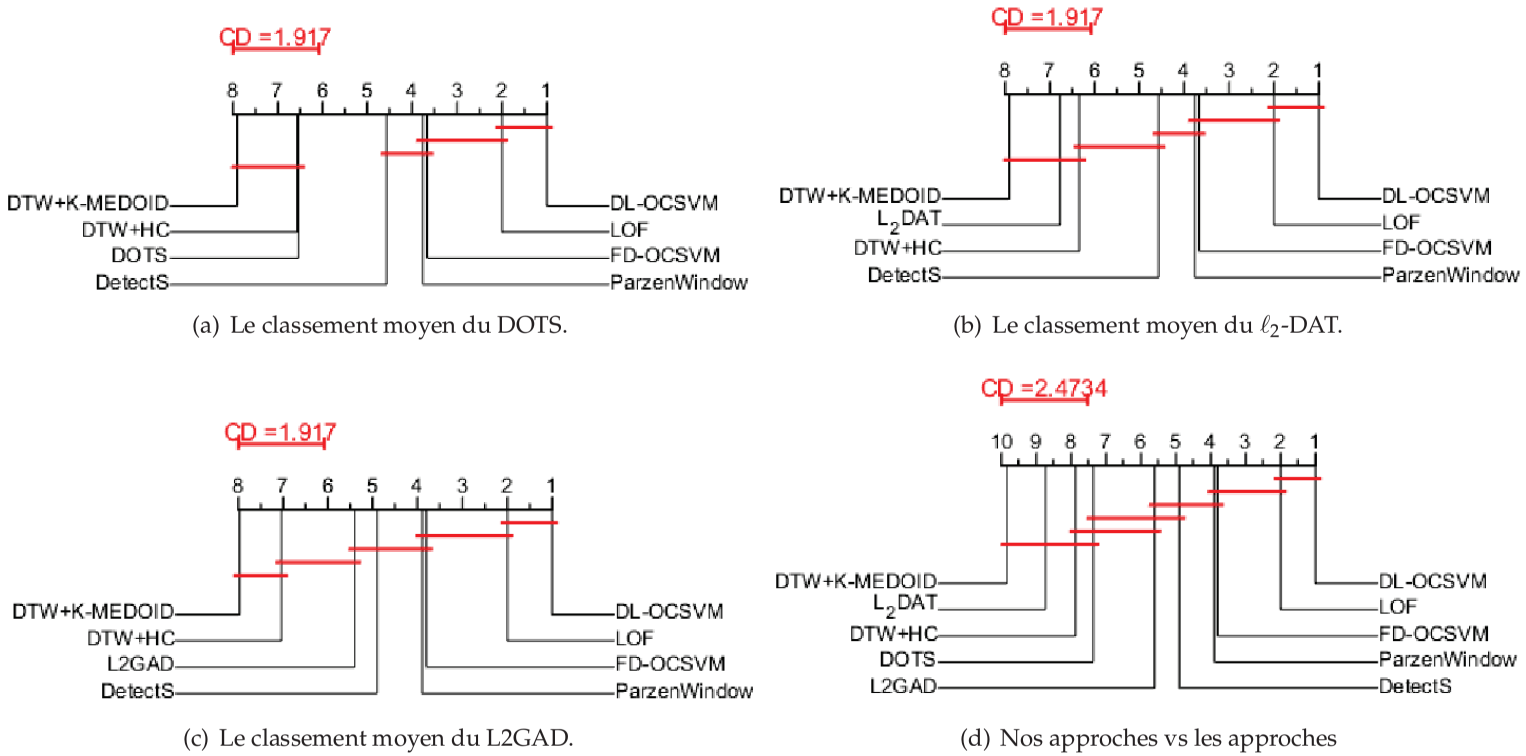
3.5 Discussion

Les résultats numériques que nous avons obtenues sont encourageants et montrent que notre idée de la pondération, locale qu'elle soit ou globale, améliore significativement la performance des approches basées sur la similarité et plus particulièrement, celles qui sont basées sur le clustering. Nous nous intéressons à présent à la comparaison de nos approches avec les autres approches selon plusieurs critères que nous jugeons pertinents pour la détection :

- **Perte d'information :** ce critère est utilisé pour savoir si une approche traite les séries temporelles dans leur représentation originale ou qu'elle passe par une étape de transformation. Cette dernière étape impliquant souvent une perte d'information, nous pensons qu'une bonne approche devrait se passer de cette étape.
- **Construction d'un modèle :** une bonne approche devrait apprendre un modèle de normalité de telle sorte qu'elle puisse confronter les nouvelles séries temporelles de test avec ce modèle.
- **Normalité (multi-profils) :** la normalité des données est souvent représentée sous forme de plusieurs profils. Les approches de détection devraient prendre en considération cette information de telle sorte qu'elle puisse détecter, si elles existent, des anomalies par profil.
- **Robustesse :** ce critère est important notamment dans la phase de création du modèle de normalité. Les approches de détection ne devraient pas être influencées par les anomalies présentes dans le jeu de données.
- **Régions de densité différentes :** ce critère est assez important notamment pour les approches à base de clustering. En, effet, les données appartiennent souvent à des régions de densités différentes. Ainsi, la dispersion des séries temporelles dans un cluster moins dense (sparse) ne devrait pas motiver à elle seule l'anormalité de ces séries.
- **Clusters de formes arbitraires :** les profils de normalité (clusters) sont parfois représentés dans l'espace sous différentes formes arbitraires (pas que convexes et sphériques). Ainsi, il est intéressant que les approches prennent cette spécificité des données en considération.
- **Passage à l'échelle :** ce critère est évidemment important pour l'exécution sur des données massives.

Dans la Table 3.5, les approches sont comparées par rapport aux critères ci-dessus.

Tout d'abord, les approches que nous proposons traitent les séries temporelles directement sans passer par une étape de transformation, ce qui implique qu'il n'y a pas de perte d'information contrairement aux approches basées sur le OC-SVM. Face aux approches de clustering séquentielles telles que la DTW+KM ou la DTW+HC, nos approches de pondération sont robustes par rapport aux séries anormales présentes dans les jeux de données. Par rapport aux approches basées sur la densité (LOF), qui ne



(a) Le classement moyen du DOTS.

(b) Le classement moyen du ℓ_2-DAT.

(c) Le classement moyen du L2GAD.

(d) Nos approches vs les approches

FIGURE 3.11 – Les différents diagrammes de classement moyen des approches de détection en terme de complexité temporelle et de passage à l'échelle. Les Figures 3.11(a), 3.11(b) et 3.11(c), présentent le classement moyen de chacune de nos approches, prises individuellement, par rapport aux autres approches de l'état de l'art. La dernière Figure 3.11(d), dresse un classement moyen global de toutes les approches où nous pouvons constater que nos trois approches de pondérations font partie des approches ayant une faible complexité temporelle. A noter que la distance critique (CD) dans les trois premières figures est égale à 1.917 alors que dans la dernière figure elle est égale à 2.4734. En effet, sa valeur dépend du nombre d'approches mises en comparaison comme indiqué dans l'équation (3.43).

Critères	DOTS	ℓ_2-DAT	L2GAD	DetectS	K-Medoid	HC	DL-OCSVM	FD-OCSVM	LOF	ParzenW
Perte d'information	NO	NO	NON	NON	NON	NON	OUI	OUI	NON	OUI
Construction d'un modèle	OUI	OUI	OUI	OUI	OUI	OUI	OUI	OUI	NON	OUI
Normalité (multi-profil)	OUI	OUI	OUI	OUI	OUI	OUI	NON	NON	OUI	OUI
Robuste aux anomalies	OUI	OUI	OUI	NON	NON	NON	OUI	OUI	OUI	OUI
Densité différente	NO	NO	OUI	OUI	NON	NON	NON	NON	OUI	OUI
Cluster sous forme arbitraire	NO	NO	NO	OUI	NON	NON	NON	NON	OUI	NON
Passage à l'échelle	OUI	OUI	OUI	NON	OUI	OUI	NON	NON	NON	NON
Critères Remplis	**5/7**	**5/7**	**6/7**	**5/7**	**4/7**	**4/7**	**2/7**	**2/7**	**5/7**	**4/7**

TABLE 3.5 – Comparaison des différentes approches en fonction de plusieurs critères.

créent pas un modèle de normalité à l'issue de la détection, nos approches apprennent un modèle représentant plusieurs profils de normalité. De même, les approches basées sur le OC-SVM assument l'existence d'un seul modèle de normalité, alors que nos approches sont capables de détecter des anomalies à partir de plusieurs profils de normalité. L'approche L2GAD permet de traiter le cas des clusters de différentes densités, alors que ce n'est pas le cas pour les approches DOTS et ℓ_2-DAT. Toutefois, L2GAD ne permet pas de traiter le cas des clusters de forme arbitraires alors que notre approche séquentielle DetectS le permet via la décomposition spectrale. Afin de répondre à tous ces critères nous pouvons par exemple intégrer L2GAD dans l'approche DetectS. En effet, cette dernière est basée sur le clustering Spectral, qui est elle même basée sur la décomposition spectrale et le clustering des vecteurs propres. Une nouvelle piste serait d'utiliser le mécanisme de la pondération dans la phase de clustering de vecteurs propres.

3.6 Conclusion

Dans ce chapitre, nous avons proposé quatre méthodes pour détecter, globalement, des séries temporelles anormales dans un contexte non-supervisé. La première approche, DetectS, dite *séquentielle*, est basée sur le clustering spectral suivi d'une étape de détection via un score basé sur l'inertie. Les trois autres approches, dites *embedded*, sont basées sur un clustering pondéré. Toutes nos approches ont été évaluées et comparées par rapport à plusieurs méthodes représentatives de l'état de l'art. Des résultats sur plusieurs bases ont été menés et ont montré que les approches *embedded* que nous avons proposées, sont efficaces et plus performantes que leur concurrentes.

Toutes les contributions présentées dans ce chapitre permettent de détecter des séries temporelles qui sont relativement anormales par rapport à la totalité d'une base de données disponible, d'où leur aspect global dans le processus de détection. Dans le chapitre suivant, nous nous focaliserons sur le caractère contextuel d'une série temporelle pour cibler la détection sur des points locaux au sein de cette série.

4

Détection contextuelle non-supervisée : Modélisation Matricielle

▷ *Dans ce chapitre, nous nous intéressons à la détection contextuelle des observations anormales au sein d'une série temporelle. Nous proposons une approche basée sur la reconstruction des observations de la série via ses observations les plus représentatives. Ainsi, la détection d'anomalies se fait en analysant les résidus de la reconstruction et l'aspect temporel des données est pris en considération et modélisé avec un paradigme probabiliste.* ◁

Plan du chapitre

4.1 Introduction

Dans le chapitre précédent, nous avons proposé plusieurs approches permettant de détecter des séries temporelles anormales dans un mode non-supervisé. A l'issue de cette tâche qui permet de mettre en évidence des séries ayant un profil anormal par rapport à la majorité, il serait judicieux de développer des outils intelligibles permettant d'expliquer l'anomalie de telles séries. Cela nous mène à étudier de près la détection d'anomalies contextuelle dans les séries temporelles. Cette tâche vise à détecter des observations anormales au sein de chacune de ces séries, qu'elle soit uni ou multi variée. A savoir que ces observations peuvent être normales dans un contexte temporel donné et anormales dans un autre comme le montre la Figure 4.1

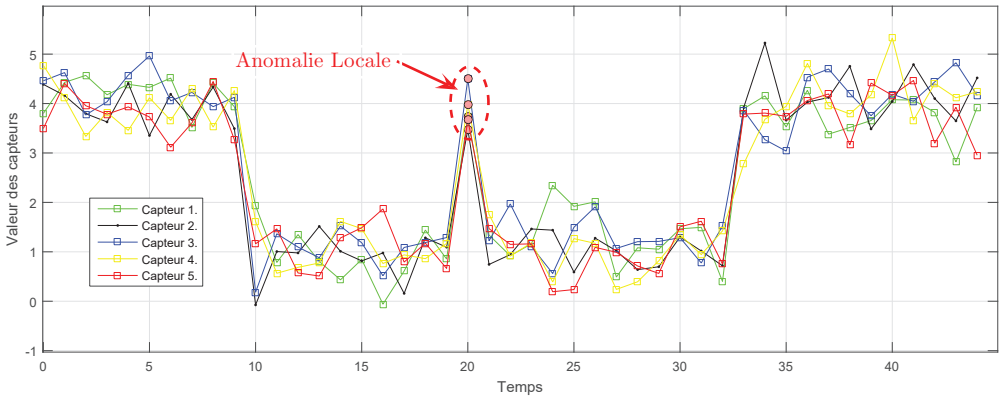


FIGURE 4.1 – Exemple d'une anomalie locale dans une série temporelle multivariée.

La majorité des approches que nous avons présentées dans l'état de l'art concernant la détection globale, peuvent être adaptées facilement à la détection contextuelle. Toutefois, les approches paramétriques à base de régression linéaire multiple [Basu07, Hill10] ou à base des SVR (machines à support pour la régression) [Ma03b] ont été particulièrement plus utilisées que les autres approches.

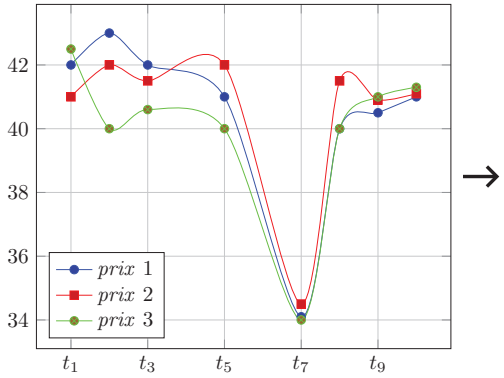
L'objectif de la détection contextuelle, est de cibler les observations anormales au sein d'une série en prenant en considération l'aspect temporel. Par exemple, dans la Figure 4.1, l'observation à l'instant $t = 20$ ne serait pas détectée comme anormale, si ses observations voisines dans le temps ne sont pas prises en considération. Plusieurs approches ont été proposées pour détecter les observations anormales au sein d'une série temporelle, notamment celles qui sont basées sur les modèles de prédiction. Toutefois, il est possible d'adapter la majorité des approches de la détection globale, comme celles présentées dans de l'état de l'art (à base de clustering, proximité, fenêtres) au problème de la détection contextuelle.

Les méthodes basées sur les modèles de prédiction mesurent l'écart entre la prédiction faite à un instant t et la vraie valeur observée au même instant. Si cet écart est assez important, l'observation est considérée comme anormale. Ces techniques diffèrent principalement par rapport au modèle de prédiction utilisé, par exemple, la valeur de la série à un instant t peut être prédite comme étant la médiane des valeurs contenues dans l'intervalle $[t-k, t+k]$ [1] [Basu07] ou comme étant la moyenne de toutes les instances appartenant au même cluster à l'instant t [Hill10]. D'autres techniques, utilisent des modèles de régression, des réseaux de neurones et des machines à support pour la régression (SVR) [Ma03b].

Contrairement aux méthodes citées ci-dessus, nous proposons dans ce chapitre une nouvelle approche de détection contextuelle non-supervisée. Notre proposition est basée sur la notion de la reconstruction des données et la détection d'anomalies par l'analyse des résidus [She11]. Nous présentons dans la section suivante les notations utilisées dans ce chapitre, et nous détaillons notre contribution dans les sections qui suivent.

4.2 Notations et définitions

Dans cette section, nous présentons les notations que nous allons utiliser tout au long de ce chapitre. Rappelons que l'objectif est de présenter un nouvel algorithme pour la détection d'anomalies au sein d'une série temporelle uni ou multi-variée par l'analyse des résidus.



	p_1	p_2	p_3
t_1	41	42	42.5
t_2	42	43	40
.	.	.	.
.	.	.	.
.	.	.	.
.	.	.	.
.	.	.	.
.	.	.	.
t_n	41	41.1	41.3

FIGURE 4.2 – Représentation matricielle de la série temporelle.

Soit T une série temporelle de taille N. Nous utilisons t_i pour designer les observations de la série T. Chaque observation t_i est un vecteur de taille p. Ainsi, notre série est

1. le paramètre k représente le degrée de voisinage dans le deux sens.

représentée sous forme d'une matrice de n lignes et p colonnes (i.e, $T \in R^{n \times p}$, cf. Figure 4.2).

Pour une matrice quelconque $A = (a_{ij}) \in R^{n \times p}$, nous désignons sa $i^{\text{ème}}$ ligne et $j^{\text{ème}}$ colonne par a^i et a_j, respectivement. Dans ce qui suit, nous définissons les différentes normes utilisées dans ce chapitre :

La norme ℓ_p d'un vecteur $x \in R^n$ is définie par :

$$\|x\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{\frac{1}{p}}. \tag{4.1}$$

La norme ℓ_0 d'un vecteur $x \in R^n$ est définie par :

$$\|x\|_0 = \sum_{i=1}^{n} |x_i|^0. \tag{4.2}$$

La norme Frobenius d'une matrice $A \in R^{n \times p}$ est définie par :

$$\|A\|_F = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{p} a_{ij}^2} = \sqrt{\sum_{i=1}^{n} \|a^i\|_2^2}. \tag{4.3}$$

La norme $\ell_{2,0}$ de la matrice A est définie par :

$$\|A\|_{2,0} = \sum_{i=1}^{n} \sqrt{\sum_{j=1}^{p} |a_{ij}|^0} = \sum_{i=1}^{n} \|a^i\|_0. \tag{4.4}$$

La norme $\ell_{2,1}$ de la matrice A est définie par :

$$\|A\|_{2,1} = \sum_{i=1}^{n} \sqrt{\sum_{j=1}^{p} a_{ij}^2} = \sum_{i=1}^{n} \|a^i\|_2. \tag{4.5}$$

4.3 Reconstruction des observations d'une série temporelle

L'idée de la reconstruction des observations d'une base d'apprentissage par quelques observations représentatives a été introduite par [Tang13]. Elle consiste à décrire toute instance comme étant une combinaisons linéaire de certaines instances pertinentes de la base. L'objectif est donc de calculer les coefficients de la combinaison linéaire pour chaque observation dans l'ensemble des données. Dans notre cas, ce problème peut se formuler comme la minimisation de l'erreur quadratique entre la série

temporelle T et sa reconstruction $W^\top T$ où $W \in \mathbb{R}^{n \times n}$ est la matrice des coefficients à déterminer :

$$\min_W \|T - W^\top T\|_F^2 + \alpha \|W\|_{2,0} \tag{4.6}$$

Chaque observation t_i est reconstruite par une combinaison linéaire de toutes les observations de la série T. Le vecteur colonne w_i de la matrice W contient les coefficients de cette combinaison linéaire. Ces coefficients reflètent la contribution de chaque observation dans la reconstruction de t_i. Par exemple, si $W_{ji} = 0$, cela veut dire que la série t_j ne contribue pas à la reconstruction de la série t_i. La version reconstruite de la série t_i est obtenue par $w_i T$. La contrainte de sparsité $\|W\|_{2,0}$ sur les lignes de W est utilisée pour restreindre le nombre des observations représentatives. Ainsi, le paramètre de régularisation α est employé pour contrôler le nombre d'observations à considérer comme représentatives dans T. Le but étant d'avoir une matrice W où tous les vecteurs lignes w^i soient nuls, sauf ceux qui correspondent aux séries temporelle représentatives.

Toutefois, le problème de minimisation dans l'équation (4.6) est difficile à cause de l'utilisation la norme $\ell_{2,0}$. Pour cela, nous proposons de relaxer le problème par l'utilisation de la norme $\ell_{2,1}$ qui représente l'enveloppe convexe de la norme $\ell_{2,0}$. Ainsi, le problème de minimisation ci-dessus est réécrit de la façon suivante :

$$\min_W \|T - W^\top T\|_F^2 + \alpha \|W\|_{2,1} \tag{4.7}$$

4.4 Analyse des résidus de la reconstruction

La détection d'anomalies par l'analyse des résidus a été initialement proposée par [She11] pour des problèmes de régression avec pénalités.

Dans notre cas, nous pouvons utiliser la matrice résiduelle $R = T - W^\top T - \Theta$ où Θ représente la matrice d'erreur aléatoire, supposée être normalement distribuée [She11], pour détecter les observations anormales. Dans un premier temps, nous supposons que les observations anormales sont assez différentes et s'écartent largement de l'ensemble des observations de la série temporelle T. En se basant sur cette hypothèse, les observations anormales ne devraient pas être bien représentées à l'issue de la phase de reconstruction. Ceci impliquerait une grande erreur de reconstruction, et cela peut être analysé via la matrice résiduelle R. Ainsi, une large norme ℓ_2 d'une ligne r^i de R indiquerait une déviation significative par rapport à la majorité des observations. Pour cette raison, nous intégrons la matrice R à notre fonction objective et le problème d'optimisation devient :

$$\min_{W,R} \|T - W^\top T - R\|_F^2 + \alpha \|W\|_{2,1} + \beta \|R\|_{2,1} \tag{4.8}$$

Partant du principe que les observations anormales sont minoritaires, nous proposons de rajouter une contrainte de sparsité sur la matrice R, représentée dans (4.8) par la norme $\ell_{2,1}$, de telle sort que la majorité des lignes soient nulles et uniquement quelques unes (qui correspondraient à anomalies) soient non nulles.

4.5 Modélisation de l'aspect temporel

Le problème de minimisation décrit ci-dessus, nous permet de détecter des observations anormales en exploitant la matrice résiduelle issue de la reconstruction. Cependant, l'aspect temporel et séquentiel des observations de la série ne sont pas pris en considération. En nous limitant uniquement à l'analyse des résidus sans prise en considération de l'aspect temporel, notre approche ne détecterait jamais l'observation anormale à l'instant $t = 20$ dans la Figure 4.1, parce que d'autres observations du même modèle sont présentes dans la série temporelle (e.g., les observations dans $[t_0, t_{10}]$ et $[t_{33}, t_{60}]$).

En considérant l'aspect temporel d'une série donnée, nous supposons que pour être normales, ses observations proches dans le temps devraient être similaires, ce qui se traduit par des résidus similaires à l'issue de la reconstruction. Ainsi, si l'observation t_i est proche dans le temps de l'observation t_j, la différence entre leurs résidus $\|r^i - r^j\|_2$ devrait être proche ou égale à zéro. En généralisant cette notion de proximité temporelle pour toutes les observations de la série, la somme des différences entre les résidus des observations proches devrait être minimale :

$$\frac{1}{2}\sum_{i,j}\left(\|r^i - r^j\|_2^2 \times \Delta(i,j)\right) \tag{4.9}$$

où $\Delta(i,j)$ est une fonction modélisant la dépendance temporelle entre les observations t_i et t_j, censée renvoyer une valeur réelle ≥ 0 reflétant la proximité entre les deux observations sur l'axe temporel. Plus cette valeur est importante, plus les observations t_i et t_j sont proches dans le temps. Si les deux observations sont assez éloignées sur l'axe temporel, la fonction Δ renvoi 0, ce qui implique que la différence entre les résidus de ces deux observations ne contribuerait pas à la minimisation de la quantité exprimée dans l'équation (4.9).

Pour une observation t_i, les valeurs renvoyées par $\Delta(i,.)$ devraient être décroissantes, plus on avance sur l'axe temporelle, plus ces valeurs sont proches de zéro. En effet, nous proposons de représenter la fonction $\Delta(.,.)$ via les probabilités de la loi de Poisson dans l'équation (4.10) :

$$P(k) = P(X = k) = \frac{\lambda^k}{k!}\,\mathrm{e}^{-\lambda} \tag{4.10}$$

Si on prend le cas de $\lambda = 1$ on aura :

$$\Delta(i,j) = P(k = |i-j|) = \frac{e^{-1}}{|i-j|!} \tag{4.11}$$

de telle sorte que plus on s'éloigne de la position i (dans les deux sens de l'axe temporel) plus la probabilité d'être proche de cette position décroit rapidement. Ainsi, l'équation (4.9) peut se réécrire de la façon suivante :

$$\begin{aligned}
\frac{1}{2}\sum_{i,j}\|r^i - r^j\|_2^2 \frac{e^{-1}}{|i-j|!} &= \frac{1}{2}\sum_{i,j}\|r^i\|_2^2 \frac{e^{-1}}{|i-j|!} + \|r^j\|_2^2 \frac{e^{-1}}{|i-j|!} - \sum_{i,j} < r^i, r^j > \frac{e^{-1}}{|i-j|!} \\
&= \frac{1}{2}\sum_{i}\|r^i\|_2^2 d_{ii} + \frac{1}{2}\sum_{i}\|r^i\|_2^2 d_{ii} - \sum_{i,j} < r^i, r^j > \frac{e^{-1}}{|i-j|!} \\
&= tr(R^\top DR - R^\top MR) = tr(R^\top (D-M)R) \\
&= tr(R^\top LR).
\end{aligned} \tag{4.12}$$

Où $M \in R^{n\times n}$ est une matrice symétrique modélisant l'aspect temporel de la série telle que :

$$m_{ij} = \begin{cases} \Delta(i,j) = \frac{e^{-1}}{|i-j|!}, & \text{si } i \neq j \\ 0, & \text{si } i = j\,. \end{cases} \tag{4.13}$$

D est une matrice diagonale tel que $d_{ii} = \sum_{j=1}^{n} m_{ij}$ et $L = (D - M)$ est la matrice Laplacienne.

En intégrant la modélisation temporelle (4.12) dans la fonction objective (4.8), nous obtenons la formulation finale de notre fonction que nous proposons pour détecter les anomalies contextuelles au sein d'une série multi-variée :

$$\overbrace{\min_{W,R}\|T - W^\top T - R\|_F^2 + \alpha\|W\|_{2,1}}^{\text{reconstruction des observations}} + \overbrace{\beta\|R\|_{2,1} + \gamma \underbrace{tr(R^\top LR)}_{\text{aspect temporel}}}^{\text{détection d'anomalies contextuelles}} \tag{4.14}$$

4.6 Approche de résolution

Dans cette section, nous détaillons dans un premier temps le modèle d'optimisation utilisé pour résoudre le problème de minimisation décrit dans l'équation (4.14). Nous présentons ensuite l'algorithme issu de la résolution avec l'analyse de sa convergence et sa complexité dans les sous-sections 4.6.2 et 4.6.3.

4.6.1 Modèle d'optimisation

La fonction objective présentée dans l'équation (4.14) n'est pas convexe par rapport à W et R, simultanément. De plus, la pénalisation via la norme $\ell_{2,1}$, rend cette fonction non-régulière. Ceci, nous mène donc à utiliser l'optimisation alternative sur W et R pour minimiser la fonction. Ce type d'optimisation consiste à fixer une variable et optimiser par rapport à l'autre. Par exemple, en fixant W, la fonction devient convexe par rapport à R et vice versa.

Optimisation par rapport à R :

On commence d'abord par fixer W et supprimer tous les termes indépendants de R. Ainsi, le problème de minimisation devient :

$$\min_R \Phi(R) = \|T - W^\top T - R\|_F^2 + \beta\|R\|_{2,1} + \lambda tr(R^\top LR) \tag{4.15}$$

En annulant la dérivée de $\Phi(R)$ par rapport à R, nous obtenons :

$$\frac{\partial}{\partial R}\Phi(R) = W^\top T - T + R + \beta D_R R + \lambda LR = 0 \tag{4.16}$$

où D_R est une matrice diagonale tel que ses éléments $d_{Rii} = \frac{1}{2\|r^i\|_2+\epsilon}$. Les matrices I et βD_R étant des matrices diagonales avec des valeurs positives, elles sont donc semi-définies positives. La matrice Laplacienne L est aussi semi-définie positive. Ainsi, la somme de ces trois matrices, $I + \beta D_R + \lambda L$, est aussi semi-définie positive. Cela implique que R possède une solution analytique :

$$R = (I + \beta D_R + \lambda L)^{-1}(T - W^\top T). \tag{4.17}$$

Optimisation par rapport à W :

De même pour W, en fixant la variable R et en supprimant tous les termes indépendants de W, le problème d'optimisation devient :

$$\min_W \Phi(W) = \|T - W^\top T - R\|_F^2 + \alpha\|W\|_{2,1} \tag{4.18}$$

En annulant la dérivée de $\Phi(W)$ par rapport à W, nous obtenons :

$$\frac{\partial}{\partial W}\Phi(W) = (TT^\top + \alpha D_W)W - TT^\top + TR^\top = 0 \tag{4.19}$$

où D_W est une matrice diagonale tel que $d_{Wii} = \frac{1}{2\|w^i\|_2+\epsilon}$. Les matrices $TT^\top$ et αD_W sont des semi-définies positives, ainsi que leur somme $\alpha D_W + TT^\top$. Cela implique que W possède aussi une solution analytique :

$$W = (TT^\top + \alpha D_W)^{-1}(TT^\top - TR^\top) \tag{4.20}$$

Par conséquent, nous pouvons résumer tous les développements ci-dessus dans l'algorithme 6. Nous l'appelons **LADOP** (pour *Local Anomaly Detection based On Poisson model*). Tout d'abord, on commence par initialiser les deux matrices D_R et D_W par la matrice d'identité et la matrice des résidus R par une approximation de (4.17), $(I + \beta D_R + \lambda L)^{-1}T$. Ensuite, on déclenche l'optimisation alternative. Ainsi, on fixe R pour mettre à jour W et vice versa dans un processus itératif jusqu'à ce que la fonction objectif dans l'équation (4.14) converge. Une fois la convergence atteinte, on calcule le score d'anormalité pour chaque observation t_i selon sa norme dans la matrice des résidus ($\|r^i\|_2$). Les observations t_i avec une large norme sont plus susceptibles d'être des anomalies. Enfin, on trie les t_i selon leur score dans un ordre décroissant.

Algorithme 6 : LADOP

Entrées : Matrice T de taille $n \times p$, les paramètres α, β et γ
Résultat : les points t_i de la série T triés selon leur score d'anormalité

Initialiser la matrice de dépendance temporelle M de taille $n \times n$, par $m_{ij} = \frac{e^{-1}}{|i-j|!}$
Calculer la matrice diagonale D par $d_{ii} = \sum_{j=1}^{n} m_{ij}$
Calculer la matrice Laplacienne $L = D - M$
Initialiser le compteur d'itération $h = 0$
Initialiser les matrices diagonales D_{R_h} et D_{W_h} par la matrice d'identité I
Initialiser $R_h = (I + \beta D_R + \gamma L)^{-1}T$
Répéter
  Calculer $W_{h+1} = (TT^\top + \alpha D_{W_h})^{-1}(TT^\top - TR_h^\top)$
  Calculer la matrice diagonale $D_{W_{h+1}}$ où la $i^{\text{ème}}$ diagonale est $\frac{1}{2\|w^i_{h+1}\|_2+\epsilon}$
  Calculer $R_{h+1} = (I + \beta D_{R_h} + \lambda L)^{-1}(T - W_{h+1}^\top T)$
  Calculer la matrice diagonale $D_{R_{h+1}}$ où la $i^{\text{ème}}$ diagonale est $\frac{1}{2\|r^i_{h+1}\|_2+\epsilon}$
  $h = h + 1$
Jusqu'à *Convergence*;
Calculer le score d'anormalité de chaque point t_i dans la série T par $\|r^i_h\|_2$
Trier les t_i selon leurs scores d'anormalité.

4.6.2 Analyse de la convergence

Dans cette partie, nous montrons que l'optimisation alternative proposée dans l'algorithme 6 permet de faire décroître la fonction objectif dans (4.14) à chaque itération de façon monotone.

Pour prouver la convergence de notre approche, nous suivons la démarche proposée par les auteurs dans [Nie10] et qui consiste à utiliser le *lemme* suivant :

Lemme 1. *Pour tout vecteur non-nul $x, y \in R^p$, l'inégalité suivante est vérifiée :*

$$\|x\|_2 - \frac{\|x\|_2^2}{2\|y\|_2} \leq \|y\|_2 - \frac{\|y\|_2^2}{2\|y\|_2} \tag{4.21}$$

Démonstration. Partant de l'inégalité évidente, $(\|x\|_2 - \|y\|_2)^2 \geq 0$, l'équation (4.21)

$$\begin{gathered}
\|x\|_2^2 + \|y\|_2^2 - 2\|x\|_2\|y\|_2 \geq 0 \\
\Rightarrow \\
2\|x\|_2\|y\|_2 - \|x\|_2^2 \leq \|y\|_2^2 \\
\Rightarrow \\
-\frac{\|x\|_2^2}{2} + \|x\|_2\|y\|_2 \leq \frac{\|y\|_2^2}{2} \\
\Rightarrow \\
-\frac{\|x\|_2^2}{2\|y\|_2} + \|x\|_2 \leq \frac{\|y\|_2^2}{2\|y\|_2} \\
\Rightarrow \\
-\frac{\|x\|_2^2}{2\|y\|_2} + \|x\|_2 \leq \frac{2\|y\|_2^2 - \|y\|_2^2}{2\|y\|_2} \\
\Rightarrow \\
\|x\|_2 - \frac{\|x\|_2^2}{2\|y\|_2} \leq \|y\|_2 - \frac{\|y\|_2^2}{2\|y\|_2}.
\end{gathered} \tag{4.22}$$

□

Théorème 4. *Le processus d'optimisation alternative présenté dans l'algorithme 6 permet de faire décroitre la fonction objectif (4.14) de façon monotone à chaque itération.*

Démonstration. A l'itération h, on fixe R_h pour obtenir W_{h+1} par l'équation (4.20). En obtenant W_{h+1}, on peut calculer R_{h+1} par l'équation (4.17). Ensuite, on alterne entre ces deux opérations jusqu'à la convergence. Pour prouver que la minimisation de la fonction objectif Φ par rapport à W et R dans l'équation (4.15) converge, il suffit de montrer que :

$$\Phi(W_{h+1}, R_{h+1}) \leq \Phi(W_{h+1}, R_h) \leq \Phi(W_h, R_h). \tag{4.23}$$

— $\Phi(W_{h+1}, R_h) \leq \Phi(W_h, R_h)$:
Après avoir fixé R_h, on obtient W_{h+1} par l'équation (4.20), ce qui implique que W_{h+1} devienne le minimiseur de Φ comme indiqué dans l'équation (4.18) :

$$W_{h+1} = \underset{W}{\operatorname{argmin}} \|T - W^\top T - R_h\|_F^2 + \alpha\|W\|_{2,1} \tag{4.24}$$

La norme $\ell_{2,1}$ de W n'étant pas lisse, on peut relaxer le problème par l'utilisation des traces des matrices $(W_{h+1}D_W W_{h+1})$ et $(W_h D_W W_h)$, ce qui implique que l'inégalité suivante soit vérifiée :

$$\begin{array}{c} \|T - W_{h+1}^\top T - R_t\|_F^2 + \alpha tr(W_{h+1}D_W W_{h+1}) \\ \leq \\ \|T - W_h^\top T - R_h\|_F^2 + \alpha tr(W_h D_W W_h) \end{array} \tag{4.25}$$

D'autre part, selon le lemme 1, pour chaque vecteur ligne on a :

$$\begin{array}{c} \|w_{h+1}^i\|_2 - \dfrac{\|w_{h+1}^i\|_2^2}{2\|w_h^i\|_2} \leq \|w_h^i\|_2 - \dfrac{\|w_h^i\|_2^2}{2\|w_h^i\|_2} \\ \Rightarrow \\ \displaystyle\sum_{i=1}^{n}\left(\|w_{h+1}^i\|_2 - \dfrac{\|w_{h+1}^i\|_2^2}{2\|w_h^i\|_2}\right) \leq \sum_{i=1}^{n}\left(\|w_h^i\|_2 - \dfrac{\|w_h^i\|_2^2}{2\|w_h^i\|_2}\right) \\ \Rightarrow \\ \left(\|W_{h+1}\|_{2,1} - \displaystyle\sum_{i=1}^{n}\dfrac{\|w_{h+1}^i\|_2^2}{2\|w_h^i\|_2}\right) \leq \left(\|W_h\|_{2,1} - \sum_{i=1}^{n}\dfrac{\|w_h^i\|_2^2}{2\|w_h^i\|_2}\right) \end{array} \tag{4.26}$$

A partir de l'équation (4.25), on peut déduire l'inégalité suivante :

$$\begin{array}{c} \|T - W_{h+1}^\top T - R_h\|_F^2 + \alpha\|W_{h+1}\|_{2,1} \\ - \\ \alpha\left(\|W_{h+1}\|_{2,1} - \displaystyle\sum_{i=1}^{n}\dfrac{\|{w^i}_{h+1}\|_2^2}{2\|{w^i}_h\|_2}\right) \\ \leq \\ \|T - W_h^\top T - R_h\|_F^2 + \alpha\|W_h\|_{2,1}. \\ - \\ \alpha\left(\|W_h\|_{2,1} - \displaystyle\sum_{i=1}^{n}\dfrac{\|{w^i}_h\|_2^2}{2\|{w^i}_h\|_2}\right) \end{array} \tag{4.27}$$

Finalement, en intégrant l'inégalité (4.26) dans l'équation (4.27), on obtient :

$$\|T - W_{h+1}^\top T - R_h\|_F^2 + \alpha\|W_{h+1}\|_{2,1} \leq \|T - W_h^\top T - R_h\|_F^2 + \alpha\|W_h\|_{2,1}. \tag{4.28}$$

Ce qui implique que :

$$\Phi(W_{h+1}, R_h) \leq \Phi(W_h, R_h) \tag{4.29}$$

— $\Phi(W_{h+1}, R_{h+1}) \leq \Phi(W_h + 1, R_h)$:

En obtenant W_{h+1}, on peut mettre à jour R_h pour obtenir R_{h+1} via l'équation (4.17), ce qui implique que R_{h+1} devienne le minimiseur de Φ comme indiqué dans l'équation (4.15) :

$$R_{h+1} = \underset{R}{\operatorname{argmin}} \|T - W_{h+1}^{\top} T - R\|_F^2 + \beta \|R\|_{2,1} + \lambda tr(R^{\top} L R) \tag{4.30}$$

Ce qui implique l'inégalité suivante :

$$\begin{gathered} \|T - W_{h+1}^{\top} T - R_{h+1}\|_F^2 + \lambda tr(R_{h+1}^{\top} L R_{h+1}) + \beta tr(R_{h+1} D_R R_{h+1}) \\ \leq \\ \|T - W_{h+1}^{\top} T - R_h\|_F^2 + \lambda tr(R_h^{\top} L R_h) + \beta tr(R_h D_R R_h) \end{gathered} \tag{4.31}$$

Et ce qui est équivalent à :

$$\begin{gathered} \|T - W_{h+1}^{\top} T - R_{h+1}\|_F^2 + \lambda tr(R_{h+1}^{\top} L R_{h+1}) \\ + \\ \beta \|R_{h+1}\|_{2,1} - \beta \left(\|R_{h+1}\|_{2,1} - \sum_{i=1}^{n} \frac{\|{r^i}_{h+1}\|_2^2}{2\|{r^i}_t\|_2} \right) \\ \leq \\ \|T - W_{h+1}^{\top} T - R_h\|_F^2 + \lambda tr(R_h^{\top} L R_h) \\ + \\ \beta \|R_h\|_{2,1} - \beta \left(\|R_h\|_{2,1} - \sum_{i=1}^{n} \frac{\|{r^i}_h\|_2^2}{2\|{r^i}_h\|_2} \right) \end{gathered} \tag{4.32}$$

Finalement, en intégrant l'inégalité du Lemme (1),

$$\|R_{h+1}\|_{2,1} - \sum_{i=1}^{n} \frac{\|{r^i}_{h+1}\|_2^2}{2\|{r^i}_h\|_2} \leq \|R_h\|_{2,1} - \sum_{i=1}^{n} \frac{\|{r^i}_h\|_2^2}{2\|{r^i}_h\|_2},$$

dans l'équation (4.32) on obtient :

$$\begin{gathered} \|T - W_{h+1}^{\top} T - R_{h+1}\|_F^2 + \lambda tr(R_{h+1}^{\top} L R_{h+1}) + \beta \|R_{h+1}\|_{2,1} \\ \leq \\ \|T - W_{h+1}^{\top} T - R_h\|_F^2 + \lambda tr(R_h^{\top} L R_h) + \beta \|R_h\|_{2,1} \end{gathered} \tag{4.33}$$

Ce qui implique que :

$$\Phi(W_{h+1}, R_{h+1}) \leq \Phi(W_{h+1}, R_h) \tag{4.34}$$

Ainsi, à partir des équations (4.34) et (4.29) on peut déduire que l'inégalité présentée dans (4.23) est vérifiée :

$$\Phi(W_{h+1}, R_{h+1}) \leq \Phi(W_{h+1}, R_h) \leq \Phi(W_h, R_h).$$

Ce qui prouve que le processus d'optimisation alternative présenté dans l'algorithme (6) fait décroitre la fonction objectif à chaque itération et finit par converger.

□

4.6.3 Analyse de la Complexité

Le but de cette section est d'analyser la complexité temporelle de l'algorithme proposé. En effet, la mise à jour des deux matrices W, et R représente les deux opérations les plus coûteuses dans l'algorithme à cause des inversions des matrices.

La mise à jour la matrice W, requiert $O(n^3)$ opérations pour inverser la matrice $(TT^\top + \alpha D_W)$. De même pour la mise à jour de R, on a besoin de $O(n^3)$ pour inverser la matrice $(I + \beta D_R + \lambda L)$. Toutefois, concernant la mise à jour de R et vu que dans notre cas le nombre de dimensions p est inférieur à la taille de la série n, l'inversion de la matrice $(I + \beta D_R + \lambda L)$ peut être évitée en résolvant le système d'équations linéaires déduit de (4.17) : $(I + \beta D_R + \lambda L)R = (T - W^\top T)$ où l'algorithme nécessite uniquement $O(n^2 p)$ opérations.

Par ailleurs, calculer le score d'anormalité pour chaque t_i, revient à calculer la norme ℓ_2 de toutes les lignes de la matrice résiduelle R ($\|r^i\|_2$). Ce qui nécessite $O(n \times p)$ opérations pour calculer les scores pour toutes les observations de la série temporelle T. La dernière étape consiste à trier les observations en fonction de leur score d'anormalité en effectuant $n \log n$ opérations.

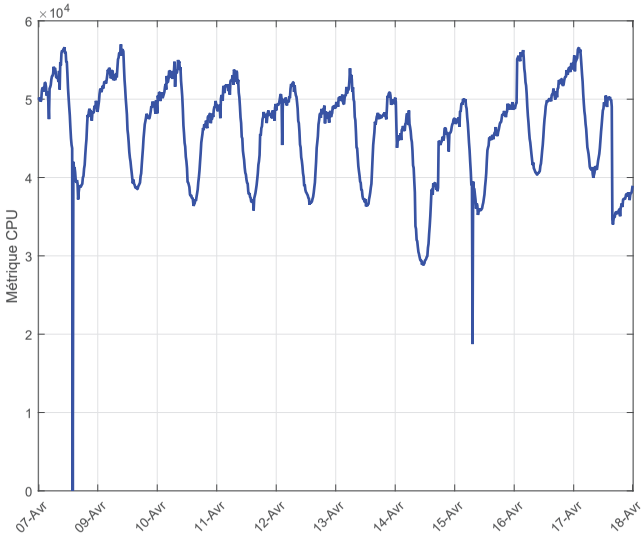
Par conséquent, la complexité temporelle du **LADOP** est de l'ordre de :

$$[(O(n^3) + O(n^2 * p)) \times h] + O(n \times p) + O(n \log n) \tag{4.35}$$

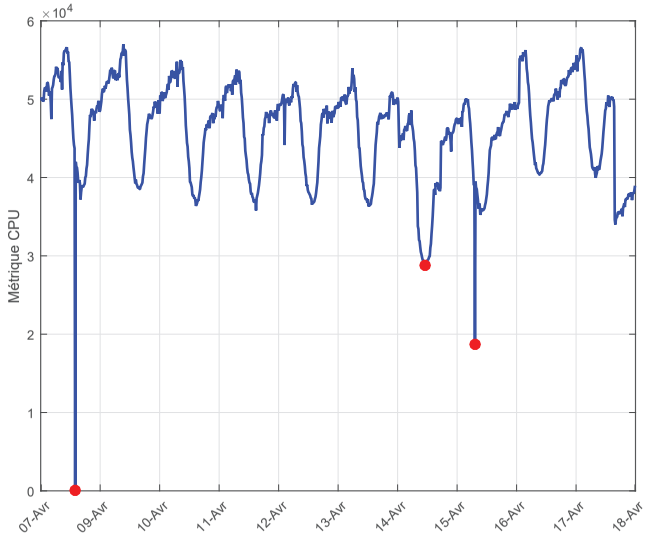
où h représente le nombre d'itérations.

4.7 Validation expérimentale

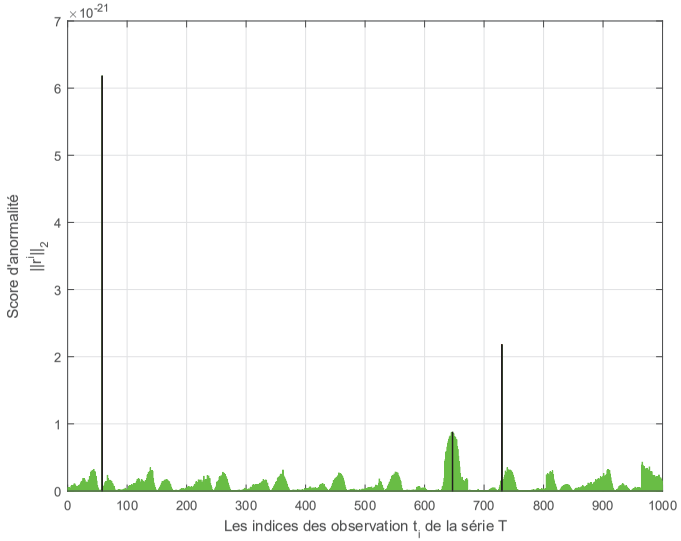
Pour évaluer l'efficacité de notre approche, nous avons mené une expérimentation sur un jeu de données récemment publié par les chercheurs de Yahoo ! [Laptev15] dans le but de disposer d'un benchemark riche pour l'évaluation des différentes techniques de détection. La Figure 4.3(a) présente une série temporelle uni-variée de 1000 observations. Chaque observation correspond à une métrique liée à l'utilisation des différents CPU des serveurs de Yahoo !. L'application de l'algorithme a permis de mettre en évidence des observations déviantes par rapport à la majorité des métriques. Ces observations sont visibles en couleur rouge dans la Figure 4.3(b), et correspondent aux résidus les plus important par rapport à toutes les observations de la série temporelle (cf. 4.3(c)). Enfin, dans la Figure 4.3(d), la courbe de convergence est clairement décroissante et converge au bout de quelques itérations.



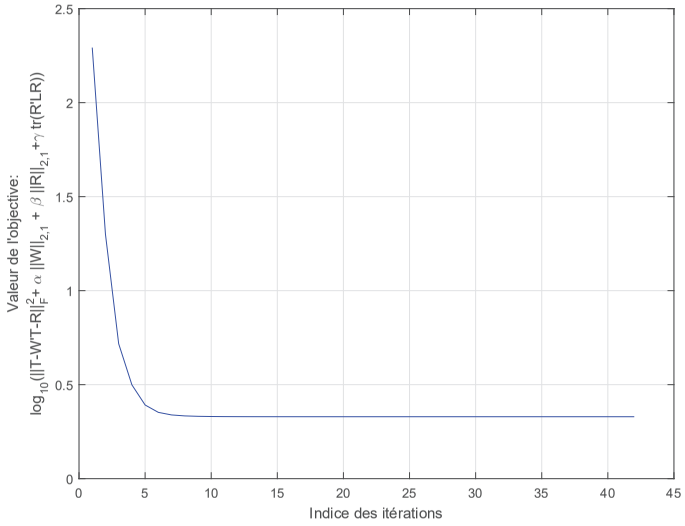
(a) Série temporelle d'une métrique CPU des serveurs Yahoo!. La figure montre la variation de 1000 valeurs prises chaque 15 min durant 12 jours.



(b) Détection des métriques anormales.



(c) Distribution des scores des observations de la série temporelle.



(d) Courbe de convergence.

FIGURE 4.3 – Validation expérimentale de notre approche LADOP sur une série temporelle de Yahoo!.

4.8 Conclusion

Dans ce chapitre, nous avons proposé une nouvelle approche pour détecter les observations anormales au sein d'une série temporelle uni ou multi-variée. L'approche proposée se base sur la notion de la reconstruction des observations de la série d'une part et de l'analyse des résidus d'autre part. Chaque observation est reconstruite via une combinaison linéaire de toutes les observations de la série où uniquement quelques-unes (dites représentatives) contribuent réellement à la reconstruction. En se basant sur les résidus de la reconstruction, et sur la modélisation probabiliste de l'aspect temporel, une observation est jugée anormale si son résidu est important vis à vis de ses observations les plus proches.

Outre l'aspect fondamental que nous avons développé, nous avons validé notre approche sur un jeu de données publique avec des résultats encourageants, qui méritent d'être confrontés à des méthodes représentatives de l'état de l'art. Ainsi, un protocole expérimental est en cours de développement avec des scénarios de comparaisons à l'appui.

5

Application aux données sur les pneumatiques

▷ *Dans ce chapitre, nous présentons le framework de détection d'anomalies que nous avons développé pour notre partenaire industriel LIZEO ONLINE MEDIA GROUP. Cette détection concerne particulièrement les prix aberrants sur les pneumatique. En effet, le but consiste à détecter pour un pneu mis en vente par plusieurs sites web marchands, les séries temporelles de son prix qui sont anormales par rapport à l'ensemble des autres prix. Nous proposons un POC (Proof of Concept) de détection, composé de deux modules, a) une chaine de pré-traitement, de nettoyage et de transformation des données permettant de les rendre exploitables et b) un module de détection comprenant l'ensemble des approches que nous avons décrites dans les chapitres précédents. Étant donnée la nature massive des données, le framework proposé a été implémenté et déployé dans un environnement distribué (Spark/Scala) de telle sorte qu'il puisse passer à l'échelle. Les résultats des expérimentations sont assez encourageants et ont permis de mettre en évidence certaines défaillances dans le système de collecte des données de l'entreprise.* ◁

Plan du chapitre

5.1 Introduction

Outre l'aspect théorique abordé dans les chapitres précédents, il est toujours intéressant de confronter les approchées développées à des données réelles, surtout quand celles-ci sont particulièrement complexes, dynamiques et volumineuses. Pour ce faire, nous proposons de décortiquer le framework évoqué ci-dessus en plusieurs étapes. Tout d'abord, une description de l'entreprise LIZEO est introduite dans la section 5.2. Ensuite, dans la section 5.3, nous présentons le framework de détection d'anomalies que nous avons mis en place et qui se compose de deux modules principaux, a) le module de pré-traitement et de transformation des données et b) le module de détection qui comprend l'ensemble des approches proposées dans cette thèse. Nous discutons aussi en détail des données de l'entreprise et de leur nature temporelle et massive 5.3.1, de la chaine de pré-traitement des données et de l'implémentation de nos différentes approches (5.3.2). Nous abordons ainsi l'écosystème *Big Data* dans lequel évolue l'ensemble des algorithmes, et nous discutons des outils que nous avons utilisés pour les implémenter, à savoir l'environnement de calcul distribué Spark et le paradigme fonctionnel à travers le langage Scala (5.3.3). Dans la section 5.4, nous présentons les résultats de la détection d'anomalies sur certains pneus avant de conclure dans (5.5)

5.2 L'entreprise

Le métier de *Lizéo Online Media Group* (LOMG) est de développer des outils issus des nouvelles technologies de l'information et de la communication qui permettent de comprendre et d'anticiper le fonctionnement du marché du pneumatique. En effet, l'activité de LOMG repose sur sa capacité à fournir à ses clients des données issues principalement d'Internet sur les pneumatiques. Actuellement le marché couvert par la so-

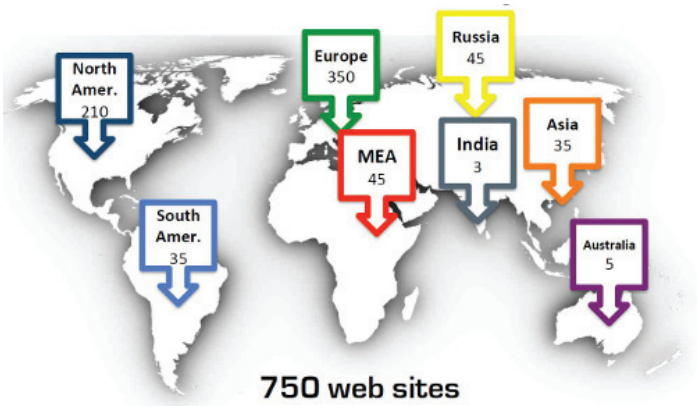


FIGURE 5.1 – Une cartographie de l'ensemble des sites web aspirés par LIZEO.

ciété (Europe, Russie, Amérique du nord, Brésil et Chine) en pneumatique tourisme, camionnette, poids-lourd, motocyclette est constitué de plus de 510 marques qui proposent plus de 220000 pneus différents. Chacun de ces pneus est caractérisé par sa boîte dimensionnelle (cinq caractéristiques) ainsi que par d'autres paramètres telles que ses conditions d'utilisation et ses performances. L'entreprise analyse quotidiennement 750 (643 en 2016) sites web de vente en ligne de pneus dans le monde dont 653 (554 en 2016) en Tourisme/Camionnette, 39 (38 en 2016) en Poids lourds et 58 (51 en 2016) en deux-roues.

5.3 Cadre applicatif

Parmi les différentes solutions business que propose LIZEO à ses clients (dans une vision B2B, manufacturiers, distributeurs ou sites-web marchands), nous nous intéressons tout particulièrement à l'outil qui consiste à fournir aux clients la variation des prix de chaque pneu à travers tous les sites web marchands. Cet outil permet aux clients de suivre l'évolution des prix mais surtout de se positionner par rapport aux concurrents. Pour ce faire, LIZEO utilise des collecteurs d'informations (appelés *crawler* en anglais) afin d'extraire quotidiennement des informations concernant les différents pneus à partir de tous les sites web marchands. Ainsi, pour un pneu donné, le crawler permet d'extraire tous les prix de ce pneu à partir de tous les sites web le mettant en vente. En effet, les prix sont récupérées quotidiennement, ce qui fait que pour chaque pneu, nous avons autant de séries temporelles de prix que de sites-web marchands. Toutefois, des erreurs de matching sont récurrentes dans la phase de crawling, comme par exemple matcher une série temporelle de prix avec le mauvais pneu (e.g. ; une série temporelle de prix d'un pneu "Michelin" matchée par erreur à un pneu "Pirelli"). Ainsi, les données fournies aux clients pourraient être erronées et fausseraient in fine l'analyse et l'intepretation. Pour pallier les différents problèmes liés à cette phase, nous avons proposé un framework (cf. Figure5.2) qui permet de répondre à cette problématique en utilisant les différentes approches que nous avons conçues pour la détection globale.

Le problème peut se formuler ainsi : soit $P = \{p_1, \ldots, p_d\}$ l'ensemble des pneus répertoriés dans le référentiel de l'entreprise, et $S = \{s_1, \ldots, s_r\}$ l'ensemble des sites web spécialisés dans le marché des pneumatiques. Pour un pneu donné p_i et à raison d'un prix par jour, la base des séries temporelles $T = \{t_1, \ldots, t_r\}$ représente les séries temporelles de prix du pneu p_i par rapport à chaque site web marchand. Ainsi, la série temporelle t_i représente la variation du prix de vente du pneu p_i par le site s_i. Étant donné l'ensemble des séries temporelles T d'un pneu p_i, le but est de détecter les séries anormales par rapport à la majorité des séries dans T.

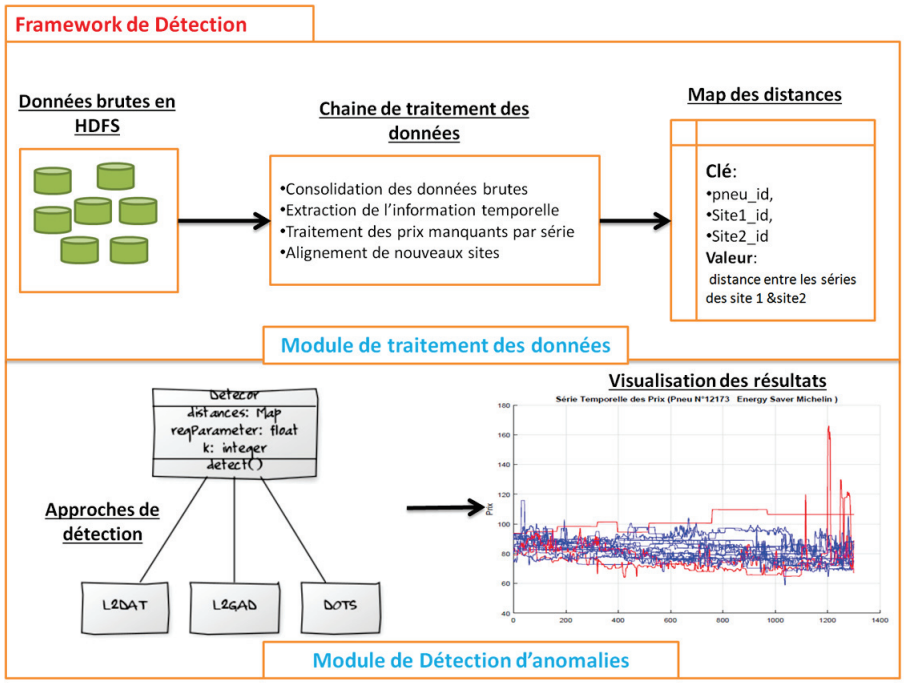


FIGURE 5.2 – Framework de détection des séries temporelles des prix anormales.

5.3.1 Description des données des pneumatiques

Les données concernant les prix des pneus sont collectées quotidiennement via les différents crawlers. Un échantillon des données prix est représenté dans la Table 5.1.

TYREID	WEBSITEID	COUNTRYID	PRICEYEAR	PRICEMONTH	PRICEDAY	AVGPRICE
12173	767	5	2015	4	22	77.27
12396	767	5	2015	1	30	65.84
12396	767	5	2015	4	7	64.05
12396	767	5	2015	8	20	66.68
12396	767	5	2015	9	23	63.47
12173	767	5	2014	1	2	83.93
12173	767	5	2017	3	21	76.46
12173	767	5	2014	2	9	88.0
12396	767	5	2014	3	15	69.03

TABLE 5.1 – Le schéma des données brutes à analyser.

Les informations concernant les prix sont décrites via les variables suivantes :

Variables	Description
TyreID	L'identifiant d'un pneu.
TyreID	L'identifiant d'un site web marchand.
countryID	L'identifiant du pays d'un site web.
PriceYear	L'année de l'extraction du prix.
PriceMonth	Le mois de l'extraction du prix.
PriceDay	Le jour de l'extraction du prix.
AvgPrice	Le prix de vente d'un pneu par un site web dans une date précise. A noter que certaines sites mettent à jours les prix plusieurs fois par jour d'où l'utilisation d'un prix moyen.

TABLE 5.2 – Description des variables du jeu de données.

Toutefois, les données collectées ne sont pas directement exploitables pour plusieurs raisons, notamment les valeurs manquantes et l'émergence de nouveaux sites. En effet, les sites web aspirés par LZIEO changement fréquemment de template pour bloquer les *crawlers*. En faisant ainsi, les collecteurs manquent souvent des prix, ce qui implique un nombre important de valeurs manquantes dans la base. L'autre problème, c'est l'émergence de nouveaux sites web. En effet, les séries temporelles des nouveaux sites web ont tellement peu de prix qu'il est insignifiant de les comparer sans un pré-traitement préalable, avec les séries temporelles des sites anciens. Pour pallier aux problèmes décrits ci-dessus, nous proposons une chaine de pré-traitement permettant de rendre les données exploitables pour l'analyse et la détection.

5.3.2 Module de traitement et de transformation des données

Nous proposons dans cette partie, une chaine de traitements des données brutes qui permet de rendre les données de l'entreprise exploitable pour la détection. La chaine que nous proposons permet de a) transformer les données brutes en séries temporelles de prix, b) traiter les prix manquants dans ces séries via plusieurs techniques de modélisation de séries temporelles, c) décentraliser le calcul des distances entre les séries afin de réduire la complexité temporelle des approches de détection globale et d) traiter de gros volumes de données dans un environnement distribué. La chaine proposée est donc constituée de plusieurs étapes. La première consiste à récupérer les données brutes (présentées dans la Table 5.1) et de les agréger et grouper par pneu et par site web. Ensuite, les prix de chaque pneu par rapport à chaque site web sont collectés sous forme de séries temporelles comme le montre la Table 5.3.

L'étape suivante consiste à réarranger les séries par dates et à traiter les prix manquants dans chaque série temporelle (Table 5.4). Plusieurs techniques de modélisation comme les modèles des moyennes mobiles ou auto-régressifs ont été implémentés dans le framework. De plus, nous avons rajouté au framework une technique simple consistant à affecter la médiane des prix voisins au prix manquants.

TYREID	WEBSITEID	COUNTRYID	PRICEYEAR	PRICEMONTH	PRICEDAY	AVGPRICE	DATE
12173	767	5	2015	4	22	77.27	2015-04-22
12396	767	5	2015	1	30	65.84	2015-01-30
12396	767	5	2015	4	7	64.05	2015-04-07
12396	767	5	2015	8	20	66.68	2015-08-20
12396	767	5	2015	9	23	63.47	2015-09-23
12173	767	5	2014	1	2	83.93	2014-01-02
12173	767	5	2017	3	21	76.46	2017-03-21
12173	767	5	2014	2	9	88.0	2014-02-09
12396	767	5	2014	3	15	69.03	2014-03-15

TABLE 5.3 – Traitement de l'information temporelle à partir des données brutes.

TYREID	WEBSITEID	COUNTRYID	SÉRIES TEMPORELLE
12173	366	178	$\{(89.99, 2011-), \ldots\}$
12396	46	4	$\{(92.02, 2009-), \ldots\}$
12396	226	200	$\{(74.20, 2011-), \ldots\}$
12396	987	67	$\{(88.51, 2015-), \ldots\}$
12173	628	67	$\{(135.2, 2012-), \ldots\}$
12173	771	2	$\{(84.10, 2013-), \ldots\}$
12173	859	67	$\{(128.2, 2014-), \ldots\}$

TABLE 5.4 – Réarrangement des séries temporelles et traitement des prix manquants.

La dernière étape consiste à calculer les distances entre chaque paire de séries temporelles d'un pneu et de les stocker dans une map après les avoir groupées par paire de sites web comme le montre la Table 5.5. En effet, la nature massive des données et le besoin de détection en mode instantané, nous a mené à décentraliser le calcul des distances des approches proposées afin de réduire la complexité temporelle.

TYREID	WEBSITES	COUNTRYID1	COUNTRYID2	SÉRIE TEMPORELLE 1	SÉRIE TEMPORELLE 2
12173	(366,915)	178	204	$\{(89.99, 2011-), \ldots\}$	$\{(97.71, 2014-), \ldots\}$
12173	(366,896)	178	1	$\{(89.99, 2011-), \ldots\}$	$\{(83.60, 2014-), \ldots\}$
12173	(366,581)	178	181	$\{(89.99, 2011-), \ldots\}$	$\{(105.5, 2013-), \ldots\}$
12173	(366,367)	178	178	$\{(89.99, 2011-), \ldots\}$	$\{(82.03, 2011-), \ldots\}$
12396	(366,670)	178	199	$\{(89.99, 2011-), \ldots\}$	$\{(69.91, 2013-), \ldots\}$

TABLE 5.5 – Groupement des séries temporelles par pneu et par couple de sites web.

En effet, comme nous l'avons montré dans le chapitre de la détection globale (cf. section 3), la complexité temporelle de nos approches se composent principalement de deux termes :

$$\overbrace{O((n(n-1)/2)L^2)}^{calul\ de\ distance} + \overbrace{O(n \times max(hk, \log n))}^{optimlisation}$$

Le premier terme représente le nombre maximum d'opérations nécessaires pour le calcul de toutes les distances entre toutes les séries temporelles. Le deuxième terme décrit la complexité temporelle requise pour la partie l'optimisation itérative de l'approche.

Pour réduire la complexité des approches, nous proposons de calculer les distances en amont de la phase de détection. Ainsi, ces distances peuvent être calculées indépendamment de l'approche. La dernière étape consiste à calculer les distances (e.g., DTW) entre les séries temporelles de chaque pneu et de les stocker dans une *map*. La clé de cette *map* est composée de l'identifiant du pneu et des identifiants de deux sites web. La valeur de chaque clé dans la *map* représente la distance (e.g., DTW) entre les séries de prix des deux sites web pour le pneu en question. La Table 5.6 montre le structure finale de la map des distances.

TYREID	WEBSITES	DTW_DISTANCE
12173	(366,915)	178
12173	(366,896)	DIST
12173	(366,581)	DIST
12173	(366,367)	DIST
12396	(366,670)	DIST

TABLE 5.6 – La Map des distances entre les séries temporelles.

En effet, en décentralisant le calcul des distances, la complexité de détection se réduit uniquement au nombre d'opérations nécessaires pour la phase d'optimisation. Ainsi, étant donnée une map de distances, la complexité de la détection est égale à :

$$O(n \times max(hk, \log n))$$

La Figure 5.3 représente succinctement l'ensemble des traitements effectués par la chaine proposée.

5.3.3 Module de détection et environnement technique

La nature massive des données dont nous disposons, nous a mené à concevoir un framework capable de traiter un nombre important de données en se basant sur des technologies spécialisées en traitement des gros volumes de données. Rappelons que le marché couvert par LIZEO est constitué de plus de 510 marques proposant plus de 212.865 pneus différents vendus par 1382 sites web. Cette quantité massives des données représente exactement 16.258.168 séries temporelles ayant chacune en moyenne une longueur sur 1640 prix différents.

Nous avons utilisé le framework Spark [Zaharia10] pour paralléliser l'ensemble des opérations de notre module de traitement des données, que ce soit pour les transformations, l'extraction de l'information temporelle ou le calcul de distances. Les approches

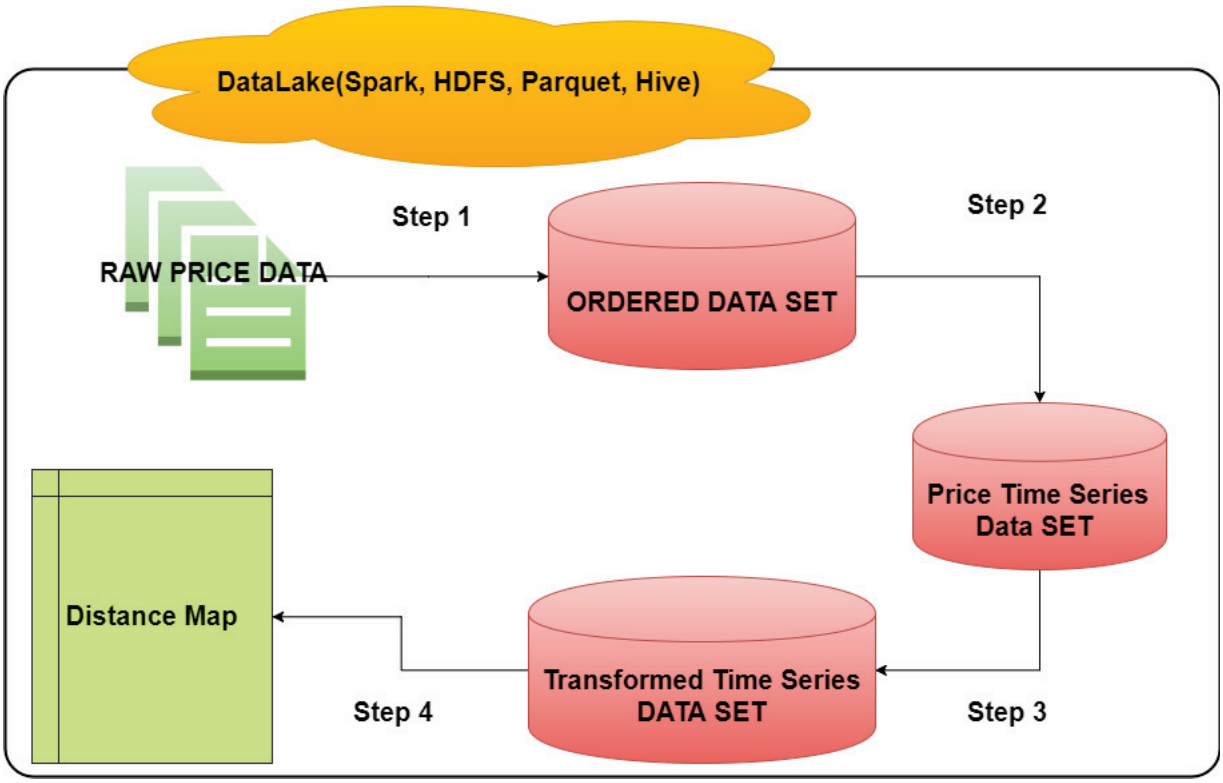


FIGURE 5.3 – Chaine de traitement et de transformation des données prix.

de détection ainsi que les différents scripts ont été implémentés en Scala [1] qui est un langage à la fois fonctionnel (ce qui est pratique dans un environnement distribué et pour le calcul concurrentiel) et orienté objet.

Spark est un framework open source de calcul en clusters. Il permet de faire de l'analyse de données rapide, à la fois en écriture et en exécution. Il a été développé à l'Université de Berkeley [Zaharia10]. Comme le montre la Figure 5.4, une application Spark est un programme pilote (driver) qui exécute la fonction principale et diverses opérations parallèles sur un cluster en s'appuyant sur une structure de donnée RDD (Resilient Distributed Dataset). La structure RDD est une collection d'éléments partitionnés sur les nœuds du cluster fonctionnant en parallèle. Elle est capable de récupérer automatiquement les défaillances de nœuds. Ce type de structure supporte deux types d'opération a) les *transformations* qui créent un nouvel ensemble de données à partir d'un existant et b) les *actions* qui renvoient une valeur au programme *pilote* après calcul sur l'ensemble de données. Le framework Spark est écrit en Scala et persiste un jeu de

1. `https://www.scala-lang.org/`

données en mémoire à travers toutes les opérations.

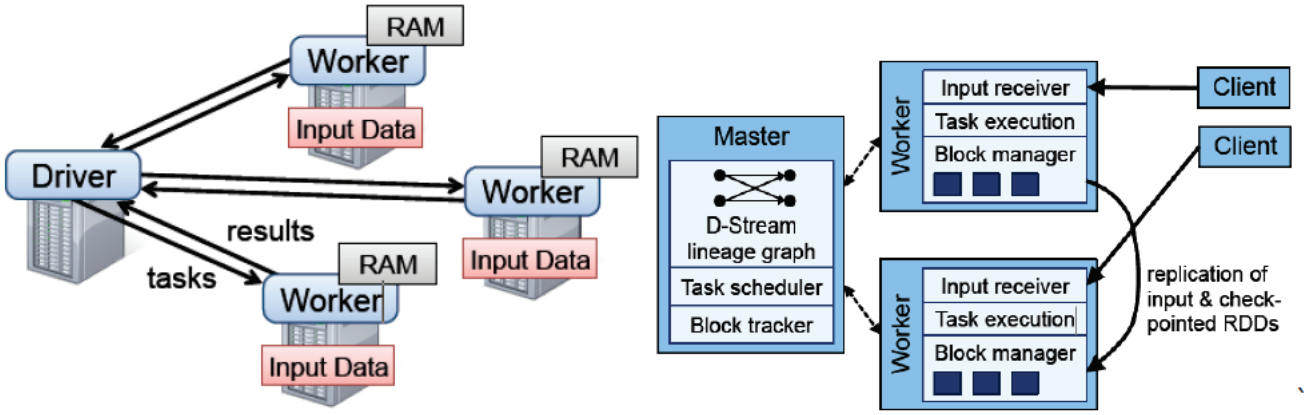


FIGURE 5.4 – Le fonctionnement d'une application implémentée en Spark. ([Zaharia10])

En effet, nous avons implémenté d'une façon fonctionnelle les algorithmes de détections ainsi que l'ensemble des opérations de transformation pour mieux bénéficier de la partie optimisation qu'offre Spark concernant le calcul parallèle. Le choix du langage Scala a été fait pour son aspect à la fois fonctionnel et orienté objet. En effet, l'aspect fonctionnel nous permet de décrire à l'aide de fonctions élémentaires (*map*,*reduce*, *filter*,...) nos algorithmes de détection et de bénéficier de la stratégie d'optimisation offerte par Spark. Quant à l'aspect orienté objet du langage, l'intérêt est d'avoir des outils efficaces, comme les patrons de conception, pour exprimer les liens de dépendances et les relations entre nos différentes approches de détection globale. Dans notre cas, nous avons utilisé le patron de conception *stratégie* pour modéliser le module de détection. Ce patron permet de définir une famille d'algorithmes (e.g., nos trois approches de détection), d'encapsuler chacun d'eux en tant qu'objet, et de les rendre interchangeables. Concrètement cela consiste à exprimer via une interface et d'une façon abstraite, l'ensemble des fonctionnalités communes des différents algorithmes, et d'en proposer les différentes implémentations dans des classes concrètes.

Dans notre cas, l'interface *Detector* décrit en détail le contrat que doit remplir une approche de détection (cf. Figure 5.5), à savoir prendre en entrée, un paramètre de régularisation (α pour L2GAD, et λ pour DOTS et ℓ_2-DAT), le nombre de profils k dans la base ainsi que la Map de distances entre les séries temporelles du pneu concerné. La fonction *detect()* permet de lancer le processus de détection selon l'approche choisie. Ainsi, le client *Spark Driver* peut switcher d'une approche à une autre à la volée.

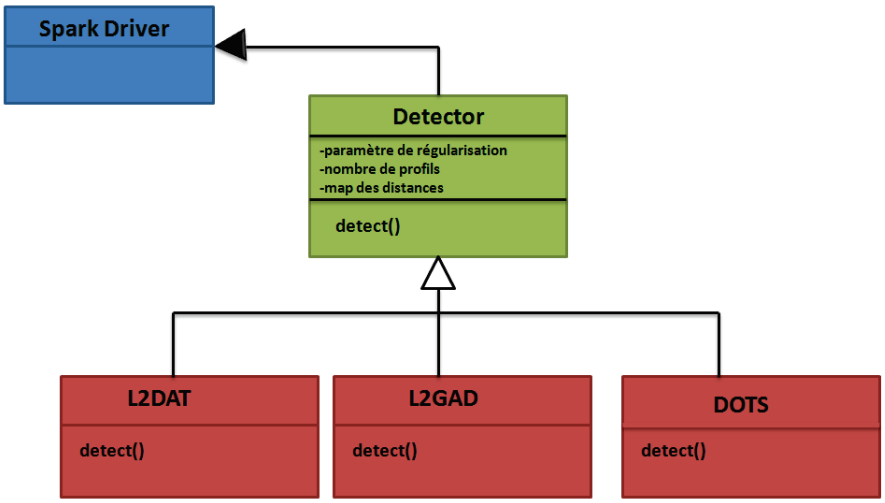


FIGURE 5.5 – Implémentation des approches de détection globale selon le patron de conception *stratégie*.

5.4 Expérimentations et résultats

Dans cette section, nous discutons des résultats obtenus à l'issue des expérimentations que nous avons menées sur les séries temporelles de prix de plusieurs pneus. Comme nous l'avons mentionné, un pneu est mis en vente par plusieurs site web marchand à la fois, durant une certaine période. Par exemple, si un pneu est mis en vente par 500 sites durant 300 jours, l'ensemble de séries temporelles de ce pneu est représenté par, $T = \{t_1, t_2, \ldots, t_{500}\}$. Le nombre de séries correspond au nombre de sites (e.g ; chaque site étant représenté par une série temporelle t_i) et la taille de chaque série de prix t_i est en moyenne 300, à raison d'un prix par jour, sauf dans certains cas où des sites web changent le prix des pneus plusieurs fois par jour, ou par exemple des prix manquants à cause des *crawlers* mis en place. L'objectif de cette expérimentation est de détecter pour un pneu donné, les séries temporelles de prix anormales dans la base T. Ces anomalies peuvent correspondre à des séries matchées par erreur au pneu en question, ou aux séries ayant des tendances assez différentes par rapport à la majorité.

A titre illustratif, nous nous limitons dans cette partie à montrer quelques exemples de détection faites par notre framework sur certains pneus de référence mis en vente dans le continent Européen, comme le *Michelin Energy Saver*, le *Cinturato P7 de Pirelli* ou le *P-Zéro de Pirelli*. La Table 5.7 représente succinctement les caractéristiques de chacune des bases de séries utilisées.

PNEU	NOMBRE DE SÉRIES	TAILLE MOYENNE DES SÉRIES
MICHELIN ENERGY SAVER	701	1300
CINTURATO P7 PIRELLI	600	1100
P-ZERO PIRELLI	800	180

TABLE 5.7 – Caractéristiques des séries temporelles des trois pneus considérés par l'expérimentation.

Protocole expérimental et résultats

Tout d'abord, les séries temporelles de prix de chaque pneu ont été traitées et transformées par le module de traitement des données de notre framework. La deuxième étape consiste à utiliser le module de détection pour mettre en évidence les séries susceptibles d'être anormales. Ayant considéré dans nos expérimentations que des sites européens, le nombre de profils k a été fixé comme étant le nombre des différents pays originaires des sites web du continent. Néanmoins, dans le cas d'une détection à l'échelle du pays où la valeur du k n'est pas évidente (e.g, uniquement les sites web de France), des mesures d'évaluation de clustering, comme l'indice de *davies-bouldin* [Davies79], peuvent être utilisés pour mieux estimer ce paramètre. Le paramètre de régularisation ou de pondération (selon l'approche choisie) est aléatoirement pris dans l'intervalle $]0, 1[$. A l'issue de la détection, les séries temporelles sont réordonnées selon leur score d'anormalité et les séries ayant les scores les plus importants sont communiquées aux experts pour validation.

Michelin Energy Saver

Les séries temporelles de prix du pneu *Michelin* sont présentées dans la Figure 5.6(a). Ce pneu est mis en vente par 700 sites web européens et la taille de ses séries temporelles est de 1300 en moyenne, ce qui correspond à une période de mise en vente de 3 ans et demi. Les résultats de la détection (cf. Figure 5.6(b)) montre effectivement qu'il y a quelques séries anormales (en rouge) par rapport à la majorité et qui seraient matchées par erreur à ce pneu.

Cinturato P7 Pirelli

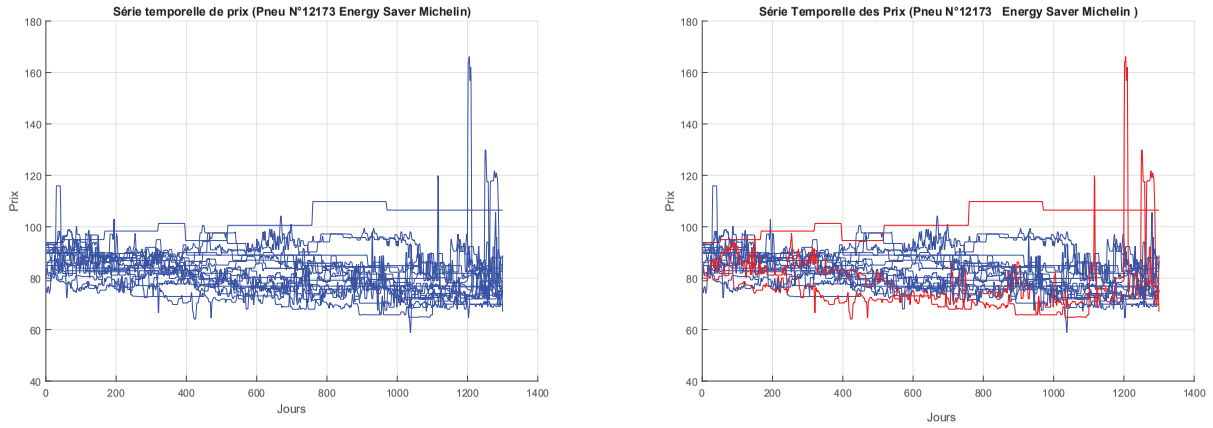
Les séries temporelles de prix du pneu *Cinturato P7 Pirelli* sont présentées dans la Figure 5.7(a). Ce pneu est mis en vente par 600 sites web européens et la taille de ses séries temporelles est environ 1100 en moyenne, ce qui correspond à une période de mise en vente de 3 ans. La détection concernant le pneu Cinturato P7 Pirelli (cf. Figure 5.7(b)) montre qu'il y a quelques séries qui ont été faussement matchées vu qu'elles s'écartent significativement par rapport à la majorité des autres séries de la base.

P-zéro Pirelli

Les séries temporelles de prix du pneu *P-Zéro Pirelli*, qui est relativement plus récent par rapport aux deux précédents, sont présentées dans la Figure 5.8(a). Ce pneu est mis en vente par 800 sites web européens et la taille de ses séries temporelles est environ 180 en moyenne, ce qui correspond à une période de mise en vente d'à peu prés un 6 mois. Les résultats de la détection concernant le pneu P-Zéro Pirelli (cf. Figure 5.8(b)) montre aussi l'existence de certaines séries anormales.

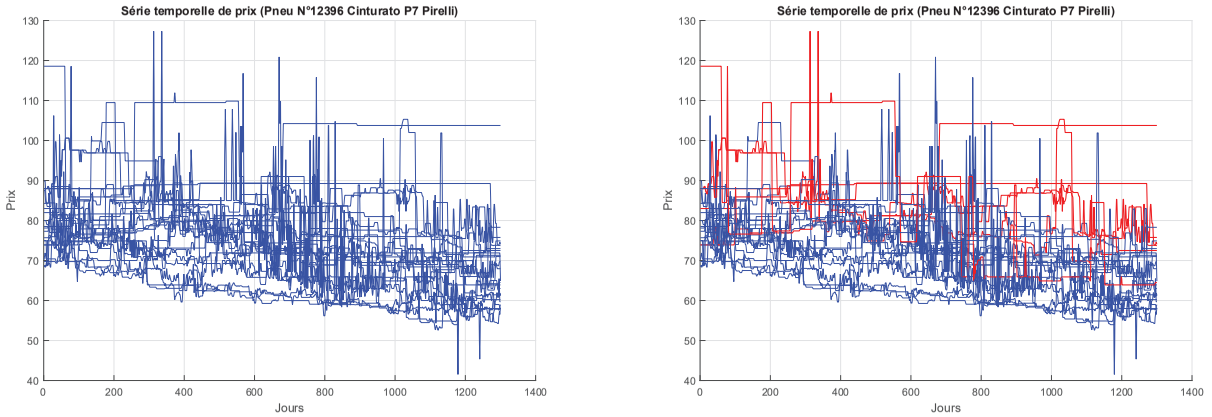
5.5 Conclusion

La validation de n'importe quelle méthode de détection sur des données réelles demeure une tache relativement difficile, qui nécessite un travail considérable pour la préparation du terrain. Tout d'abord, une étape conséquente de pré-traitement était nécessaire pour exploiter au mieux les données sur les prix de pneumatiques. Ensuite, nous avons développé la procédure de détection en deux modules. Le premier permet de pré-traiter et transformer efficacement les données. Le deuxième module comprend l'ensemble de nos approches de détection (globale et contextuelle). La série de traitement proposée a été implémentée et déployée dans un environnement distribué via le le paradigme Spark et le langage fonctionnel Scala. Les résultats obtenus ont permis de détecter des séries anormales pour certains pneus et de mettre ainsi en évidence quelques dysfonctionnements dans la phase de collecte de données.



(a) Séries temporelles de prix du pneu Energy Saver Michelin (b) Séries temporelles anormales (en rouges) par rapport à la majorité des séries

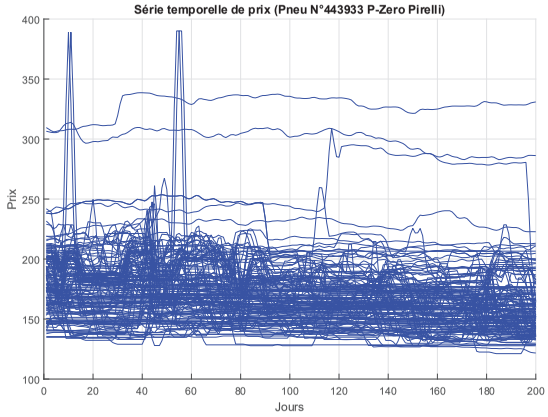
FIGURE 5.6 – Exemple de détection sur les séries temporelles de prix du pneu Energy Saver Michelin.



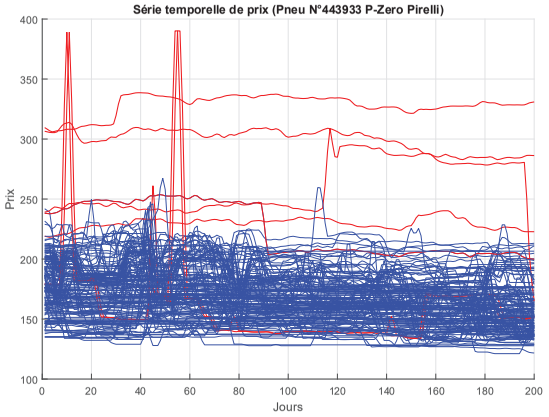
(a) Séries temporelles de prix du pneu Cinturato P7 Pirelli

(b) Séries temporelles anormales (en rouges) par rapport à la majorité des séries

FIGURE 5.7 – Exemple de détection sur les séries temporelles de prix du pneu Cinturato P7 Pirelli.



(a) Séries temporelles de prix du pneu P-Zéro Pirelli



(b) Séries temporelles anormales (en rouges) par rapport à la majorité des séries

FIGURE 5.8 – Exemple de détection sur les séries temporelles de prix du pneu P-Zéro Pirelli.

6

Conclusion et Perspectives

Dans cette thèse, nous nous sommes intéressés à deux aspects particuliers de la détection d'anomalies non-supervisée dans les séries temporelles uni et multi-variées. Le premier est global et consiste à mettre en évidence des séries relativement anormales par rapport à une base entière. Le second est contextuel et vise à détecter localement, les observations anormales par rapport à la structure globale de la série étudiée. Pour ce faire, nous avons dressé un état de l'art complet concernant les différentes approches de détection dans les données temporelles. Nous nous sommes tous particulièrement intéressés aux approches basées sur la structure des données et l'utilisation de la notion de similarité. Nous avons ensuite discuté des différentes techniques existantes pour mesurer la similarité entre les séries temporelles.

Concernant la détection globale, nous avons proposé des approches d'optimisation à base de clustering pondéré et de déformation temporelle. La différence entre ces approches réside principalement dans la manière de pondérer les séries ainsi qu'à la vison de la détection, globale (DOTS et ℓ_2-DAT) ou locale (L2GAD). Les résultats expérimentaux ont montré que l'idée de la pondération améliore nettement la performance de détection par rapport aux différentes approches de l'état de l'art.

Concernant la détection contextuelle non-supervisée des observations anormales au sein d'une série uni ou multi-variée, nous avons proposé une nouvelle approche (LADOP) où la dépendance temporelle entre les observations de la série, est modélisée par la loi de Poisson. L'approche est basée sur le principe de la reconstruction des données et l'analyse des résidus pour la détection d'anomalies.

Enfin, pour répondre aux besoins exprimés par l'entreprise LIZEO, nous avons présenté une validation expérimentale sur un problème réel, concernant la détection des séries de prix anormaux sur les pneumatiques. Ainsi, nous avons développé un framework de détection composé de deux modules, a) une chaine complète de prétraitement, de nettoyage et de transformation des données permettant de les rendre exploitables et b) un module de détection comprenant l'ensemble des approches que nous avons décrites dans les chapitres précédents. Étant donnée la nature massive des données, le framework proposé a été implémenté et déployé dans un environnement distribué (Spark/Scala) pour garantir un passage à l'échelle. Les résultats expérimentaux ont permis de mettre en évidence certaines défaillances dans le système de collecte des données (e.g., des prix) de l'entreprise.

L'ensemble des approches développées durant cette thèse, a permis de dégager des pistes intéressantes pour de futurs travaux de recherche, d'autant plus que nos approches ne sont pas exemptés de limites. Les travaux futurs comprennent, sans s'y limiter, les points suivants :

- La performance de détection de nos approches de pondération dépend en partie de la mesure de similarité utilisée. Pour le moment, nous nous appuyons sur la DTW qui donne globalement de très bons résultats sans pour autant qu'elle soit parfaite sur tous les jeux de données. Toutefois, nous pensons que l'apprentissage de cette métrique pourrait améliorer nettement la détection là où une mesure arbitraire ne saurait capter les caractéristiques subtiles d'un jeu de données. Ainsi, la détection et l'apprentissage de métrique peuvent être combinés dans un seul framework.

- Les approches de pondération que nous avons proposées ne permettent pas de traiter le cas des clusters de forme arbitraires (e.g., non-convexes) alors que l'approche séquentielle (DetectS) le permet via la décomposition spectrale. Nous pensons qu'il serait intéressant d'étudier comment intégrer la pondération dans cette approche. Comme cette dernière est basée sur la décomposition spectrale et le clustering des vecteurs propres, une nouvelle piste serait d'utiliser le mécanisme de la pondération dans la phase de clustering de ces vecteurs propres.

- Un autre point important est de savoir comment intégrer efficacement le feedback des experts sur une détection. On peut par exemple, basculer du mode complètement non-supervisé vers un mode semi-supervisé où les retours d'expérience seraient exprimés sous forme de contraintes.

- Enfin, concernant la détection contextuelle où l'aspect temporel est modélisé par la loi de Poisson, ce qui représente en soit, une hypothèse assez forte sur la dépendance temporelle entre les observations, nous pensons qu'il serait judicieux de voir comment apprendre l'aspect temporel sans aucun a priori.

A

Liste des Publications

Revue internationale

- **Seif-Eddine Benkabou**, Khalid Benabdeslem & Canitia Bruno. *Unsupervised Outlier Detection for Time Series by Entropy and Dynamic Time Warping*. **Knowledge and Information Systems**, vol. 54, no. 2, pages 463-486, Feb 2018.

Conférences internationales

- **Seif-Eddine Benkabou**, Khalid Benabdeslem & Bruno Canitia. *Local-to-Global Unsupervised Anomaly Detection from Temporal Data*. **In Advances in Knowledge Discovery and Data Mining - 21st Pacific-Asia Conference, PAKDD** 2017, Jeju, South Korea, May 23-26, 2017, Proceedings, Part I, pages 762–772, 2017.

- **Seif-Eddine Benkabou**, Khalid Benabdeslem & Bruno Canitia. *L2-type regularization based unsupervised anomaly detection from temporal data*. **In 2017 International Joint Conference on Neural Networks, IJCNN** 2017, Anchorage, AK, USA, May 14-19, 2017, pages 2354–2361, 2017.

Workshops Internationaux

- **Seif-Eddine Benkabou**, Khalid Benabdeslem & Canitia Bruno. *Entropy-based clustering for anomaly detection from time-series data*. In **ICML Workshop on Anomaly detection**, New York, United States, 2016.

Conférences nationales

- **Seif-Eddine Benkabou**, Khalid Benabdeslem & Canitia Bruno. *Régularisation Ridge pour la détection non-supervisée à partir de séries temporelles*. In **Société Francophone de Classification (SFC)**, Lyon, France, June 2017.

- **Seif-Eddine Benkabou**, Khalid Benabdeslem & Canitia Bruno. *Une approche embedded pour la détection de nouveautés à partir de séries temporelles*. In **Société Francophone de Classification (SFC)**, Marrakech, Morocco, 2016.

- **Seif-Eddine Benkabou**, Khalid Benabdeslem & Canitia Bruno. *Une approche à deux niveaux séquentiels pour la détection de nouveautés à partir de séries temporelles*. In **Société Francophone de Classification (SFC)**, Nantes, France, September 2015.

Travaux en cours

- **Seif-Eddine Benkabou**, Khalid Benabdeslem & Canitia Bruno. *Local Anomaly Detection for time series by temporal dependency based On Poisson model*.

Bibliographie

[Aggarwal01] Charu C. Aggarwal & Philip S. Yu. *Outlier Detection for High Dimensional Data*. In Proceedings of the 2001 ACM SIGMOD international conference on Management of data, Santa Barbara, CA, USA, May 21-24, 2001, pages 37–46, 2001.

[Aggarwal05] Charu C. Aggarwal. *On Abnormality Detection in Spuriously Populated Data Streams*. In Proceedings of the 2005 SIAM International Conference on Data Mining, SDM 2005, Newport Beach, CA, USA, April 21-23, 2005, pages 80–91, 2005.

[Aggarwal08] Charu C. Aggarwal & Philip S. Yu. *Outlier Detection with Uncertain Data*. In Proceedings of the SIAM International Conference on Data Mining, SDM 2008, April 24-26, 2008, Atlanta, Georgia, USA, pages 483–493, 2008.

[Aggarwal11] Charu C. Aggarwal, Yuchen Zhao & Philip S. Yu. *Outlier detection in graph streams*. In Proceedings of the 27th International Conference on Data Engineering, ICDE 2011, April 11-16, 2011, Hannover, Germany, pages 399–409, 2011.

[Aggarwal12] Charu C. Aggarwal & Karthik Subbian. *Event Detection in Social Streams*. In Proceedings of the Twelfth SIAM International Conference on Data Mining, Anaheim, California, USA, April 26-28, 2012., pages 624–635, 2012.

[Aggarwal13] Charu C. Aggarwal. Outlier analysis. Springer Publishing Company, Incorporated, 2013. 1, 7

[Angiulli07] Fabrizio Angiulli & Fabio Fassetti. *Detecting Distance-based Outliers in Streams of Data*. In Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management, CIKM '07, pages 811–820, New York, NY, USA, 2007. ACM. 30, 31

[Aßfalg06] Johannes Aßfalg, Hans-Peter Kriegel, Peer Kröger, Peter Kunath, Alexey Pryakhin & Matthias Renz. *Similarity Search on Time Series Based on Threshold Queries*. In EDBT, volume 3896 of *Lecture Notes in Computer Science*, pages 276–294. Springer, 2006. 25

[Basu07] Sabyasachi Basu & Martin Meckesheimer. *Automatic outlier detection for time series : an application to sensor data*. Knowl. Inf. Syst., vol. 11, no. 2, pages 137–154, 2007. 15, 30, 75, 76

[Baum70] Leonard E. Baum, Ted Petrie, George Soules & Norman Weiss. *A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains*. Ann. Math. Statist., vol. 41, no. 1, pages 164–171, The Institute of Mathematical Statistics, 02 1970. 17

[Benkabou15] Seif-Eddine Benkabou, Khalid Benabdeslem & Canitia Bruno. *Une approche à deux niveaux séquentiels pour la détection de nouveautés à partir de séries temporelles*. In 22ème Rencontres de la Société Francophone de Classification, Société Francophone de Classification, pages 129–132, Nantes, France, September 2015.

[Benkabou16a] Seif-Eddine Benkabou, Khalid Benabdeslem & Canitia Bruno. *Entropy-based clustering for anomaly detection from time-series data*. In ICML Workshop on Anomaly detection, New York, United States, 2016. 39

[Benkabou16b] Seif-Eddine Benkabou, Khalid Benabdeslem & Canitia Bruno. *Une approche embedded pour la détection de nouveautés à partir de séries temporelles*. In Société Francophone de Classification (SFC), Marrakech, Morocco, 2016.

[Benkabou17a] Seif-Eddine Benkabou, Khalid Benabdeslem & Canitia Bruno. *Régularisation Ridge pour la détection non-supervisée à partir de séries temporelles*. In SFC : Société Francophone de Classification, Lyon, France, June 2017.

[Benkabou17b] Seif-Eddine Benkabou, Khalid Benabdeslem & Bruno Canitia. *L2-type regularization-based unsupervised anomaly detection from temporal data*. In 2017 International Joint Conference on Neural Networks, IJCNN 2017, Anchorage, AK, USA, May 14-19, 2017, pages 2354–2361, 2017. 47, 53

[Benkabou17c] Seif-Eddine Benkabou, Khalid Benabdeslem & Bruno Canitia. *Local-to-Global Unsupervised Anomaly Detection from Temporal Data*. In Advances in Knowledge Discovery and Data Mining - 21st Pacific-Asia Conference, PAKDD 2017, Jeju, South Korea, May 23-26, 2017, Proceedings, Part I, pages 762–772, 2017.

[Benkabou18] Seif-Eddine Benkabou, Khalid Benabdeslem & Bruno Canitia. *Unsupervised outlier detection for time series by entropy and dynamic time warping*. Knowledge and Information Systems, vol. 54, no. 2, pages 463–486, Feb 2018. 39

[Berndt94] Donald J. Berndt & James Clifford. *Using Dynamic Time Warping to Find Patterns in Time Series*. In Knowledge Discovery in

Databases : Papers from the 1994 AAAI Workshop, Seattle, Washington, July 1994. Technical Report WS-94-03, pages 359–370, 1994.

[Birant06] D. Birant & A. Kut. *Spatio-temporal outlier detection in large databases*. In 28th International Conference on Information Technology Interfaces, 2006., pages 179–184, 2006. 30

[Breunig00] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng & Jörg Sander. *LOF : Identifying Density-based Local Outliers*. In Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, SIGMOD '00, pages 93–104, New York, NY, USA, 2000. ACM. 58

[Bu07] Yingyi Bu, Oscar Tat-Wing Leung, Ada Wai-Chee Fu, Eamonn J. Keogh, Jian Pei & Sam Meshkin. *WAT : Finding Top-K Discords in Time Series Database*. In SDM, pages 449–454. SIAM, 2007. 30

[Budalakoti06] Suratna Budalakoti, Ashok Srivastava, R Akella & Eugene Turkov. *Anomaly Detection in Large Sets of High-Dimensional Symbol Sequences*. 01 2006. 13, 23

[Budalakoti09] Suratna Budalakoti, Ashok Srivastava & Matthew Otey. *Anomaly Detection and Diagnosis Algorithms for Discrete Symbol Sequences with Applications to Airline Safety*. vol. 39, pages 101 – 113, 02 2009. 13, 23, 57

[Cabrera01] João B. D. Cabrera, Lundy Lewis & Raman K. Mehra. *Detection and Classification of Intrusions and Faults Using Sequences of System Calls*. SIGMOD Rec., vol. 30, no. 4, pages 25–34, ACM, New York, NY, USA, December 2001. 11

[Chandola08] Varun Chandola, Varun Mithal & Vipin Kumar. *A Comparative Evaluation of Anomaly Detection Techniques for Sequence Data*. In Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008), December 15-19, 2008, Pisa, Italy, pages 743–748, 2008. 13, 16, 23, 56

[Chandola09] Varun Chandola, Arindam Banerjee & Vipin Kumar. *Anomaly detection : A survey*. ACM Comput. Surv., vol. 41, no. 3, pages 15 :1–15 :58, 2009. ix, 2, 7, 8, 9

[Chandola12] Varun Chandola, Arindam Banerjee & Vipin Kumar. *Anomaly Detection for Discrete Sequences : A Survey*. IEEE Trans. Knowl. Data Eng., vol. 24, no. 5, pages 823–839, 2012.

[Chen04] Lei Chen & Raymond T. Ng. *On The Marriage of Lp-norms and Edit Distance*. In VLDB, pages 792–803. Morgan Kaufmann, 2004. 23, 24

[Chen05] Lei Chen, M. Tamer Özsu & Vincent Oria. *Robust and Fast Similarity Search for Moving Object Trajectories*. In SIGMOD Conference, pages 491–502. ACM, 2005. 23

[Chen09] Xiao-yun Chen & Yan-yan Zhan. *Erratum to : "Multi-scale anomaly detection algorithm based on infrequent pattern of time series" [J. Comput. Appl. Math. 214(1) (2008) 227-237].* J. Computational Applied Mathematics, vol. 231, no. 2, page 1004, 2009.

[Chen15] Yanping Chen, Eamonn Keogh, Bing Hu, Nurjahan Begum, Anthony Bagnall, Abdullah Mueen & Gustavo Batista. *The UCR Time Series Classification Archive*, July 2015. `www.cs.ucr.edu/~eamonn/time_series_data/`. 44, 56

[Cho08] HyungJun Cho, Yang-jin Kim, Hee Jung Jung, Sang-Won Lee & Jae Won Lee. *OutlierD : an R package for outlier detection using quantile regression on mass spectrometry data*. Bioinformatics, vol. 24, no. 6, pages 882–884, 2008.

[Davies79] D. L. Davies & D. W. Bouldin. *A Cluster Separation Measure*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-1, no. 2, pages 224–227, April 1979. 100

[Dempster77] A. P. Dempster, N. M. Laird & D. B. Rubin. *Maximum likelihood from incomplete data via the EM algorithm*. JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B, vol. 39, no. 1, pages 1–38, 1977. 13

[Demsar06] Janez Demsar. *Statistical Comparisons of Classifiers over Multiple Data Sets*. Journal of Machine Learning Research, vol. 7, pages 1–30, 2006. 59, 60

[Endler98] D. Endler. *Intrusion Detection Applying Machine Learning to Solaris Audit Data*. In Proceedings of the 14th Annual Computer Security Applications Conference, ACSAC '98, pages 268–, Washington, DC, USA, 1998. IEEE Computer Society. 11, 31

[Eskin01] E. Eskin, Wenke Lee & S. J. Stolfo. *Modeling system calls for intrusion detection with dynamic window sizes*. In DARPA Information Survivability Conference amp ; Exposition II, 2001. DISCEX '01. Proceedings, volume 1, pages 165–175 vol.1, 2001.

[Eskin02] Eleazar Eskin, Andrew Arnold, Michael Prerau, Leonid Portnoy & Sal Stolfo. A geometric framework for unsupervised anomaly detection, pages 77–101. Springer US, Boston, MA, 2002. 13

[Esling12] Philippe Esling & Carlos Agón. *Time-series data mining*. ACM Comput. Surv., vol. 45, no. 1, pages 12 :1–12 :34, 2012. 17

[Evangelista] Paul F. Evangelista, Piero Bonnisone, Mark J. Embrechts & Boleslaw K. Szymanski. *FUZZY ROC CURVES FOR THE 1 CLASS SVM : APPLICATION TO INTRUSION DETECTION*. 13

[Fawcett06] Tom Fawcett. *An introduction to ROC analysis*. Pattern Recognition Letters, vol. 27, no. 8, pages 861–874, 2006. 59

[Florez-Larrahondo05] German Florez-Larrahondo, Susan M. Bridges & Rayford Vaughn. *Efficient Modeling of Discrete Events for Anomaly Detection Using Hidden Markov Models*. In Proceedings of the 8th International Conference on Information Security, ISC'05, pages 506–514, Berlin, Heidelberg, 2005. Springer-Verlag. 16

[fou]

[Fox72] A.J. Fox. *Outliers in Time Series*. Journal of the Royal Statistical Society. Series B (Methodological), vol. 34, no. 3, pages 350–363, 1972.

[Frentzos07] Elias Frentzos, Kostas Gratsias & Yannis Theodoridis. *Index-based Most Similar Trajectory Search*. In ICDE, pages 816–825. IEEE Computer Society, 2007. 18

[Friedman37] Milton Friedman. *The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance*. Journal of the American Statistical Association, vol. 32, no. 200, pages 675–701, [American Statistical Association, Taylor & Francis, Ltd.], 1937. 59

[Gao02] Bo Gao, Hui-Ye Ma & Yu-Hang Yang. *HMMs (Hidden Markov models) based on anomaly intrusion detection method*. In Proceedings. International Conference on Machine Learning and Cybernetics, volume 1, pages 381–385 vol.1, 2002. 11, 16

[Gao10] Jing Gao, Feng Liang, Wei Fan, Chi Wang, Yizhou Sun & Jiawei Han. *On community outliers and their efficient detection in information networks*. In KDD, pages 813–822. ACM, 2010.

[Ghosh99a] Anup K. Ghosh, Aaron Schwartzbard & Michael Schatz. *Learning Program Behavior Profiles for Intrusion Detection*. In Proceedings of the 1st Conference on Workshop on Intrusion Detection and Network Monitoring - Volume 1, ID'99, pages 6–6, Berkeley, CA, USA, 1999. USENIX Association. 11

[Ghosh99b] Anup K. Ghosh, Aaron Schwartzbard & Michael Schatz. *Using Program Behavior Profiles for Intrusion Detection*. In In Proceedings of the SANS Third Conference and Workshop on Intrusion Detection and Response, 1999. 11

[Ghoting04] Amol Ghoting, Matthew Eric Otey & Srinivasan Parthasarathy. *LOADED : Link-Based Outlier and Anomaly Detection in Evolving*

Data Sets. In ICDM, pages 387–390. IEEE Computer Society, 2004.

[Golay98] Xavier Golay, Spyros Kollias, Gautier Stoll, Dieter Meier, Anton Valavanis & Peter Boesiger. *A new correlation-based fuzzy logic clustering algorithm for FMRI*. Magnetic Resonance in Medicine, vol. 40, no. 2, pages 249–260, Wiley Subscription Services, Inc., A Wiley Company, 1998. 25

[Goldstein16] Markus Goldstein & Seiichi Uchida. *A Comparative Evaluation of Unsupervised Anomaly Detection Algorithms for Multivariate Data*. PLOS ONE, vol. 11, no. 4, pages 1–31, Public Library of Science, 04 2016. 59

[González03] Fabio A. González & Dipankar Dasgupta. *Anomaly Detection Using Real-Valued Negative Selection*. Genetic Programming and Evolvable Machines, vol. 4, no. 4, pages 383–403, Dec 2003. 13

[Gu11] Quanquan Gu & Jiawei Han. *Towards feature selection in network*. In CIKM, pages 1175–1184. ACM, 2011. 40

[Gupta12a] Manish Gupta, Jing Gao, Yizhou Sun & Jiawei Han. *Community Trend Outlier Detection Using Soft Temporal Pattern Mining*. In ECML/PKDD (2), volume 7524 of *Lecture Notes in Computer Science*, pages 692–708. Springer, 2012. 31

[Gupta12b] Manish Gupta, Jing Gao, Yizhou Sun & Jiawei Han. *Integrating community matching and outlier detection for mining evolutionary community outliers*. In KDD, pages 859–867. ACM, 2012.

[Gupta13] Manish Gupta, Abhishek Singh, Haifeng Chen & Guofei Jiang. *Context-Aware Time Series Anomaly Detection for Complex Systems*. In Proc. of the SDM Workshop on Data Mining for Service and Maintenance, January 2013. 13

[Gupta14] Manish Gupta, Jing Gao, Charu C. Aggarwal & Jiawei Han. *Outlier Detection for Temporal Data : A Survey*. IEEE Trans. Knowl. Data Eng., vol. 26, no. 9, pages 2250–2267, 2014. ix, 8, 12

[Hastie01] Trevor Hastie, Robert Tibshirani & Jerome Friedman. The elements of statistical learning. Springer, 2001. 47

[Hill09] David J. Hill, Barbara S. Minsker & Eyal Amir. *Real-time Bayesian anomaly detection in streaming environmental data*. Water Resources Research, vol. 45, no. 4, pages n/a–n/a, 2009. W00D28. 30

[Hill10] David J. Hill & Barbara S. Minsker. *Anomaly Detection in Streaming Environmental Sensor Data : A Data-driven Modeling Approach*. Environ. Model. Softw., vol. 25, no. 9, pages 1014–1022,

Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, September 2010. 15, 30, 75, 76

[Hodge04] Victoria J. Hodge & Jim Austin. *A Survey of Outlier Detection Methodologies*. Artif. Intell. Rev., vol. 22, no. 2, pages 85–126, 2004.

[Hofmeyr98] Steven A. Hofmeyr, Stephanie Forrest & Anil Somayaji. *Intrusion Detection Using Sequences of System Calls*. J. Comput. Secur., vol. 6, no. 3, pages 151–180, IOS Press, Amsterdam, The Netherlands, The Netherlands, August 1998. 12, 31

[Jagadish99] H. V. Jagadish, Nick Koudas & S. Muthukrishnan. *Mining Deviants in a Time Series Database*. In VLDB, pages 102–113. Morgan Kaufmann, 1999.

[Jiang06] G. Jiang, H. Chen & K. Yoshihira. *Modeling and Tracking of Transaction Flow Dynamics for Fault Detection in Complex Systems*. IEEE Transactions on Dependable and Secure Computing, vol. 3, no. 4, pages 312–326, Oct 2006. 31

[Keogh05a] E. Keogh, J. Lin & A. Fu. *HOT SAX : efficiently finding the most unusual time series subsequence*. In Fifth IEEE International Conference on Data Mining (ICDM'05), pages 8 pp.–, Nov 2005. 30, 31

[Keogh05b] Eamonn J. Keogh & Chotirat (Ann) Ratanamahatana. *Exact indexing of dynamic time warping*. Knowl. Inf. Syst., vol. 7, no. 3, pages 358–386, 2005. 21

[Keogh07] Eamonn J. Keogh, Jessica Lin, Sang-Hee Lee & Helga Van Herle. *Finding the most unusual time series subsequence : algorithms and applications*. Knowl. Inf. Syst., vol. 11, no. 1, pages 1–27, 2007. 29

[Lakhina04] Anukool Lakhina, Mark Crovella & Christiphe Diot. *Characterization of Network-wide Anomalies in Traffic Flows*. In Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement, IMC '04, pages 201–206, New York, NY, USA, 2004. ACM. 31

[Lane97a] Terran Lane & Carla Brodley. *Sequence Matching and Learning in Anomaly Detection for Computer Security*. 05 1997. 12, 13

[Lane97b] Terran Lane & Carla E. Brodley. *An Application of Machine Learning to Anomaly Detection*. In In Proceedings of the 20th National Information Systems Security Conference, pages 366–380, 1997. 12, 31

[Lane99] Terran Lane & Carla E. Brodley. *Temporal Sequence Learning and Data Reduction for Anomaly Detection*. ACM Trans. Inf. Syst. Se-

cur., vol. 2, no. 3, pages 295–331, ACM, New York, NY, USA, August 1999. 12, 31

[Laptev15] Nikolay Laptev, Saeed Amizadeh & Ian Flint. *Generic and Scalable Framework for Automated Time-series Anomaly Detection*. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '15, pages 1939–1947, New York, NY, USA, 2015. ACM. 86

[Li17] Jundong Li, Harsh Dani, Xia Hu & Huan Liu. *Radar : Residual Analysis for Anomaly Detection in Attributed Networks*. In IJCAI, pages 2152–2158. ijcai.org, 2017.

[Lin03] Jessica Lin, Eamonn Keogh, Stefano Lonardi & Bill Chiu. *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, pages 2–11, New York, NY, USA, 2003. ACM. 28, 29

[Liu08] Fei Tony Liu, Kai Ming Ting & Zhi-Hua Zhou. *Isolation Forest*. In ICDM, pages 413–422. IEEE Computer Society, 2008.

[Ma03a] J. Ma & S. Perkins. *Time-series novelty detection using one-class support vector machines*. In Proceedings of the International Joint Conference on Neural Networks, 2003., volume 3, pages 1741–1745 vol.3, July 2003. 13

[Ma03b] Junshui Ma & Simon Perkins. *Online novelty detection on temporal sequences*. In KDD, pages 613–618. ACM, 2003. 15, 75, 76

[Malinowski13] Simon Malinowski, Thomas Guyet, René Quiniou & Romain Tavenard. *1d-SAX : A Novel Symbolic Representation for Time Series*. In IDA, volume 8207 of *Lecture Notes in Computer Science*, pages 273–284. Springer, 2013. 29

[Marceau00] Carla Marceau. *Characterizing the Behavior of a Program Using Multiple-length N-grams*. In Proceedings of the 2000 Workshop on New Security Paradigms, NSPW '00, pages 101–110, New York, NY, USA, 2000. ACM.

[Marco15] Bevilacqua Marco & Tsaftaris A. Sotirios. *Dictionary-decomposition-based one-class svm for unsupervised detection of anomalous time series*. In Proceedings of *23rd European Signal Processing Conference (EUSIPCO)*, pages 1776–1780, Sept 2015. 58

[Michael00] C. C. Michael & A. Ghosh. *Two state-based approaches to program-based anomaly detection*. In Computer Security Applications, 2000. ACSAC '00. 16th Annual Conference, pages 21–30, Dec 2000.

[Möller-Levet03] Carla S. Möller-Levet, Frank Klawonn, Kwang-Hyun Cho & Olaf Wolkenhauer. *Fuzzy Clustering of Short Time-Series and Unevenly Distributed Sampling Points*. In IDA, volume 2810 of *Lecture Notes in Computer Science*, pages 330–340. Springer, 2003. 18

[Mori16] Usue Mori, Alexander Mendiburu & Jose A. Lozano. *Distance Measures for Time Series in R : The TSdist Package*. R journal, vol. 8, no. 2, pages 451–459, 2016. ix, 19, 20, 22, 26

[Nairac99] Alexandre Nairac, Neil W. Townsend, Roy Carr, Steve King, Peter Cowley & Lionel Tarassenko. *A System for the Analysis of Jet Engine Vibration Data*. Integrated Computer-Aided Engineering, vol. 6, no. 1, pages 53–66, 1999. 13, 30

[Ng01] Andrew Y. Ng, Michael I. Jordan & Yair Weiss. *On Spectral Clustering : Analysis and an algorithm*. In NIPS, pages 849–856. MIT Press, 2001. 36

[Nie10] Feiping Nie, Heng Huang, Xiao Cai & Chris H. Q. Ding. *Efficient and Robust Feature Selection via Joint $\ell_{2,1}$ -Norms Minimization*. In NIPS, pages 1813–1821. Curran Associates, Inc., 2010. 82

[Otey06] Matthew Eric Otey, Amol Ghoting & Srinivasan Parthasarathy. *Fast Distributed Outlier Detection in Mixed-Attribute Data Sets*. Data Min. Knowl. Discov., vol. 12, no. 2-3, pages 203–228, Kluwer Academic Publishers, Hingham, MA, USA, May 2006. 32

[Pan09] Xinghao Pan, Jiaqi Tan, Soila Kavulya, Rajeev Gandhi & Priya Narasimhan. *Ganesha : blackBox diagnosis of MapReduce systems*. SIGMETRICS Performance Evaluation Review, vol. 37, no. 3, pages 8–13, 2009. 13

[Portnoy01] Leonid Portnoy, Eleazar Eskin & Salvatore Stolfo. *Intrusion Detection with Unlabeled Data Using Clustering*. 11 2001. 13, 31, 58

[Qiao02] Y. Qiao, X. W. Xin, Y. Bin & S. Ge. *Anomaly intrusion detection method based on HMM*. Electronics Letters, vol. 38, no. 13, pages 663–664, Jun 2002. 16

[Ratanamahatana04] Chotirat (Ann) Ratanamahatana & Eamonn J. Keogh. *Making Time-Series Classification More Accurate Using Learned Constraints*. In SDM, pages 11–22. SIAM, 2004. ix, 44, 45

[Rebbapragada09] Umaa Rebbapragada, Pavlos Protopapas, Carla E. Brodley & Charles R. Alcock. *Finding anomalous periodic time series*. Machine Learning, vol. 74, no. 3, pages 281–313, 2009. 13

[Sakoe78] H. Sakoe & S. Chiba. *Dynamic programming algorithm optimization for spoken word recognition*. IEEE Transactions on Acoustics,

Speech, and Signal Processing, vol. 26, no. 1, pages 43–49, Feb 1978. 21

[Salvador05] Stan Salvador & Philip Chan. *Learning States and Rules for Detecting Anomalies in Time Series*. Applied Intelligence, vol. 23, no. 3, pages 241–255, Dec 2005.

[Schölkopf99] Bernhard Schölkopf, Robert Williamson, Alex Smola, John Shawe-Taylor & John Platt. *Support Vector Method for Novelty Detection*. In Proceedings of the 12th International Conference on Neural Information Processing Systems, NIPS'99, pages 582–588, Cambridge, MA, USA, 1999. MIT Press. 12, 58

[Scholkopf01] Bernhard Scholkopf & Alexander J. Smola. Learning with kernels : Support vector machines, regularization, optimization, and beyond. MIT Press, Cambridge, MA, USA, 2001. 57

[Sequeira02] Karlton Sequeira & Mohammed Javeed Zaki. *ADMIT : anomaly-based data mining for intrusions*. In Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, July 23-26, 2002, Edmonton, Alberta, Canada, pages 386–395, 2002. 13, 31

[Shahabi00] Cyrus Shahabi, Xiaoming Tian & Wugang Zhao. *TSA-Tree : A Wavelet-Based Approach to Improve the Efficiency of Multi-Level Surprise and Trend Queries on Time-Series Data*. In SSDBM, pages 55–68. IEEE Computer Society, 2000.

[She11] Yiyuan She & Art B. Owen. *Outlier Detection Using Nonconvex Penalized Regression*. Journal of the American Statistical Association, vol. 106, no. 494, pages 626–639, Taylor and Francis, 2011. 76, 78

[Silvestri94] G. Silvestri, F. B. Verona, M. Innocenti & M. Napolitano. *Fault detection using neural networks*. In Neural Networks, 1994. IEEE World Congress on Computational Intelligence., 1994 IEEE International Conference on, volume 6, pages 3796–3799 vol.6, June 1994.

[Smola03] Alexander J. Smola & Risi Kondor. *Kernels and Regularization on Graphs*. In COLT, volume 2777 of *Lecture Notes in Computer Science*, pages 144–158. Springer, 2003.

[Sun06] Pei Sun, Sanjay Chawla & Bavani Arunasalam. *Mining for Outliers in Sequential Databases*. In SDM, pages 94–105. SIAM, 2006. 16

[Szymanski04] B. K. Szymanski & Y. Zhang. *Recursive data mining for masquerade detection and author identification*. In Proceedings from the Fifth Annual IEEE SMC Information Assurance Workshop, 2004., pages 424–431, June 2004. 13, 31

[Tang13] Jiliang Tang & Huan Liu. *CoSelect : Feature Selection with Instance Selection for Social Media Data*. In SDM, pages 695–703. SIAM, 2013. 77

[Vintsyuk68] T. K. Vintsyuk. *Speech discrimination by dynamic programming*. Cybernetics, vol. 4, no. 1, pages 52–57, Jan 1968. 19, 36

[Vlachos02] Michail Vlachos, Dimitrios Gunopulos & George Kollios. *Discovering Similar Multidimensional Trajectories*. In ICDE, pages 673–684. IEEE Computer Society, 2002. 23

[Warren Liao05] T Warren Liao. *Clustering Time Series Data â A Survey*. vol. 38, pages 1857–1874, 11 2005. 25

[Warrender99] C. Warrender, S. Forrest & B. Pearlmutter. *Detecting intrusions using system calls : alternative data models*. In Proceedings of the 1999 IEEE Symposium on Security and Privacy (Cat. No.99CB36344), pages 133–145, 1999. 31

[Wei05] Li Wei, Nitin Kumar, Venkata Nishanth Lolla, Eamonn J. Keogh, Stefano Lonardi & Chotirat (Ann) Ratanamahatana. *Assumption-Free Anomaly Detection in Time Series*. In SSDBM, pages 237–240, 2005.

[Wei06] L. Wei, E. Keogh & X. Xi. *SAXually Explicit Images : Finding Unusual Shapes*. In Sixth International Conference on Data Mining (ICDM'06), pages 711–720, Dec 2006. 31

[Williams07] Andrew W. Williams, Soila M. Pertet & Priya Narasimhan. *Tiresias : Black-Box Failure Prediction in Distributed Systems*. In IPDPS, pages 1–8. IEEE, 2007.

[Yang03] J. Yang & W. Wang. *CLUSEQ : efficient and effective sequence clustering*. In Proceedings 19th International Conference on Data Engineering (Cat. No.03CH37405), pages 101–112, March 2003. 16

[Yankov08] Dragomir Yankov, Eamonn Keogh & Umaa Rebbapragada. *Disk Aware Discord Discovery : Finding Unusual Time Series in Terabyte Sized Datasets*. Knowl. Inf. Syst., vol. 17, no. 2, pages 241–262, Springer-Verlag New York, Inc., New York, NY, USA, November 2008. 30

[Ye00] Nong Ye. *A Markov Chain Model of Temporal Behavior for Anomaly Detection*. In In Proceedings of the 2000 IEEE Workshop on Information Assurance and Security, pages 171–174, 2000.

[Yeung02] Dit-Yan Yeung & C. Chow. *Parzen-window network intrusion detectors*. In Object recognition supported by user interaction for service robots, volume 4, pages 385–388 vol.4, 2002. 58

[Yuxiang05] Sun Yuxiang, Xie Kunqing, Ma Xiujun, Jin Xingxing, Pu Wen & Gao Xiaoping. *Detecting spatio-temporal outliers in climate dataset : a method study*. In Proceedings. 2005 IEEE International Geoscience and Remote Sensing Symposium, 2005. IGARSS '05., volume 2, pages 4 pp.–, July 2005. 30

[Zaharia10] Matei Zaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker & Ion Stoica. *Spark : Cluster Computing with Working Sets*. In Proceedings of the 2Nd USENIX Conference on Hot Topics in Cloud Computing, HotCloud'10, pages 10–10, Berkeley, CA, USA, 2010. USENIX Association. x, 96, 97, 98

[Zhang03] Xiaoqiang Zhang, Pingzhi Fan & Zhongliang Zhu. *A new anomaly detection method based on hierarchical HMM*. In Proceedings of the Fourth International Conference on Parallel and Distributed Computing, Applications and Technologies, pages 249–252, Aug 2003. 16

[Zhang10] Yang Zhang, Nirvana Meratnia & Paul J. M. Havinga. *Outlier Detection Techniques for Wireless Sensor Networks : A Survey*. IEEE Communications Surveys and Tutorials, vol. 12, no. 2, pages 159–170, 2010.

[Zhou05] Dengyong Zhou, Jiayuan Huang & Bernhard Schölkopf. *Learning from labeled and unlabeled data on a directed graph*. In ICML, volume 119 of *ACM International Conference Proceeding Series*, pages 1036–1043. ACM, 2005.