Contribution au pronostic de pannes d'actionneurs électromécaniques : application en milieu aéronautique

Alexandre Eid

▶ **To cite this version:**

Alexandre Eid. Contribution au pronostic de pannes d'actionneurs électromécaniques : application en milieu aéronautique. Energie électrique. Université de Lyon, 2022. Français. NNT : 2022LYSE1022 . tel-04725059

HAL Id: tel-04725059
https://theses.hal.science/tel-04725059v1

Submitted on 8 Oct 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

N° d'ordre NNT : **2022LYSE1022**

THÈSE de DOCTORAT DE L'UNIVERSITÉ DE LYON
Opérée au sein de :
l'université Claude-Bernard — Lyon 1

École doctorale n°160
Électronique, électrotechnique, automatique et traitement du signal

Spécialité de doctorat : Génie électrique

Soutenue à huit clos le 17/03/2022, par :
Alexandre EID

Contribution au pronostic de pannes d'actionneurs électromécaniques

Application en milieu aéronautique

Devant le jury composé de :

Marie-Cécile PÉRA Professeur des universités (HDR), université de Franche-Comté / Institut de recherche FEMTO-ST	**Présidente**
Mohamed BENBOUZID Professeur des universités (HDR), université de Brest / IRDL — CNRS UMR 6027	**Rapporteur**
Mitra FOULADIRAD Professeur des universités (HDR), École Centrale Marseille	**Rapporteur**
Nicolas BONNEEL Chargé de recherche CNRS (HDR), université de Lyon / LIRIS — CNRS UMR 5205	**Examinateur**
Laetitia CHAPEL Maître de conférence, université de Bretagne-Sud / IRISA — CNRS UMR 6074	**Examinatrice**
Hubert RAZIK Professeur des universités, université de Lyon / Ampère — CNRS UMR 5005	**Examinateur**
Guy CLERC Professeur des universités, université de Lyon / Ampère — CNRS UMR 5005	**Directeur de thèse**
Badr MANSOURI Ph.D. Ingénieur systèmes asservis, Safran Electronics & Defense	**Invité (encadrant)**
Babak NAHIDMOBARAKEH Professeur des universités (HDR), université de Lorraine / GREEN	**Invité**
Noureddine TAKORABET Professeur des universités (HDR), université de Lorraine / GREEN	**Invité**

Université Claude Bernard - Lyon 1

Président de l'université	M. Frédéric Fleury
Président du conseil académique	M. Hamda ben Hadid
Vice-Président du conseil d'administration	M. Didier Revel
Vice-Président du conseil des études de la vie universitaire	Mme Céline Brochier
Vice-Président de la commission recherche	M. Petru Mironescu
Directeur général des services	M. Pierre Rolland

Composantes santé

Département de formation et centre de recherche en biologie humaine	Mme Anne-Marie Schott	Directrice
Faculté d'ontologie	Mme Dominique Seux	Doyenne
Faculté de médecine et maïeutioque Lyon-Sud — Charles Mérieux	Mme Carole Burillon	Doyenne
Faculté de médecine Lyon-Est	M. Gilles Rode	Doyen
Institut des Sciences et Techniques de la Réadaptation (ISTR)	M. Xavier Perrot	Directeur
Institut des Sciences Pharmaceutiques et Biologiques (ISBP)	M. Claude Dussart	Directeur

Composantes & départements de sciences & technologies

Département Génie Électrique et des Procédés (GEP)	Mme Rosaria Ferrigno	Directrice
Département informatique	M. Behzad Shariat	Directeur
Département mécanique	M. Marc Buffat	Directeur
École supérieure de Chimie, Physique, Électronique (CPE Lyon)	M. Gérard Pignault	Directeur
Institut de Science Financière et d'Assurances (ISFA)	M. Nicolas Leboisne	Directeur
Institut national du professorat et de l'éducation	M. Pierre Chareyron	Directeur
Institut universitaire de technologie de Lyon 1	M. Christophe Viton	Directeur
Observatoire de Lyon	Mme Isabelle Daniel	Directrice
Polytechnique Lyon	M. Emmanuel Perrin	Directeur
UFR biosciences	Mme Kathrin Gieseler	Administratrice provisoire
UFR des Sciences et Techniques des Activités Physiques et Sportives (STAPS)	M. Yannick Vanpoulle	Directeur
UFR faculté des sciences	M. Bruno Andrioletti	Directeur

À ma femme, Camille.

Ce que l'on conçoit bien s'énonce clairement,
Et les mots pour le dire arrivent aisément.
(…)
Travaillez à loisir, quelque ordre qui vous presse.
Et ne vous piquez point d'une folle vitesse :
(…)
Hâtez-vous lentement ; et, sans perdre courage,
Vingt fois sur le métier remettez votre ouvrage :
Polissez-le sans cesse et le repolissez ;
Ajoutez quelquefois, et souvent effacez.

Boileau, *L'Art poétique*, Chant I

Table des matières

Table des figures

Liste des tableaux

Liste des hypothèses

Glossaire

ADADELTA Adaptive Learning Rate Method. 26

ADAGRAD Adaptive Subgradient Method. 26

AE Auto-Encoder. 19, 28, 32, 33, 34, 39, 40, 41, 43, 44, 49, 55, 88

ATO Aborted Take-Off. 6

BGA Ball Grid Array. 86

BM Boltzmann Machine. 33, 34

BM Brownian Motion. 90

BPNN Back Propagation Neural Network. 50

CCPF Constant Continuous Piecewise Function. 66, 67

CID Complexity Invariant Distance. 89

C-MAPSS Commercial Modular Aero-Propulsion System Simulation. 64, 86, 87

CBM Condition-Based Maintenance. 16

CDBN Convolutional Deep Belief Network. 49

CDBN-CS Convolutional Deep Belief Network with Compressed Sensing. X, 48, 49, 50

CNN Convolutional Neural Network. 19, 28, 31, 32, 36, 39, 40, 41, 42, 43, 48, 49, 52, 54

CNN-CF Convolutional Neural Network with Class speficied Fully connected components. 55

COCO Common Object in Context. 65

CS Compressed Sensing. 49

CTO Chief Technical Officer. 3

CVA Canonical Variate Analysis. 88

DAE Denoising Auto-Encoder. 40, 44, 45, 46

DBM Deep Boltzmann Machine. 35, 40, 41

DBN Deep Belief Network. 28, 35, 39, 40, 47, 48, 49, 55

DCAE Deep Convolutional Auto-Encoder. 55

DL Deep Learning. 19

DNN Deep Neural Network. 43, 55

DTW Dynamic Time Wrapping. 55, 89, 91, 92, 137, 138

ECS-DBN Evolutionary Cost Sensitive Deep Belief Network. 50, 51

EDAE Ensemble Deep Auto-Encoder. X, 44, 45

EDEA Ensemble Deep Auto-Encoder. 46, 47

EMD Empirical Mode Decomposition. 50

E-TRAS Electric Thrust Reverser Actuation Systems. IX, 4, 6, 7, 8, 9, 56, 57, 76, 136, 137, 138, 139

EvT Evidence Theory. 89

FLMSN Fédérationh Lyonnaise de Modélisation et Sciences du Numérique. 97

GAF Gramiam Angular Field. 54

GAN Generative Adversarial Network. 20, 140

GASF Gramiam Angular Summation Fields. 54

GGM-VAE GRU-based Gaussian Mixture Variational Auto-Encoder. 55

GIGO Garbage In, Garbage Out. 12

GM Gaussian Mixture. 19

GPR Gaussian Process Regression. 19

GPU Graphics Processing Unit. 31, 39, 67

GRU Gate Recurrent Unit. 37, 52

HI Health Indicator. 87, 88, 89, 90

HIVE-COTE Hierarchical Vote Collective of Transformation-Based Ensembles for Time Series Classification. 55

HM Health Monitoring. X, 15, 39

HPC High Performance Computing. 68, 97

HSMM Hidden Semi-Markov Model. 89

IA intelligence artificielle. 12

IATA International Air Transport Association. 6

IEA International Energy Agency. 3

ILSVRC ImageNet Large Scale Visual Recognition Challenge. 30

KDE Kernel Density Estimation. 74, 89, 139

KTST Kernel Two Sample Test. 89

LCPF Linear Continuous Piecewise Function. 66, 67

LLA Lower Left Actuator. X, XI, XII, 56, 59, 60, 61, 62, 103, 104, 106, 107, 108, 110, 111, 112, 114, 127, 128

LMD Local Mean Decomposition. 50

LR Linear Regression. 43

LRA Lower Right Actuator. XI, XII, 56, 62, 103, 104, 106, 108, 127, 128

LSTM Long Short Term Memory. X, 36, 37, 40, 52, 54, 88

MCTG Maintenance Cost Technical Group. 6

MDP Multi-state Diagnosis end Prognosis. 51

MHMS Machine Health Monitoring System. 40

MKAD Mutliple Kernel Based Anomaly Detection. 52

ML Machine Learning. 19

MLR Multiple Linear Regression. 88

MMD Maximum Mean Discrepancies. 89, 92, 120

MRO Maintenance Repair and Overhaul. 5, 6

MSAP machine synchrone à aimants permanents. 7, 56, 138, 139

MSCRED Multi-Scale Convolutional Recurrent Encoder-Decoder. 54

MTBF Mean Time Between Failure. 14, 15, 16

NADAM Nesterov-accelerated Adaptive Moment Estimation. 26

NASA National Aeronautics and Space Administration. 8

NN completly connected Neural Network. 43, 49

PARTITA Partial Time scaling Invariant Temporal Alignement. XXI, XXII, 133, 137, 138, 140

PARTITA-RULE Partial Time scaling Invariant Temporal Alignement for Remaining Useful Life Estimation. XXI, XXII, 133, 137

PCA Principal Component Analysis. 31, 32, 50, 55, 88

PF Particle Filter. 88, 89

PHM Prognostics and Health Management. IX, XII, XXI, XXII, 4, 5, 6, 8, 9, 12, 13, 14, 15, 16, 17, 18, 19, 20, 27, 33, 35, 39, 40, 41, 43, 49, 50, 52, 54, 56, 57, 58, 64, 79, 84, 86, 87, 131, 136, 137, 138, 139, 140

PLS-SVR Partial Least Square Support Vector Regression. 87

PSO Particle Swarm Optimization. 88

R2F Run To Failure. 67, 76, 87, 88, 90, 92, 127

RAM Random Access Memory. 68, 97

RBM Restricted Boltzmann Machine. 31, 33, 34, 35, 47, 88

RBPF Rao-Blackwellized Particle Filter. 89

RCM Reliability Centered Maintenance. 16

ReLU Rectified Linear Unit. 31

RF Random Forest. 19, 40, 47

RGB Red Green Blue. 30, 54, 63

RMS Root Mean Square. XI, 60, 103, 104, 106, 107, 108

RNN Recurrent Neural Network. X, 26, 28, 35, 36, 40, 88

RP Recurrent Plot. 54

RUL Remaining Useful Life. XXI, XXII, 5, 14, 15, 86, 87, 88, 89, 90, 91, 92, 94, 112, 120, 122, 127, 128, 129, 131, 132, 133, 137, 138

SAE Sparse Auto-Encoder. 43, 44, 49

SAE-DNN Sparse Auto-Encoder with Deep Neural Network. 44

SbRLP Similarity-based Residual Life Prediction. 86

SDAE Stacked Denoised Auto-Encoder. 55

SHERLOC Structural HEalth monitoring, manufacturing and Repair technologies for Life management Of Composite fuselage. 4

SPOT Sliced Partial Optimal Transport. 94, 95, 99

SQL Structured Query Language. 137

ST-CNN Spatio-Temporal Convolutional Neural Network. 43

SUNRISE Soft clUsteriNg foR tIme SEries. XXI, XXII, 80, 81, 82, 83, 133, 136, 137, 138, 139, 140

SVM Support Vector Machine. 19, 42, 43, 47, 49, 50, 52, 83, 88, 89

SVR Support Vector Regression. 19, 40, 88

TCM Tool Condition Monitoring. X, 51

TPESampler Tree-structured Parzen Estimator Sampler. 97

TRL Technology Readiness Level. 15

TSBP Trajectory Similarity-Based Prediction. XI, 86, 89

ULA Upper Left Actuator. XII, 56, 62, 106, 108, 127, 128

URA Upper Right Actuator. XII, 56, 62, 106, 107, 108, 127, 128

VRAM Video Random Access Memory. 67, 68

WPD Wavelet Packet Decomposition. 50

Remerciements

Je tiens à remercier M. Benbouzid ainsi que M^{me} Fouladirad d'avoir accepté de rapporter mes travaux sur le pronostic d'actionneurs électromécaniques en milieu aéronautique. Leur lecture attentive et détaillée m'a permis d'apporter des modifications pertinentes afin d'améliorer ce manuscrit. Je tiens aussi à remercier M^{me} Péra d'avoir accepté de présider ce jury avec un bonne humeur communicative. Je remercie également M. Bonneel pour son aide lors de la réalisation de ce projet ainsi que sa lecture attentive de cette thèse qui m'a permis d'en améliorer certains points. Je remercie M^{me} Chapel pour avoir pris le temps d'examiner ces travaux ainsi que pour nos échanges lors de la soutenance. Je remercie M. Razik d'avoir examiné mes recherches et d'avoir contribué à mon intégration au laboratoire. Enfin je remercie M. Takorabet d'avoir assisté à la présentation de cette thèse.

Parfois l'on peut penser qu'une thèse est un projet de recherche de trois années durant lesquelles le doctorant est cloîtré dans son laboratoire poussiéreux à résoudre des équations en pleine nuit. Sans grande surprise, je peux vous confirmer qu'il n'en est rien. Une thèse c'est avant tout un travail d'équipe magnifié par les interactions avec autrui. Au sein de cette équipe : Guy Clerc, Badr Mansouri, Babak Nahidmobarakeh et moi-même. Aussi, je remercie chaleureusement Babak pour son accueil au sein du laboratoire Green à Nancy et pour les échanges que nous avons pu avoir par la suite. Je remercie Badr qui m'a fait confiance depuis mon projet de fin d'étude pour mener à bien des travaux de recherche en pronostic et surveillance d'état de santé et nos échanges riches m'ont permis de mener à bien cette thèse. J'ai pu apprendre et découvrir le monde du PHM grâce à lui. Enfin, je remercie du fond du cœur Guy Clerc, mon directeur de recherche. Depuis mon arrivé au laboratoire j'ai reçu un soutien inconditionnel de sa part, et ai vu en lui un scientifique exemplaire d'une gentillesse sans bornes. J'ai pu, grâce à lui, devenir une graine de chercheur. Son usage de la maïeutique nous a permis de proposer des idées originales dans le domaine de la surveillance d'état de santé ; nous avons pu noircir ensemble quelques tableaux blancs. J'aurais encore beaucoup de choses à dire sur mon directeur mais, de peur de perdre mon fil rouge, je vais abréger en ces quelques mots : un grand merci Guy. Cette thèse a été effectuée en partie au sein de Safran Electronics & Defense, c'est pourquoi je remercie François Guillot pour sa bienveillance, son suivi, ainsi que nos échanges.

Au cours de ces trois années, j'ai eu la chance d'encadrer Stella Roux, une étudiante talentueuse sans laquelle je n'aurais pu mener à bien la refonte de la méthode de partitionnement ainsi que la rédaction d'un article de revue. Qu'elle soit remerciée ici pour sa contribution active, nos échanges, nos réunions passées à griffonner sur un tableau blanc et nos séances de *brainstorming*.

J'ai effectué ces travaux dans la magnifique ville de Lyon au sein du laboratoire Ampère sur le site de l'université Claude Bernard - Lyon 1. Sur ce site, au sein du bâtiment Oméga, se trouve une équipe hors du commun dans laquelle j'ai eu la chance d'être intégré. C'est avec eux que j'ai pu rire, partager des moments de convivialité et apprendre. Je remercie du fond du cœur Christian, Riccardo, Fabien S., Luong-Viet, Charles, Pascal, Ali, Laurent, Fabien M., Michelle, Younes et Olivier. Je remercie mes compagnons de labeur : Edgar, William, Sofia, Etienne, Rania, Marwan, Philippe, Mohamed, Loup, Claire, Jami, Clémentine. Merci à eux, merci pour nos conversations, nos pauses cafés, nos cohésions. Leur soutien m'a été indispensable. Merci à Romain, mon prédécesseur, pour son aide et nos échanges.

Un grand merci à Henri avec lequel j'ai pu réaliser mon cycle d'ingénieur, dont l'influence m'a très probablement conduit à me lancer dans un projet de recherche et grâce auquel j'ai pu m'octroyer quelques pauses bien méritées dans cet emploi du temps bien rempli.

Enfin, je n'en serais pas là sans mes parents que je remercie pour leur soutien indéfectible, pour leurs encouragements et l'aide qu'ils m'ont apportée. Il ont forgé ce que je suis aujourd'hui et je les en remercie profondément. Merci aussi à mon frère Thomas ainsi qu'à ma sœur Mélanie ; les liens qui nous unissent depuis l'enfance me sont chers et nécessaires pour entreprendre de grands projets.

Pour finir, je tiens à remercier ma femme, Camille, à laquelle je dédicace ces travaux. Le chemin parcouru pour arriver à ce manuscrit a été loin d'être simple et tu m'as toujours soutenu à bout de bras, tu as toujours été présente pour moi. Depuis les concours nous évoluons ensemble pour mon plus grand bonheur. La thèse n'a pas été une épreuve que pour moi et je te remercie de ta patience ainsi que de ton aide incommensurable. Grâce à toi, j'ai pu me concentrer sur ce que je fais le mieux : travailler, travailler, travailler. Merci, ce manuscrit marque en soi... le commencement d'un nouveau chapitre !

Résumé

Aujourd'hui, afin de répondre à l'enjeu climatique de demain, l'industrie aéronautique est en phase de transition. Elle s'est engagée à réduire ses émissions de CO_2 de 30% d'ici 2050. Au-delà de l'utilisation toujours plus importante de carburants durables, de la réduction de consommation des turboréacteurs ou encore de l'étude de systèmes de propulsions hybrides, cet engagement ne pourra être réalisé sans le développement d'actionneurs électriques. Ces derniers possèdent un meilleur rendement énergétique que les actionneurs hydrauliques et pneumatiques ainsi qu'une meilleure puissance massique. De plus, ils ne possèdent plus de liquide de compression corrosif, ce qui en plus d'être un atout environnemental, simplifie leur maintenance. Cependant, ils sont principalement réalisés à base de technologies de transmission de mouvement qui sont susceptibles de gripper. En tant que méthodologie reposant sur la surveillance et le pronostic d'état de santé en temps réel d'un système, le Prognostics and Health Management (PHM) permettrait l'adoption généralisée d'actionneurs électromécaniques en anticipant tout défaut de grippage. Ces méthodes permettent d'inférer l'état de santé d'un système ainsi que le temps de vie restant avant l'apparition d'un défaut majeur. De surcroît, au-delà du volet sécuritaire, le PHM agit aussi sur le volet économique. Ainsi, la prédiction de futurs défauts permettrait aux compagnies aériennes d'optimiser l'exploitation du produit en dimensionnant au plus juste l'immobilisation des aéronefs pour la maintenance des équipements.

Cette thèse développe une méthode de PHM complète allant du pré-traitement de la donnée à la détermination de durée de vie restante d'un inverseur de poussée électrique. Ce faisant, son objectif est de fournir un point de départ à la mise en place d'une stratégie plus globale de maintenance prédictive industrielle.

Après la définition du contexte industriel de ces travaux, nous présenterons un état de l'art sur les méthodes de PHM. La méthodologie vise à résoudre deux types de problèmes : la classification ou le partitionnement pour effectuer une étape de diagnostic et la régression couplée à une classification pour effectuer le pronostic. Depuis ces dernières années, les méthodes d'intelligence artificielle et notamment les réseaux de neurones profonds permettent d'apporter une solution plus pertinente à ces deux grandes classes de problèmes, c'est pourquoi ils serviront au développement de la méthode PHM. Des signaux vibratoires mesurés sur l'inverseur de poussée sont alors extraits des descripteurs de défauts. Ces descripteurs de défauts, ou signatures, sont des grandeurs statistiques dont les dynamiques doivent être corrélées au vieillissement du système. Cependant, ces descripteurs ne possèdent pas de labels permettant d'assigner un degré de sévérité à un point donné. Autrement dit, leur état de santé est inconnu au cours du temps. C'est pourquoi, un algorithme de partitionnement de séries temporelles est développé pour réaliser cette tâche. Chaque descripteur est encodé sous forme d'une image qui sera ensuite segmentée par un réseau de neurones profond entraîné par une génération de données artificielles. Des informations unidimensionnelles sur les frontières de groupes sont ensuite extraites et filtrées. Puis, une estimation de densité à noyau transforme le signal résultat en une densité de probabilité empirique. Enfin, un modèle de mélange gaussien extrait les composantes indépendantes de la distribution créée pour obtenir des frontières de groupes probables. Cette méthode, Soft clUsteriNg foR tIme SEries (SUNRISE), permet de révéler différents degrés de sévérité de défauts dans les données étudiées, avec leur vraisemblance respective et ce, sans connaissance *a priori* de la structure des données. De plus, afin d'évaluer les résultats de partitionnement obtenus, nous avons développé un nouvel indicateur de qualité χ. Il mesure la cohérence temporelle ainsi que les similarités des formes de groupes obtenues par le schéma de partitionnement. Une fois que la labélisation des données est effectuée, la détermination de la durée de vie restante du système est réalisée par alignement de séries temporelles par la méthode Partial Time scaling Invariant Temporal Alignement for Remaining Useful Life Estimation (PARTITA-RULE). L'idée est de déterminer l'état de santé d'un nouvel actionneur en mesurant la similarité entre ses signaux vibratoires et les précédents, contenus en base de données. Pour ce faire, une méthode d'alignement de séries temporelles est mise au point : Partial Time scaling Invariant Temporal Alignement (PARTITA). Une fois que la série inconnue est alignée sur tous les candidats possibles en base de données, un schéma de pondération est créé afin d'affecter un score de vraisemblance à chaque candidat. Enfin, un modèle de fusion résolvant un problème de transport optimal permet d'obtenir le temps de vie restant ou Remaining Useful Life (RUL) de l'actionneur avec une mesure d'incertitude. Par la suite, la robustesse de la méthode aux segments temporels incomplets est testée.

Cette thèse a permis la réalisation d'une méthode de PHM créée spécifiquement pour fonctionner en milieu industriel avec un ensemble de données d'entrées de petite taille mais qui pourrait être enrichi au fur et à mesure de l'exploitation du produit. Enfin, la méthode est robuste aux données manquantes et reste prometteuse pour un fonctionnement sous de multiples conditions opérationnelles.

Abstract

Today, to meet tomorrow's climate challenges, the aeronautics industry is in a transition phase. It has committed to reducing its CO2 emissions by 30 % by 2050. In addition to the ever-increasing use of sustainable fuels, the reduction in consumption of turbojet engines, and the study of hybrid propulsion systems, aeronautics cannot achieve this commitment without the development of electric actuators. Electric actuators are more energy-efficient than hydraulic and pneumatic actuators and have a higher power-to-weight ratio. In addition, they no longer have a corrosive compression fluid, which is an environmental advantage and simplifies their maintenance. However, they are mainly based on motion transmission technologies that are susceptible to seizure. A methodology based on real-time monitoring and prognosis of a system's health status should allow the widespread adoption of electromechanical actuators by anticipating any seizure defect. These methods enable inferring the health state of a system and the remaining lifetime before the appearance of a significant fault. In addition, beyond the safety aspect, the system also acts on the economic part. Thus, the prediction of future defects would allow airlines to optimize the operation of the product by sizing the immobilization of aircraft for equipment maintenance as accurately as possible.

This thesis develops a complete predictive maintenance method from data pre-processing to the determination of the remaining lifetime of an electric thrust reverser. In doing so, its objective is to provide a starting point for implementing a more global strategy of industrial predictive maintenance.

After defining the industrial context of this work, we will present the state of the art on methods of PHM. The methodology aims at solving two types of problems: classification or partitioning to perform a diagnostic step and regression coupled with classification to perform the prognosis. Since the last few years, artificial intelligence methods and, in particular, deep neural networks have made it possible to provide a more relevant solution to these two major classes of problems. That is why we will use artificial intelligence algorithms to develop the whole method. From vibratory signals measured on the thrust-reverser are then extracted descriptors of defects. These are statistical quantities whose dynamics are expected to be correlated with the system's aging. However, these descriptors do not have a label allowing to assign a degree of severity to a given point. In other words, the state of health is unknown over time. Therefore, a time series partitioning algorithm is developed to perform this task. Each descriptor is encoded as an image segmented by a deep neural network trained by artificial data generation. One-dimensional information on the group boundaries is then extracted and filtered. Then, a kernel density estimation transforms the resulting signal into an empirical probability density. Finally, a Gaussian mixture model extracts the independent components of the created distribution to obtain probable group boundaries. This method, SUNRISE, allows revealing different degrees of severity of faults in the studied data, with their respective likelihood, without any prior knowledge of the data structure. Moreover, to evaluate the obtained partitioning results, we have developed a new quality indicator χ. It measures the temporal consistency and the similarities of the group shape obtained by the partitioning scheme. Once labeled data are possessed, the determination of the remaining lifetime of the system is performed by time series alignment using the method PARTITA-RULE. The idea is to determine the health status of a new actuator by measuring the similarity between its vibration signals and the previous ones contained in the database. A time series alignment method is developed for this: PARTITA. Once the unknown series is aligned with all possible candidates in the database, a weighting scheme is created to assign a likelihood score to each candidate. Finally, a fusion model solving an optimal transport problem is used to obtain the remaining lifetime or RUL of the actuator with an uncertainty measure. Subsequently, the robustness of the method to incomplete time segments is tested.

This thesis resulted in a method of PHM created specifically to operate in an industrial setting with a small input dataset but which could be enriched as the product is used. Finally, the method is robust to missing data and remains promising under multiple operational conditions.

Chapitre 1

Introduction et contexte

1.1 Le domaine aéronautique en mutation pour répondre aux défis de demain

Aujourd'hui, afin de limiter l'augmentation globale de la température terrestre sous le seuil des 2 °C, la communauté internationale souhaite diminuer, entre autres choses, la production de gaz à effet de serre. Cette volonté avait déjà été affirmée, sans objectif chiffré cependant, par l'accord de Kyoto [UNFCCC, 1997] ratifié en décembre 1997 par les Nations unies avec une prise d'effet le 16 février 2005. Ce rapport impose la création d'instances mesurant les émissions au niveau nationale, la taxonomie des différents secteurs à caractériser, ainsi que la définition de l'ensemble des gaz à mesurer. L'ensemble des gaz considérés comme *à effet de serre*, de ce fait influant sur le réchauffement climatique, est constitué de dioxyde de carbone (CO_2), de méthane (CH_4), de protoxyde d'azote (N_2O), d'hydrofluorocarbures (HFC), de perfluocarbures (PFC) ainsi que d'hexafluorure de souffre (SF_6). L'estimation de la production de ces gaz est communiquée annuellement aux Nations unies par chaque nation signataire et c'est sur ces données que s'appuient les résultats qui seront présentés ici. Ainsi, la figure 1.1 permet d'obtenir une vision d'ensemble de l'année 2016 en fonction des secteurs ratifiés.

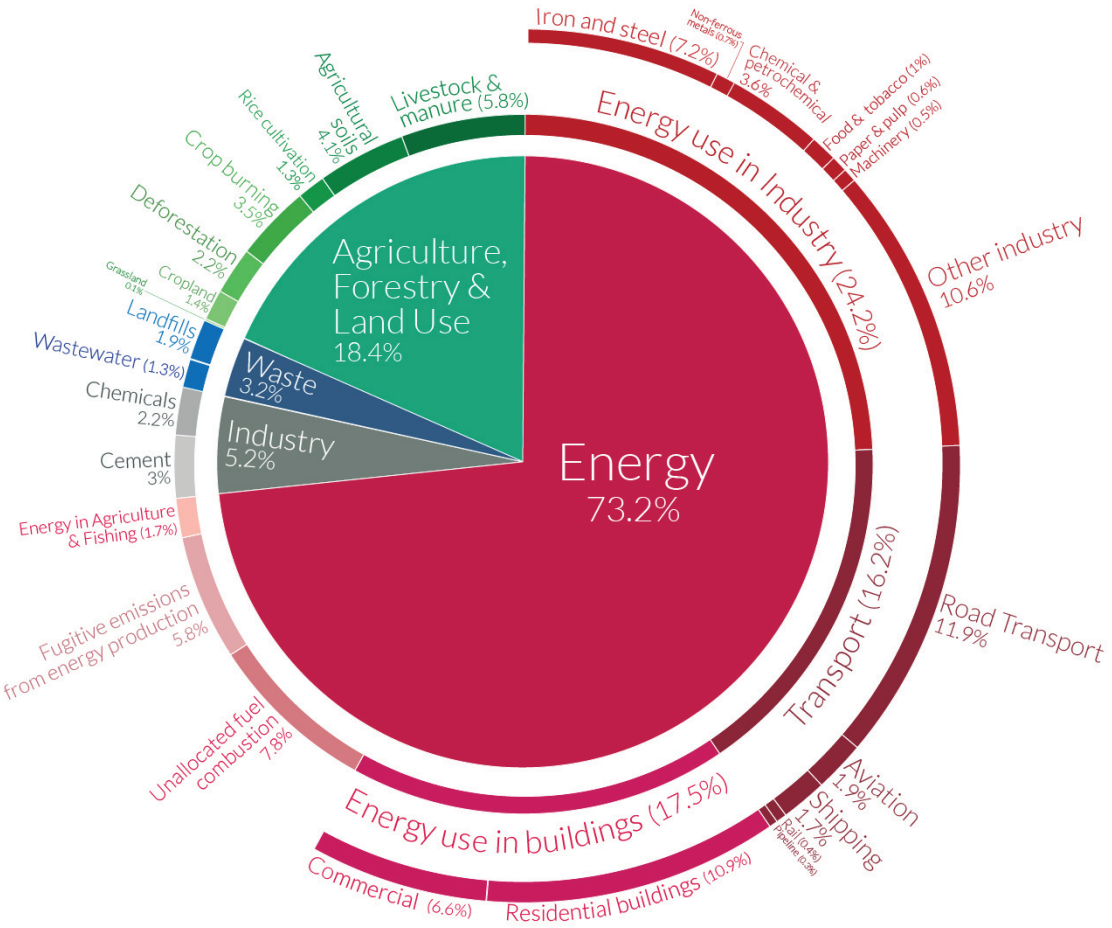


FIGURE 1.1 – Émissions mondiales de gaz à effet de serre par secteur, rapportées en 2016, représentant 49, 4 milliards de tonnes d'équivalent CO_2 (graphique modifié d'après [RITCHIE, 2020])

Notons que la mesure des émissions de gaz à effet de serres est un problème complexe. Aussi, l'information pertinente à extraire de la figure 1.1 est contenue dans les proportions relatives des secteurs les uns par rapport aux autres. En effet, selon [GÜTSCHOW, JEFFERY *et al.*, 2016] avec le dernier ensemble de données [GÜTSCHOW, GÜNTHER et PFLÜGER, 2021] couvrant les années 2016 et 2018, le secteur de l'énergie représente 74 % des émissions de gaz à effet de serre, soit respectivement 34.60 Gt et 35.60 Gt. En 2019, il représente à 73.7 % des émissions totales soient 35.60 Gt. C'est pourquoi les chiffres présentés par la figure 1.1 pour 2016 affectent au domaine de l'énergie 73, 2 % des parts des émissions totales. Les données utilisées par [RITCHIE, 2020] sont issues, quant à elles, de deux ensembles différents de données d'où les quelques disparités avec les

chiffres annoncés précédemment. Dans un premier temps, une version antérieure de [Gütschow, Günther et Pflüger, 2021] était utilisée et l'évaluation était alors de 75 % en 2016 et 2018. De plus, *Climate Watch* a créé son propre ensemble de données [Watch, 2020], qui affecte 76 % pour les années 2016 et 2018. Au final, l'apport du domaine de l'énergie est légèrement majoré par rapport aux données présentes dans la figure 1.1. Il est alors nécessaire de conserver un certain recul dans l'analyse plus fine des émissions imputées au domaine du transport et plus précisément, au domaine de ces travaux : l'aéronautique.

Tandis que la figure 1.1 s'intéressait aux proportions de rejets de gaz à effets de serre au niveau mondial, la figure 1.2 présente la contribution détaillée du secteur des transports au niveau des émissions de CO_2. Les données sont issues des relevés de l'International Energy Agency (IEA) [IEA, 2021].

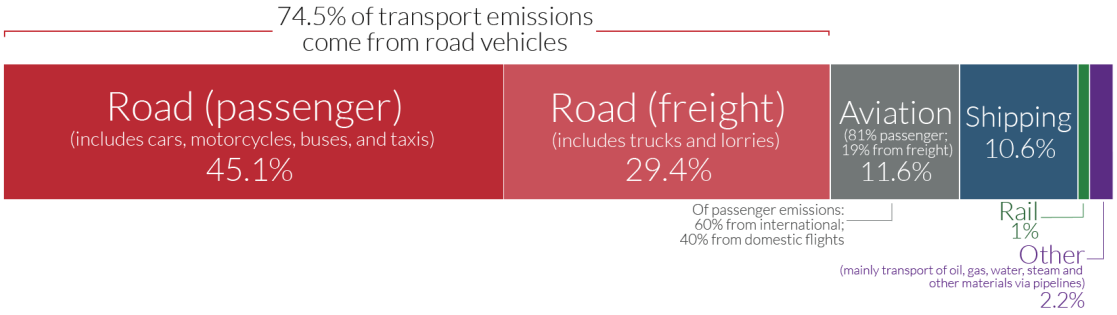


Figure 1.2 – Émissions mondiales de CO_2 imputées au domaine des transports, rapportées en 2018, représentant 8 Gt de CO_2 soit 24 % des émissions du pôle *énergie* (graphique modifié d'après [Ritchie et Roser, 2021])

Ainsi, pour le domaine aéronautique, ce sont donc 93.9 Mt de CO_2 équivalent rejetés dans l'atmosphère pour l'année 2016. Dans les 1.92 Gt de CO_2 comptabilisées en 2018 par le domaine des transports, sont imputées à l'aéronautique 223 Mt de CO_2. Notons la différence majeure entre les chiffres calculés à partir des émissions de gaz à effet de serre selon l'accord de Kyoto [UNFCCC, 1997]. Ainsi, l'aviation ne produit dans les deux cas de figure (présentés par la figure 1.1 et la figure 1.2) qu'une faible partie des émissions imputées, en comparaison avec les autres moyens de transport. Leurs émissions sont bien en deça des émissions générées par le transport routier et produit autant que le domaine du transport maritime international (voir les deux figures 1.1 et 1.2). Une étude plus détaillée sur la ventilation des différents pôles d'intérêt dans ces calculs globaux a été rédigée par [Graver, K. Zhang et Rutherford, 2019]. Remarquons tout de même que l'étude ne porte ici que sur l'émission de gaz à effet de serre. D'autres phénomènes comme, par exemple, les traînées de condensation, ne sont pas présentés ici. En effet, comme le montre la figure 1.3, bien que le CO_2 représente la grande majorité des émissions provenant du flux d'air chaud d'un turboréacteur, la vapeur d'eau est aussi présente en grande quantité.

Bien que limitée par rapport aux autres domaines industriels mesurés, la contribution du domaine aéronautique au réchauffement climatique n'en est pas moins bien réelle, c'est pourquoi la filière est dans une phase de transition et prend des engagements pour répondre au défi climatique. C'est dans ce contexte que les sept Chief Technical Officer (CTO) d'acteurs aéronautiques majeurs (Airbus, Boeing, Dassault Aviation, GE Aviation, Rolls Royce, **Safran** et Pratt & Whitney) ont signé le 26 octobre 2021 une déclaration commune [SafranGroup, 2021] visant notamment à développer les nouvelles technologies embarquées dans les aéronefs pour permettre une réduction de l'impact carbone du secteur aérien. De plus, Safran est un acteur majeur du projet *Clean Sky* 2 qui a pour objectif notamment, de réduire les émissions de CO_2 de 20 % à 30 % (par rapport au meilleur aéronef en 2014) à l'horizon 2050 [CleanSky, 2021].

L'actionnement électrique fait partie des axes de recherche privilégiés du secteur industriel pour créer une aviation plus respectueuse de l'environnement. En effet, partie intégrante de l'avion plus électrique, les actionneurs électromécaniques possèdent plusieurs avantages par rapport à leurs homologues pneumatiques ou hydrauliques. Ils sont dans un premier temps plus légers que les systèmes usuels, ce qui permet de diminuer la masse embarquée de l'aéronef et par ce fait, de réaliser des économies de carburant. Ainsi, d'après [Geffrier, 2021], le remplacement du système de freinage hydraulique de l'Airbus A220 par un système électrique permet un gain de masse de 500 kg, ce qui permet à cet appareil de consommer environ 25 % de carburant en moins que l'A319, un avion

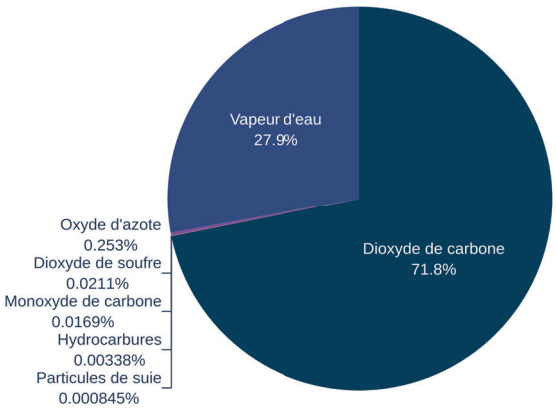


FIGURE 1.3 – Composés issus de la combustion de 850 t d'air avec 2 700kg pour un avion standard à deux turboréacteurs durant une heure de vol avec 150 passagers, d'après [EUROPEAN AVIATION SAFETY AGENCY. et EAA., 2019, figure 1.8] (les proportions ont été calculées pour un flux d'air chaud de 130 t en sortie de turboréacteur)

de même calibre, pour une même distance et un même nombre de passagers. Dans le domaine des commandes de vol, d'après [GOMEZ *et al.*, 2017], l'inverseur de poussée innovant *O-Duct* Electric Thrust Reverser Actuation Systems (E-TRAS) présent dans la nacelle du COMAC C919 ferait économiser 0, 5 % de carburant par rapport à ses concurrents hydrauliques. Au-delà du poids, ces systèmes possèdent aussi un design plus compact ce qui permet d'embarquer une plus grande puissance dans un même volume. De plus, contrairement à leurs homologues hydrauliques, par nature, les systèmes électromécaniques n'ont pas besoin de liquide de compression. Or, ces liquides doivent conserver leurs propriétés sur de larges plages de températures et surtout avoir une température d'inflammabilité très haute, comme le *Skydrol*. Toutes ces propriétés étant difficilement présentes dans la nature, ce sont des composés de synthèse qui sont corrosifs, dangereux pour l'homme et l'environnement. En s'en affranchissant, les actionneurs électriques permettent la création de systèmes plus respectueux de l'environnement et la simplification des procédures de maintenance pour les compagnies aériennes. De plus, mieux intégrés dans le harnais électrique de l'aéronef, ce type d'actionnement est propice à la mise en place de capteurs et circuits de surveillance. Cependant, les systèmes électromécaniques sont sujets à de potentiels défauts non rencontrés dans les systèmes actuels pneumatiques ou hydrauliques, notamment les défauts de grippage. Pour un actionneur de commande de vol primaire (ou secondaire), cet événement redouté est à éviter par tous les moyens. Il convient alors de mettre en place des stratégies de surveillance pour pallier l'apparition de ces potentiels défauts.

1.2 L'approche du PHM (*Prognostics & Health Management*) en aéronautique

Ces stratégies de surveillances sont développées au sein d'un ensemble de méthodes, parties intégrantes du PHM. Elles sont aujourd'hui principalement utilisées dans l'industrie aéronautique pour s'enquérir de l'état de santé de turboréacteurs. Plus récemment, le projet Européen *Clean Sky* 2 soutient le projet de recherche Structural HEalth monitoring, manufacturing and Repair technologies for Life management Of Composite fuselage (SHERLOC) avec un budget de 9 294 198€ qui prendra fin en juin 2022 [*SHERLOC – SHERLOC Project* 2020]. Ce projet vise à créer un ensemble complet de méthodes allant de la captation de données à la prise de décision, pour surveiller et prédire l'état de santé des structures composites qui composent le fuselage ainsi que les ailes d'un avion.

Le PHM est une méthodologie de surveillance et de prédiction d'état de santé plutôt adaptée aux systèmes industriels, soit un système à surveiller sur lequel sont placés des capteurs. Des

informations issues de capteurs seront extraites des signaux représentatifs de l'état en temps réel du système étudié. À partir de la valeur de ces signaux, une déviation à un état nominal est calculée afin de pouvoir déterminer l'état de santé actuel du système et, dans un second temps, son temps de vie restant. D'après la norme ISO 13372 : 2012 [Organisatsion Internationale de Normalisation, 2012, section 10.2], le résultat du pronostic est alors défini comme "l'estimation de la durée de fonctionnement avant défaillance et du risque pour un ou plusieurs modes de défaillance naissant". Ainsi, l'information obtenue à l'issue du processus est normalement double. Elle est constitué du temps de vie restant — RUL ainsi qu'une mesure de confiance. Cette mesure de confiance quantifie le risque associé à cette estimation. Le niveau de confiance est quant à lui redéfini dans la norme ISO 13381-1 :2015 [Normalisation, 2015, section 3.3] comme le "chiffre indiquant le degré de certitude que le diagnostic/le pronostic est correct". Cette information est alors utilisée pour la gestion de l'état de santé (*Health Management*) du système où des décisions sont prises quant à la mise en maintenance de la pièce. C'est pourquoi le PHM permettrait d'avoir des systèmes plus sûrs en exploitation. L'ensemble des étapes de l'approche peut être résumé par la figure 1.4.

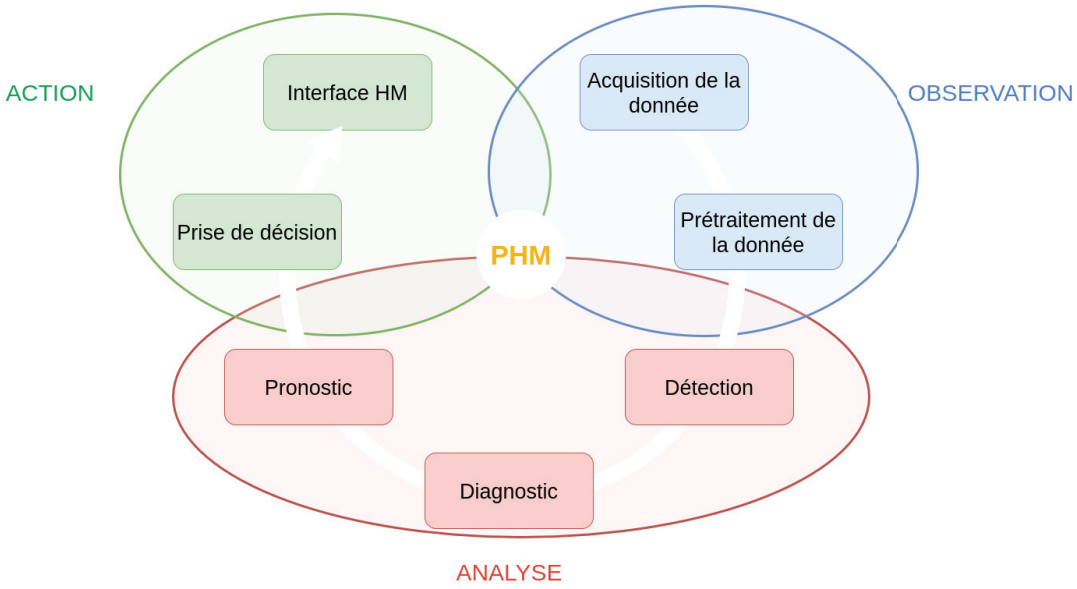


Figure 1.4 – Différentes étapes du processus de PHM, d'après [Atamuradov *et al.*, 2017, figure 1]

Cet ensemble de méthodes, ce *framework*, présente un réel intérêt économique pour les compagnies aériennes. En effet, en Europe, [AIRMES, 2019] estime à $2,8$ milliards d'euros annuel le coût d'incidents techniques imprévus sur aéronefs, c'est pourquoi les compagnies aériennes ont un réel intérêt à limiter ou prédire l'occurrence de défauts dans les systèmes [Koornneef, Verhagen et Curran, 2020]. Au-delà des maintenance imprévues, suite à avaries, la fréquence mal calculée des opérations de Maintenance Repair and Overhaul (MRO) et, plus particulièrement, un nombre de maintenance excessif résulte pour la compagnie en un pôle important de dépenses ainsi qu'une immobilisation non pertinente de l'aéronef d'où l'intérêt de pouvoir estimer au mieux ces intervalles [Mofokeng, Mativenga et Marnewick, 2020]. Cette dernière approche peut être notamment le résultat de maintenances préventives. La maintenance *préventive* est définie par la norme ISO 13372 :2012 [Organisatsion Internationale de Normalisation, 2012, p. 1.18] comme étant une "maintenance effectuée selon un calendrier établi ou selon des critères prescrits qui détectent ou qui empêchent la dégradation d'une structure fonctionnelle", contrairement à la maintenance *prédictive* qui détermine la nécessité de maintenance en fonction des résultats de pronostic du système. De fait, sous couvert d'un algorithme de pronostic qui produirait peu de faux positifs, le PHM permettrait de mieux anticiper les maintenances nécessaires et *de facto* de mieux estimer les intervalles d'immobilisation de l'aéronef. Après retour d'expérience, le problème de l'estimation se pose car les maintenances non planifiées dépendent de la proba-

bilité d'occurrence de défauts et sont donc, par nature, stochastiques [Dinis, Barbosa-Póvoa et Teixeira, 2019a] [Dinis, Barbosa-Póvoa et Teixeira, 2019b]. L'intérêt du PHM dans ce cas est de pouvoir mieux prédire les potentiels défauts et ainsi diminuer le taux de maintenances imprévues [Nicchiotti, 2018].

Selon le groupe technique des coûts de maintenance — Maintenance Cost Technical Group (MCTG) de l'Association internationale du transport aérien — International Air Transport Association (IATA), 69 milliards de dollars US ont été dépensés en maintenance, réparation et révision — MRO en 2018 pour une flotte de 27 535 avions provenant de 54 compagnies aériennes [MCTG, 2019]. Dans son rapport prospectif sur l'étude du marché MRO [Cooper *et al.*, 2021] estime un forte croissance du marché pour les trois prochaines années à venir, poussée par les reports de maintenance dûs à la pandémie. En 2030, [Cooper *et al.*, 2021] évalue ainsi le marché à 130 milliards de dollars US.

Au-delà de l'intérêt économique, le PHM s'inscrit dans une démarche de développement durable pour Safran Electronics & Defense. En effet, de fait la gestion du cycle de vie du produit en opération est optimisée et les données obtenues lors de l'utilisation du produit sur aéronefs pourront servir lors de la phase de conception d'actionneurs pour dimensionner au mieux les systèmes en fonction des défauts déjà rencontrés. De ce fait, un cercle vertueux est créée pour développer des systèmes intelligents toujours plus sûrs. C'est en cela que le PHM pourrait permettre l'adoption d'actionneurs électromécaniques en masse et de ce fait, participer à la réduction de l'impact carbone du domaine aéronautique.

1.3 Présentation du système d'étude

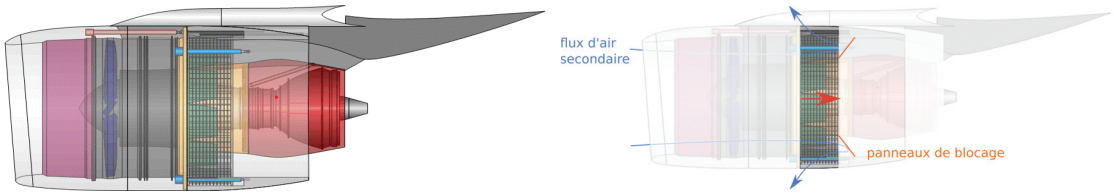
Ces travaux se concentreront sur l'étude de signaux vibratoires provenant d'un inverseur de poussée électrique : l'E-TRAS produit par Safran Electronics & Defense. L'inverseur de poussée, ou également appelé « aérofrein » permet d'améliorer les performances de freinage d'un avion lors d'un décollage avorté — Aborted Take-Off (ATO) ou, plus couramment, d'un atterrissage sur piste courte, mouillée, enneigée ou verglacée.

(a) Inverseur de poussée en position déploiement lors de l'atterrissage d'un A330 *neo* ©Adrien Daste

(b) Nacelle d'A330neo avec inverseur de poussée en position ouverte ©Christel Sasso

Figure 1.5 – Inverseur de poussée à grille sur A330neo

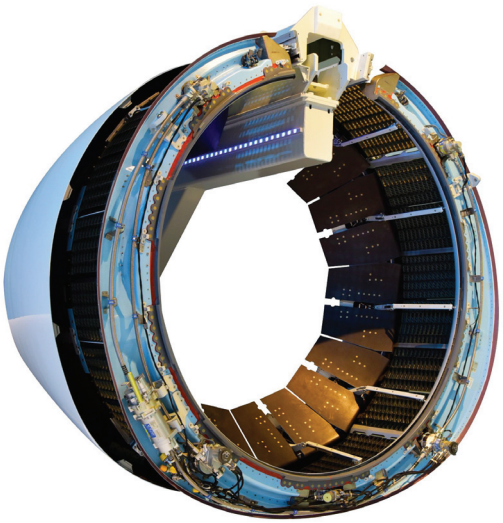
Dans un turboréacteur, le flux d'air entrant se divise en deux parties. La première partie, le flux d'air primaire, traverse le compresseur basse pression ainsi que tout le réacteur pour créer la poussée principale. Le flux d'air secondaire, quant à lui, est accéléré par le compresseur basse pression pour générer une poussée additionnelle. C'est ce flux d'air qui est dévié par l'inverseur de poussée. Comme représenté par la figure 1.5, les inverseurs de poussée étudiés ici sont des inverseurs à grille. Ainsi, à la suite de l'ordre d'actionnement des aérofreins, un capot mobile coulisse le long de la nacelle pour dévoiler les aubes d'une grille de déviation comme le montre la figure 1.6b. Dans le même temps, des panneaux de blocage se déploient à l'intérieur du canal d'éjection pour empêcher l'écoulement de l'air à travers le circuit secondaire. Dès lors, le flux est dévié et éjecté à travers les grilles mises à jour. Une force contraire à la poussée principale des turboréacteurs est créée, d'où la qualification d'*inverseur de poussée*. Notons que, de par sa fonction, un tel système ne doit jamais s'ouvrir en plein vol, c'est pourquoi des loquets de sécurité maintiennent la partie mobile de la nacelle lors du parcours de l'avion.

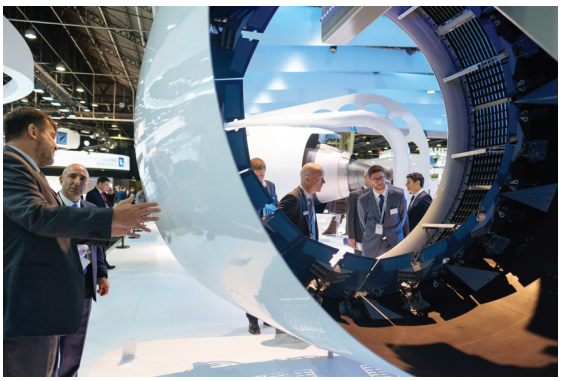


(a) Nacelle fermée (b) Nacelle ouverte

FIGURE 1.6 – Schéma d'une nacelle avec inverseur de poussée (actionneurs en bleu) en position fermée (gauche) et ouverte (droite)

Dans la famille des inverseurs de poussée, Safran Electronics & Defense innove en proposant un inverseur de poussée électromécanique *O-Duct* : l'E-TRAS. Ces travaux se concentrent sur l'étude de l'E-TRAS développé pour le C919, un bi-réacteur moyen courrier de l'avionneur chinois *Comac*. La figure 1.7





(b) E-TRAS vue inversée avec volets de déviation du flux d'air, grille et actionnement, ©Cyril Abad

(a) E-TRAS intégré sur la partie arrière de la nacelle d'un COMAC C919, ©Eric Drouin

FIGURE 1.7 – E-TRAS au salon du Bourget 2017

Comme la figure 1.7a le montre, l'E-TRAS est une structure composée de quatre actionneurs linéaires, des vis à billes, mis en mouvement par l'intermédiaire de liaisons mécaniques souples par une machine synchrone à aimants permanents (MSAP). La MSAP étant repérée sur l'image 1.7a en bas à gauche de la nacelle avec un élément jaune. Ainsi, lorsque l'axe de la MSAP effectue un mouvement de rotation, celui-ci est transmis par les liaisons souples puis transformé en déplacement linéaire par les vis à billes. La partie mobile de la nacelle étant solidaire des écrous de chaque vis à bille, elle translatera pour laisser apparaître la grille d'échappement du flux d'air secondaire. Les deux figures 1.7a et 1.7b font apparaître les volets de déviation interne du flux d'air.

Les données récoltées qui seront étudiées dans ces travaux, ont été enregistrées sur un système E-TRAS pour lequel les quatre actionneurs électromécaniques ont été poussés jusqu'à fin de vie, autrement dit, jusqu'à ce que le système ne puisse plus remplir sa fonction. Dans notre cas, la nacelle ne pouvait plus translater suite à un grippage des vis à billes. Pour réaliser les essais de vieillissement sur le système complet, ce dernier a été monté sur une structure de type *iron bird* faisant office de nacelle. Pour simuler les charges aérodynamiques qu'auront à subir le système, des vérins de contre-charge sont liés à chacune des quatre vis à billes. Le but de la procédure était de détecter tout précurseur d'un défaut de blocage dans chaque vis à billes C'est pourquoi un accéléromètre a été placé sur chaque structure surveillée. Cette configuration a permis de collecter les signaux de vibration provenant de l'axe longitudinal du dispositif d'actionnement. Ce sont les

signaux renvoyés par ces derniers qui seront utilisé dans ces travaux, comme il sera détaillé à la section 3.2. À partir de ces mesures le but de ces travaux est de déterminer le temps de vie restant de l'E-TRAS en simulant une captation incomplète de données.

1.4 Principaux verrous scientifiques

Fortement ancrés dans le domaine aéronautique industriel, ces travaux en partagent les contraintes. Ainsi, la plupart des verrous scientifiques de cette thèse sont inhérents aux problématiques du domaine. En particulier, le premier des verrous est le manque de données représentatives de vieillissement.

Verrou 1. *Manque de données d'exploitations en volume et en qualité pour l'étude des dégradations d'actionneurs électromécaniques.*

Le verrou 1 est partagé par l'ensemble du domaine de PHM, c'est pourquoi la plupart des travaux sont réalisés avec des données provenant de bancs de tests académiques ou encore de données simulées produites principalement par le centre d'excellence sur le pronostic de la National Aeronautics and Space Administration (NASA). Pour répondre au mieux aux problématiques industrielles, nous avons pris le partie de travailler uniquement avec les données d'exploitation récoltées sur E-TRAS. En plus de la quantité de données disponibles, l'aéronef sera plongé dans différents environnements au cours de son exploitation, *de facto* les systèmes le composant aussi. Ainsi, l'inverseur de poussée doit par exemple fonctionner sur une plage de température allant de $-55\,°C$ à $125\,°C$. De plus, lors des différents trajets, la charge aérodynamique s'exerçant sur le système ne sera pas constante, à la différence d'un scénario d'essais, d'où le verrou 2.

Verrou 2. *Modification des conditions opérationnelles des actionneurs en usage dans l'aéronef.*

Enfin, la perspective de ses travaux étant d'être implémentés sur cible en exploitation, il convient de prendre en compte la problématique d'implémentabilité de la méthode développée, d'où le verrou 3.

Verrou 3. *Contrainte d'applicabilité et de mise à l'échelle en production des algorithmes développés.*

L'ensemble de ces verrous scientifiques n'appelant pas une solution simple qui tiendrait dans les quelques lignes de ce paragraphe, ils ont permis de donner une direction privilégiée à ces travaux. Ainsi, les méthodes développées aux chapitres 3 et 4 l'ont été pour apporter un élément de réponse à ces verrous. Ces derniers seront repris dans la discussion des résultats obtenus, résumée à la conclusion 5 de ce mémoire.

1.5 Structure du manuscrit

Ce manuscrit de thèse est organisé en trois grandes parties principales. Une première avec le chapitre 2 concernant un état de l'art sur les méthodes de pronostic et les deux derniers chapitres 3 et 4 exposent la méthode de PHM développée dans ces travaux.

Le chapitre 2 présente un état de l'art des méthodes d'intelligence artificielle adaptées au PHM. Dans un premier temps, la section 2.1 défini ce qu'est le PHM et les méthodes principalement utilisées dans le domaine aujourd'hui. Puis les méthodes d'intelligence artificiel, et notamment celles liées au connexionnisme sont présentées à la section 2.2 comme une amélioration des processus standards. Enfin, la section 2.3 abordera l'application de l'apprentissage profond aux problématiques de PHM. Ces deux dernières sections donneront les pré-requis nécessaires à l'application de telles méthodes dans le domaine industriel, comme par exemple, la nécessité d'avoir une base d'apprentissage de taille significative et labellisée.

Les données utilisées ici, provenant des signaux vibratoires de l'E-TRAS ne possédant pas de tels labels *à priori* il convient dans un premier temps de développer une méthode pour assigner une étiquette, c'est à dire dans ces travaux, un degré de sévérité, à chaque point obtenu. Par conséquent, une méthode de partitionnement de séries temporelles à été créée au chapitre 3 pour apporter une réponse à cette problématique. Afin de déterminer les points d'intérêt de ce type de méthodes, une revue de la littérature a été menée sur les algorithmes de partitionnement (ou *clustering*) spécifiques à l'analyse de séries temporelles dans la section 3.1. Plusieurs tendances

en sont alors extraites en commençant par la faible quantité de méthodes dédiées et d'indicateurs afférents permettant l'évaluation de leurs résultats. La section 3.2 décrit l'ensemble des données extraites des signaux vibratoires issus du système E-TRAS. A partir des signaux bruts, sont calculés des descripteurs, grandeurs mathématiques qui constituent des séries temporelles. C'est à partir de ces signaux temporels que seront développés tous les algorithmes de ces travaux. Une fois que les descripteurs sont prêts, la section 3.3 présente la nouvelle méthode de *clustering* qui intègre, en son cœur, un réseau de neurones profond pour effectuer de la segmentation d'images ayant appris sur des données artificielles. Ensuite, pour évaluer la performance du partitionnement obtenu, un nouvel indicateur de qualité est développé à la section 3.4. Ce dernier servira alors à comparer notre méthode aux méthodes classiques d'apprentissage statistiques de l'état de l'art à la section 3.5.

Maintenant que la méthode de partitionnement a permis la création d'une base de données labellisée, il est possible d'utiliser ces résultats pour développer les méthodes de diagnostic et de pronostic pour des séries temporelles. L'idée est de pouvoir assigner un degré de sévérité à un intervalle de série de taille quelconque, inconnu pour l'étape de diagnostic. Puis pour l'étape de pronostic, qui vise à déterminer le temps de vie restant de l'actionneur, l'objectif est d'extrapoler le comportement de la série étudiée pour conclure quant à un potentiel futur défaut.

Le dernier chapitre 4 propose une méthode pour résoudre ces deux problématiques. Comme l'ensemble des données étudiées n'a pas été enrichi expérimentalement, le problème du manque de données pertinentes est toujours d'actualité. Aussi, en regard des informations apportées par le chapitre 2 et compte tenu de faible nombre de données, il ne nous a pas apparu souhaitable de conserver dans cette partie des architectures de classifieurs et régresseurs à base de réseaux de neurones profonds. Nous avons donc opté pour une méthode à base d'alignement de séries temporelles pour effectuer le diagnostic et le pronostic de l'état de santé de l'actionneur E-TRAS. La base de données précédemment obtenue avec la méthode du chapitre 3 couvre les comportements possibles lors d'un vieillissement. Toute série inconnue, sera alors comparée à cette base de comportements potentiels. Le comportement futur de la série inconnue sera alors un mélange des séries temporelles les plus similaires. De plus, lorsque l'alignement sera effectué, la phase de diagnostic sera réalisée car, chaque point de la base de données possédant un degré de sévérité, l'état de santé probable sera de même déterminé par mesure de similarité. L'avantage d'une telle méthode est alors de pouvoir améliorer les performances de pronostic en enrichissant au fur et à mesure de l'utilisation de systèmes la base de comparaison. Afin d'apprécier les limites rencontrées par la littérature à l'utilisation de ce type de méthodes pour le PHM, un état de l'art des méthodes de pronostic par alignement est effectué à la section 4.1. La section 4.2 développe ensuite la méthode de diagnostic/pronostic de ces travaux qui lève les principaux verrous des algorithmes de l'état de l'art précédemment effectué. Enfin, cette dernière est appliquée aux données expérimentales que sont nos signaux vibratoires captés sur E-TRAS pour obtenir le temps de vie restant de l'actionneur.

La dernière section 5 synthétisera les différents points abordés dans ces travaux de thèse afin de les situer dans la méthodologie globale de PHM présentée dans ce manuscrit. Elle détaillera ensuite l'apport de ces travaux à la littérature avec notamment la discussion de la résolution des verrous scientifiques présentés à la section 1.4 ainsi que les hypothèses fondamentales faites, facteur limitant pour la généralisation de cet ensemble de méthodes. Ce dernier point, permettra de proposer des perspectives pour de futurs travaux dans ce domaine.

Chapitre 2

État de l'art sur les méthodes de surveillance d'état de santé

Dans ce chapitre seront développées les principales méthodes utilisées aujourd'hui pour réaliser des tâches de diagnostic et de pronostic dans le domaine industriel. Ce premier état de l'art a permis de faire émerger une tendance à la convergence actuelle des méthodes classiques à base de modèles physiques, vers des méthodes à base de données et plus particulièrement, des méthodes à base d'intelligence artificielle (IA). Dans ces dernières, avec l'utilisation accrue des réseaux profonds depuis 2010, l'architecture connexionniste, représentée aujourd'hui par les réseaux de neurones, a connu un développement considérable. En effet, à partir des signatures de défauts basées sur les signaux temporels, fréquentiels ou exploitant la cyclostationnarité, les réseaux de neurones sont de plus en plus utilisés dans le domaine de la surveillance d'état de santé. Par la nature des opérations à effectuer, cette architecture s'y prête particulièrement. Ainsi, l'étape de diagnostic qui s'intéresse à déterminer l'état de santé actuel d'un système peut être reformulée comme un problème de classification de données, tandis que l'étape de pronostic peut être redéfinie comme un problème de régression (et plus précisément comme la classification de résultats issus d'une extrapolation). Nous verrons que ces structures sont adaptées à l'élaboration de méthodes de Prognostics and Health Management (PHM). Avant toute chose, dans un premier temps, les travaux de la littérature concernant les méthodes de pronostic et d'état de santé seront présentés à la section 2.1. Cette première section permettra d'expliciter ce qu'est le PHM, d'en présenter le concept et les grands axes, puis de faire un inventaire des méthodes utilisées principalement pendant ces trois dernières années. La sous-section 2.1.2 permettra alors de dégager la tendance principale du domaine à migrer progressivement vers des méthodes à base d'IA, c'est pourquoi la section 2.2 suivante présente l'approche connexionniste et la raison de son utilisation dans le domaine de ces travaux. Seront alors abordées la nature d'un réseau de neurones simple à la sous-section 2.2.1 et la nature de ce qui a fait le succès de ces architectures avec l'apprentissage automatique à la sous-section 2.2.2. Puis un rapide état de l'art est constitué pour introduire des exemples d'usage en PHM à la sous-section 2.2.3. Enfin, pour terminer cette section, la transition vers les architectures profondes est présentée en sous-section 2.2.4 et les principales architectures profondes qui sont utilisées dans le domaine du PHM sont détaillées. La section 2.3 finale se concentre alors sur la présentation d'un certain nombre de travaux de diagnostic et pronostic utilisant ces dernières architectures.

2.1 Définition de la surveillance et du pronostic d'état de santé

2.1.1 Qu'est-ce que le PHM ?

Le PHM est le nom d'un domaine d'ingénierie qui s'intéresse au management de l'état de santé de systèmes industriels. À l'origine, il a été principalement introduit par le département américain de la Défense et de l'Énergie dans l'optique de diminuer les coûts de maintenance et d'augmenter la disponibilité des systèmes militaires [Ramezani, Moini et Riahi, 2019] au début des années 2000 [Ranasinghe *et al.*, 2022, figure 1]. Comme cela a été introduit dans la section 1.2, son nom dévoile les deux intérêts principaux de la discipline que sont le pronostic et le management d'état de santé. Sans revenir sur la définition des termes, déjà introduite à la section 1.2, l'idée est d'une part de déterminer le temps de vie restant d'un système, d'autre part, en fonction de cette nouvelle information, de déterminer quelles seraient les actions à mener au regard des résultats précédents. À ces objectifs principaux peut aussi être rajouté le fait de détecter une défaillance le plus tôt possible [Atamuradov *et al.*, 2017]. Les différentes étapes constituant le PHM peuvent être schématisées par la figure 2.1 reprenant la figure 1.4.

La figure 2.1 met en évidence les étapes qui seront développées dans ces travaux afin de mieux cerner les contours du PHM. Détaillons, ci-dessous, l'ensemble de ces étapes :

Acquisition de la donnée Cette première étape consiste à définir et à mettre en œuvre la stratégie de récupération de données sur le système étudié. Il est alors nécessaire de bien dimensionner la chaîne d'acquisition, et en particulier les capteurs, en fonction des défauts à surveiller. Par exemple, la précision des capteurs, leurs fréquences d'échantillonnage, leurs positions sur le système, le format des données renvoyées doivent être choisis avec soin. Toute cette première étape est capitale car c'est à partir de ces données que les autres algorithmes de la chaîne de traitement extrairont des informations pertinentes pour le diagnostic et/ou le pronostic de pannes. L'importance de cette étape est soulignée par [Jia, B. Huang *et al.*, 2018]. En effet, nous verrons par la suite que pour les descripteurs très bruités que nous utilisons, un résultat pertinent pour la prise de décision est difficile à obtenir, ce qui fait écho au principe bien connu du Garbage In, Garbage

Out (GIGO). Cette étape est bien plus importante qu'un simple choix technique pour la chaîne de captation. En effet, dans les limites des méthodes développées depuis les années 2000, [Jia, B. Huang *et al.*, 2018] notent que les systèmes étudiés sont utilisés sous plusieurs régimes de fonctionnement et que les méthodes sont en général développées pour des régimes opérationnels fixes. Cette tension entre modélisation et pratique reste à résoudre. Pour l'instant, la seule solution est de tester les méthodes développées dans ces différents régimes afin de s'assurer de leur pertinence et leur robustesse. Un autre axe d'exploration serait la création d'une flotte de systèmes dédiée aux algorithmes de PHM. Effectivement, aujourd'hui les bancs d'essais industriels sont coûteux à mettre en place et à exploiter aussi, et ces travaux n'y font pas exception, la plupart du temps les algorithmes sont testés sur un faible nombre de réalisations des systèmes étudiés. Il conviendrait alors, afin d'améliorer et surtout d'étudier les résultats de généralisation des méthodes développées, de créer des ensembles de données provenant d'une flotte de systèmes sous surveillance. À l'ère de l'industrie 4.0, de l'internet des objets et, de manière générale, des objets plus connectés, le flux d'information transitant entre systèmes n'a jamais été aussi important et pourtant une solution à ce problème n'est que rarement proposée dans l'état de l'art, au mieux il est constaté, et ces travaux n'y font pas exception. Certains travaux comme ceux de [Saxena et Goebel, 2008] fournissent aux équipes de recherche un ensemble de données leur permettant de comparer le résultat de leurs algorithmes. Seulement, cet ensemble reste produit à partir de simulations et non à partir d'une flotte de systèmes industriels en opération et sous surveillance. Depuis vingt ans que le problème existe, les applications industrielles ne permettent pas, de par leur nature et leur confidentialité, de créer des *pipelines* de données qui permettraient le développement massif de méthodes de pronostic d'état de santé. L'objectif est de créer des systèmes capables de déterminer leur propre état de santé, leur état futur ainsi que les décisions à prendre par un opérateur pour optimiser leur cycle de vie. Malheureusement, nous avons aussi été confrontés au manque de données applicatives, ce qui a conduit à certains de nos choix.

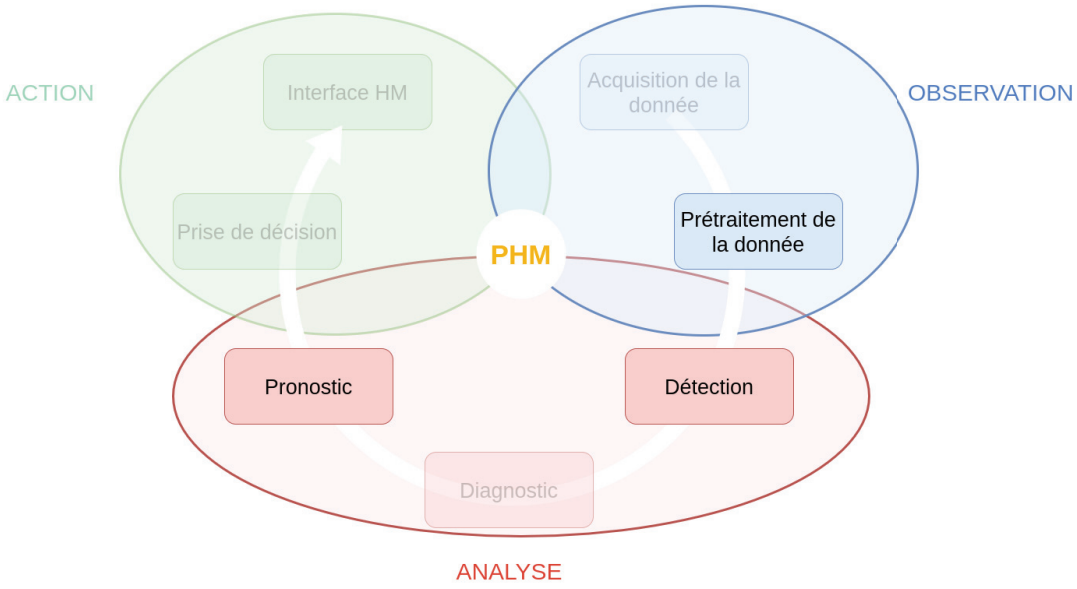


Figure 2.1 – Schéma regroupant les différentes étapes de la méthode de PHM

Pré-traitement de la donnée Contrairement à l'acquisition des données, la partie concernant le *traitement de la donnée* est abordée à la section 3.2 de ce manuscrit. L'idée de cette partie est, dans un premier temps, de nettoyer les données brutes récoltées à l'étape précédente. En effet, les données récoltées sur systèmes sont parfois inutilisables parce qu'incomplètes, ou constituant des données aberrantes suite à un problème d'instrumentation. Cette première phase de nettoyage est nécessaire, de surcroît lorsque les méthodes qui seront adoptées dans la suite du *framework* sont dîtes « à base de données » ou *data-driven*. Cette première étape faite, les signaux bruts peuvent être filtrés pour améliorer le rapport signal sur bruit. Puis une couche de mise en forme de l'information peut être réalisée à partir des signaux bruts. Dans la pratique, et dans ces travaux, cela

représente un ensemble de grandeurs statistiques qui sont supposées avoir une dynamique corrélée au vieillissement de l'actionneur surveillé. Ainsi, à partir des signaux vibratoires, la moyenne, le kurtosis ou encore un ensemble de percentiles seront extraits. De plus, des critères de sélection permettant d'évaluer la cohérence des signaux obtenus avec le vieillissement d'un actionneur peuvent être développés afin de ne conserver que les descripteurs pertinents pour, notamment, la détermination de durée de vie restante. Notons que cette étape est optionnelle lorsqu'une approche à base d'apprentissage profond est sélectionnée pour fournir des résultats de PHM. Dans ce cas, ce seront les premières couches d'un réseau de neurones qui extrairont une représentation de l'information pertinente au regard des opérations de diagnostic et de pronostic. Cette approche est de plus en plus utilisée et possède l'avantage d'être bien moins coûteuse que la création de descripteurs à partir de savoirs experts, en témoignent les nombreux travaux de l'état de l'art dédiés à ce sujet et notamment [Boileau, 2010], [Leboeuf, 2012], [Breuneval, 2017] ou encore [Favin-Lévêque *et al.*, 2019]. Notons que dans la revue de l'état de l'art de [Hamadache *et al.*, 2019, table 1], une liste plutôt exhaustive de signatures de défauts calculées pour la surveillance d'état de santé de roulement est fournie.

Détection Tout comme dans la partie précédente, la *détection* sera étudiée dans ces travaux. Une fois que les données ont été récupérées du système sous surveillance et que le pré-traitement de la donnée a été effectué, cette dernière peut être explorée dans le but de détecter l'apparition de potentiels défauts ou comportements anormaux. L'implantation faite de cette phase sera présentée au chapitre 4. Dans ces travaux, elle est réalisée conjointement avec la phase de pronostic. Les méthodes de détection s'intéressent plus généralement à déterminer toute déviation du système par rapport à son état nominal. Elles font partie du domaine de la *reconnaissance de la nouveauté*. La phase de détection est, en général, étroitement liée à la phase de diagnostic. Dans cette phase, le constat de défaut est réalisé. Il peut être binaire dans le cas d'une mesure de déviation uniquement, comme il peut être multi-classe dans le cas, ici, d'une détection de plusieurs degrés de sévérités de défaut. Cette approche est utilisée, par exemple, lorsqu'aucun modèle physique n'est développé ou que les bases d'apprentissage du système ne permettent pas de représenter plusieurs défauts. De même, au-delà de la quantité de données récoltées, il est courant, dans le domaine industriel, de ne pas posséder les journaux de tous les essais effectués, ou que l'analyse *post mortem* n'ait pas été réalisée. C'est pourquoi aucun label, quant au défaut rencontré, n'est fourni avec la base de données. Sans connaissance de cette information, la meilleur information possible rendue par les algorithmes *ad hoc* reste seulement la gradation des différents états par lesquels le système est supposé être passé.

Diagnostic En complément de la phase de détection, la phase de diagnostic s'intéresse à déterminer les causes d'une défaillance ou du changement de comportement d'un système. Ainsi, le diagnostic est une phase plutôt utilisée en analyses *post mortem* [Atamuradov *et al.*, 2017]. Le diagnostic s'intéresse à la causalité. Il est alors plus naturel de l'associer à une méthode à base de modélisation de système physique où une déviation par rapport au modèle peut être imputée à l'une de ses composantes bien précise. Dans le cas des approches à base de données, l'idée est alors d'utiliser le plus souvent une classification multi-classes, l'algorithme, extrayant la représentation de la donnée, ayant appris avec des ensembles de données disjoints représentant différents défauts probables du système surveillé. Le résultat courant est alors comparé aux différents défauts déjà connus et en fonction de la quantification de la ressemblance par rapport à la base d'apprentissage, la causalité du défaut sera inférée.

Pronostic Contrairement à la phase précédente, le pronostic aurait plus vocation à s'effectuer pendant l'exploitation du système [Atamuradov *et al.*, 2017], mais pas nécessairement de manière embarquée. Maintenant que le système est mieux connu, que son comportement lors de défauts probables a été caractérisé, la méthodologie de PHM s'intéresse à prédire sa durée de vie en conditions opérationnelles. C'est à cette étape que la durée de vie restante du système, autrement appelée Remaining Useful Life (RUL), est estimée. Cette dernière sera détaillée au chapitre 4 pour l'inverseur de poussée à l'étude ici. À la différence de la Mean Time Between Failure (MTBF) qui détermine le temps moyen restant avant la prochaine défaillance, donc un scalaire, la RUL a pour objectif d'associer une valeur du temps de vie restant avec sa mesure d'incertitude associée [Nguyen *et al.*, 2019]. Notons que cette définition est assouplie dans l'état de l'art où, comme il sera vu par la suite, bon nombre de travaux ne calculent pas de mesure d'incertitude lors de la détermination de la

RUL. La différence fondamentale entre la MTBF et la RUL réside alors dans la manière de calculer ces deux quantités. La MTBF résulte d'une approche purement statistique au niveau d'une flotte de produits similaires alors que la RUL effectue sa prédiction en fonction des données récoltées (en temps réel) sur le système surveillé. Cette dernière doit prendre en compte les variations de conditions opérationnelles du produit ou les événements non prévus tels qu'une dégradation subite du système suite à un stress intense.

Prise de décision L'étape de prise de décision n'est pas étudiée dans ces travaux. Cette étape complète les méthodes précédentes, elle est nécessaire au processus de PHM. C'est l'aboutissement de la méthode qui donne son sens au PHM. En fonction des résultats obtenus précédemment, la décision d'actions correctives ou préventives est effectuée. De manière concrète, c'est lors de cette étape que l'emploi du temps de maintenance d'une compagnie aérienne pourrait être modifiée [Jia, B. Huang *et al.*, 2018], [Mofokeng, Mativenga et Marnewick, 2020] ou encore [Zhou *et al.*, 2020]. On pourra aussi trouver les travaux de [Wesendrup et Hellingrath, 2020] ou de [Y. Hu *et al.*, 2022] sur l'étude de la prise de décision dans le domaine. Enfin, l'étape de conversion des résultats des méthodes de PHM en décision de maintenance reste encore un sujet ouvert [Jia, B. Huang *et al.*, 2018]. Dans ces travaux, nous faisons l'hypothèse que les résultats de pronostic obtenus permettront uniquement d'obtenir le temps de vie restant d'un système.

Interface Health Monitoring (HM) L'interface de HM vise à présenter les résultats du processus de manière synthétique et organisée à un opérateur qui serait décisionnaire. De manière générale, ces deux dernières étapes sont réalisées lors d'un processus de PHM avec un Technology Readiness Level (TRL) [Tzinis, 2015] bien plus élevé que celui de ces travaux, atteignant un TRL de niveau 3 uniquement.

L'objectif du PHM est d'optimiser les cycles de maintenance d'un système surveillé. Aussi, il est important de situer ce *framework* au sein des différents types de maintenances existant dans le domaine industriel. Les principales maintenances considérées sont représentées à la figure 2.2.

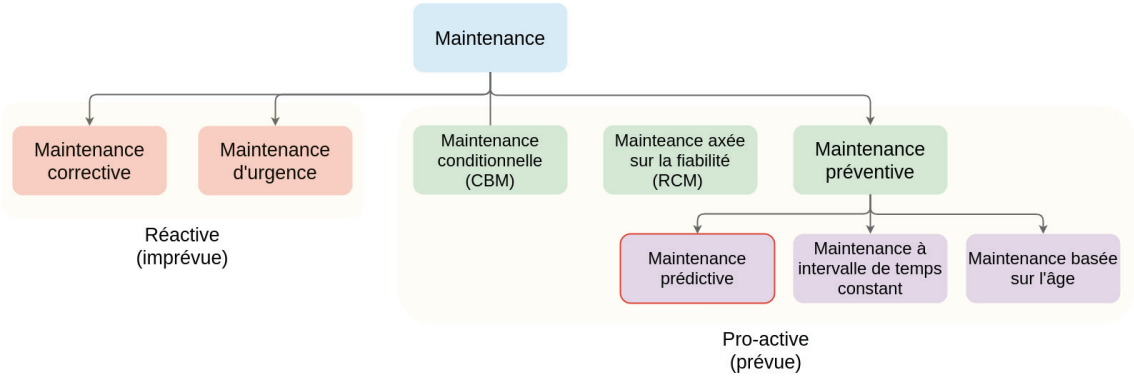


Figure 2.2 – Taxonomie des différents types de maintenance, d'après une figure modifiée de [Fei *et al.*, 2020, figure 1]

La figure 2.2 présente deux grandes familles de plans de maintenances : les maintenances réactives ou subies, et les maintenances résultant d'un comportement pro-actif donc anticipé. Dans les maintenances imprévues (en rouge), deux cas sont distingués, à savoir les maintenances correctives et les maintenances d'urgence. Les premières sont effectuées lorsqu'une défaillance est détectée de manière à remettre le système en état de fonctionnement ou en état optimal. Les secondes, quant à elles, sont les maintenances les plus coûteuses car devant être réalisées dans l'urgence. Cela signifie en général que l'emploi du temps de maintenance de la compagnie aérienne est bousculé pour effectuer des réparations de dernière minute, par exemple, lorsqu'une avarie est déclarée sur un aéronef avant que les passagers n'embarquent. Ne pouvant complètement supprimer de telles occurrences, cette famille de maintenance est cependant à limiter le plus possible. Pour ce faire, une politique et des méthodes de surveillance d'état de santé peuvent être mises en place à travers une méthode de PHM. Concernant les maintenances pro-actives (en vert), il y en a de deux sortes : les maintenances préventives et les maintenances prédictives. Les premières sont déjà largement mises en place par les compagnies aériennes. La maintenance à intervalles de temps constants permet de faire une visite de maintenance tous les intervalles de temps (plus précisément dès qu'un certain nombre de

cycles de vols a été dépassé) définis par le constructeur ou par une étude statistique de la flotte de systèmes. Avec la maintenance basée sur l'âge du système, il est aisé de voir que ces dernières politiques génèrent un plan de maintenance statique qui ne tient pas compte des variations ou des inconnues de l'environnement dans lequel est plongé un système. La maintenance prédictive permet alors de pallier cette limite avec la maintenance fondée sur la fiabilité ou Reliability Centered Maintenance (RCM), et la maintenance conditionnelle ou Condition-Based Maintenance (CBM). La première s'intéresse à déterminer les emplois du temps de maintenance en fonction du calcul de la MTBF, supposant, de fait, que le système surveillé évolue dans des conditions opérationnelles déterministes et statiques. Cette première méthode ne peut alors pas prendre en compte les différents régimes de fonctionnement d'un système en exploitation et n'est donc pas optimale. La seconde méthode, ou CBM, ressemble le plus au PHM et en partage les caractéristiques, si ce n'est qu'elle ne s'intéresse pas qu'à des conditions opérationnelles déterministes et bien définies. C'est donc la CBM, composante de la maintenance prédictive, qui sera développée dans ces travaux.

Afin de mieux illustrer les conséquences qu'ont les différents plans sur les coûts de maintenance en général, la figure 2.3 est réalisée.

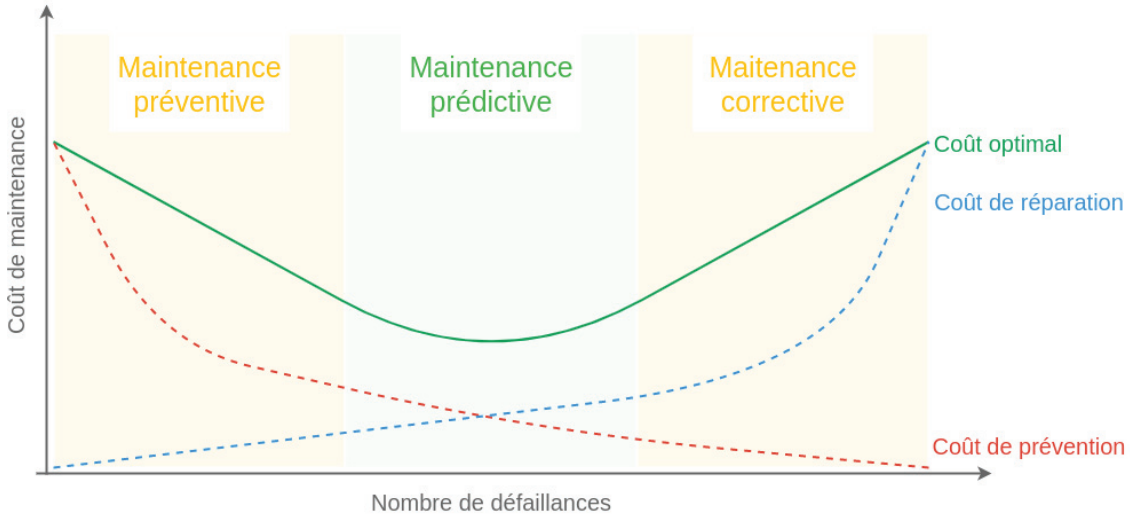


FIGURE 2.3 – Schéma de principe sur les coûts de maintenance, d'après une figure modifiée de [SPRONG, X. JIANG et POLINDER, 2019, figure 1] reprise de [TCHAKOUA *et al.*, 2014, figure 6]

Sur la figure 2.3, les trois familles de maintenance principales sont représentées, avec la maintenance corrective représentant toute la famille des maintenances imprévues. Le nombre de défaillances en abscisse permet de déterminer la famille dans laquelle on se trouve et en ordonnée, le coût de l'opération de maintenance est représenté. Dans le cas d'une maintenance préventive (en rouge), le coût de prévention associé est élevé au début puis finit par devenir nul au fur et à mesure du nombre de défaillances. Ce comportement témoigne du fait qu'au début, des systèmes encore en bon état peuvent être changés ce qui engendre un coût supérieur pour les compagnies aériennes. Il diminue naturellement au fur et à mesure du temps. Concernant le coût de réparation (en bleu), il est symétrique au coût de prévention. En effet, si la compagnie attend jusqu'au dernier moment pour effectuer des réparations, le coût final est plus important mais quasi inexistant au début. Notons que dans ce cas, en plus du coût, le risque de défaut du système en opération augmente, ce qui aurait pour incidence, si jamais un défaut apparaissait en fonctionnement, de décupler le prix d'une réparation. La maintenance prédictive, quant à elle, permet d'obtenir la courbe de coût optimale (en vert). Le système passe alors en maintenance ni trop tôt, ni trop tard, mais au bon moment de sa durée de vie.

Cette dernière approche de maintenance prédictive est permise par le développement de méthodes de PHM. La sous-section 2.1.2 suivante présente les différents algorithmes utilisés dans la littérature afin d'implémenter les méthodes de diagnostic et de pronostic permettant d'obtenir des résultats de maintenance prédictive pertinents.

2.1.2 Les tendances des méthodes développées ces trois dernières années

Le PHM utilise des méthodes qui pourraient être décomposées en trois familles différentes : les méthodes à base de données, les méthodes à base de modèles et les méthodes hybrides.

Les méthodes à base d'analyse de données extraient des informations concernant l'état de santé du système ainsi que son état futur uniquement à partir des données d'entrées fournies aux algorithmes. Ces méthodes suscitent un intérêt croissant, car elle permettent de produire des résultats de manière moins coûteuse que les méthodes à base de modèles [Nguyen *et al.*, 2019]. Cependant, cette économie de coût à un prix. Ces méthodes nécessitent un grand volume de données dans un domaine où l'on manque cruellement de données récoltées à partir d'essais de vieillissement sur systèmes. D'une certaine manière, le coût de développement d'un modèle physique est alors reporté sur le coût de collecte de données pertinentes.

De manière général, l'usage de méthodes à base de données et, plus spécifiquement, de méthodes d'intelligence artificielle, reste une nouveauté pour le domaine du PHM, qui a évolué pendant de nombreuses années avec l'analyse de modèles physiques. Ces derniers permettent, au-delà de la détection de défauts, de déterminer leurs causes, tandis que le seul moyen pour les modèles dont les paramètres sont inférés à partir des données, de renseigner sur la cause d'une défaillance, est de détecter un défaut déjà caractérisé dans les données surveillées et ainsi, par similarité, de conclure quant à la cause possible de défaillance. Les méthodes à base de données permettent de prendre en compte plus facilement que les autres méthodes les non linéarités dans les données et s'étendent facilement à de larges bases de données d'apprentissage [Vrignat, Kratz et Avila, 2022]. Seulement, elles ont besoin de plus de données pour l'obtention de résultats pertinents que celles à base de modèles physiques et elles sont, par essence, bien trop dépendantes des dynamiques de l'information d'entrée qui leur est fournie. Si les données sont incomplètes, les algorithmes ne pourront pas fournir de résultats pertinents.

Les méthodes qui sont fondées sur des modèles physiques ne possèdent pas cette limite. Plusieurs approches sont alors utilisées. Soit un jumeau numérique est développé pour représenter le système sous surveillance et simuler son comportement dans diverses conditions opérationnelles, soit des modèles de dégradation sont utilisés dans le *pipeline* de PHM afin de rendre des résultats de diagnostic et de pronostic. L'avantage des modèles physiques est qu'ils permettent, notamment dans une phase de diagnostic, de remonter plus facilement aux causes d'une panne, si la modélisation en tient compte. Bien que les modèles soient coûteux à développer, ils sont réutilisables pour des systèmes similaires, ils peuvent, de plus, générer des données d'entraînement afin de développer des méthodes à base de données. Enfin, ils permettent d'isoler les différentes défaillances sur le système. Leur inconvénient réside dans les hypothèses simplificatrices requises pour leur développement. Ces dernières limitent fortement le modèle à un point de fonctionnement qui risque de ne pas être représentatif des variations opérationnelles. Les paramètres d'un modèle sont difficiles à identifier sur des systèmes industriels coûteux pour lesquels les tests sont difficiles à mener, et, comme dit précédemment, ils sont coûteux à développer [Vrignat, Kratz et Avila, 2022].

L'approche hybride, quant à elle, vise à allier le meilleur de chaque méthode en utilisant des approches à base de données et des modèles. Les travaux de [Jia, Cai *et al.*, 2019] en sont un exemple. En effet, l'étape de prédiction de durée de vie restante est réalisée en deux étapes. La première consiste à comparer des descripteurs tests à ceux contenus dans une base de données, représentatifs d'un comportement en exploitation du système surveillé, avec un test statistique couplé à un critère de corrélation non linéaire. La seconde étape consiste, à partir de ces informations de similarité, d'inférer les paramètres d'une distribution de Weibull pour obtenir un temps de vie restant avec une mesure d'incertitude. Plus récemment, les travaux de [Arias Chao *et al.*, 2022] peuvent aussi être cités. Dans ces derniers, un modèle physique est utilisé pour effectuer l'apprentissage d'un réseau de neurones profond.

Ainsi, ces trois approches forment l'ensemble des méthodes utilisées en PHM pour diagnostiquer et pronostiquer des systèmes. Afin de fournir une image des principales méthodes utilisées jusqu'à maintenant dans le domaine, la figure 2.4 a été créée. Les données de la figure 2.4 ont été extraites à partir des revues de la littérature de [Atamuradov *et al.*, 2017], [Z. Zhang *et al.*, 2018], [R. Liu *et al.*, 2018], [Nannan, Xiaohao et Baolong, 2018], [Liangwei Zhang *et al.*, 2019], [Nguyen *et al.*, 2019] et [Ranasinghe *et al.*, 2022]. Notons que cette figure ne prétend pas à l'exhaustivité, mais synthétise les principales méthodes utilisées ces dernières années. Le domaine du PHM est large et, faisant appel à plusieurs disciplines scientifiques, de nombreuses méthodes sont décrites dans la littérature.

FIGURE 2.4 – Taxonomie des méthodes de PHM utilisées dans l'état de l'art pour du diagnostic et/ou du pronostic

La figure 2.4 est composée de trois parties. Notons tout d'abord que la taxonomie présentée à la figure 2.4 est une vue imparfaite et incomplète de l'ensemble des méthodes observées dans l'état de l'art. Elle nous permet néanmoins d'étayer notre raisonnement développé par la suite. Il en sera de même pour les taxonomies extraites de l'état de l'art. Une première partie rassemble les méthodes à base de modèles contenant des lois de dégradations spécifiques ainsi que des modèles physiques de systèmes tels que des turboréacteurs ou encore des actionneurs de commande de vol. Le filtre de Kalman et ses variantes sont, elles, utilisées afin d'estimer le comportement d'un système dynamique sain pour ensuite détecter toute défaillance si la mesure sur système dévie de la prédiction. La deuxième partie est constituée des modèles ne rentrant pas dans les cases précédentes. Nous retrouvons alors les modèles à base de logique floue et leur variante en réseau

de neurones, les systèmes experts qui sont composés de conditions algorithmiques cherchant à reproduire un raisonnement humain, ou encore la théorie des graphes avec les graphes acycliques. Ces dernières méthodes sont alors moins utilisées que celles de la dernière partie de la figure 2.4 : les méthodes à base de données. Elles sont de plusieurs sortes. Les méthodes à base d'inférence bayésienne permettent notamment d'évaluer une mesure d'incertitude dans les résultats de rendus. Par exemple, les mélanges de gaussiennes ou Gaussian Mixture (GM) permettent d'effectuer des tâches de regroupement en assignant à chaque groupe une distribution associée. La méthode de régression des processus gaussiens ou Gaussian Process Regression (GPR) permet, elle, de créer un résultat d'extrapolation avec une mesure d'incertitude associée. Parallèlement à la famille des méthodes d'inférence bayésienne, les approches statistiques plus classiques sont aussi souvent utilisées. La modélisation de séries temporelles par processus auto-régressifs (et leurs variantes) sont pertinents lors de la modélisation de processus stationnaires stochastiques. Enfin, les architectures connexionnistes sont séparées à la figure 2.4 en *apprentissage statistique* ou Machine Learning (ML) et *apprentissage profond* ou Deep Learning (DL). Notons que ces méthodes seront détaillées pour certaines dans les sections suivantes. Dans les méthodes à base d'apprentissage statistique, citons les Support Vector Machine (SVM) et les Support Vector Regression (SVR) utilisées dans les travaux de [Breuneval, 2017] ou encore les forêts aléatoires ou Random Forest (RF) qui, avant l'avènement des architectures profondes, possédaient des résultats de classification pertinents pour le diagnostic. Ces méthodes ont été dernièrement remplacées par l'étude des architectures profondes qui seront étudiées aux sections suivantes.

Parmi les méthodes à base de données, l'approche connexionniste est de plus en plus utilisée dans le domaine du PHM et parmi ces dernières, l'usage des méthodes basées sur l'apprentissage profond reste récent mais de plus en plus développé. En effet, dans leurs travaux, [Rezaeianjouybari et Shang, 2020, figure 5] ont remarqué que le nombre de publications utilisant une architecture d'apprentissage profond a doublé de 2016 à 2017 pour passer de 20 publications à 44 publications. En 2018 il a atteint un maximum de 78 publications pour redescendre à 40 en 2019. Ces chiffres dans l'absolu témoignent du fait que les méthodes connexionnistes, et plus précisément, les méthodes profondes sont encore peu utilisées dans ce domaine. La figure 2.5 représente la proportion d'architectures profondes utilisées pour obtenir des résultats de diagnostic et de pronostic.

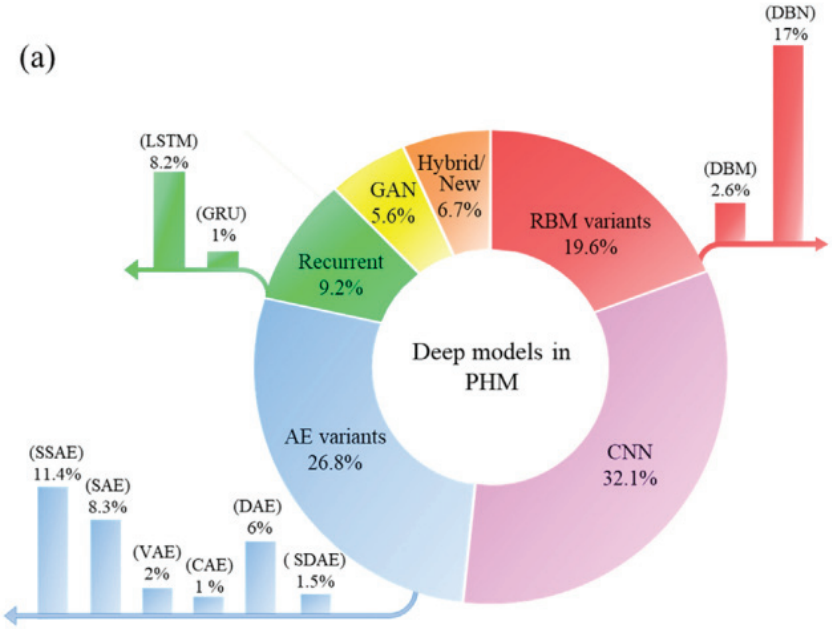


Figure 2.5 – Proportion des différentes architectures utilisées entre 2013 et septembre 2019, d'après [Rezaeianjouybari et Shang, 2020, figure 5]

La figure 2.5 montre que les Convolutional Neural Network (CNN) ainsi que les Auto-Encoder (AE) sont, de loin, les architectures les plus utilisées. Notons qu'elles sont d'ailleurs utilisées principalement pour les étapes de diagnostic. De plus, la partie *Hybrid*, non négligeable, témoigne du fait que les algorithmes à base de réseaux de neurones sont souvent utilisés en combinaison avec

d'autres approches plus classiques. Afin de développer l'utilisation de l'apprentissage profond pour ces travaux, la section 2.2 suivante présente, dans un premier temps, les réseaux de neurones peu profonds. Puis les principales architectures utilisées en PHM décrites à la figure 2.5 sont détaillées à la sous-section 2.2.4, à l'exception faite des Generative Adversarial Network (GAN).

2.2 L'approche connexionniste de l'intelligence artificielle : les réseaux de neurones

2.2.1 Pourquoi les réseaux de neurones ?

Un réseau de neurones peut être défini comme un ensemble de primitives inter-connectées de manière parallèle dont la structure globale permet de réaliser une fonction déterminée. Le rassemblement de ces primitives, simples, permet ainsi de résoudre des problèmes complexes. Ainsi l'appellation « réseau de neurones » est un terme générique qui décrit une structure de calcul hautement parallèle. Cette structure permet, en théorie, d'estimer une fonction mathématique quelconque, ce qui la rend intéressante pour déterminer des modèles à partir d'un ensemble de données.

En effet, cette capacité construite, entre autres choses, sur les travaux de [Kolmogorov, 1957] sur la décomposition de fonctions continues à plusieurs variables par une combinaison linéaires d'autres fonctions continues a servi de base à la démonstration de l'utilité des réseaux de neurones pour l'approximation de fonctions continues. Première pierre à l'élaboration d'un théorème d'approximation universelle, ces résultats ont été utilisés presque simultanément par [Cybenko, 1989], [Funahashi, 1989] et [Hornik, Stinchcombe et White, 1989]. L'idée de [Cybenko, 1989] était alors qu'une combinaison linéaire finie de fonctions affines pouvait approximer le comportement d'une fonction continue à plusieurs variables ayant comme support l'hypercube unité. Une telle combinaison de fonctions serait permise par un réseau de neurones a une couche cachée avec une fonction d'activation non linéaire de type sigmoïde. La notion d'*approximation universelle* est, elle, apportée (et de même démontrée) par [Hornik, Stinchcombe et White, 1989] qui concluent en annonçant que toute difficulté pour vérifier ce théorème par l'exemple viendrait du fait que soit l'apprentissage du réseau était mauvais, soit le nombre de couches cachées était insuffisant ou soit la relation entre les entrées et les cibles de la base d'apprentissage n'était pas déterministe. Précédent la descente de gradient pour obtenir les paramètres d'un tel réseau, la « méthode des tamis » de [Geman et Hwang, 1982] est d'ailleurs citée. Avec des notations contemporaines, le théorème d'approximation universel 2.2.1 développé par ces travaux est présenté ici.

Théorème 2.2.1 : Approximation universelle

Soit f une fonction continue sur un compact C une fonction à estimer, Φ une fonction bornée, croissante et continue de $\mathbb{R}^d$. Soit $\varepsilon > 0$, $\exists\,(N, v_i, b_i) \in \mathbb{R}^3$, $\exists w_i \in \mathbb{R}^d$, $\forall i \in [\![1; n]\!]$ tel que :

$$\forall x \in C,\ F(x) = \sum_{i=1}^{N} v_i \Phi\left(w_i^\top x + b_i\right)$$

F est une approximation de f à ε près sur C telle que :

$$\forall x \in C,\ |F(x) - f(x)| \leq \varepsilon$$

Remarque. *Comme indiqué par [Cybenko, 1989], le théorème d'approximation universel ne fait que statuer l'existence d'une solution. Il ne caractérise ni la complexité de l'algorithme d'apprentissage pour aboutir à un tel résultat, ni la structure du réseau de neurones le vérifiant et surtout pas sa capacité de généralisation. Ce résultat permet uniquement d'affirmer qu'une combinaison linéaire de fonctions bien choisies peut en approximer une autre.*

Ces capacités générales d'approximation de fonctions sont, dans un premier temps utilisées pour déterminer l'équation de surfaces séparatrices entre les données d'entrée. Cela revient à effectuer une classification. Expliquons et détaillons le fonctionnement d'une telle structure avec l'exemple primordial en apprentissage statistique de la classification supervisée. Pour cela, considérons un

problème de classification binaire dans un espace à deux dimensions. Une paramétrisation d'une surface séparatrice, dans ce cas une droite, de l'ensemble de données d'entrée peut être donnée par l'équation 2.1.

$$y = \frac{\Delta y}{\Delta x} x + b \tag{2.1}$$

Dans l'équation 2.1, $b \in \mathbb{R}$ est l'ordonnée à l'origine, les symboles Δy et Δx représentent les variations sur les deux dimensions du problème. Notons que pour un espace d'entrée à n dimension la surface séparatrice est une hypersurface. Cette formulation peut alors être écrite sous forme cartésienne à l'équation 2.2.

$$\Delta y x + \Delta x b - \Delta x y = 0 \tag{2.2}$$

En introduisant les vecteurs nommés $\mathcal{W} = \begin{bmatrix} w_0 & w_1 & w_2 \end{bmatrix}^\top \in \mathcal{M}_{3,1}(\mathbb{R})$ et $X = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^\top \in \mathcal{M}_{2,1}(\mathbb{R})$ l'équation 2.2 peut être généralisée par l'équation 2.3.

$$w_1 x_1 + w_2 x_2 + w_0 = 0 \tag{2.3}$$

Dans un repère orthonormal en deux dimensions muni de la base canonique $\mathcal{B} = (e_0, e_1)$, l'équation cartésienne d'une droite $\mathcal{D}$ est définie par : $w_1 x_1 + w_2 x_2 + w_0 = 0$. À partir de l'équation de cette droite, une fonction f peut être définie à l'équation 2.4.

$$\begin{aligned} f_{\mathcal{W}} : \mathcal{M}_{2,1}(\mathbb{R}) &\to \mathbb{R} \\ X &\mapsto w_1 x_1 + w_2 x_2 + w_0 \end{aligned} \tag{2.4}$$

Le vecteur de coordonnées $\mathcal{N} = (w_1, w_2)$ étant normal à la droite $\mathcal{D}$ permet de définir un sens pour effectuer la disjonction de cas suivante :

$$f(X) = \begin{cases} > 0, \text{ si } X \text{ est au-dessus de la droite } \mathcal{D} \\ = 0, \text{ si } X \text{ appartient à la droite } \mathcal{D} \\ < 0, \text{ si } X \text{ est en dessous de la droite } \mathcal{D} \end{cases} \tag{2.5}$$

Cette configuration est représentée par le schéma de principe de la figure 2.6.

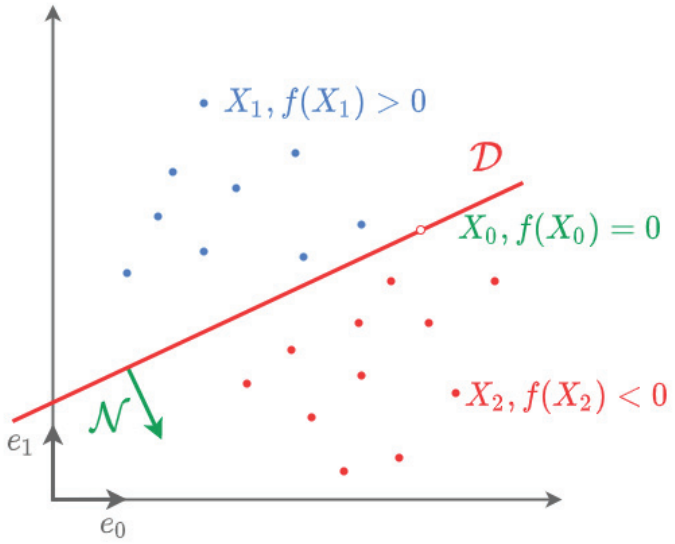


FIGURE 2.6 – Schéma de principe d'une séparation linéaire

De fait, la droite $\mathcal{D}$ ainsi définie scinde l'espace des données en deux dimensions en deux ensembles. La fonction $f_{\mathcal{W}}$ définie à l'équation 2.4 peut alors être modifiée pour lui rajouter un comportement de fonction indicatrice au regard de la position d'un point par rapport à la surface séparatrice. Cette fonction serait donc à valeurs dans $\mathbb{N}$ telle que $f : \mathcal{M}_{2,1}(\mathbb{R}) \to \{0, 1\}$. Une fonction de classification est créée. À chaque élément d'entrée sera ainsi affecté un indice de classe.

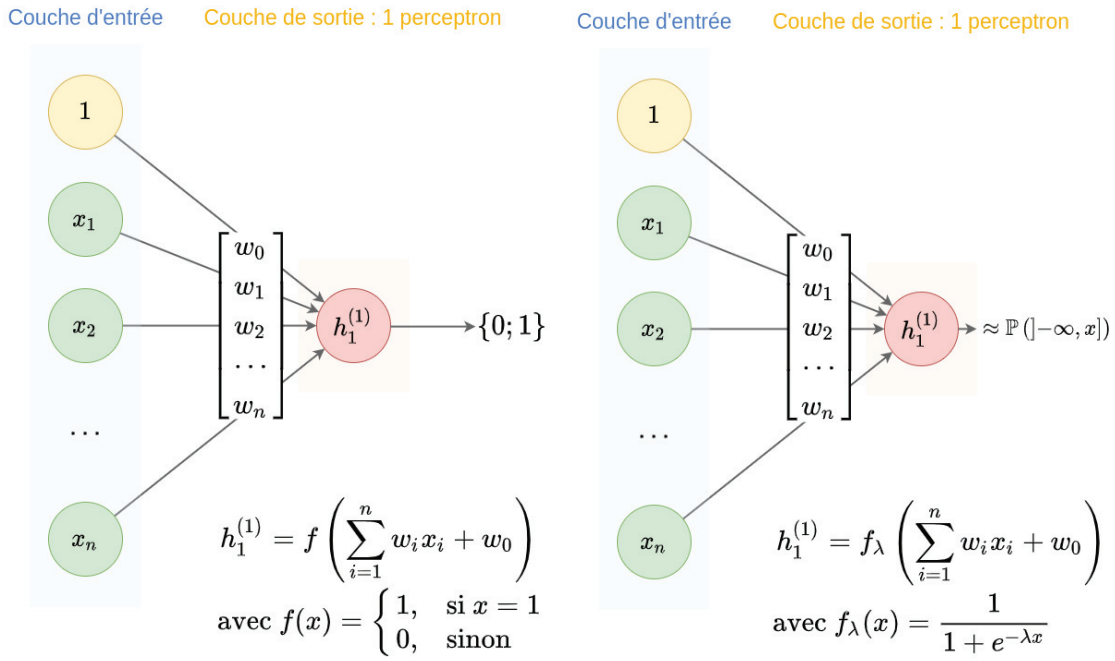
Afin de traduire l'équation 2.3 sous forme de produit scalaire (voir les définitions du produit scalaire .0.3 et du produit matriciel .0.4 en annexes 5) pour généraliser l'approche à un espace à n dimensions, le vecteur d'entrée précédent X est augmenté de l'état constant 1 et devient :

$$\mathcal{X} = \begin{bmatrix} 1 & x_1 & x_2 \end{bmatrix}^\top \in \mathcal{M}_{3,1}(\mathbb{R}) \tag{2.6}$$

En modifiant la fonction $f_\mathcal{W}$ pour prendre ses valeurs dans $\mathcal{M}_{3,1}(\mathbb{R})$, l'équation de la surface séparatrice devient donc l'équation 2.7.

$$\begin{aligned} f_\mathcal{W}(\mathcal{X}) &= \mathcal{W}^\top \mathcal{X} \\ &= \begin{bmatrix} w_0 & w_1 & w_2 \end{bmatrix}^\top \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix} \\ &= w_0 + w_1 x_1 + w_2 x_2 \end{aligned} \tag{2.7}$$

Notons que le fait d'augmenter le vecteur d'entrée permet d'introduire un biais constant w_0 qui représente, dans le cas de la figure 2.6, l'ordonnée à l'origine de la droite séparatrice $\mathcal{D}$. Cela permet de généraliser l'équation de la surface de séparation car sans la présence du biais, cette dernière serait restreinte à l'ensemble des surfaces (droites en deux dimensions) passant uniquement par l'origine. Ainsi, pour discriminer deux classes dans les données d'entrée, 3 pondérations w_i sont au moins nécessaires dans le cas d'un problème en dimension 2. Pour discriminer cette fois-ci n classes, il convient de trouver les $(w_i)_{\forall i \in [\![1;n+1]\!]}$, paramètres de l'hypersurface de dimension au moins n. Seulement, l'hypersurface à trouver n'est pas nécessairement linéaire, c'est pourquoi il convient de rajouter une non linéarité dans le processus. C'est l'idée qui a conduit au développement du *perceptron* par [Rosenblatt, 1958]. Un perceptron représente un des premiers éléments constitutifs d'un réseau de neurones. Il est formé d'une combinaison linéaire et d'une fonction d'activation non linéaire. Le problème précédent peut alors être reformulé sous forme d'une structure parallèle : un réseau de neurones. Dans un premier temps, la figure 2.7 est une architecture constituée d'un seul perceptron permettant la réalisation d'une classification binaire.



(a) Architecture d'un perceptron pour une classification binaire

(b) Architecture d'une régression logistique pour une classification binaire

Figure 2.7 – Architecture basique de réseaux de neurones pour une classification binaire

Sur la figure 2.7 les entrées sont représentées par le vecteur $\mathcal{X} = \begin{bmatrix} 1 & x_1 & \cdots & x_n \end{bmatrix}^\top \in \mathcal{M}_{n+1,1}(\mathbb{R})$ et l'ensemble des poids du réseau sont représentés par $\mathcal{W} = \begin{bmatrix} w_0 & w_1 & \cdots & w_n \end{bmatrix}^\top \in \mathcal{M}_{n+1,1}(\mathbb{R})$.

Les couches du réseau sont indexées par l'exposant (.). La sous-figure 2.7a représente l'architecture initiale développée par [Rosenblatt, 1958]. Pour des entrées binaires, le but du réseau est de réaliser une fonction logique. Pour déterminer cette fonction, le vecteur des poids $\mathcal{W}$ était initialisé de manière aléatoire à l'étape initiale. Un ensemble d'entraînement constitué de données permettait alors de calibrer le réseau. Lors du passage d'une entrée dans le réseau, si la sortie devait être 1 et le réseau produisait 0 alors les poids correspondant aux entrées unitaires étaient diminués, et inversement lorsque la sortie devait être nulle. Au bout de quelques passages le réseau était entraîné. En plus de la simple fonction logique, une sigmoïde est par la suite rajoutée en fonction d'activation du perceptron pour réaliser une régression logistique, la différence entre les problèmes de classification et de régression étant la nature de la variable de sortie obtenue. En effet, dans la classification, la variable de sortie est une catégorie, le plus souvent un entier naturel, tandis que dans la régression le résultat souhaité est une valeur réelle. Remarquons cependant que malgré son nom trompeur, la régression logistique est considérée comme un algorithme de classification [Chollet, 2019]. Notons que la paternité de l'ajout d'une sigmoïde en tant que fonction d'activation pour des réseaux de neurones artificiels n'est pas clairement définie dans la littérature, cependant la fonction était utilisée en neurosciences par [Wilson et Cowan, 1972]. La sous-figure 2.7b représente une telle structure. Ainsi, le résultat obtenu n'est plus binaire mais donne la probabilité d'occurrence d'un évènement en considérant la sigmoïde comme une fonction de répartition d'une loi logistique. Autrement dit, le modèle appris prédit la probabilité qu'un événement arrive en considérant l'information contenue dans l'ensemble d'entrée $\mathcal{X}$.

Chacun des réseaux présenté à la figure 2.7 peut alors être modifié pour réaliser une classification à plusieurs classes ou prédire l'occurrence de plusieurs événements. Pour ce faire, de multiples perceptrons sont rajoutés dans la couche unique de sortie, comme le montre les figures 2.8 et 2.9. L'ensemble des poids du réseau n'est plus considéré comme un vecteur mais comme une matrice $\mathbb{W}$. En effet, dans ce type de structures complètement connectées, chaque liaison à un perceptron possède un poids spécifique et pour l'instant, ils sont tous considérés comme différents. La figure 2.8 met en exergue le fait que chaque perceptron correspond à une classe, c'est pourquoi la couche de sortie en possède K. Remarquons qu'ici parler de K *perceptrons* est un abus de langage car l'unique couche du réseau possède une seule fonction d'activation multi-entrées. Cette fonction d'activation f sélectionne le perceptron (ou le *neurone* par abus de langage) qui possède la valeur d'activation la plus élevée. La classification est ainsi faite.

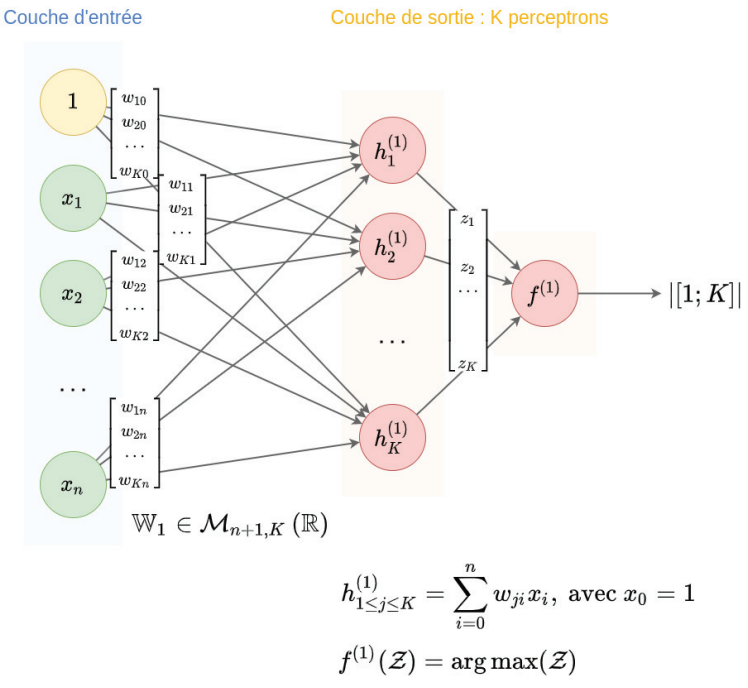
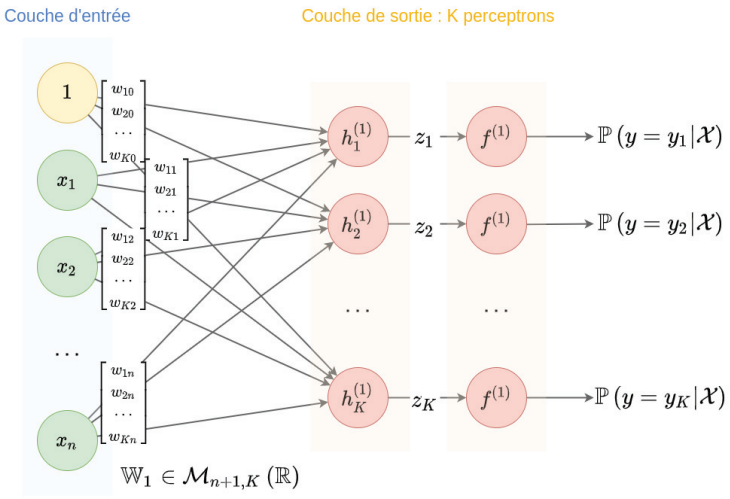


Figure 2.8 – Schéma de principe d'un perceptron pour une classification à K classes



$$h^{(1)}_{1\leq j\leq K} = \sum_{i=0}^{n} w_{ji}x_i, \text{ avec } x_0 = 1$$

$$f^{(1)}(z_j) = \frac{e^{z_j}}{\sum_{j=1}^{K} e^{z_j}}$$

FIGURE 2.9 – Schéma de principe d'une régression logistique pour une classification à K classes

Remarque. *Pour éviter toute confusion, notons la différence entre* $\arg\max f(x)$ *et* $\max f(x)$ *pour une fonction* $f : E \to F$ *avec* (E, F) *deux espaces vectoriels quelconques. La valeur maximale de* $f(x)$, $\max f(x)$, *est à valeurs dans l'espace image* F *de* f, *sous réserve d'existence. On a alors* $f(x) \in F$. *Tandis que* $\arg\max f(x)$ *désigne la valeur de l'antécédent* x *pour laquelle le maximum de la fonction est atteint, ainsi* $\arg\max f(x) \in E$. *Notons qu'il peut y avoir plusieurs valeurs d'antécédents pour lesquels* $f(x)$ *atteint un maximum, dans ce cas* $\arg\max f(x)$ *est un ensemble.*

Notons que la figure 2.9, généralisation de la sous-figure 2.7b, est un réseau ne possédant toujours qu'une seule couche. Mais, à la différence de la figure 2.8, l'architecture a bien K perceptrons différents, détenant chacun leur propre fonction d'activation. Cette fois-ci, la fonction d'activation est la généralisation de la fonction de répartition sigmoïde. C'est la fonction exponentielle normalisée ou *softmax* [BRIDLE, 1989]. Le résultat obtenu peut être interprété comme la probabilité qu'aurait un élément d'appartenir à une certaine classe.

Ainsi, le problème initial de classification revient à déterminer un vecteur de paramètres $\mathcal{W}$ ou une matrice de paramètres $\mathbb{W}$ permettant de caractériser l'hypersurface séparatrice entre les ensembles à classifier. Comme vu précédemment, le choix de ces poids se faisait « à la main » en comparant la sortie obtenue du réseau à la sortie souhaitée. En ce sens, l'apprentissage était manuel et supervisé. La sous-section 2.2.2, présentera les techniques d'apprentissage automatique qui ont fait le succès des réseaux de neurones et participé à l'adoption de ces structures par une audience plus large.

2.2.2 L'apprentissage automatique

Dans la section 2.2.1, l'apprentissage d'un réseau à une couche était réalisé manuellement. De fait, l'adoption de ce type d'architectures pour approximer des fonctions plus complexes en utilisant les résultats du théorème universel 2.2.1 conduit à ce qui sera appelé plus tard par [NEAL, 1996], un « réseau infini ». Ce sont les prémices de l'apprentissage profond et de la prolifération de couches augmentant ainsi la profondeur des réseaux de neurones afin d'améliorer leurs capacité d'approximation de fonctions complexes. Avant de réaliser un bref historique des méthodes d'apprentissages automatiques, explicitons la notion même d'*apprentissage* dans ce cadre.

L'ouvrage de [CORNUÉJOLS et MICLET, 2010] fournit une explication du fonctionnement de l'apprentissage dans un cadre général. Dans le cas des réseaux de neurones, celui-ci peut être qualifié d'*incrémental* dans le sens où chaque étape de l'apprentissage nécessite une nouvelle donnée à partir de laquelle les poids des connexions sont modifiés. Pour fournir cette donnée, plusieurs méthodes de sélection sont principalement utilisées. Soit les données sont fournies à l'algorithme à

la suite d'un tirage aléatoire avec (ou sans) remise, soit elles sont fournies de manière séquentielle au réseau. La première méthode, ne considérant que le caractère séquentiel des données d'apprentissage, ne doit pas être apprise par le réseau tandis que la seconde se concentre sur ce dernier point. Trois types d'apprentissages sont alors discernés : l'apprentissage supervisé, l'apprentissage semi-supervisé et l'apprentissage non supervisé. Dans l'apprentissage supervisé, les données d'entrée sont couplées avec une classe d'appartenance, cette dernière étant représentée par un vecteur de dimension égale au nombre de classes contenues dans l'ensemble d'apprentissage. De fait, cette méthode pourrait être utilisée pour l'apprentissage des structures présentées aux figures 2.7, 2.8 et 2.9. Ainsi, l'ensemble d'apprentissage est augmenté de la sortie désirée du réseau pour chaque entrée. À chaque entrée correspond un label. Dans le cas de l'apprentissage semi-supervisé, l'ensemble des labels des données d'entrées est incomplet et dans le cas de l'apprentissage non supervisé, les labels sont inexistants. L'algorithme d'apprentissage doit alors déterminer la structure des données d'entrée. C'est la différence entre un problème de classification et un problème de partitionnement (ou *clustering*). Ces trois cas d'apprentissages possèdent cependant un point commun : ils doivent avoir une structure qui permette la mise à jour des poids du réseau en fonction d'une quantification d'erreur. Le cas échéant, l'ensemble des poids est statique et le réseau n'apprend pas. Cette propagation de l'erreur aux couches d'un réseau afin de mettre à jour ses poids est appelée la *rétro-propagation de gradient* ou *descente de gradient*.

Une première mention de la descente de gradient faite pour trouver le zéro d'une fonction positive continue a été trouvée dans un compte rendu hebdomadaire de l'académie des sciences par Cauchy [Cauchy, 1847]. Bien plus tard, [Robbins et Monro, 1951] ont développé une méthode pour l'approximation stochastique d'une fonction monotone permettant d'estimer de manière itérative la solution d'une équation et démontrant alors que la séquence de solutions estimées converge en probabilité vers la solution théorique du problème. Leurs travaux ont ensuite été repris par [Kiefer et Wolfowitz, 1952] qui reformulent le problème initial de la régression d'une fonction pour arriver à la descente de gradient stochastique. La méthode de résolution d'équations par rétro-propagation de gradient a, quant à elle, était définie algorithmiquement en Fortran par [Linnainmaa, 1976] mais non appliquée à des réseaux de neurones. Plus tard, avec l'augmentation des puissances de calculs, la *généralisation de la loi δ* afin de mettre à jour les poids de réseaux de neurones simulant des fonctions logiques a été développée par [Rumelhart, Geoffrey E. Hinton et Williams, 1986]. En résumé, cette approche développée de manière itérative au fil des années peut être résumée de la manière suivante :

1. Un ensemble de données contenant une entrée ainsi que la sortie souhaitée (le label) est constitué.

2. Le système utilise une entrée pour produire une sortie.

3. La sortie produite est comparée au label, cette comparaison est réalisée à l'aide d'une fonction de coût qui, dans les travaux de [Rumelhart, Geoffrey E. Hinton et Williams, 1986], représente l'écart quadratique entre les deux vecteurs d'entrée et le label.

    - Si la sortie est identique à l'entrée alors l'apprentissage n'est pas nécessaire.
    - Le cas échéant, les poids sont mis à jour en fonction d'une quantité proportionnelle à la différence entre la sortie et le label du réseau.

Lors de la descente de gradient la mise à jour des poids n'est pas réalisée avec une quantité aléatoire. Afin de réaliser un apprentissage d'une fonction, la contribution de chaque poids au résultat est calculée lors d'une première étape de l'apprentissage puis la valeur de ces derniers est modifiée selon le sens qui leur permet de contribuer à se rapprocher d'un optimum représentant la sortie théorique voulue du réseau. De manière plus générale, et comme celle utilisée initialement par Cauchy [Cauchy, 1847], cette technique était employée dans l'optimisation de fonction de classes $\mathcal{C}^1$, soit des fonctions continues et au moins une fois dérivable. Le gradient est présenté à la définition 2.2.1.

La valeur d'un poids à l'étape d'itération $k+1$ courante est alors déterminée à partir de sa valeur à l'instant k et le gradient calculé à l'étape courante. Il peut être modélisé par l'équation 2.8.

$$w_{k+1} = w_k - \alpha \nabla f(w_k) \tag{2.8}$$

Définition 2.2.1 : Gradient

Soit f une fonct ion à n variables différentiables en tout point d'un domaine $\mathcal{D} \subset \mathbb{R}^n$, soit $X = \begin{bmatrix} x_1 & x_2 & \dots & x_n \end{bmatrix}^\top$ un vecteur de $\mathcal{D}$. On appelle gradient de f le champ de vecteur défini sur $\mathcal{D}$ par :

$$\nabla f : X \mapsto \begin{bmatrix} \dfrac{\partial f}{\partial x_1}(X) \\ \dfrac{\partial f}{\partial x_2}(X) \\ \dots \\ \dfrac{\partial f}{\partial x_n}(X) \end{bmatrix}$$

Afin de déterminer l'apport d'une valeur de pondération à l'erreur en sortie du réseau, le gradient est calculé. De manière plus spécifique au domaine, un réseau de neurones peut être considéré comme une fonction à plusieurs variables. Avec le théorème de dérivation des fonctions composées, il est alors possible d'exprimer la variation de la sortie du réseau en fonction de son entrée. Pour ce faire, tous les poids des connexions sont introduits dans le calcul comme des variables intermédiaires. Cela permet de quantifier l'apport de chaque poids à la variation du résultat final. Soit f la fonction représentée par le réseau de neurones et x_i une sortie, l'équation 2.9 représente cette relation.

$$\frac{\partial f}{\partial x_i} = \sum_j \frac{\partial f}{\partial w_j} \frac{\partial w_j}{\partial x_i} \tag{2.9}$$

Pour rétro-propager l'erreur, les librairies de différentiation automatique d'aujourd'hui telles que [Abadi *et al.*, 2015] ou [Paszke *et al.*, 2019] représentent la structure du réseau sous forme de graphe dont chaque noeud pourrait représenter un poids, la donnée d'entrée étant alors propagée aux autres noeuds du réseau. C'est en partie pour cela que les calculs d'apprentissages sont extrêmement coûteux en ressources. Afin justement d'améliorer les performances de l'apprentissage, qui est une optimisation de fonction, des variantes de la méthode originelle de descente de gradient ont été développées. Dans le cas des réseaux de neurones profonds, d'après [R. Zhao *et al.*, 2019], le pré-apprentissage non supervisé d'un réseau profond présenté par [Geoffrey E. Hinton, 2007] est une avancée importante dans le domaine de la recherche. La procédure consiste à initialiser les paramètres d'un réseau profond non pas par des poids aléatoires mais par des poids résultats d'une première phase d'optimisation non supervisée. Puis il convient de modifier plus finement ces derniers par l'approche standard de rétro-propagation. [Geoffrey E. Hinton, 2007] a alors démontré que cette approche accélérait la convergence et la précision des résultats obtenus par le réseau.

Ainsi, l'ajout d'un terme de régularisation, un *moment*, pour accélérer la convergence de la descente de gradient stochastique a été fait par [Qian, 1999]. De nombreux autres algorithmes ont été développés pour optimiser la phase d'apprentissage du réseau. En rajoutant un terme permettant de prédire la solution itérée suivante [Nesterov, 1983] permet d'améliorer la convergence de l'apprentissage du Recurrent Neural Network (RNN). Plus récemment, [Duchi, Hazan et Singer, 2011] créent une nouvelle approche appelée Adaptive Subgradient Method (ADAGRAD) en adaptant le taux d'apprentissage localement, en fonction de chaque paramètre du réseau. Deux ans plus tard, Adaptive Learning Rate Method (ADADELTA) [Zeiler, 2012] est créé comme une extension de ADAGRAD qui, au lieu de conserver toutes les valeurs précédentes de gradients, les annule au delà d'une certaine fenêtre. La dynamique de convergence est encore modifiée par [Kingma et Ba, 2017] qui développent l'algorithme *Adam* stockant non plus les gradients passés mais une décroissance exponentielle de ces derniers. Cette méthode utilise la norme $\mathcal{L}_2$ à la différence de *AdaMax* [Kingma et Ba, 2017] qui utilise la norme infinie. Une combinaison de toutes ces dernières dynamiques est alors apportée par Nesterov-accelerated Adaptive Moment Estimation (NADAM) [Dozat, 2016]. Enfin, pour les cas où *Adam* convergerait vers une solution sous optimale, [Reddi, Kale et Kumar, 2018] ont développé l'algorithme *AMSGRAD*, classiquement utilisé, aujourd'hui, dans l'apprentissage profond. Pour une illustration plus complète des différents algorithmes de descente de gradient, le lecteur pourra se référer à l'état de l'art de [Ruder, 2017]. Des méta-heuristiques à partir d'algorithmes génétiques peuvent aussi être utilisées [Liao, Lei Zhang

et W. Li, 2017]. De plus, pour l'apprentissage de réseaux de neurones complètement connectés de petites tailles, l'algorithme d'estimation des moindres carrés non linéaires de [Levenberg, 1944] et [Marquardt, 1963] est parfois utilisé.

Seulement, bien que cette méthode d'apprentissage ait été améliorée au fil des années, le principe de la descente de gradient reste coûteux et parfois non pertinent pour la résolution de certains problèmes. Notons par exemple les problèmes bien connus d'explosion (ou *exploding gradient*) ou de disparition (*vanishing gradient*) du gradient lors de la rétro-propagation permettant l'apprentissage de certaines structures de réseaux de neurones dès lors que de nombreuses couches dites « cachées » sont rajoutées. Les travaux de [Metz *et al.*, 2021] montrent alors que la descente de gradient n'est pas adaptée lors de la résolution de certains problèmes physiques, notamment pour la simulation de systèmes dynamiques que sont des processus de type récurrents. Le calcul de variations à travers toutes les couches du réseau nécessite un Jacobien (voir définition .0.2 en annexes 5) qui lorsque son spectre possède des valeurs propres supérieures à 1 conduit, lors du processus itératif, à une explosion du gradient lors de l'optimisation. Au-delà de cette explosion, le problème réside dans le fait que la descente de gradient ne permet pas alors de converger vers une solution mais présente une variation plutôt chaotique.

2.2.3 Les réseaux de neurones peu profonds appliqués à la surveillance d'état de santé

Comme cela a été vu à la sous-section 2.2.1, les réseaux de neurones permettent d'apprendre les paramètres de surfaces séparatrices linéaires ou non linéaires. Ils sont principalement utilisés dans des problèmes de classification supervisés et non supervisés. De ce fait, ils sont intéressants pour les problématiques de diagnostic à partir de reconnaissance de formes. Ils présentent ainsi des résultats prometteurs en diagnostic de défauts de roulements dans [Liao, Lei Zhang et W. Li, 2017] où un réseau de neurones est couplé avec un algorithme d'optimisation par essaim de particules. Dans le PHM où la physique des systèmes possède une place importante afin de pouvoir remonter aux sources des défauts, ces approches de type « boîtes noires » permises par l'apprentissage d'un réseau de neurones artificiel n'en demeurent pas moins performantes pour la détection et la prédiction d'anomalies sans pour autant en déterminer la cause.

Les réseaux de neurones sont des outils d'algèbre linéaire bien adaptés à la résolution des problématiques de diagnostic. En effet, le diagnostic est avant tout un problème de classification supervisée ou non. En associant un certain de nombre de classes à un certain nombre de défauts, déterminer l'état de santé d'un système revient à assigner une classe à un ensemble de données. En résumé, dans le cas du diagnostic on cherche à résoudre un problème de classification de manière automatique et fiable. Autrement dit, on cherche à discriminer des ensembles dans un univers choisi. Un ensemble étant caractérisé par les éléments qui le composent, on cherche alors à discriminer des éléments entre eux. En prenant l'exemple de la classification binaire étudiée à la section 2.2.1, une classe pourrait être attribuée à un état sain et une autre à un état en défaut. Mais la nature du défaut ne sera cependant pas déterminée. Pour ce faire, et tenter de pallier les désavantages d'un modèle de type « boîte noire », il serait nécessaire de contenir l'information de plusieurs défauts dans la dynamique de l'espace des données d'entrée. Ainsi, lors de l'apprentissage supervisé, le réseau pourra modifier la valeur de ses poids pour apprendre à détecter les défauts que l'utilisateur aurait demandé de reconnaître. Le pronostic de l'état de santé d'un système, quant à lui, revient à classifier des données extrapolées, en faisant l'hypothèse que les données utilisées sont représentatives du comportement du système étudié.

Cependant, afin de lier le monde de la physique et celui de l'apprentissage artificiel, les travaux de [Raissi et Karniadakis, 2018] s'intéressent à développer une approche hybride. En effet, dans [Raissi et Karniadakis, 2018] les auteurs s'attachent à résoudre des équations de la physique aux dérivées partielles par des méthodes d'apprentissage machine à base de processus gaussiens. Là où un réseau de neurones classique modélise une fonction déterministe, un processus gaussien permet de modéliser un processus stochastique. Un moyen de remédier à cette limite fondamentale des structures à peu de couches serait d'augmenter la profondeur des réseaux considérés, partant ainsi du principe qu'un réseau de neurones profond peut représenter toute fonction aussi complexe soit elle. Dans le cas d'un processus gaussien permettant de représenter des dynamiques stochastiques, [Neal, 1996] démontre qu'un réseau de neurones bayésien « infini » (dont les poids sont des distributions et non des réels), possédant une largeur de couche illimitée, converge vers un processus gaussien.

2.2.4 Les réseaux de neurones profonds

Cette sous-section 2.2.4 présente les structures de réseaux de neurones « élémentaires » les plus utilisées dans le domaine de l'apprentissage profond. Dans un premier temps, les CNN seront présentés au paragraphe 2.2.4.1, puis les AE au paragraphe 2.2.4.2, les Deep Belief Network (DBN) au paragraphe 2.2.4.3 et enfin, dédiés aux informations séquentielles, les RNN au paragraphe 2.2.4.4. Notons que dans la littérature, les architectures sont souvent des variantes ou des combinaisons de ces derniers réseaux d'où le terme d'« élémentaire » utilisé ici.

2.2.4.1 Le réseau de neurones à convolutions ou *Convolutional Neural Network - CNN*

La sous-section 2.2.1 introduit la notion de classification avec des réseaux connexionnistes peu profonds. Dans le cas d'un problème de classification binaire, en considérant l'architecture de la figure 2.7a avec un vecteur d'entrée $\mathcal{X} \in \mathcal{M}_{(3,1)}(\mathbb{R})$ il y a 3×1 paramètres à optimiser. Dans le cas de la figure 2.7b, il y a toujours 3 paramètres mais un hyperparamètre est rajouté avec la sigmoïde. En considérant toujours le même vecteur d'entrée $\mathcal{X}$ avec un réseau discriminant K classes comme celui de la figure 2.8, ou même de la figure 2.9, il y a $4K$ paramètres à optimiser. Notons l'ajout du biais en entrée du réseau d'où le fait qu'il y ait $4K$ paramètres à optimiser et non $3K$. En extrapolant le résultat pour une entrée de taille n, cela revient à calculer $K(n+1)$ poids pour optimiser un réseau du même type que ces deux derniers. Notons alors que ces architectures ne possèdent aucune couche cachée. Intuitivement, pour approximer des hypersurfaces sur des topologies complexes, les calculs d'apprentissage des réseaux auront alors tendance à exploser. Prenons l'exemple de la reconnaissance d'images pour nous en convaincre. Soit une image en niveaux de gris de dimension 256×256 pixels pour effectuer, par exemple, de la segmentation, *i.e.* trouver et détourer des zones de l'image correspondant à une certaine caractéristique. Dans le cadre des architectures complètement connectées, en considérant l'image comme un matrice $\mathbb{I} \in \mathcal{M}_{(256,256)}([0;255])$, il convient de la transformer en un vecteur de taille $256 \times 256 = 65\,536$. Imaginons que l'architecture en question possède une couche avec autant de neurones qu'il y a d'entrées, il y aura donc $65\,536^2 \approx 4 \times 10^9$ poids à optimiser par rétro-propagation de gradient. Pour un réseau de neurones complètement connecté à une seule couche, au moins 4 milliards de poids sont nécessaires et l'objectif d'effectuer une segmentation ne sera pas atteint.

Avant d'être utilisé dans des réseaux avec un grand nombre de couches cachées, le CNN tire son origine du *Cognitron* de [Fukushima, 1975][1]. L'idée sous-jacente d'une telle structure était de pouvoir reproduire le fonctionnement d'un champ réceptif du cortex visuel des mammifères, et, plus précisément, du chat [Hubel et Wiesel, 1962]. L'hypothèse principale était que les connexions synaptiques (les liens avec les poids dans les architectures artificielles), proches physiquement les unes des autres, catalysaient leurs activations mutuelles. L'architecture a ensuite évolué dans [Fukushima, 1980] sous le nom de *Neocognitron* pour permettre de reconnaître des schémas structurels dans des données et ce de manière invariante aux translations des images de digits qui lui étaient fournies en entrée. Plus tard, [Y. LeCun *et al.*, 1989], à partir de ses travaux à l'Université de Toronto [Y. LeCun, 1989], portant notamment sur le partage des poids pour optimiser l'apprentissage d'une structure neuronale, adapta le principe de rétro-propagation pour l'apprentissage à cette structure. Le réseau de neurones à convolution noté CNN était né. Il permet alors de mutualiser les poids de certaines connexions. Dans le cas d'une image, au lieu de considérer l'image complète pour réaliser une transformation, des fenêtre successives sont utilisées. La structure permet de réduire drastiquement le nombre de connexions, d'axones, et ainsi peut être utilisée pour la détection de schémas structurels dans des données d'entrées de plus grande dimension. Le réseau apprend donc à effectuer des produits de convolution et chaque couche extrait plusieurs images résultantes correspondant chacune à un ensemble de poids et donc dans la pratique, à un filtrage. Le principe est illustré à la figure 2.10.

[1] Le *Cognitron* a été défini dans un premier temps dans l'article du même nom une année plus tôt mais uniquement en langue japonaise. Ici seule la référence anglo-saxonne est citée à cause de la barrière de la langue.

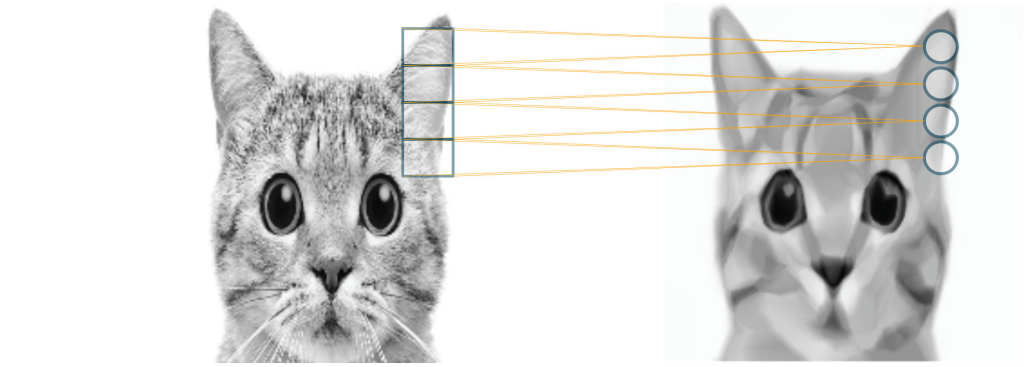


FIGURE 2.10 – Illustration de ce que pourrait être le résultat d'une couche de convolution

La figure 2.10 présente un résultat qui pourrait être produit par une couche de convolution prenant en entrée une image à un canal pour créer une sortie filtrée. Les deux fenêtres en gris sur l'image originale (à gauche) définissent le champ visuel d'intérêt considéré pour l'opération de convolution discrète. La couleur du trait en jaune représente l'ensemble des poids définissant le filtre discret. Notons que pour une image filtrée donnée, comme expliqué précédemment, un ensemble de poids fixé est utilisé sur l'entièreté de l'image. Le résultat de l'apprentissage d'une telle structure est de déterminer le noyaux de convolution, qui est une matrice, comme exposé à la définition 2.2.2.

Définition 2.2.2 : Produit de convolution discret

Soit $(k_1, k_2) \in \mathbb{N}^2$ *les dimensions du noyau de convolution, autrement dit, du filtre* $K \in \mathcal{M}_{(2k_1+1,2k_2+1)}(\mathbb{R})$. *Soit* $(n, m) \in \mathbb{N}^2$, $I \in \mathcal{M}_{(nm)}(\mathbb{R})$ *la matrice associée d'une fonction quelconque. Soit* $(i, j) \in \mathbb{N}^2$ *les indices de la position de l'élément considéré dans la matrice résultante du produit de convolution. Le produit de convolution discret de* I *par le filtre* K *est défini par :*

$$(I * K)_{i,j} = \sum_{u=-k_1}^{k_1} \sum_{v=-k_2}^{k_2} K_{u,v} I_{r+u,s+v}$$

Avec K *le noyau ou* kernel *défini comme :*

$$K = \begin{bmatrix} K_{-k_1,-k_2} & \cdots & K_{-k_1,k_2} \\ \vdots & K_{0,0} & \vdots \\ K_{k_1,-k_2} & \cdots & K_{k_1,k_2} \end{bmatrix}$$

La figure 2.11 extraite de [LOPEZ PINAYA *et al.*, 2020] représente un exemple de convolution discrète avec un noyau de taille 3×3 sur une matrice de taille 5×6.

D'après la représentation de la figure 2.11, une couche de convolution applique de ce fait un filtre à une image d'entrée (ou plus généralement une matrice) en effectuant la somme des éléments de la matrice obtenue par le produit de Hadamard entre le filtre (ou le noyau de convolution) et l'image d'entrée de couche. La taille de la matrice noyau est interprétée comme le champ récepteur de la couche. En faisant le parallèle avec la biologie, le champ récepteur est une région sensorielle dont l'activité peut influencer l'activité électrique de cellules sensorielles et plus particulièrement de neurones, *via* des connexions synaptiques [LEVITT, BHUTIA et ROGERS, 2018]. L'inspiration du cortex visuel des mammifères pour créer cette structure de convolution est bien présente. Ce parallèle permet en plus de diminuer drastiquement le nombre de poids du réseau. Notons que le filtre ainsi créé translatera pour s'appliquer à l'entièreté de l'image d'entrée. Cette translation par application successive peut se faire de plusieurs manières. Les propriétés principales qui la caractérisent sont : le remplissage (ou *padding*), l'enjambement (ou *stride*) ou la dilatation. Avec la figure 2.11, en appliquant le noyau de convolution tel quel, sans remplissage, avec un enjambement unitaire et sans dilatation, l'image filtrée résultante (*Feature map*) serait de taille 3×4. Remarquons que ce dernier calcul est un exemple dont le résultat n'est pas représenté à la figure 2.11. Rajouter un remplissage avec, par exemple, des zéros, permet d'augmenter la dimension de la donnée d'entrée

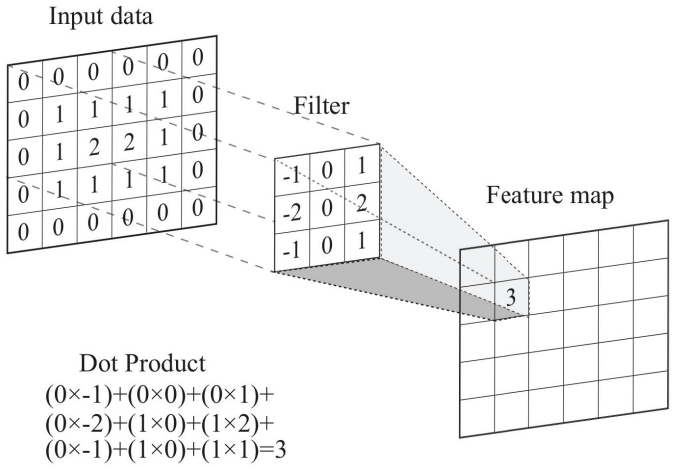


FIGURE 2.11 – Exemple de convolution discrète, d'après [LOPEZ PINAYA *et al.*, 2020, figure 10.2]

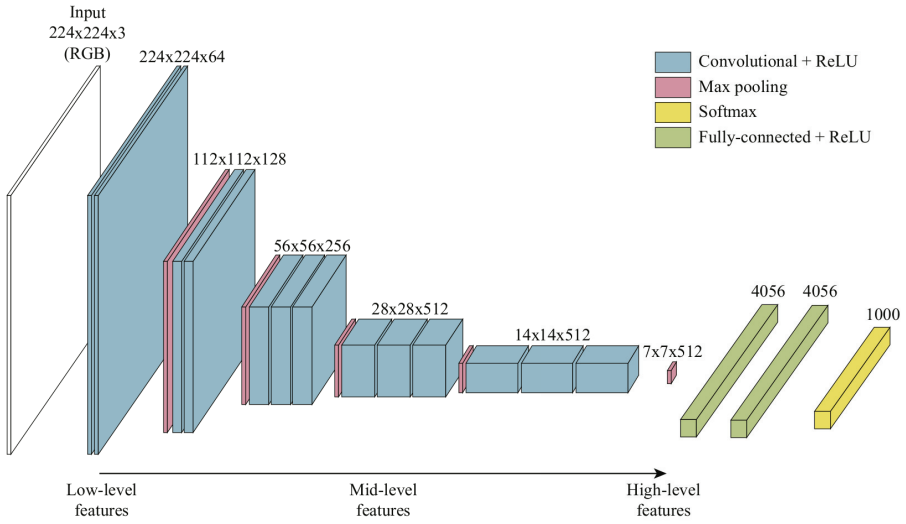


FIGURE 2.12 – Architecture de réseau de neurones profond à convolution VGG16 [SIMONYAN et ZISSERMAN, 2015], d'après une représentation de [LOPEZ PINAYA *et al.*, 2020]

en ajoutant deux lignes et deux colonnes de part et d'autre de l'image. De fait, certains motifs présents aux limites de l'image d'entrée sont mieux reconnus et la sortie serait de dimension 5×6. L'enjambement, lui, définit le nombre de pixels à sauter lors de la translation du noyau. La dilatation permet de contrôler l'espacement entre chaque élément du noyau de convolution. Ce qui lui permet d'être appliqué à des éléments non adjacents de la matrice filtrée. Un résumé visuel des opérations permises par les couches de convolution a été réalisé par [DUMOULIN et VISIN, 2016]. Afin de découvrir comment ces couches de convolutions pourraient être agencées dans un réseau de convolution profond, la figure 2.12 représente l'architecture gagnante deux années de suite sur le challenge ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2012 et 2013 [RUSSAKOVSKY *et al.*, 2015].

La figure 2.12 présente une architecture complète d'un réseau de neurones profond avec des couches de convolution dont l'entrée est une image de taille 224×224 à trois canaux (Red Green Blue (RGB)). La première couche ne réduit pas la taille de l'entrée mais extrait 64 autres images filtrées d'où la troisième dimension au dessus du schéma de chaque groupe. Notons que le premier groupe de couches est constitué de deux couches de convolutions. Avec, par exemple un noyau de dimension 4×4, une couche de convolution nécessiterait $4 \times 4 \times 64 = 1\,024$ poids à déterminer lors de la phase d'apprentissage. C'est cette juxtaposition d'opérations élémentaires qui permet au réseau d'extraire plusieurs dimensions de représentation d'information. De ce fait, l'apprentissage profond, automatise l'étape coûteuse de la création de descripteurs à partir des données brutes (*feature engineering*). Le modèle apprend alors les différentes représentations de l'information

de manière jointe et lorsque le modèle doit être modifié pour servir un objectif final, toute les caractéristiques qui en dépendent s'adaptent automatiquement [Chollet, 2021, section 1.2.6]. Au delà du premier groupe de couches de convolutions, une couche de sous-échantillonnage en rose (*max pooling*) permet de réduire la dimension de l'information propagée. Au lieu d'utiliser un noyau de convolution, et donc une transformation linéaire apprise, la couche de sous-échantillonnage permet de réduire la dimension de l'image avec une opération simple telle que le maximum ou la moyenne. Ces deux techniques sont représentées à la figure 2.13.

(a) Technique dite de *max pooling* pour sous-échantillonner une matrice

(b) Technique dite d' *average pooling* pour sous-échantillonner une matrice

Figure 2.13 – Techniques principales de sous-échantillonnage d'images

Enfin, après les différents groupes de couches de convolution et de sous-échantillonnage, le réseau de la figure 2.12 se termine par deux couches complètement connectées et une couche *softmax* (voir la définition de la fonction $f^{(1)}$ à la figure 2.9) d'une dimension égale au nombre de classes à déterminer dans les données d'entrée.

Dans la littérature, le réseau de neurones à convolutions fait ainsi partie des méthodes d'apprentissage profond. Il est principalement utilisé dans la reconnaissance d'images ou encore dans la compréhension du langage pour la reconnaissance de parole (le mot est alors un vecteur réel) [Y. Kim, 2014]. Leur succès, et celui des architectures profondes au sens large, est en partie dû à l'usage des Graphics Processing Unit (GPU) pour effectuer les calculs d'apprentissage. De plus, une nouvelle génération de fonctions d'activation permet d'améliorer les résultats d'apprentissage de tels réseaux [Glorot, Bordes et Bengio, 2011]. La fonction d'activation non linéaire Rectified Linear Unit (ReLU), a été introduite par [Hahnloser *et al.*, 2000] dans ses travaux sur la simulation d'un réseau de neurones avec un circuit de composants électroniques en justifiant l'activation par des hypothèses biologiques et mathématiques, puis elle fut utilisée par [Jarrett *et al.*, 2009] pour la détection de formes avec un réseau de convolutions. Elle fut enfin popularisée à partir de son usage pour les machines de Boltzmann resctrictives — Restricted Boltzmann Machine (RBM) dans [Nair et Geoffrey E. Hinton, 2010]. Les techniques de *dropout* [Geoffrey E. Hinton, Srivastava *et al.*, 2012] permettent aussi d'améliorer l'apprentissage de structures profondes[2] en ignorant certaines parties du réseau dans la boucle d'optimisation. Enfin, les techniques de génération artificielle de données permettent encore de pallier les manques dans l'ensemble de données d'entraînement et ainsi d'obtenir une meilleure performance du réseau [Yann LeCun, Bengio et G. Hinton, 2015]. Aujourd'hui, les CNN sont des réseaux de neurones incontournables dans le domaine de l'apprentissage profond.

2.2.4.2 L'auto-encodeur ou *Auto-Encoder - AE*

L'auto-encodeur ou auto-associateur a été défini pour la première fois par [Rumelhart, Geoffrey E. Hinton et Williams, 1986] en tant que structure avec une couche cachée pour encoder l'information à son entrée. Il a été ensuite repris par [Bourlard et Kamp, 1988], puis par [Baldi et Hornik, 1989] afin de réaliser une opération similaire à une Principal Component Analysis (PCA). Une équipe française l'a alors redéveloppé sous le nom de structure *Diabolo* presque simultanément par [Geoffrey E Hinton et Zemel, 1994] et [Schwenk et Milgram, 1994]. Enfin, les différences

[2]Notons qu'un brevet a été déposé par Google [Geoffrey E. Hinton, Krizhevsky *et al.*, 2016] sur cette méthode d'apprentissage artificielle, tout comme l'apprentissage de CNN sur des GPU déposé par Microsoft [Puri, 2010].

fondamentales avec la PCA ont été traitées dans [Japkowicz, Jose Hanson et Gluck, 2000]. Plus tard, cette structure fut utilisée pour l'initialisation des paramètres d'un CNN par [Ranzato *et al.*, 2006]. En effet, au début initialisés par des valeurs aléatoires, aujourd'hui afin d'optimiser la convergence de l'apprentissage du réseau de neurones profond, les points initiaux sont souvent choisis comme étant eux-mêmes les résultats d'une optimisation. L'auto-encodeur est constitué de trois parties distinctes : un encodeur, une couche cachée et un décodeur. Il peut être représenté par le schéma de la figure 2.14.

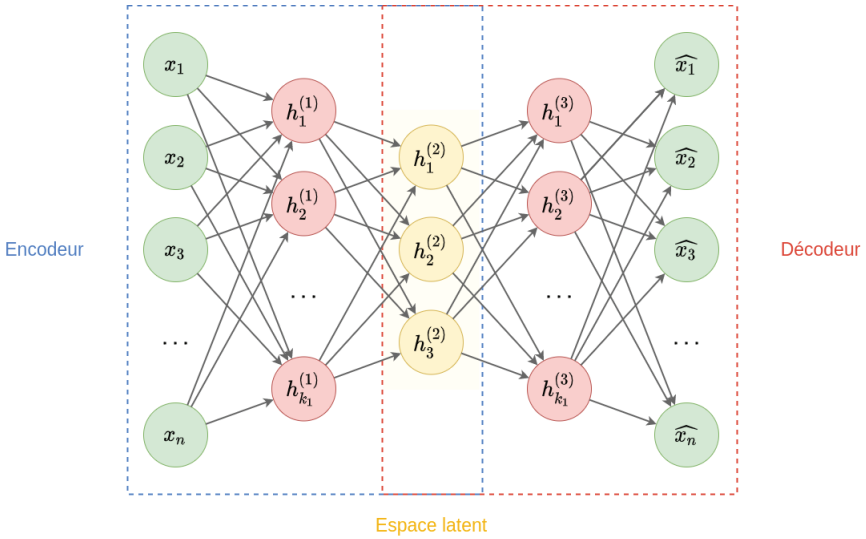


Figure 2.14 – Schéma de principe d'un auto-encodeur à une couche cachée

La figure 2.14 représente un auto-encodeur à 3 couches cachées. La partie gauche entourée en bleu représente l'encodeur qui extrait une première représentation de l'information à partir du vecteur d'entrée X et la partie de droite entourée en rouge représente la partie décodeur. À l'intersection des deux ensembles se trouve l'espace latent qui contient la représentation de l'information finale de l'architecture. À la différence d'un CNN, l'AE possède autant de neurones sur sa couche d'entrée que sur sa couche de sortie. Cette particularité est le reflet de sa fonction. En effet, les premières couches de la partie *encodeur* extraient des représentations de l'information successives pour arriver à une information compressée dans l'espace latent. La partie *décodeur* essaie ensuite de reconstruire l'entrée du réseau à partir de l'information de l'espace latent. Le processus d'apprentissage revient alors à minimiser l'erreur entre l'entrée de l'encodeur et la sortie du décodeur. De manière plus formelle avec l'exemple de la figure 2.14, soit $X = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}^\top \in \mathcal{M}_{n,1}(\mathbb{R})$ et $\hat{X} = \begin{bmatrix} \hat{x}_1 & \hat{x}_2 & \cdots & \hat{x}_n \end{bmatrix}^\top \in \mathcal{M}_{n,1}(\mathbb{R})$ respectivement le vecteur d'entrée du réseau et le vecteur de sortie. Soit $\mathcal{C} = \begin{bmatrix} c_1 & c_2 & c_3 \end{bmatrix}^\top$ le vecteur de l'espace latent produit par les neurones $\left(h_1^{(2)}, h_2^{(2)}, h_3^{(2)}\right)$. Définissons une fonction permettant de représenter les opérations effectuées par une couche de réseau à l'équation 2.10.

$$\begin{aligned} f^{(x)}_{m\to n,\mathcal{W}} : (\mathbb{R}, \mathcal{M}_{m,1}(\mathbb{R}) &\mapsto \mathcal{M}_{n,1}(\mathbb{R}) \\ (x, \mathcal{X}) &\to \mathcal{W}^{(x)} \cdot \mathcal{X} \end{aligned} \tag{2.10}$$

Dans l'équation 2.10, $x \in \mathbb{N}$ représente l'indice de la couche du réseau considérée, $n \in \mathbb{N}$ représente le nombre de neurones d'une couche (et donc la dimension du vecteur de sortie de la couche), $\mathcal{W}^{(x)}$ représente le vecteur de poids affecté à la x^{e} couche et $\mathcal{X}$ représente le vecteur d'entrée d'une couche, augmenté d'un biais constant. Dans notre exemple, le vecteur $\mathcal{C}$ obtenu dans l'espace latent ainsi que la sortie du réseau peuvent alors être définis par l'équation 2.11.

$$\begin{aligned} \mathcal{C} &= f^{(2)}_{k_1\to 3,\mathcal{W}} \circ f^{(1)}_{n\to k_1,\mathcal{W}} \\ \hat{X} &= f^{(3)}_{k_1\to n,\mathcal{W}} \circ f^{(2)}_{3\to k_1,\mathcal{W}} \end{aligned} \tag{2.11}$$

L'apprentissage du réseau s'effectue alors en minimisant l'erreur d'estimation entre X et $\hat{X}$. Deux fonctions de coûts sont principalement utilisées selon [Hoang et H.-J. Kang, 2019] :

- La minimisation de l'erreur moyenne quadratique définie telle que :

$$\min J(\mathcal{W}^1, \mathcal{W}^2, \mathcal{W}^3) = \min \frac{1}{n} \sum_{i=1}^{n} (\hat{x_i} - x_i)^2 \tag{2.12}$$

- La minimisation de l'entropie croisée permet de mesurer la distance (ou plutôt *divergence*) entre deux distributions de variables aléatoires. Ici elle est définie telle que :

$$\min J(\mathcal{W}^1, \mathcal{W}^2, \mathcal{W}^3) = \min \frac{1}{n} \sum_{i=1}^{n} x_i \log(\hat{x_i}) + (1 - x_i) \log(1 - \hat{x_i}) \tag{2.13}$$

En fin d'apprentissage, la couche représentant l'espace latent produit une représentation de l'information compressée possédant en théorie la quintessence de l'information d'entrée du réseau. Ainsi, lors de la phase d'inférence, seule la partie encodeur est utilisée, le décodeur ne sert qu'à l'apprentissage pour diriger le réseau vers une représentation latente optimale. Dans notre domaine, le parallèle peut être fait entre la *représentation* de l'information apprise, qui est jusqu'ici un concept un peu abstrait, et les signatures de défauts ou *descripteurs*. Ainsi, un AE pourrait extraire des descripteurs de l'information d'entrée et donc extraire des signatures pertinentes pour notre problématique. Dans la section 2.3.3, des exemples seront donnés sur leur utilisation dans le domaine du PHM. La profondeur d'un AE dépend alors du nombre de couches neuronales avant obtention de l'espace latent.

2.2.4.3 Le réseau à conviction profonde ou *Deep Belief Network - DBN*

Les machines de Boltzmann — Boltzmann Machine (BM) (nommées ainsi pour faire référence à la distribution de probabilité du même Boltzmann) tirent leur inspiration des réseaux de Hopfield [Hopfield, 1982], un réseau à deux couches permettant de créer des séquences binaires. C'est une année plus tard que [Geoffrey E. Hinton et J. Sejnowski, 1983] et [Fahlman, Geoffrey E. Hinton et T. J. Sejnowski, 1983] développent une méthode d'apprentissage pour ces architectures, à base de calcul d'énergie. Puis la structure a été démocratisée par [Ackley, Geoffrey E. Hinton et T. J. Sejnowski, 1985]. La méthode d'apprentissage a été modifiée pour permettre l'optimisation du réseau de manière bien plus efficace. Afin de réduire la complexité d'un tel réseau, [Smolensky, 1986] introduit ce qu'il baptisera un *harmonium*, la machine de Boltzmann restreinte — RBM était créée. Elle sera remise au goût du jour avec les travaux de [Geoffrey E. Hinton, 2002] adressant le problème épineux de leur apprentissage, puis étendue à des structures profondes dans [Salakhutdinov et G. Hinton, 2009].

Avant de présenter leur structure, introduisons le concept de modèle générateur et discriminant avec les définitions respectives 2.2.3 et 2.2.4.

Définition 2.2.3 : Modèle générateur

Soit X et Y deux évènements observables d'un espace probabilisé $(\Omega, \mathcal{A}, \mathbb{P})$ avec Ω l'univers, $\mathcal{A}$ une tribu et $\mathbb{P}$ la mesure de probabilité définie sur $\mathcal{A}$. Un modèle générateur *est définit comme un modèle qui s'intéresse à la recherche de la probabilité conditionnelle de l'observation X en sachant l'attendu Y. De manière plus formelle, c'est un modèle tel que :*

$$\mathbb{P}(X \mid Y = y)$$

Avec y une réalisation de Y.

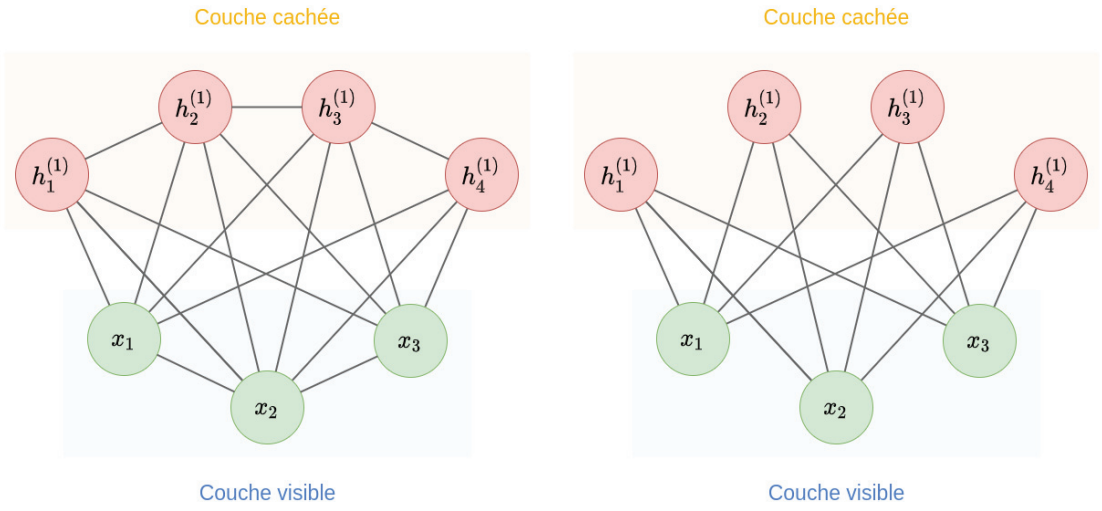
Définition 2.2.4 : Modèle discriminant

Soit X et Y deux évènements observables d'un espace probabilisé $(\Omega, \mathcal{A}, \mathbb{P})$ avec Ω l'univers, $\mathcal{A}$ une tribu et $\mathbb{P}$ la mesure de probabilité définie sur $\mathcal{A}$. On définit un modèle discriminant comme un modèle qui s'intéresse à la recherche de la probabilité conditionnelle de l'attendu Y en sachant l'observation X. De manière plus formelle, c'est un modèle tel que :

$$\mathbb{P}(Y \mid X = x)$$

Avec x une réalisation de X.

Une RBM est alors un réseau de neurones stochastique générateur qui apprend une distribution de probabilités à partir des informations contenues dans ses entrées. Elles font partie des algorithmes à apprentissage non supervisé mais, à la différence des AE vus à la sous-section 2.2.4.2, ce sont des modèles probabilistes. Dans leur plus simple structure, les BM et RBM sont des structures à deux couches, l'une visible et l'autre cachée. Chacune de ces couches est constituée de deux types d'états : des états observables et des états cachés. Le but de l'apprentissage est alors d'estimer les distributions de probabilités de passage d'un état à un autre. Autrement dit, une BM ou RBM est un réseau dont l'activation de chaque neurone est le résultat d'un processus stochastique. Une représentation de ces architectures est donnée à la figure 2.15.



(a) Machine de Boltzmann (b) Machine de Boltzmann restreinte

FIGURE 2.15 – Architecture à trois états visibles et quatre états cachés (une couche)

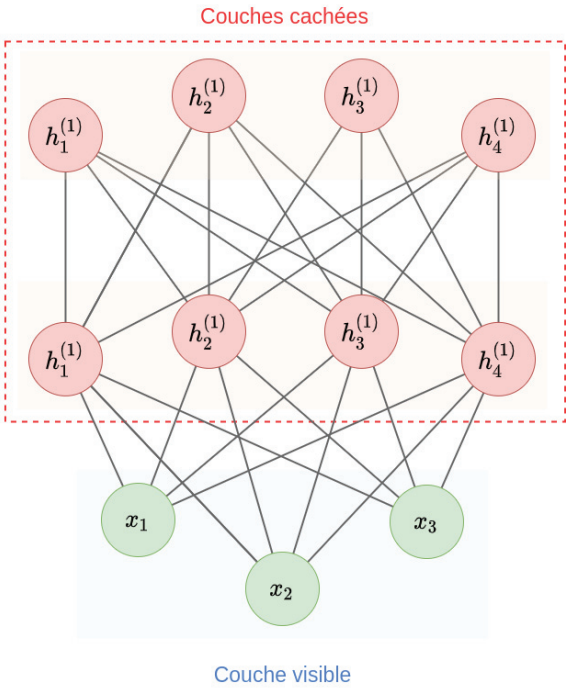


FIGURE 2.16 – Machine de Boltzmann restreinte à deux couches cachées

Contrairement aux infrastructures développées jusqu'ici, les machines de Boltzmann simples et restreintes possèdent des connexions entre neurones bi-directionnelles. La figure 2.15 représente deux architectures avec une seule couche cachée. Notons la différence entre la BM et la RBM qui

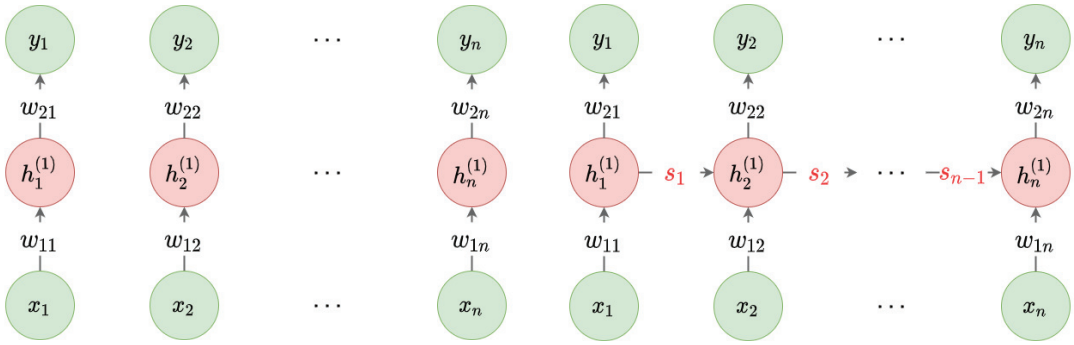
réside dans la manière dont les neurones sont connectés entre eux. En effet, le terme « restreinte » est utilisé car dans la RBM, chaque neurone d'une couche ne peut être connecté qu'aux neurones d'une autre couche, avec la limite que les neurones de la couche d'entrée possèdent uniquement des connexions avec la couche cachée leur étant directement adjacente. C'est pourquoi, à la figure 2.15b, le neurone x_1 d'entrée ne possède plus que quatre *synapses*. Les RBM sont alors considérées comme des modèles générateurs [Khan et Yairi, 2018]. En effet, une fois l'apprentissage terminé, elles sont capables de générer des données observables en fonction de leurs entrées. Elles restent cependant difficiles à entraîner. Concernant la problématique de PHM qui nous intéresse dans ces travaux, nous verrons qu'une RBM a été utilisée afin de prédire le temps de vie restant de roulements.

Les deux réseaux présentés ici à la figure 2.15 ne possédaient qu'une seule couche cachée. La multiplication de ces couches permet alors de créer une machine de Boltzmann profonde — Deep Boltzmann Machine (DBM). Comme le montre la figure 2.16, l'empilement de couches cachées permet de créer une telle architecture. La section (2.3.4) donne des exemples d'utilisation de tels algorithmes pour le diagnostic.

Remarque. *Il convient de bien dissocier les DBN et les DBM. Un réseau de neurones à conviction profonde ou DBN possède des connexions orientées dans toutes les couches hormis la plus profonde alors qu'un DBM conserve toutes ses connexions bi-directionnelles [Salakhutdinov et G. Hinton, 2009, figure 2]. Notons que, selon l'état de l'art de [Khan et Yairi, 2018], les DBN sont bien plus développés dans le domaine du diagnostic et du pronostic d'état de santé.*

2.2.4.4 Le réseau de neurones récurrents ou *Reccurrent Neural Network - RNN*

Les réseaux de neurones récurrents ou RNN ont, quant à eux, été développés pour l'étude des séries temporelles. De par leur nature, ils seraient le candidat idéal pour le diagnostic et le pronostic à partir des signaux vibratoires qui nous intéressent dans ces travaux. En effet, ils permettent de modéliser une mémoire à plus ou moins long terme, ce qui permet au réseau d'interpréter des données au regard d'un contexte. L'approche adoptée par l'architecture peut être illustrée par la figure 2.17.



(a) Réseau de neurone simple (b) Réseau de neurones récurrent — RNN

Figure 2.17 – Architecture de réseaux de neurones pour traitement de données séquentielles

La figure 2.17 permet de mettre en évidence la différence entre un réseau de neurones simple utilisé pour des données séquentielles tel que celui de la figure 2.17a et le RNN de la figure 2.17b pour lequel l'état propagé entre chaque couche est noté $s_i, i \in [\![1; n-1]\!]$. Ainsi, lorsque le réseau standard considère les éléments d'une série de manière indépendante, le RNN propage l'information d'un état antérieur aux états postérieurs du réseau. Ceci permet de définir la notion de *profondeur* du réseau, qui indique le nombre d'états consécutifs considérés dans la mémoire du réseau. L'état propagé est alors qualifié d'état *caché*. De manière plus détaillée, la structure d'un RNN est représentée par la figure 2.18. Notons qu'un bloc de la figure 2.18 correspond à une colonne, par exemple celle de $h_1^{(1)}$ de la sous-figure 2.17b.

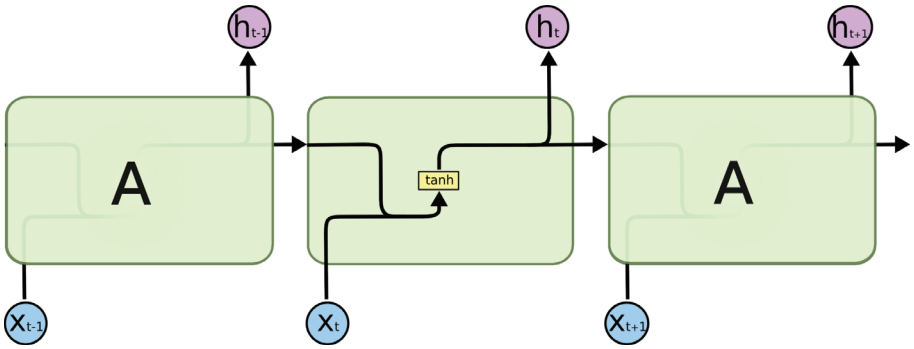
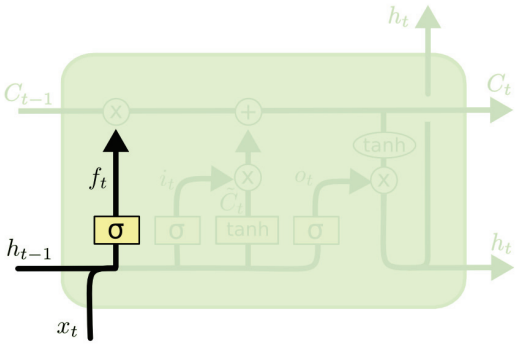
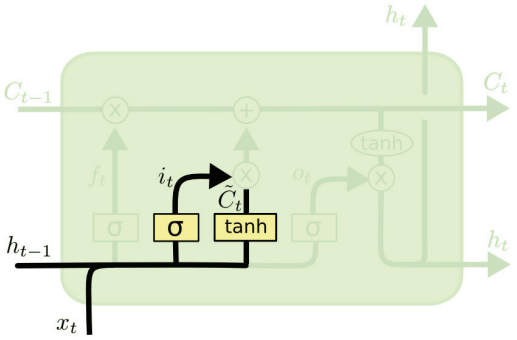


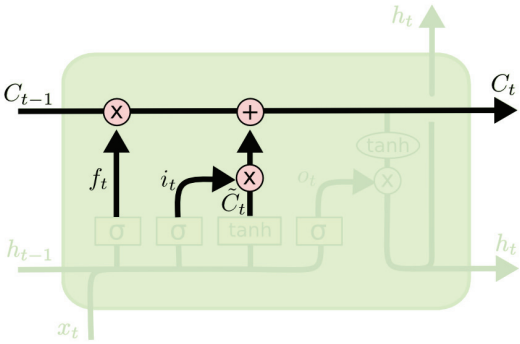
FIGURE 2.18 – Schéma de principe d'un RNN à une couche cachée, d'après [COLAH, 2015]



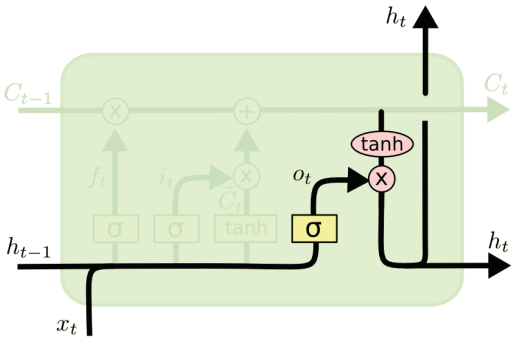
(a) Oubli du contexte en fonction de la donnée d'entrée



(b) Mise à jour de l'information en fonction de la mémoire



(c) État propagé sur la cellule



(d) Couche de sortie de la cellule

FIGURE 2.19 – Les étapes structurantes d'une cellule Long Short Term Memory (LSTM), d'après [COLAH, 2015]

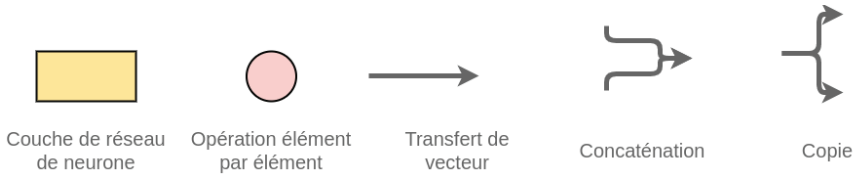


FIGURE 2.20 – Légende de la figure 2.19, d'après [COLAH, 2015]

Dans sa version la plus épurée, un RNN est constitué d'une couche d'entrée, d'une couche de sortie et d'une couche cachée avec une fonction d'activation tangente hyperbolique. Notons que cette couche cachée est reliée à la sortie du réseau précédent. On pourrait alors penser qu'au lieu de multiplier les couches, comme il paraît naturel de le faire avec des CNN, il convient cette

fois-ci de multiplier les réseaux. Cependant, une limite existe quant au chaînage de telles cellules élémentaires. En effet, plus la structure devient profonde plus l'apprentissage devient compliqué à cause de la rétro-popagation du gradient. Le problème a été étudié par [Bengio, Simard et Frasconi, 1994] et le simple algorithme de descente de gradient ne suffit pas pour l'apprentissage de cette architecture. Pour pallier ce problème et augmenter la profondeur des réseaux récurrents, les réseaux à mémoire à court et long terme ou LSTM ont été développés en 1997 par [Hochreiter et Schmidhuber, 1997]. L'idée était de rajouter un « état » qui se propage dans le réseau en plus de la sortie des étages précédents. Cet état modélise la mémoire du réseau. La figure 2.19 représente cette nouvelle architecture de manière détaillée.

Avec x_t l'entrée de la cellule LSTM soit l'entrée du réseau, h_{t-1} la sortie de la cellule à l'instant précédent, C_{t-1} l'état de la cellule à l'instant $t-1$, C_t l'état propagé de la cellule et h_t la sortie de la cellule propagée. De plus σ représente la fonction d'activation sigmoïde et tanh la tangente hyperbolique (voir la définition de la sigmoïde 1 et de la tangente hyperbolique 2 en annexe 5). Notons que les grandeurs (x, C, h), dans leur notation, sont seulement indexées par rapport au temps. Elles seront alors considérées comme des scalaires ou des matrices. De plus, chaque étage de la cellule possède des matrices de poids et un biais indépendants.

- À l'étape 2.19a, la fonction d'activation sigmoïde est appliquée (ou une fonction d'activation non linéaire), à la concaténation du vecteur d'entrée et de la sortie propagée de la cellule à l'instant $t-1$. Ainsi, soit $x_t, h_{t-1} \in \mathcal{M}_{(1,n)}(\mathbb{R})$, soit $[x_t, h_{t-1}]$ la concaténation de deux vecteurs, est alors obtenu $[x_t, h_{t-1}] \in \mathcal{M}_{(1,2n)}(\mathbb{R})$. La sigmoïde renvoyant un résultat compris entre 0 et 1,[3] on peut interpréter cette grandeur f_t comme la proportion de mémoire à garder de l'étape précédente.

- À l'étape 2.19b, la combinaison de la concaténation de l'entrée et de l'état propagé par $t-1$ est effectuée. Le résultat représente alors une matrice dont les valeurs sont comprises entre -1 et 1.

- À l'étape 2.19c, il s'agit d'actualiser l'état interne de la cellule en fonction des informations des deux étapes précédentes. f_t indiquant le pourcentage de mémoire à garder, il est multiplié à l'état propagé. À la grandeur résultante est ajoutée la mise à jour de l'information. L'état courant peut alors être transmis aux cellules suivantes.

- Enfin, l'étape 2.19d, permet de calculer la sortie de la cellule qui est alors une fusion entre l'état courant et la concaténation de l'entrée et de la sortie de la cellule précédente.

Ainsi, un LSTM permet d'apprendre une représentation des données d'entrée séquentielle capturant à la fois le contexte ainsi que la dépendance des données entre elles.

On peut alors modifier cette architecture en fonction des besoins de notre problème. Ces initiatives donnèrent naissance notamment aux portes récurrentes ou Gate Recurrent Unit (GRU) [Cho *et al.*, 2014].

2.3 L'approche connexionniste profonde utilisée dans le pronostic et la surveillance d'état de santé

2.3.1 État de l'art général

Pour faire un résumé, en l'état actuel du développement des techniques d'apprentissage profond utilisées dans la surveillance de l'état de santé de systèmes, [Khan et Yairi, 2018] présentent une revue détaillée des méthodes de la littérature. Ils introduisent alors une classification des méthodes utilisant « l'intelligence artificielle » [4] pour résoudre les problèmes inhérents au domaine :

[3] Attention dans notre cas, ce n'est pas un scalaire mais bien une matrice $\in \mathcal{M}_{(1,2n)}([0;1])$

[4] Le terme est ici utilisé pour désigner aussi bien les algorithmes de première génération que les derniers algorithmes de l'état de l'art.

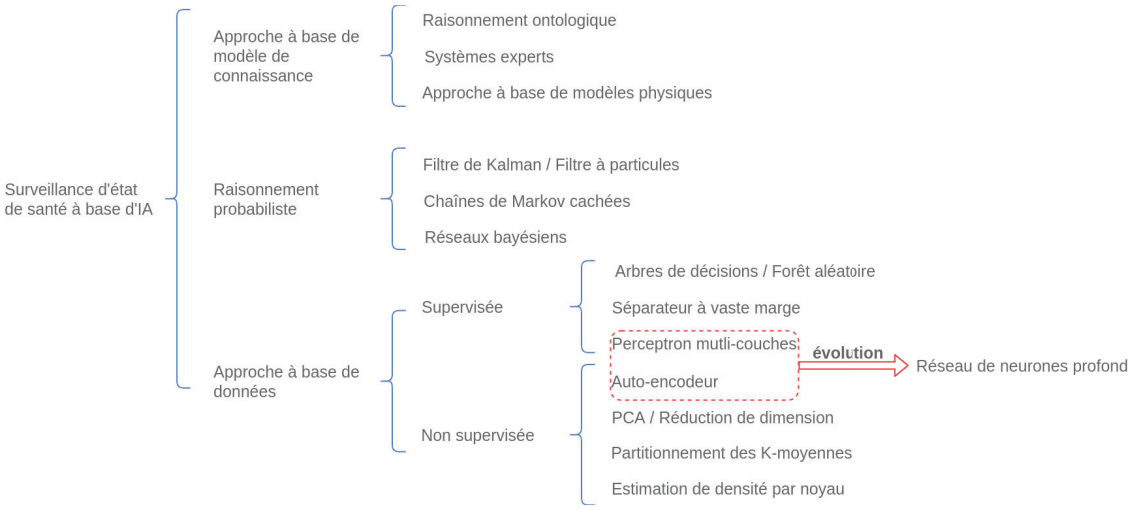


FIGURE 2.21 – Classification des méthodes utilisées dans la surveillance de l'état de santé des systèmes utilisant de l'intelligence artificielle, d'après [KHAN et YAIRI, 2018]

Technique	Strengths	Limitations	References
Supervised learning			
Decision trees	Decision using branches Easy to understand Non-parametric Good visualisation	Previous knowledge of systemMay over fit datalt can get stuck in local minimaHeuristic	Ash et al. [71], Rao et al. [72], Spina et al. [73], Bajwa and Kulkarni [74]
Random forest	Improved performance to decision treesFast and scalable Generally, trains faster than SVM	Increase in bias	Yan [75], Yang et al. [76]
Naïve Bayes	Simple Require less data	Does not allow for rich hypothesisAssumptions of attribute independence can be too constrainingFeatures need to be correlated	Muralidharan and Sugumaran [77], Elangovan et al. [78], Zhang et al. [79], Kumar et al. [80], Ng et al. [81]
[a]Bayesian networks	Probabilistic models Reduces no of parameters to learn Easy to visualise dependency links	Previous knowledge of system Learning unknown structure is complex	Hu and Hao [82], Weber et al. [83], Zaidan et al. [84]
Support vector machines	High accuracy Robust against noise Efficient for large datasets	Requires training data labels Previous knowledge of system Results can be incomprehensible Fundamentally a binary classifier Memory intensive No standard for choosing the kernel function	Fuertes et al. [85], Salem et al. [86], Li et al. [87]
[b]Neural networks (multilayer perceptron)	Small number of parameters needs to be optimised for training Adaptive system	May require greater computational resources Prone to overfitting No standard to determine network structure	Jakubek and Strasser [88], Kobayashi and Simon [89], Marsland [90], McDuff et al., [91], Zhang and Ganesan [92]
Unsupervised learning			
Clustering (k-means)	Short training times	Prior knowledge is needed i.e. the number of cluster, K, must be determined before hand	Soualhi et al. [93]
Adaptive resonance theory	Short training times Ability to model non-linear shaped clusters Robust against outliners	Less suitable for function fitting May require suitable data pre-processing scheme	Lei et al. [94], He et al. [95]
Self-organising maps	Data mapping is easily interpreted Capable of organising large complex data sets	Difficult to determine what input weights to use, mapping can result in divided clusters Requires that nearby points behave similarly	Prabakaran et al. [96], Lacaille et al. [97], Tibaduiza et al. [98], Cho et al. [99]
Hidden Markov models	Statistical models Scalable	Previous knowledge of system, can become complex Large amount of data needed for developing an accurate model	Yu [100], Zhou et al. [101]

FIGURE 2.22 – Algorithmes pour effectuer du diagnostic à partir de données, d'après [KHAN et YAIRI, 2018]

D'après [KHAN et YAIRI, 2018], la majorité des méthodes d'intelligence artificielle ont convergé vers le domaine du connexionnisme avec les réseaux de neurones et leurs variantes, au détriment du symbolisme qui est constitué de raisonnements experts. Notons que [KHAN et YAIRI, 2018] développent aussi une section taxonomique dans laquelle ils définissent les termes utilisés dans

le domaine. Ils présentent notamment une définition de l'intelligence artificielle : "L'intelligence artificielle tend à permettre aux machines de réaliser des tâches à la manière d'un expert. Une machine intelligente percevra son environnement et pourra prendre des initiatives pour maximiser son propre fonctionnement. Les problèmes centraux d'aujourd'hui incluent des phénomènes de déduction, de raisonnement, de résolution de problèmes, de représentation de la connaissance et enfin d'apprentissage". Ces travaux permettent de créer une image des algorithmes principalement utilisés par la littérature. Ce tableau est représenté à la figure 2.22.

Plus récemment, les travaux de [Chalapathy et Chawla, 2019] se sont intéressés à réaliser un état de l'art sur les méthodes d'apprentissage profond utilisées pour la détection d'anomalies. Comme il sera vu par la suite, les algorithmes les plus utilisés sont des algorithmes à base de CNN, d'AE et de DBN. Autrement dit, des méthodes de filtrage, des méthodes de compression de l'information et des méthodes s'apparentant à des pseudo chaînes de Markov.

D'après [R. Zhao *et al.*, 2019], la tendance de l'industrie est aujourd'hui au développement de systèmes de surveillance d'état de santé à base de données. Ils pallient les principaux défauts de l'approche à base de modèles, à savoir la difficulté de modélisation et leur manque de robustesse vis-à-vis de la variabilité de leur environnement. Les techniques d'apprentissage artificiel sont alors utilisées pour faire du diagnostic et du pronostic à partir des données captées lorsque le système à surveiller est en opération. L'apprentissage artificiel profond, lui, est devenu une orientation viable grâce à l'avènement des GPU et à la kyrielle de données de plus en plus disponibles. D'après les tests de [Raina, Madhavan et Ng, 2009], sur un DBN à 100 millions de paramètres, un GPU permet de réduire une semaine de calculs à une journée. Ainsi pour l'entraînement de modèles d'apprentissage artificiel, aujourd'hui le goulot d'étranglement calculatoire devient le processeur graphique lorsque le processeur central est bien dimensionné. De par la nature des calculs d'apprentissage artificiel, il y a donc un réel intérêt économique et scientifique à investir dans un matériel performant.

Quel est alors l'intérêt de l'apprentissage profond pour les méthodes de surveillance d'état de santé ? Aujourd'hui la majeure partie des indicateurs de défauts, de grandeurs qui permettent de représenter l'information brute en une information corrélée au défaut à détecter, est principalement créée par des experts [R. Zhao *et al.*, 2019]. Les différentes étapes du processus de PHM (voir figure 1.4) sont effectuées de manière presque indépendantes. D'un point de vue *projet*, cette segmentation des étapes permet d'en optimiser le développement. En effet, des briques technologiques indépendantes les unes des autres peuvent alors être développées. Par conséquent, le retard ou la non réalisation d'une tâche n'impactera que très peu le développement du bloc suivant. Seulement, d'après [R. Zhao *et al.*, 2019], le problème pourrait être optimisé en considérant la réalisation de ces étapes de manière parallèle. L'approche utilisant de l'apprentissage profond permet de réaliser une telle tâche. Ainsi, un modèle développé pour le diagnostic pourrait être utilisé pour du pronostic en modifiant la (ou les) dernière(s) couche(s) du réseau. Par exemple, dans le cas de [Pan et Q. Yang, 2010], la dernière couche du réseau *softmax* qui réalise le diagnostic est remplacée par une couche de régression linéaire afin d'effectuer une tâche de pronostic. Il convient alors d'entraîner à nouveau le réseau sans faire table rase des paramètres de sa précédente fonction. Cette méthode est appelée *l'apprentissage par transfert*. La figure 2.23 rassemble les différentes approches qui coexistent dans le domaine du PHM.

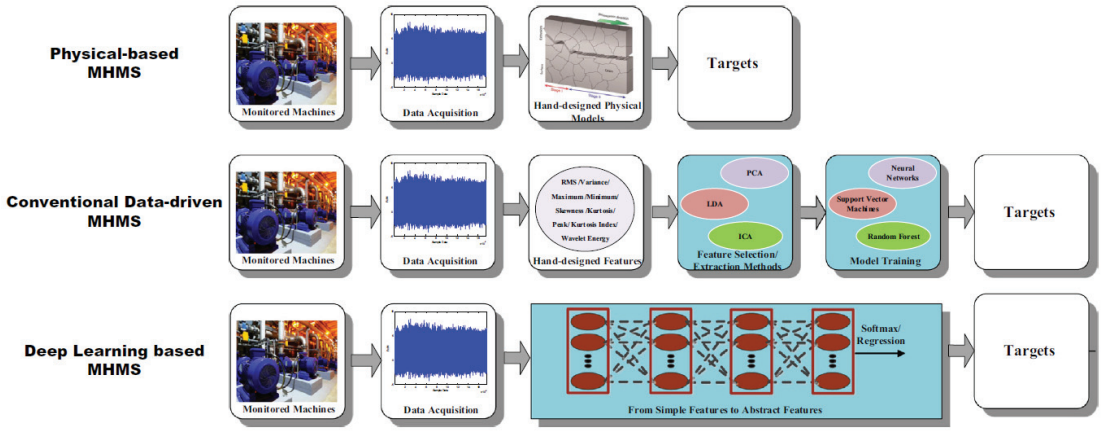


Figure 2.23 – Les différentes approches de HM, d'après [R. Zhao *et al.*, 2019]

La première approche de la figure 2.23 représente l'approche classique à base de modèles physiques de surveillance d'état de santé des machines ou Machine Health Monitoring System (MHMS). Les descripteurs de défauts sont créés par savoir expert et leur variation est surveillée tout au long de la durée de vie du système. Dès qu'un seuil défini est dépassé, le système est considéré comme étant en défaut. La deuxième approche, à base d'apprentissage machine conventionnel (au sens *peu profond*) utilise toujours des descripteurs obtenus par savoir expert, seulement, de ces derniers sont extraits une autre représentation de l'information. De plus, cette étape permet la sélection automatique de descripteurs pertinents tels que des descripteurs monotones et corrélés au vieillissement. Par la suite, un modèle d'apprentissage statistique apprend sa propre représentation de l'information pour rendre une conclusion quant à l'état du système. Enfin, la dernière approche utilisant de l'apprentissage profond permet de rassembler ces trois dernières étapes en un réseau de neurones complet qui se chargera d'extraire une représentation de l'information pertinente au fur et à mesure de la propagation de l'information dans ses couches. Ce sera un mélange de la deuxième et de la troisième approche qui sera utilisé dans ces travaux de thèse. Aujourd'hui principalement utilisée dans le domaine de la vision par ordinateur ou la reconnaissance vocale, la communauté PHM s'intéresse au connexionnisme, et notamment à l'approche de l'apprentissage profond, et des travaux de recherche émergent en ce sens. Les architectures principalement utilisées dans la littérature sont les suivantes :

- Les réseaux de neurones à convolutions ou CNN (développés à la sous-section 2.2.4.1).
- Les auto-encodeurs ou AE (développés à la sous-section 2.2.4.2).
- Les réseaux à convictions profondes ou DBN (introduits à la sous-section 2.2.4.3).
- Les machines de Boltzmann profondes ou DBM (développés à la sous-section 2.2.4.3).
- Les réseaux de neurones récurrents ou RNN (développés à la sous-section 2.2.4.4).

Afin d'illustrer le gain de performances acquis entre les algorithmes d'apprentissage machine standards et les algorithmes d'apprentissage profond, [R. Zhao *et al.*, 2019] réalisent une étude comparative entre les différents algorithmes suivants [5] :

- Régression par séparateur à vaste marge ou SVR.
- Régression par séparateur à vaste marge avec noyaux fonction à base radiale.
- Forêts aléatoires ou RF.
- Réseau de neurones à deux couches cachées.
- Auto-encodeur ou AE.
- Auto-encodeur avec « débruitement » ou Denoising Auto-Encoder (DAE).
- Réseau à conviction.
- Réseau de neurones à convolutions ou CNN.
- Réseau à mémoire à court et long terme ou LSTM.
- Réseau à mémoire à court et long terme bidirectionnel.

[R. Zhao *et al.*, 2019] calculent les indicateurs de défauts suivants à partir de chaque donnée capteur, à savoir une force dans les trois dimensions, une mesure de vibration dans les trois dimensions et la valeur efficace d'un capteur de tension :

[5] Notons que les codes Python des programmes implémentés sont accessibles à l'adresse suivante : https://github.com/ClockworkBunny/MHMS_DEEPLEARNING. En ligne à l'écriture de ce rapport le 23/11/2021.

Domain	Features	Expression
Statistical	RMS	$z_{rms} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} z_i^2}$
	Variance	$z_{var} = \frac{1}{n}\sum_{i=1}^{n}(z_i - \bar{z})^2$
	Maximum	$z_{max} = max(z)$
	Skewness	$z_{skew} = E[(\frac{z-\mu}{\sigma})^3]$
	Kurtosis	$z_{kurt} = E[(\frac{z-\mu}{\sigma})^4]$
	Peak-to-Peak	$z_{p-p} = max(z) - min(z)$
Frequency	Spectral Skewness	$f_{skew} = \sum_{i=1} k(\frac{f_i - \bar{f}}{\sigma})^3 S(f_i)$
	Spectral Kurtosis	$f_{kurt} = \sum_{i=1} k(\frac{f_i - \bar{f}}{\sigma})^4 S(f_i)$
Time–Frequency	Wavelet Energy	$E_{WT} = \sum_{i=1}^{N} wt_{\phi}^2(i)/N$

FIGURE 2.24 – Indicateurs de défauts calculés, d'après [R. ZHAO *et al.*, 2019]

Notons que [R. ZHAO *et al.*, 2019] calculent un vecteur signature de dimension 70.

Dans le domaine de l'apprentissage par ensemble, [MA et CHU, 2019] développent, dans le cas du diagnostic de roulement des moteurs, un modèle rassemblant un réseau de neurones résiduels à convolutions, un auto-encodeur ou AE et une machine à conviction profonde DBM. L'apprentissage de l'ensemble est ensuite réalisé par une méthode d'optimisation multi-objectifs évolutive.

2.3.2 Algorithmes à base de réseaux de neurones à convolutions

Toujours à l'aide des signaux bruts, [JANSSENS *et al.*, 2016] utilise un CNN pour effectuer du diagnostic sur les défauts des roulements de paliers d'une machine électrique. Ils résument alors la procédure de diagnostic par le schéma de la figure 2.25, à mettre en parallèle de la figure 2.23, car représentant aussi les différentes approches utilisées pour le PHM, en laissant une part toujours plus importante à l'apprentissage machine. Cependant, cette procédure se concentre uniquement sur les méthodes d'extraction de signatures (ou descripteurs) pour la surveillance d'état de santé.

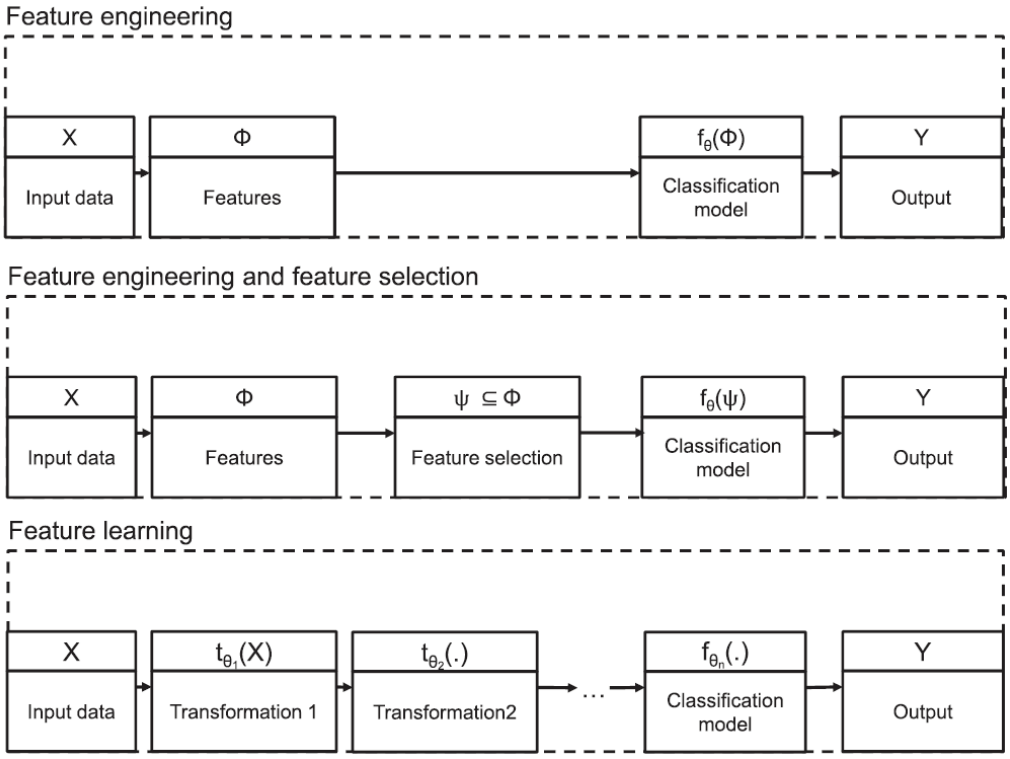


FIGURE 2.25 – Représentation schématique de l'extraction de signatures, d'après [JANSSENS *et al.*, 2016]

La figure 2.25 permet de reconnaître une première approche qui consiste à extraire des signatures à partir de données brutes. Ces dernières sont extraites par des étapes de traitement du signal et nécessitent une expertise certaine. Cette approche a été utilisée jusqu'à maintenant dans les travaux de [Boileau, 2010] et [Leboeuf, 2012]. La deuxième approche consiste à rajouter entre l'étape d'extraction et l'étape de classification une étape de sélection automatique. Ainsi, seules les signatures les plus pertinentes sont utilisées dans l'algorithme de diagnostic. Cette approche a été utilisée dans les travaux de [Breuneval, 2017]. Enfin, dans la dernière approche, l'algorithme d'apprentissage profond extrait lui-même les indicateurs de défauts qu'il juge pertinent pour la classification. Cette approche, non encore implémentée dans nos travaux, requiert un grand nombre de données. Il conviendra cependant de la tester et de caractériser l'ensemble des données nécessaires au succès d'une telle procédure. Voulant comparer ces différentes approches, [Janssens *et al.*, 2016] s'intéressent alors à détecter les différents états de santé d'un roulement de palier d'une machine électrique. Il définit alors les conditions suivantes :

- Roulement sain.
- Lubrification du roulement légèrement insuffisante.
- Lubrification du roulement largement insuffisante.
- Défaut sur bague extérieure.
- Roulement sain avec rotor déséquilibré.
- Lubrification du roulement légèrement insuffisante avec rotor déséquilibré.
- Lubrification du roulement largement insuffisante avec rotor déséquilibré.
- Défaut sur bague extérieure avec rotor déséquilibré.

À partir des signaux vibratoires échantillonnés à 20 kHz, ils extraient donc des indicateurs de défauts statistiques, tels que le kurtosis ou encore le facteur de crête. Ces indicateurs représentent les données d'entrée d'un classifieur par forêt aléatoire ou d'une SVM. Notons que les données brutes sont constituées des amplitudes du spectre des signaux vibratoires qui sont alors fournies à un CNN. Les deux matrices de confusion obtenues par [Janssens *et al.*, 2016] sont présentées à la figure 2.26.

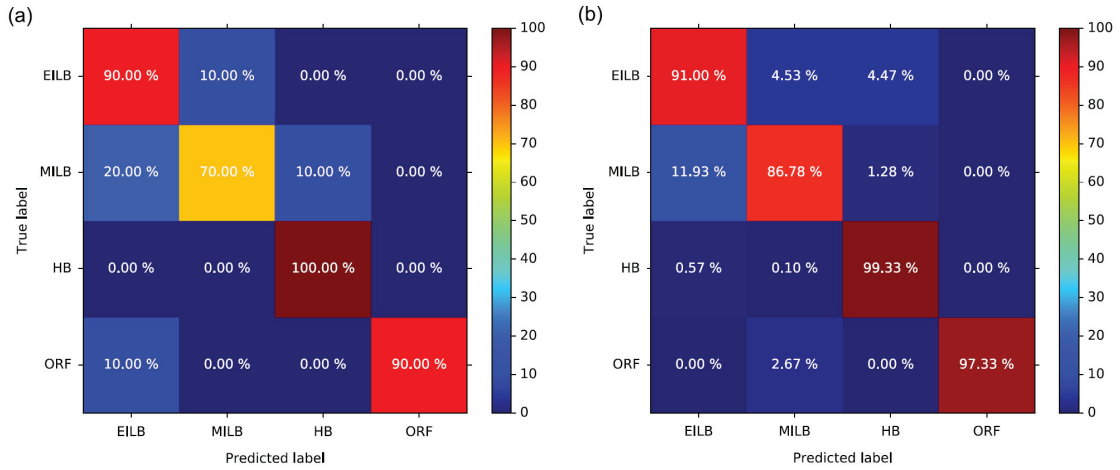


Figure 2.26 – Matrices de confusion pour la classification de défauts de roulements par extraction de signature manuelle (matrice de gauche) et automatique (matrice de droite), d'après [Janssens *et al.*, 2016]

Ainsi, d'après la figure 2.25, l'approche par réseau de neurones profond, sans extraction au préalable de signatures, possède la matrice de confusion témoin d'une meilleure performance en classification. Dans un problème de classification, la matrice de confusion permet de représenter les quantités de faux positifs, faux négatifs, vrais positifs et vrais négatifs. Le cas idéal serait donc une matrice diagonale où la diagonale serait unitaire (si elle est normalisée comme ici) et les

autres termes seraient nuls. Ainsi par rapport à une approche classique, le CNN permet d'obtenir une précision de classification de 6 % supérieure sans avoir à développer l'expertise nécessaire à l'extraction de signatures pertinentes dans le domaine considéré. Au-delà de l'usage de l'apprentissage profond afin d'améliorer les performances des méthodes de PHM, [Janssens *et al.*, 2016] concluent en ouvrant sur la perspective de fiabiliser les algorithmes de diagnostic avec la fusion de capteurs. Par exemple, en plus de signaux électriques ou vibratoires, des images thermiques ou des relevés de température pourraient servir à la détection de défauts de lubrification. En effet, l'augmentation des frottements résultant du manque de lubrifiant à l'intérieur du système génère alors une élévation de température locale, un point chaud qui pourrait être détectable.

Outre le domaine des machines tournantes et des roulements, [Abdeljaber *et al.*, 2017] s'intéressent au diagnostic des structures métalliques par analyse vibratoire. Pour ce faire, [Abdeljaber *et al.*, 2017] ont développé un CNN unidimensionnel pour l'analyse de séries temporelles. Notons d'ailleurs que dans ces deux approches, les capacités de diagnostic en temps réel de l'algorithme sont intéressantes pour notre problématique. En effet, la convolution étant une série d'opérations de type *multiplication-accumulation* [6], opérations de base d'un processeur, ce type d'algorithme se prête bien à un usage en temps réel sur cible. L'originalité de la solution de [Abdeljaber *et al.*, 2017] vient du fait que l'algorithme de diagnostic est décentralisé. Aussi, à chaque accéléromètre il y a un CNN pour diagnostiquer la structure localement.

[Han *et al.*, 2019] développent un Spatio-Temporal Convolutional Neural Network (ST-CNN) pour effectuer du diagnostic sur des données multi-variées d'une éolienne étudiée en soufflerie. Leur approche est intéressante car elle rappelle les intérêts de la cyclostationnarité pour augmenter les performances de l'algorithme de détection.

2.3.3 Algorithmes à base d'auto-encodeurs

La même année que [Janssens *et al.*, 2016], mais cette fois-ci pour la détection de défauts dans les moteurs à induction, [Sun *et al.*, 2016] utilisent des AE parcimonieux ou Sparse Auto-Encoder (SAE) et un réseau de neurones profonds. L'algorithme consiste à extraire des données non labellisées, des indicateurs de défauts à la manière d'un apprentissage non supervisé. Pour rendre plus robuste l'extraction de descripteurs, du bruit est rajouté dans les données d'entrée du SAE. Ces descripteurs sont alors fournis à un réseau de neurones qui effectuera l'étape de classification pour obtenir le résultat de diagnostic. À noter que [Sun *et al.*, 2016] utilisent la méthode de *dropout* pour l'apprentissage. Pour éviter le sur-apprentissage de l'algorithme, cette méthode, utilisée en apprentissage profond, consiste à supprimer des liaisons de manière aléatoire dans le réseau. Ainsi seulement des parties spécifiques de ce dernier seront entraînées, les autres permettront de généraliser les résultats d'apprentissage. Les données brutes proviennent d'un banc d'essais spécifique construit dans le but de précipiter des défauts de barre cassée, d'excentricité du rotor, de roulements défaillants, de court-circuits inter-spires et de balourd au rotor. Les travaux de [Sun *et al.*, 2016] permettent alors de créer une base de données constituée des signaux vibratoires du moteur échantillonnés à 20 kHz. Après avoir séparé cet ensemble de données en une base d'apprentissage et une base de test, les valeurs sont normalisées entre -1 et 1. Sachant que l'apprentissage profond permet d'extraire de manière automatique des indicateurs de défauts, [Sun *et al.*, 2016] comparent alors les performances de son algorithme à des réseaux de neurones profonds standards, pour justifier l'usage d'une architecture spécifique à l'extraction de signatures. Les taux de précision de détection sont reportés au tableau 2.1.

Classifieur	Moyenne de la précision de détection en %
completly connected Neural Network (NN)(100)	80.04 %
NN(600)	74.33 %
NN(600; 100)	96.39 %
SAE-SVM	96.42 %
SAE-Linear Regression (LR)	92.75 %
SAE-Deep Neural Network (DNN)	**97.61** %

Table 2.1 – Comparaison des précisions de classification, d'après [Sun *et al.*, 2016]

[6] Plus connues sous l'acronyme MULAC pour le dimensionnement d'unités de calculs.

Plusieurs notations sont introduites à la table 2.1. Ainsi, un réseau de neurones à une couche cachée composée de x neurones est noté NN(x). Par extension, NN($x; y$) représente un réseau de neurones à deux couches cachées, la première possédant x neurone et la seconde y neurones. Les trois infrastructures simples de la table 2.1 sont comparées à trois algorithmes de classification qui prennent en entrée les indicateurs extraits par un SAE : un séparateur à vaste marge, une régression linéaire et un réseau de neurones profonds. L'approche développée dans les travaux de [Sun *et al.*, 2016] est alors la plus performante. En plus de la mesure de précision de détection, [Sun *et al.*, 2016] ont effectué une étude de stabilité de son architecture. L'étude de stabilité consiste à corrompre les données d'entrée du réseau et, ce faisant, à tester les limites de l'algorithme. Ainsi pour une corruption des données comprise entre 0 % et 40 %, le Sparse Auto-Encoder with Deep Neural Network (SAE-DNN) réussit toujours à classifier les états de santé du moteur avec une précision supérieure à 90 % alors que l'on ne retrouve pas ce comportement dans l'approche par réseaux de neurones profonds standards.

Afin de diagnostiquer des paliers de roulement, [H. Shao, H. Jiang, Y. Lin *et al.*, 2018] propose une nouvelle méthode utilisant une combinaison d'AE profonds ou Ensemble Deep Auto-Encoder (EDAE). Un DAE peut être représenté comme un empilement d'AE ce qui est schématisé par la figure 2.27.

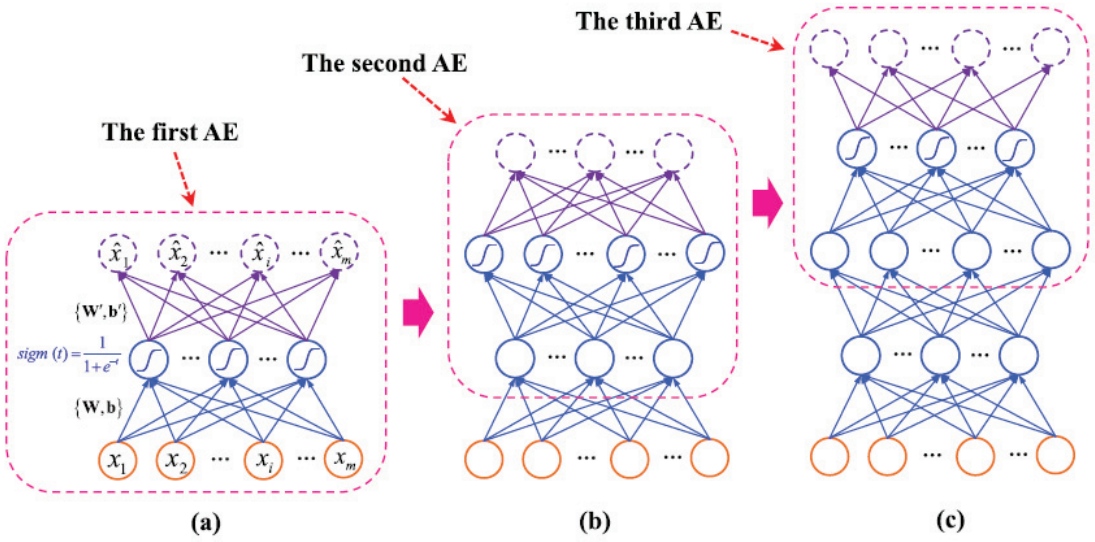


Figure 2.27 – Schéma de principe pour expliquer l'architecture d'un auto-encodeur profond. (a) Auto-encodeur classique, (b) auto-encodeur profond à une couche cachée, (c) auto-encodeur profond à trois couches cachées, d'après [H. Shao, H. Jiang, Y. Lin *et al.*, 2018]

Dans le premier modèle (a) de la figure 2.27, la partie réalisant l'encodage de l'information est représentée en bleu tandis que la partie décodage de l'information est représentée en violet. Le principe de l'approche est d'entraîner chaque couche d'encodeur séparément, à la manière d'un AE classique. L'encodeur du réseau précédemment entraîné devient par la suite l'entrée d'un nouvel AE complet (plus précisément, l'espace latent du premier AE) et ainsi de suite jusqu'à atteindre la profondeur souhaitée. L'apprentissage d'une telle structure est enfin réalisé de manière non supervisée. De fait, l'approche de [H. Shao, H. Jiang, Y. Lin *et al.*, 2018] consiste à extraire différents niveaux d'abstraction des données brutes afin de représenter plusieurs plans de descripteurs de défauts optimaux. En reformulant la procédure, le premier encodeur extrait la signature de bas niveau d'abstraction qui devient l'entrée du deuxième encodeur. Ce dernier extrait une signature d'un degré d'abstraction supérieur, signature qui devient l'entrée de l'encodeur final. L'encodeur final qui extrait finalement une signature de niveau d'abstraction maximal. Le DAE est obtenu de cette manière. C'est enfin le dernier descripteur (ou signature) qui est fourni à une couche *softmax* réalisant une étape de classification.

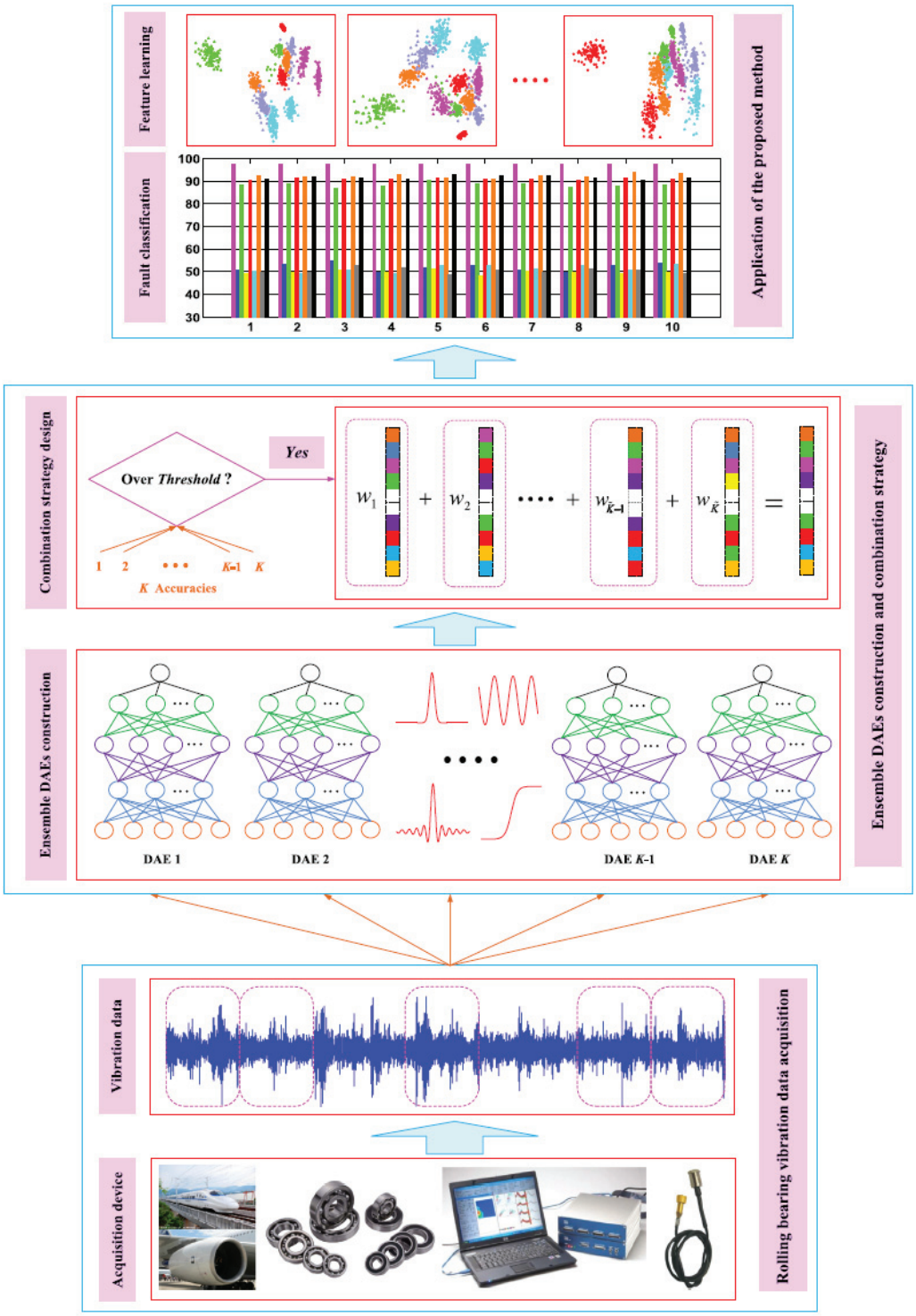


FIGURE 2.28 – Méthode de diagnostic par EDAE, d'après [H. SHAO, H. JIANG, Y. LIN *et al.*, 2018]

Afin d'améliorer les résultats de diagnostic, [H. SHAO, H. JIANG, Y. LIN *et al.*, 2018] proposent une méthode dite « ensembliste » car utilisant l'information conjointe apportée par une mutlitude de réseaux pour obtenir un résultat. Il créent alors une série de 15 DAE possédant chacun des fonctions d'activations différentes. Les DAE ainsi créés génèrent chacun un vecteur représentant un descripteur de défaut. Par la suite, l'information individuelle apportée par chacun de ces descripteurs est agrégée pour créer un vecteur de signature généralisé. Ce dernier résultat sera à ce moment là fourni en entrée d'une couche de classification. De manière plus détaillée, [H. SHAO, H. JIANG, Y. LIN *et al.*, 2018] définie plusieurs étapes pour l'algorithme de diagnostic :

1. Les signaux vibratoires sont acquis par une chaîne de capteurs.
2. Sans pré-traitement, les signaux sont divisés en une base d'apprentissage et une base de test.
3. Les signaux sont fournis à la série de DAE indexés sur l'ensemble $[\![1; K]\!]$.
4. À cette étape, il s'agit d'élaborer la structure ensembliste de l'algorithme. Chaque DAE produit un descripteur $w_{i,1\leq i\leq K}$. Si la précision du DAE est strictement supérieure à un seuil alors on peut considérer que l'apprentissage est de bonne qualité pour cette structure, et de fait, la valeur du descripteur obtenue est conservée. Dès que toutes les signatures de défauts pertinentes sont obtenues, elles sont sommées les unes avec les autres pour obtenir un vecteur représentant l'information globale d'un défaut.
5. L'étape de diagnostic est effectuée à partir du descripteur global précédent.
6. Afin de s'assurer des résultats, les données de la base de test sont projetées dans l'ensemble des signatures obtenues.

Cette approche est schématisée par la figure 2.28 extraite des travaux de [H. Shao, H. Jiang, Y. Lin *et al.*, 2018].

Cette fois-ci, [H. Shao, H. Jiang, Y. Lin *et al.*, 2018] cherchent à détecter 12 régimes différents de fonctionnement de roulement à billes :

- Fonctionnement normal.
- 3 degrés de défauts [7] sur les billes.
- 3 degrés de défauts sur la bague intérieure.
- 5 degrés de défauts sur la bague extérieure.

Pour chacun des domaines de fonctionnement présentés, 300 échantillons sont utilisés. Un total de 3 600 échantillons sont donc utilisés pour l'apprentissage de la méthode par Ensemble Deep Auto-Encoder (EDEA). Chaque échantillon possède alors entre 150 et 600 points, selon l'expérience menée. En plus de la simple comparaison de performances entre différents algorithmes, [H. Shao, H. Jiang, Y. Lin *et al.*, 2018] étudient ici la robustesse du diagnostic en fonction de la représentativité des défauts de la base de données.

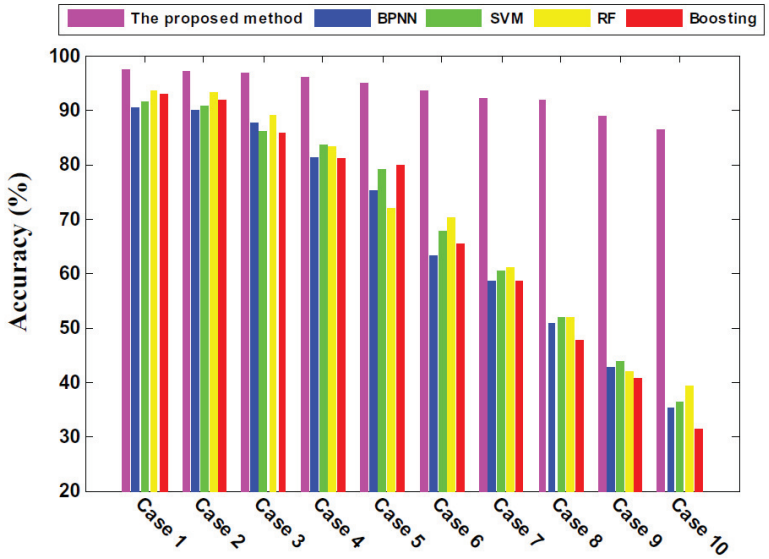


FIGURE 2.29 – Analyse de la sensibilité des algorithmes de diagnostic à la représentativité du défaut, d'après [H. Shao, H. Jiang, Y. Lin *et al.*, 2018]

[7] Le diamètre du défaut est mesuré en pouces. Les mesures sont de l'ordre du millième de pouce, soit dans le système métrique de l'ordre de 10 µm.

Dans cette expérience, pour étudier la sensibilité de l'algorithme de diagnostic à la représentativité du défaut, [H. Shao, H. Jiang, Y. Lin *et al.*, 2018] modifie le nombre de points de chaque échantillon ([150 ; 200 ; 250 ; 300 ; 350 ; 400 ; 450 ; 500 ; 550 ; 600] points). Avec la figure 2.29, on remarque que contrairement aux algorithmes standards d'apprentissage statistiques que sont les réseaux de neurones complètement connectés, la SVM, la RF et le *Gradient Boosting* qui présentent une décroissance linéaire en fonction du nombre de points, l'EDEA se stabilise entre 150 et 400 points puis décroît légèrement. La précision de détection reste voisine de 90 % contrairement aux autres algorithmes qui finissent à 35 % de précision. Cette performance ouvre la voie à deux domaines qui aujourd'hui sont à explorer, à savoir : l'apprentissage par regroupement de structure (une sorte d'apprentissage distribué, ce que la littérature qualifie d'*ensemble learning*) et l'analyse de la stabilité des algorithmes d'apprentissage machine développés.

2.3.4 Algorithmes à base de réseaux à convictions profondes

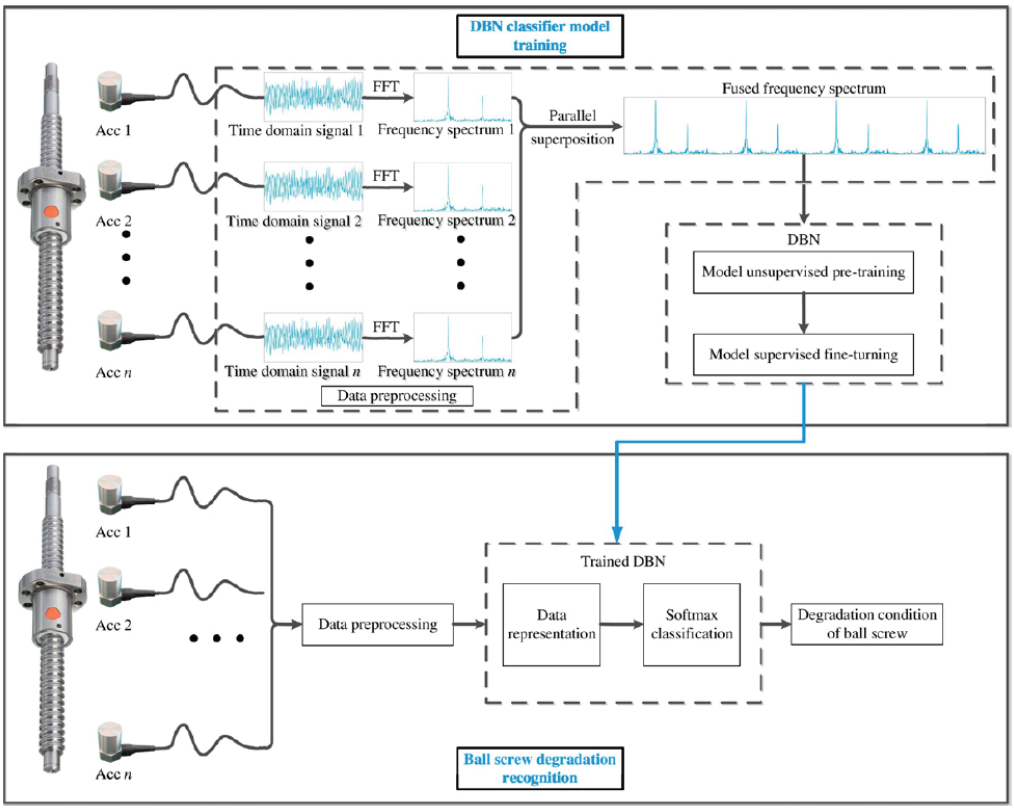


Figure 2.30 – Procédure de diagnostic d'une vis à billes, d'après [Li Zhang *et al.*, 2017]

Pour revenir à notre problématique, [Li Zhang *et al.*, 2017] utilisent de la fusion de données issues d'accéléromètres, donc toujours en utilisant des signaux vibratoires, avec un réseau à conviction profonde — DBN. Un tel réseau permet de modéliser la loi de probabilité jointe entre les données mesurées et les labels. D'une manière plus formelle, en notant ν la couche d'entrée et h_i la i^{e} couche cachée du réseau, pour un réseau à N couches la probabilité jointe peut être modélisée par l'équation 2.14.

$$\mathbb{P}(\nu, h_1, ..., h_{N-1}) = \mathbb{P}(\nu \mid h_1)\,\mathbb{P}(h_1 \mid h_2)\,...\mathbb{P}(h_{N-2} \mid h_{N-1}) \tag{2.14}$$

La fonction de l'équation 2.14 peut être décomposée en plusieurs distributions plus élémentaires. Ce sont les distributions données par les RBM dont les DBN sont un empilement, avec des connexions directionnelles entre couches. Les RBM sont alors entraînées de manière unitaire, puis elles sont assemblées pour former un réseau profond. De plus, dans le but d'améliorer la fiabilité de l'information rendue par les capteurs, [Li Zhang *et al.*, 2017] utilisent un algorithme de fusion de données. La fusion de données consiste à récupérer les données de n capteurs (pas nécessairement identiques) placés à des endroits différents du banc d'essais pour créer une nouvelle information

plus pertinente qui permettrait de supplanter les limites d'un seul capteur. Dans la pratique, l'information rendue par une fusion de capteurs est, dans les faits, plus fiable et plus robuste que l'information rendue par un unique capteur. Le spectre fréquentiel de chacune des séries (une série correspond à un capteur) est calculé puis les spectres sont concaténés pour former un signal. Ce dernier est normalisé puis utilisé pour l'apprentissage du réseau profond. L'approche est résumée par le schéma de la figure 2.30 extraite des travaux de [Li Zhang *et al.*, 2017].

Les résultats de classification sont ensuite comparés aux performances d'un réseau de neurones profond complètement connecté. La méthode développée par [Li Zhang *et al.*, 2017] est 10 % plus précise sur la base de données de test que l'approche standard, pour un temps d'apprentissage divisé par 2. La base de données créée représente alors 7 degrés de dégradation différents pour la vis à billes. Dans cette méthode, la fusion de données permet d'augmenter de 18 % la précision de leur DBN.

Concernant toujours les signaux vibratoires, [H. Shao, H. Jiang, H. Zhang *et al.*, 2018] couple alors une machine à conviction profonde ou DBN avec un CNN. Remarquons que ces deux dernières architectures sont beaucoup utilisées dans l'état de l'art depuis ces trois dernières années. [H. Shao, H. Jiang, H. Zhang *et al.*, 2018] a alors développé une machine à conviction profonde à convolutions améliorée par acquisition comprimée ou Convolutional Deep Belief Network with Compressed Sensing (CDBN-CS). L'approche adoptée est résumée par la figure 2.31.

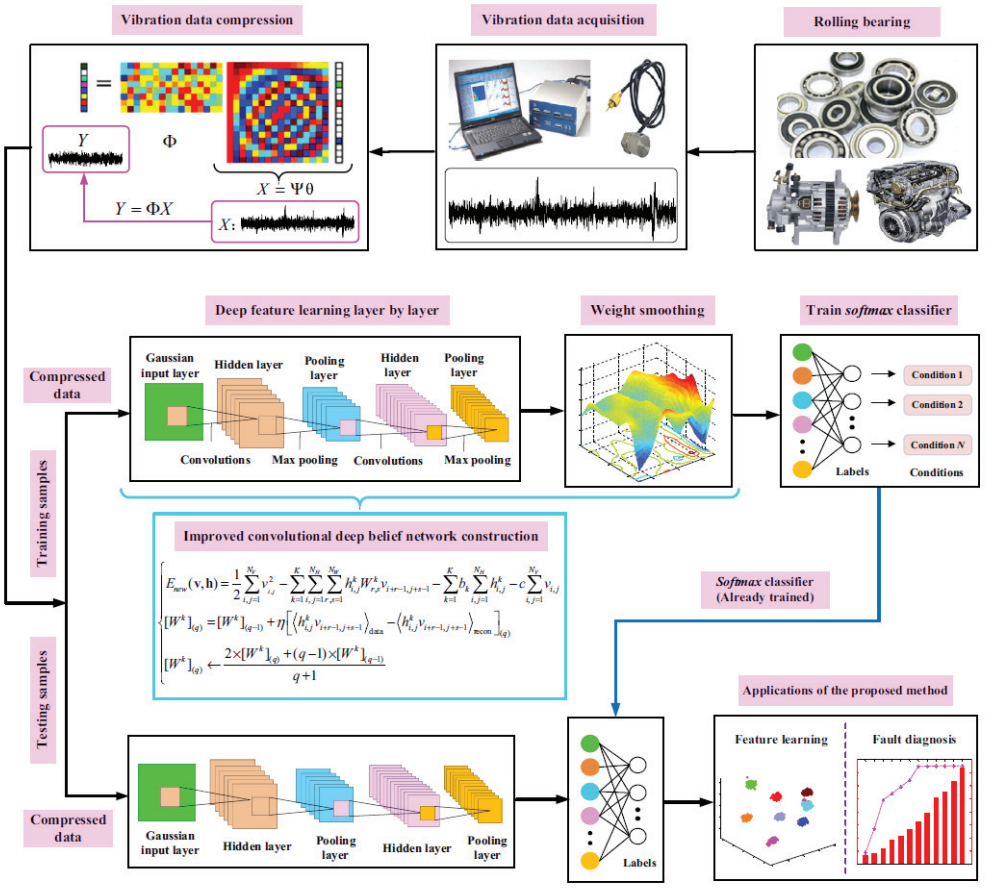


Fig. 4. The framework of the proposed method.

Figure 2.31 – Schéma récapitulatif de la méthode de diagnostic par CDBN-CS, d'après [H. Shao, H. Jiang, H. Zhang *et al.*, 2018]

Le processus décrit à la figure 2.31 peut être résumé par les étapes suivantes :

1. Sélection d'une population de paliers de roulements moteur, qui sont des roulements à rouleaux.

2. Acquisition des données vibratoires par banc d'essais.

3. Application de la technique d'acquisition comprimée. Cette technique consiste, à la manière d'un AE, à extraire des signaux bruts un vecteur représentatif du comportement initialement mesuré.
4. Apprentissage d'un Convolutional Deep Belief Network (CDBN) avec méthode de moyenne glissante exponentielle pour son optimisation.
5. Inférence avec le CDBN pour obtenir le résultat de diagnostic.

Il est à noter que les acquisitions des signaux vibratoires sont effectuées à 25 kHz, [H. Shao, H. Jiang, H. Zhang *et al.*, 2018, table 11] cherchent à détecter 8 types de domaines de fonctionnement, à savoir :

- Fonctionnement nominal du roulement, sans défaut.
- Défaut sur la bague extérieure.
- Défaut sur la bague intérieure.
- Défaut sur les rouleaux.
- 4 ensembles de défauts composés constitués de :
  - défauts sur la bague extérieure et intérieure,
  - défauts sur la bague intérieure et les rouleaux,
  - défauts sur la bague extérieure et les rouleaux,
  - défauts sur la bague extérieure, intérieure ainsi que les rouleaux.

Afin d'évaluer la performance de leur approche, [H. Shao, H. Jiang, H. Zhang *et al.*, 2018] utilisent en apprentissage 99 échantillons de données pour chaque condition, dont 25 serviront à l'inférence. Chaque échantillon est constitué d'un vecteur de dimension 2 500 compressé en un vecteur de dimension 1 024 après l'étape d'acquisition comprimée. La performance des algorithmes principalement développés dans les travaux de la littérature concernant le PHM est donnée à la table 2.2. Dans cette première expérience, [H. Shao, H. Jiang, H. Zhang *et al.*, 2018] utilisent pour l'inférence les données compressées et des indicateurs statistiques.

Catégorie	Algorithme	Précision	Écart type
Deep learning	CDBN-CS	**97.375** %	0.5336
	DBN standard	90.675 %	1.0691
	CNN standard	91.338 %	0.6720
	SAE standard	91.013 %	0.7937
Shallow learning	NN-Compressed Sensing (CS)	51.60 %	6.40
	NN handcrafted feature	83.07 %	2.16
	SVM-CS	57.93 %	3.42
	SVM handcrafted feature	84.53 %	1.77

Table 2.2 – Précision de différents algorithmes pour le diagnostic, d'après [H. Shao, H. Jiang, H. Zhang *et al.*, 2018]

D'après les résultats de la table 2.2, une amélioration de 5 % par rapport au meilleur algorithme de diagnostic, à savoir l'approche par CNN, est observée. Il est aussi important de noter la tendance confirmée, qui est que les algorithmes d'apprentissage profond sont plus performants que leurs structures équivalentes en apprentissage peu profond.

Remarque. *Notons que l'intérêt d'une publication dans le domaine est de trouver un algorithme qui possède des indicateurs statistiques meilleurs que ceux déjà développés par les pairs. Seulement, il faut étudier les résultats annoncés avec circonspection. En effet, dans le domaine de l'apprentissage machine, les paramètres des infrastructures sont d'une importance capitale pour les performances de ce dernier. Dans la littérature, lorsque les équipes comparent leur algorithme à l'état de l'art, elles n'indiquent pas tout le temps les paramètres utilisés. De plus, l'étendue des efforts que l'équipe a fournis pour l'apprentissage des réseaux peu profonds et les architectures concurrentes reste inconnue.*

Dans un second temps, [H. Shao, H. Jiang, H. Zhang *et al.*, 2018] comparent leur algorithme à des méthodes d'apprentissage machine standards couplées à des algorithmes d'extraction de signatures, à savoir : la décomposition modale empirique ou Empirical Mode Decomposition (EMD), la décomposition en ondelettes ou Wavelet Packet Decomposition (WPD) et la décomposition en modes locaux ou Local Mean Decomposition (LMD). Les résultats obtenus sont rassemblés à la table 2.3.

Algorithme	Précision	Écart type
CDBN-CS	**94.80** %	0.53
Back Propagation Neural Network (BPNN) - EMD	85.73 %	1.48
BPNN - WPD	87.20 %	1.47
BPNN - LMD	87.47 %	1.45
SVM - EMD	87.40 %	1.35
SVM - WPD	88.47 %	1.14
SVM - LMD	87.73 %	0.95

Table 2.3 – Précision de différents algorithmes pour le diagnostic et comparaison avec des algorithmes d'apprentissage machine standards, d'après [H. Shao, H. Jiang, H. Zhang *et al.*, 2018]

D'après la table 2.3, comme démontré dans les travaux de [Breuneval, 2017], l'utilisation préliminaire d'algorithmes d'extraction d'information permettant de construire des descripteurs permet d'augmenter les performances de l'algorithme de diagnostic. Seulement, l'approche utilisée par [H. Shao, H. Jiang, H. Zhang *et al.*, 2018] est toujours plus performante. Pour illustrer visuellement le gain de performances permis par l'extraction de signatures par l'algorithme d'apprentissage profond, [H. Shao, H. Jiang, H. Zhang *et al.*, 2018] utilisent une décomposition en composantes principales ou PCA et obtiennent les résultats de la figure 2.32.

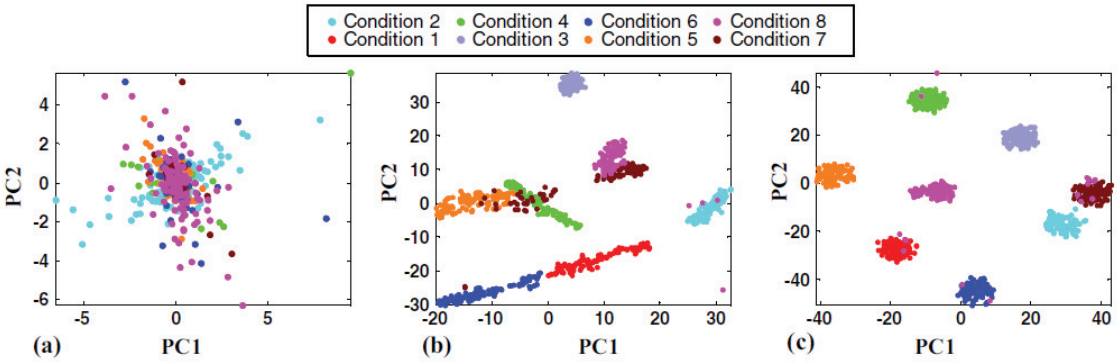


Figure 2.32 – Décomposition en composantes principales des signatures de défauts extraites par : (a) acquisition comprimée, (b) algorithme d'extraction de signatures, (c) algorithme d'apprentissage profond, d'après [H. Shao, H. Jiang, H. Zhang *et al.*, 2018]

D'après la figure 2.32, les signatures profondes ou *deep features* sont mieux séparées que les signatures obtenues par les deux méthodes concurrentes. Ainsi, la figure (*c*) présente des groupes mieux définis que les schémas (*a*) et (*b*) au sens où les distances inter-classes sont maximisées tandis que les distances intra-classes sont minimisées.

Notons cependant que le domaine du PHM souffre du problème récurrent de la mauvaise qualité de la base de données. En effet, pour diagnostiquer correctement un état, il convient d'avoir appris sur un nombre suffisant d'exemples représentatifs. Dans l'industrie, les bases de données sont souvent incomplètes et biaisées pour des raisons de coûts. Le défi est alors d'appliquer des algorithmes d'apprentissage profond à ces situations et d'obtenir des résultats fiables et exploitables. Tout comme [H. Shao, H. Jiang, Y. Lin *et al.*, 2018], [Chong Zhang *et al.*, 2018] s'attaquent à ce problème et développent l'algorithme Evolutionary Cost Sensitive Deep Belief Network (ECS-DBN). Dans ces travaux, la difficulté d'obtenir une base de données d'apprentissage représentative de défauts est soulignée. En effet, ces bases sont en général biaisées vers un état sain car il est plus difficile d'obtenir des échantillons lorsque le système est défectueux. De manière plus général, dans

l'industrie, un système est plus souvent dans un état sain qu'en défaut. Cela rejoint et justifie le verrou scientifique 1 présenté à la section 1.4. Il est intéressant de noter que l'état de l'art partage ces limitations mais réussit à y répondre par une approche d'apprentissage profond. Ce propos est cependant à nuancer et seuls les tests d'algorithmes utilisés avec nos données d'exploitation pourront statuer sur l'intérêt des méthodes d'apprentissage profond pour la résolution de la problématique de ces travaux. Rappelons que ces travaux sont ancrés dans un contexte industriel. Notons que le papier très détaillé de [Chong Zhang *et al.*, 2018] indiquent qu'ils ont utilisé pour l'entraînement de leur algorithme pas moins de 13 798 690 échantillons de données (représentant des mesures vibratoires, des forces et des couples) avec des ratio de déséquilibre allant de 1 (base de donnée équilibrée) à 1, 63 entre deux classes de défauts. L'approche adoptée est résumée par la figure 2.33.

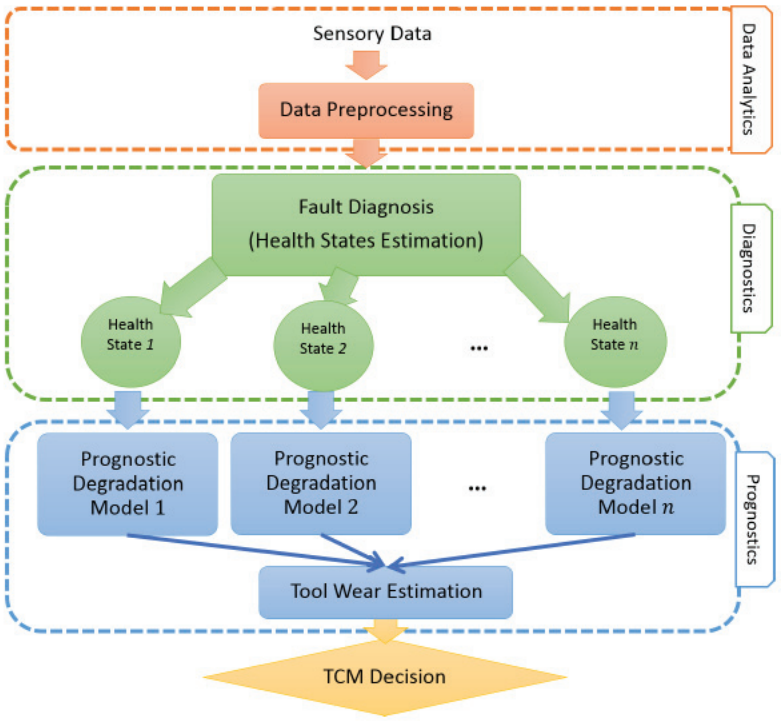


Figure 2.33 – Méthode de diagnostic/pronostic pour le Tool Condition Monitoring (TCM), d'après [Chong Zhang *et al.*, 2018]

Contrairement aux approches jusqu'ici développées, [Chong Zhang *et al.*, 2018] ajoutent la phase de pronostic au *framework* de diagnostic, ce qu'ils appellent le Multi-state Diagnosis end Prognosis (MDP). Ainsi dans l'approche classique, les données capteurs sont récupérées du système en cours de surveillance, ensuite une étape de diagnostic conduit à obtenir l'état de santé du système puis, à chaque état de santé, un modèle de dégradation est associé. Notons que l'état de santé signifie ici la probabilité d'appartenir à une classe de défaut. Aussi, la fusion des modèles de pronostic pourra conduire à l'estimation globale de l'état d'usure du système. Par exemple, en diagnostiquant deux défauts en même temps, deux modèles de dégradations seront obtenus et leur fusion conduira à une estimation de l'état de santé général. L'algorithme ECS-DBN peut donc servir à la fois à effectuer du diagnostic et à réaliser un pronostic. Leur approche est alors capable de produire un pronostic robuste et généralisable.

2.3.5 Algorithmes à base de réseaux de neurones récurrents

À la rédaction des travaux de [Nanduri et Sherry, 2016], le standard de l'industrie pour le diagnostic de défauts en aéronautique était une approche basée sur des seuils, c'est pourquoi [Nanduri et Sherry, 2016] comparent les résultats de leurs algorithmes à ces méthodes. Dans cette méthode standard, les paramètres à surveiller sont définis et les algorithmes déclenchent des alertes lorsque leurs valeurs excèdent une certaine limite. Cette approche est peu robuste, comparée aux méthodes d'intelligence artificielle. En effet, en sélectionnant *a priori* des indicateurs, la dimension

de l'information de diagnostic a été réduite et cette même réduction empêche la détection de défauts cachés. Pour tester la validité de ses algorithmes et pallier les problèmes de confidentialité inhérents à l'industrie aéronautique, [Nanduri et Sherry, 2016] ont utilisé une base de données générée par simulation de vols constituée de 21 variables discrètes et continues échantillonnées à 2 Hz pour 500 vols. Dans les 500 vols étaient alors simulés 485 vols sains et 15 en anomalie. Les données récupérées sont constituées de séries temporelles et des drapeaux d'incidents. Après avoir imposé une structure d'un neurone par défaut à détecter, [Nanduri et Sherry, 2016] ont réalisé une étude comparative avec différents algorithmes de l'état de l'art. Les défauts à détecter sont des défauts usuels tels que l'influence du vent, une variation brusque de l'angle de roulis et d'autres défauts.

- Détection d'anomalies par noyaux multiples et classification avec une machine à vecteur de support ou Mutliple Kernel Based Anomaly Detection (MKAD)-SVM
- Réseau de neurones à mémoire à court et long terme ou LSTM
- Réseau de neurones à porte récurrente ou GRU

Il en ressort que l'approche à base de réseaux de neurones récurrents (LSTM et GRU) permet de détecter 8 anomalies sur 11 injectées, contrairement à l'approche d'apprentissage machine peu profond qui en détecte 2 de moins. D'après [Nanduri et Sherry, 2016], la réduction de dimension utilisée dans MKAD-SVM occasionne une perte d'information. L'approche à base de réseaux de neurones récurrents est de fait plus sensible aux anomalies cachées et est mieux adaptée à la détection de défauts en ligne en temps réel.

Conclusion

Ce premier chapitre a permis de développer la nature du PHM et de cibler les étapes qui seront abordées dans ces travaux, à savoir : le pré-traitement de la donnée, la détection de pannes et pour finir le pronostic de pannes à partir de données provenant d'un inverseur de poussée électromécanique. L'étude des tendances du domaine a été effectuée et cette dernière a permis de démontrer l'intérêt des méthodes d'apprentissage artificiel dans notre domaine. En effet, les travaux dédiés à l'application de méthodes d'intelligence artificiel pour la surveillance et la prédiction d'états de santé de systèmes ne cessent de croître depuis ces trois dernières années, et notamment l'utilisation des méthodes d'apprentissage profond. De plus, contrairement à nos travaux qui étudient des signaux vibratoires provenant d'actionneurs électromécaniques, l'état de l'art est principalement tourné vers la détection de pannes sur des roulements, à partir de signaux vibratoires acquis à haute résolution, soit de l'ordre de la dizaine de kHz. Cette caractéristique se prête bien à l'utilisation de l'apprentissage profond où la masse de données pertinentes est primordiale. Mais nos travaux ne partagent pas cette caractéristique. Nos systèmes sont à plus basse fréquence et en petit volume. Afin d'utiliser des techniques d'apprentissage machine, une manière de pouvoir contourner le problème serait notamment l'usage de techniques de génération de données artificielles ou encore le développement de techniques d'apprentissage non supervisées. Le domaine de l'intelligence artificielle est en pleine expansion et son usage dans le domaine du PHM suit cette tendance. Aussi, cette synthèse de la littérature ne cherche pas à fournir une liste exhaustive de toutes les méthodes utilisées, mais plutôt à fournir les bases qui étayeront les méthodes qui seront développées dans les sections suivantes. Nous avons pris le parti d'extraire le plus d'informations possibles à partir uniquement des données qui étaient à notre disposition. Ces travaux ont vocation à développer une méthode complète de PHM allant de la captation de la donnée à la prédiction de la durée de vie d'un système surveillé avec peu de données et ce, de la manière la plus fiable possible. C'est pourquoi, la seule architecture profonde qui sera utilisée dans le chapitre 3 suivant est un réseau de segmentation, à base de CNN, développé dans la section 3.3. L'apport de ces travaux à l'état de l'art réside donc dans la méthode complète à partir d'algorithmes d'intelligence artificielle au sens large.

Chapitre 3

Extraction et exploration d'informations à partir des données brutes

Ce chapitre présente une nouvelle méthode pour le partitionnement de séries temporelles à base d'apprentissage non supervisé, ainsi qu'un nouvel indicateur pour mesurer la pertinence des résultats de groupement obtenus au regard de notre problématique de Prognostics and Health Management (PHM). Dans un premier temps, un état de l'art des méthodes de partitionnement de séries (*clustering*) est effectué à la section 3.1. Puis des données brutes d'un banc industriel sont extraites des descripteurs de défauts à la section 3.2. Ces données seront utilisées par la suite dans les chapitres suivants et notamment dans la section 3.3, qui présente la méthode développée pour assigner un label aux données d'exploitation. Les résultats de cette dernière sont ensuite comparés aux méthodes de partitionnement de l'état de l'art à la section 3.4. Enfin, dans cette même section, sera détaillé un nouvel indicateur de qualité pour le partitionnement de séries temporelles.

3.1 État de l'art des méthodes de partitionnement de séries temporelles

Dans le domaine du PHM, et plus particulièrement dans ces travaux, l'étude des séries temporelles est un axe d'intérêt majeur. Elles permettent de tracer l'évolution du système. Ces séries décrivent les états successifs du système sous surveillance. Dans le cadre des méthodes de diagnostic, il convient de classifier les signaux récupérés à chaque instant pour déterminer l'état de santé du système, à savoir : le mode de défaillance et le niveau de dégradation. Il est aussi intéressant d'effectuer du regroupement de données issues de ces séries (*clustering*, ou partitionnement) lorsque les labels des données (les états du système) sont inconnus. L'état de l'art récent de [Susto, Cenedese et Terzi, 2018] recense la plupart des méthodes de classification pour les séries temporelles. Pour l'approche à base d'apprentissage profond qui est adoptée ici, les travaux de [Min *et al.*, 2018] et [Ismail Fawaz, Forestier *et al.*, 2019] recensent quant à eux les méthodes de classification et de partitionnement appliquées aux séries temporelles. Les notions de classification et de regroupement peuvent être appliquées à la détection d'anomalies dans les données temporelles comme présentées dans [Braei et Wagner, 2020] ou encore dans [Fink *et al.*, 2020], cette dernière référence se focalisant davantage sur l'apprentissage profond dans le domaine spécifique du PHM.

D'après [Fink *et al.*, 2020], la visualisation du comportement des séries temporelles sous forme d'images est un sujet de recherche d'actualité. Ainsi, la dynamique des séries temporelles peut être extraite par la représentation du champ angulaire de Gram - Gramiam Angular Field (GAF) de [Zhiguang Wang et Oates, 2015b] ou le graphique de récurrence - Recurrent Plot (RP) de [Hatami, Gavet et Debayle, 2018]. Une variante du GAF, les Gramiam Angular Summation Fields (GASF) [Zhiguang Wang et Oates, 2015a], est utilisée par [Martínez-Arellano, Terrazas et Ratchev, 2019] pour transformer des séries temporelles en images. Dans leurs travaux, trois signaux sont transformés parallèlement et les trois images résultantes sont considérées comme étant chacune le canal d'une image Red Green Blue (RGB). De manière classique, dans l'état de l'art, un Convolutional Neural Network (CNN) est ensuite utilisé pour déterminer l'état de défaut des séries considérées. Notons que contrairement à notre contexte industriel, l'apprentissage du CNN utilisé a été réalisé avec des données déjà labellisées. Suivant la même tendance de transformation de séries unidimensionnelles en images, [Chuxu Zhang *et al.*, 2019] ont développé Multi-Scale Convolutional Recurrent Encoder-Decoder (MSCRED). À partir de séries temporelles multi-variées sont calculées des matrices de *signatures*. Chaque matrice est créée en calculant l'inter-corrélation entre différents segments de séries temporelles sources. *De facto* les matrices sont de tailles carrées. Leur dimension est définie en fonction de la taille de la fenêtre utilisée pour le calcul d'inter-corrélation. Ensuite, un CNN hybride contenant des couches de Long Short Term Memory (LSTM) extrait un signal intermédiaire qui, après passage dans un réseau décodeur, produit une image finale. Cette image permet la création d'un signal en une dimension représentant un score d'anomalie [Chuxu Zhang *et al.*, 2019, figure 7]. Ce signal sépare les données sources en zones d'anomalies, ce qui ressemble aux frontières de groupes que nous chercherons à obtenir par la suite. Notons que le manque de données avec labels n'est pas uniquement limité au domaine industriel du PHM. Ce constat est partagé notamment par [Mousavi *et al.*, 2019] qui cherchent à partitionner des données sismiques pour détecter des motifs. Pour ce faire, les spectrogrammes de Fourier des séries sont fournies à un auto-encodeur incomplet qui possède deux sorties : la première étant l'image reconstruite à partir des données d'entrée et la seconde étant une distribution de Student [Student, 1908] modélisant la probabilité d'appartenance à un centroïde dans les données. Notons que l'apprentissage de l'auto-encodeur incomplet est effectué uniquement avec des

données sans labels. Dans un autre domaine, [Asadi et Regan, 2020] étudient le flux de trafic de véhicules sur la côte californienne à partir d'une méthode de partitionnement utilisant la distance Dynamic Time Wrapping (DTW) qu'ils ont généralisée à un ensemble de données multidimensionnelles. Cela permet alors de générer des images des matrices obtenues dans lesquelles peuvent être détectés des groupements de données [Asadi et Regan, 2020, figure 8]. Sans passer par une transformation quelconque, [Bao *et al.*, 2019] ne considèrent pas les séries temporelles mais les images de leurs graphes en noir et blanc. À partir de celles-ci, des auto-encodeurs extraient des descripteurs. Chacun de ces descripteurs est comparé à une base de données avec labels pour déterminer l'état de santé du système étudié. Enfin, adoptant une approche moins courante, [Ali *et al.*, 2019] transforment les séries temporelles en des graphes en deux dimensions. L'algorithme de regroupement est effectué directement sur ces graphes par un Deep Convolutional Auto-Encoder (DCAE). Utilisant seulement, cette fois-ci, la concaténation de plusieurs séries temporelles pour fournir une entrée adéquate à un réseau de neurones profond, [Ismail Fawaz, Lucas *et al.*, 2020] ont développé *InceptionTime*, un réseau de neurones à convolutions, dans le but de supplanter la structure profonde Hierarchical Vote Collective of Transformation-Based Ensembles for Time Series Classification (HIVE-COTE) [Lines, Taylor et Bagnall, 2016], extrêmement coûteuse à entraîner. Son entrée est constituée de séries temporelles multi-variées concaténées dans une matrice. Cette matrice traverse de multiples couches spécifiques appelées *inception* et des couches de sorties complètement connectées permettent la production d'un nombre défini de classes. Adoptant un format de données d'entrées similaire, [Perslev *et al.*, 2019] ont développé U-TIME. Cette architecture dérivée de U-NET (la partie décodeur étant modifiée) leur permet de classifier des électroencéphalogrammes de patients en phase de sommeil. Enfin, sans encoder les séries temporelles, [F. Xu, W. t. P. Tse et Y. L. Tse, 2018] s'intéressent au diagnostic de données vibratoires de roulements. L'approche adoptée consiste à extraire des descripteurs des données brutes à partir d'un Stacked Denoised Auto-Encoder (SDAE) puis l'algorithme de Gath-Geva [Gath et Geva, 1989] est utilisé pour partitionner les données [F. Xu, W. t. P. Tse et Y. L. Tse, 2018, figure 2]. Ils présentent aussi le fait que les modèles profonds sont de plus en plus utilisés pour effectuer du diagnostic, notamment les structures à base d'auto-encodeurs - Auto-Encoder (AE). Malheureusement ces structures nécessitent en général des données avec labels pour leur entraînement ce qui est rarement le cas dans le domaine industriel. C'est pourquoi, à défaut de ne pouvoir utiliser d'algorithmes de classification, les algorithmes de partitionnement sont employés. En continuant dans les ensembles de données sans labels, [Tavakoli *et al.*, 2020] présentent une approche plus classique en extrayant des descripteurs des données brutes et en appliquant l'algorithme des *KMeans*. Ce dernier nécessite la connaissance *a priori* du nombre de groupes à trouver dans les données. Le schéma de partitionnement sera alors calculé pour correspondre au mieux à ce nombre de groupes. Afin de déterminer le schéma le plus pertinent, plusieurs valeurs de groupes sont testées et le schéma de partitionnement possédant le meilleur score de *Silhouette* [Rousseeuw, 1987] est retenu. Cette base de données ainsi constituée permet l'apprentissage d'un auto-encodeur qui sera alors capable de partitionner des données inconnues. De même avec des données issues de roulements sans labels, [F. Xu et P. W. Tse, 2019] utilisent une Deep Belief Network (DBN) pour en extraire des descripteurs, puis une Principal Component Analysis (PCA) réduit la dimension des données. Le résultat est fourni à l'algorithme de propagation d'affinité [Frey et Dueck, 2007] pour effectuer le partitionnement. Concernant la détection d'anomalies pour des capteurs de température, [Guo *et al.*, 2018] développent un réseau GRU-based Gaussian Mixture Variational Auto-Encoder (GGM-VAE) qui permet de calculer des corrélations entre séquences temporelles. Un modèle de mélange gaussien est ensuite utilisé dans l'espace latent du réseau pour obtenir une distribution des valeurs du descripteur. La comparaison à un seuil de la probabilité reconstruite permet alors de déterminer un potentiel défaut. [Munir *et al.*, 2019] fournissent également directement les séries temporelles à un réseau de neurones à convolution profond, sans nécessiter de labels. Enfin, [Hao, H. Cao et Draayer, 2020] utilisent les données mutli-variées de [Dua et Graff, 2017] pour classifier des séries temporelles avec leur Convolutional Neural Network with Class speficied Fully connected components (CNN-CF). Enfin, [J. Chen *et al.*, 2020] utilisent la logique flou pour obtenir un schéma de partitionnement pertinent.

L'ensemble de ces représentations sont utilisées car elles permettent d'améliorer les résultats de classification ou de regroupement (*clustering*, ou partitionnement) de données à partir de Deep Neural Network (DNN). C'est pourquoi, dans ces travaux, une approche similaire est utilisée pour visualiser le comportement des séries temporelles.

Dans [Jain, Murty et Flynn, 1999], Jain a défini le *partitionnement* comme "la classification non supervisée de motifs en groupes." Alors que cette technique d'analyse exploratoire est

largement utilisée dans les ensembles de données organisés dans l'espace, elle est peu utilisée dans les séries chronologiques où les données sont organisées en temps opportun. D'où en découle la nécessité de créer une nouvelle méthode avec son indicateur de qualité. Plusieurs études récentes ont été présentées, utilisant le partitionnement ou la reconnaissance des formes pour étiqueter les données en vue de leur utilisation ultérieure dans l'entraînement d'une structure de diagnostic. Dans [Hendrickx *et al.*, 2020], chaque déviation de l'état nominal des machines électriques est détectée à partir d'un algorithme de partitionnement. De plus, les conditions de fonctionnement peuvent être déduites de la structure des groupes de données [Perafán-López et Sierra-pérez, 2020]. L'association entre un groupe et la gravité d'un défaut est clairement établie dans [Zaporowska *et al.*, 2020]. Les modèles trouvés dans cette approche sont supposés être représentatifs de la détection de plusieurs défauts. Cette même hypothèse a été suivie ici : un groupe de données représente une certaine gravité de défaut.

3.2 Présentation des données d'étude

3.2.1 Présentation des données brutes

Comme montré à la sous-section 1.3, le système étudié est un inverseur de poussée électrique : l'Electric Thrust Reverser Actuation Systems (E-TRAS). Ce dernier est composé de 4 vérins qui sont les actionneurs d'étude. Pour rappel, lors de l'activation de l'aérofrein, une partie de la nacelle translate pour découvrir une grille qui permettra la déviation du flux d'air secondaire pour générer ainsi une force contraire à la poussée du turboréacteur. Cette partie mobile de la nacelle est liée à chacun des 4 vérins de l'actionneur qui sont nommés Lower Left Actuator (LLA), Lower Right Actuator (LRA), Upper Right Actuator (URA) et Upper Left Actuator (ULA), en référence à leur position sur le banc de test. Le banc de test est un banc d'intégration, ou *Iron Bird*, constitué de 4 vis à billes mises en mouvement par une machine synchrone à aimants permanents (MSAP) *via* des liaisons mécaniques souples. Chaque vis à bille est alors en liaison avec un vérin de contre charge qui simule notamment la composante des forces aérodynamiques sur l'E-TRAS. Sur chacun de ces vérins est placé un accéléromètre à axe longitudinal. Ce sont ces relevés qui seront utilisés tout au long de ces travaux de PHM.

Les signaux des 4 accéléromètres sont échantillonnés à 1 kHz. Ils sont repérés par l'actionneur dont ils mesurent les vibrations ainsi que le mode d'opération de l'inverseur de poussée lors de la captation, à savoir : le déploiement de la nacelle ou le repliement de la nacelle. Pour l'année 2017, les données utilisées représentent 1 198 cycles d'endurances réduits à $n = 1\,130$ après nettoyage de la base de données. Cette quantité de cycles représente 5 jours 21 h 51 min 3 s d'essais non continus dans le temps, *de facto* cela constitue une valeur moindre de durée de vie cumulée pour l'inverseur de poussée de 1 jour 6 h 55 min 51 s. Pendant cette phase, l'actionneur électromécanique E-TRAS, et l'ensemble des vis à billes le constituant, ont été amenés jusqu'à fin de vie. La fin de vie s'est traduite par un défaut de grippage généralisé qui a empêché l'inverseur de poussée de remplir sa fonction première : faire translater la partie mobile de la nacelle. Soit n_i et n_j le nombre d'échantillons récupérés lors d'une phase de déploiement et de repliement avec $i \in [\![1; n]\!]$ et $j \in [\![1; n]\!]$. Une base de données $\mathfrak{B}$ rassemblant les signaux de déploiement $\mathfrak{A}_{\mathcal{D}} \in \mathfrak{M}_{n_i,n,4}(\mathbb{R})$ et de repliement $\mathfrak{A}_{\mathcal{R}} \in \mathfrak{M}_{n_j,n,4}(\mathbb{R})$ peut alors être créée.

Remarque. *Notons que par la suite, dans un soucis de concision, tous les exemples seront donnés à partir des signaux extraits d'une phase de déploiement de la nacelle.*

Pour caractériser le contenue de la base de données $\mathfrak{A}_{\mathcal{D}}$, l'actionneur LLA est sélectionné. La matrice $\mathbb{A}_{\mathcal{D}}^{\text{LLA}} \in \mathcal{M}_{n_i,n}(\mathbb{R})$ rassemble alors l'ensemble des données captées sur cet actionneur.

$$\mathbb{A}_{\mathcal{D}}^{\text{LLA}} = \begin{bmatrix} \overset{1^{\text{e}}\,essai}{a_{11}\left(\tau_1 + t_0\right)} & \overset{2^{\text{e}}\,essai}{a_{12}\left(\tau_2 + \Delta_1 t_0\right)} & \cdots & \overset{n^{\text{e}}\,essai}{a_{1n}\left(\tau_n + \sum_{j=1}^{n-1} \Delta_j t_0\right)} \\ a_{21}\left(\tau_1 + \delta_1^1 t_0\right) & a_{22}\left(\tau_2 + \delta_1^2 \Delta_1 t_0\right) & \cdots & a_{2n}\left(\tau_n + \delta_1^n \sum_{j=1}^{n-1} \Delta_j t_0\right) \\ \vdots & \vdots & \vdots & \vdots \\ a_{n_i 1}\left(\tau_1 + \Delta_1 t_0\right) & a_{n_i 2}\left(\tau_2 + (\Delta_2 + \Delta_1) t_0\right) & \cdots & a_{n_i n}\left(\tau_n + \sum_{j=1}^{n} \Delta_j t_0\right) \end{bmatrix} \in \mathcal{M}_{n_i,n}\left(\mathbb{R}\right)$$

La non continuité des essais dans le temps est modélisée par les indices temporels des éléments $(a_{ij})_{1\leq i\leq n_i, 1\leq j\leq n}$ de la matrice $\mathbb{A}_{\mathcal{D}}^{\text{LLA}}$. La durée d'un essai, ici un déploiement de nacelle, est représentée par la grandeur Δ_j définie telle que : $\forall j \in [\![1;n]\!], \Delta_j = \sum_{i=1}^{n_i} \delta_i^j$. Comme les essais ne sont pas contigus dans le temps, une suite d'éléments $(\tau_j)_{1\leq j\leq n}$ modélise le décalage temporel entre deux essais. Cela permet de garder une image du temps expérimental. Cependant, cette modélisation permet de mettre en exergue le fait que l'état du système reste inconnu entre deux essais. C'est pourquoi, il est nécessaire de faire l'hypothèse 1 pour la suite des travaux.

Hypothèse 1. *Le vieillissement calendaire n'est pas pris en compte dans ces travaux. Ainsi, il est supposé que l'actionneur se dégrade uniquement en fonctionnement.*

De plus, à chacune des colonnes de la matrice $\mathbb{A}_{\mathcal{D}}^{\text{LLA}}$ est associée un numéro de cycle.

Remarque. *Notons que c'est ce numéro de cycle qui deviendra l'indice temporel des séries étudiées dans les sections suivantes, et notamment pour les séries temporelles du chapitre 4.2.*

Enfin, le mot *actionneur* a été utilisé indifféremment pour désigner l'inverseur de poussée E-TRAS comme les quatre vis à billes le constituant. Néanmoins, ne possédant pas les données de plusieurs inverseurs de poussée, considérer un unique système complet pour la suite des travaux ne permettra pas d'obtenir des résultats pertinents. C'est pourquoi, ce seront les signaux vibratoires des vis à billes qui seront considérés comme cas d'étude. Ainsi, un *actionneur* désignera une vis à bille et l'hypothèse fondamentale 2 est faite.

Hypothèse 2. *Les quatre actionneurs constituant l'inverseur de poussée sont considérés comme indépendants dans tous ces travaux.*

3.2.2 Choix des descripteurs de défauts *a priori*

Pour pouvoir effectuer le diagnostic et le pronostic d'actionneurs en PHM, l'approche standard consiste à extraire des grandeurs des signaux bruts qui seraient représentatifs d'un vieillissement. Un descripteur idéal serait une grandeur parfaitement monotone, ce qui permettrait de corréler son évolution au vieillissement du système, lorsqu'aucune maintenance n'a été effectuée. Un bon indicateur de défaut (ou *signature*, ou *descripteur*) devrait représenter, par ses variations, la perte de fonction progressive du système étudié. Notons que, comme montré par la figure 2.23, cette approche est essentiellement utilisée dans les méthodes de PHM ne faisant généralement pas appel à des réseaux d'apprentissage profond. En effet, ces dernières architectures permettent d'extraire leurs propres indicateurs pertinents à partir de leurs premières couches. Néanmoins, dans notre étude, nous avons choisi d'extraire des descripteurs à partir des données brutes pour alimenter les algorithmes qui suivront.

Ainsi, le calcul de descripteurs de défauts est une première étape de compression de l'information. En effet, à chaque cycle de déploiement (*resp.* repliement), autrement dit à chaque colonne de $\mathbb{A}_{\mathcal{D}}^{\text{LLA}}$ (*resp.* $\mathbb{A}_{\mathcal{R}}^{\text{LLA}}$), sera associée la valeur d'un descripteur pour un cycle donné. Comme dans la remarque 6, les données considérées ne seront plus indexées par rapport au temps absolu mais par rapport à un indice de cycle. Les données brutes possédaient initialement une fréquence d'échantillonnage de l'ordre de la milliseconde, les descripteurs posséderont donc une fréquence d'échantillonnage de l'ordre du cycle. Cette modélisation reste pertinente pour ce cas d'étude car

un cycle représente dans la réalité un vol complet. La procédure d'extraction de descripteur peut être schématisée par la figure 3.1.

$$\mathbb{A}_{\mathcal{D}}^{\text{LLA}} = \begin{bmatrix} a_{11}(\cdots) & a_{12}(\cdots) & \cdots & a_{1n}(\cdots) \\ a_{21}(\cdots) & a_{22}(\cdots) & \cdots & a_{2n}(\cdots) \\ \vdots & \vdots & \vdots & \vdots \\ a_{n_i1}(\cdots) & a_{n_i2}(\cdots) & \cdots & a_{n_in}(\cdots) \end{bmatrix} \quad \mathcal{S}^j$$

$$D_{\mathcal{D}}^{\text{LLA},j} = \begin{bmatrix} d_{11}(1) & d_{21}(2) & \cdots & d_{n1}(n) \end{bmatrix}^\top$$

FIGURE 3.1 – Schéma de principe pour l'extraction d'un vecteur descripteur D_D^{LLA} à partir de la fonction d'agrégation $\mathcal{S}^j$

La fonction d'agrégation $\mathcal{S}^j : \mathcal{M}_{n_i,1}(\mathbb{R}) \to \mathbb{R}$ permet de créer un vecteur D^j contenant les valeurs pour chaque cycle de l'actionneur lors d'une phase de fonctionnement. Cette opération n'étant pas réversible, il est impossible de conserver la dynamique des données brutes une fois effectué ce premier traitement. C'est pourquoi, le choix de descripteurs pertinents, autrement dit l'ensemble des fonctions $\mathcal{S}^j$ avec $1 \leq j \leq p$ est primordiale pour obtenir des résultats de diagnostic et pronostic pertinents. C'est l'approche qui est usuellement faite dans l'état de l'art en PHM avec la partie du *framework sélection de signatures de défauts* [BREUNEVAL, 2017, figure 1.23]. L'étape d'extraction de descripteurs est effectuée ici *a priori* parce que les descripteurs ne sont pas calculés en fonction de leur qualité pour le vieillissement. Les grandeurs statistiques utilisées ici ont été déterminées à partir d'une expertise du système et aucune autre étape de sélection ne sera effectuée. La figure 3.2 présente une partie des grandeurs statistiques calculées faisant office de descripteurs de défauts.

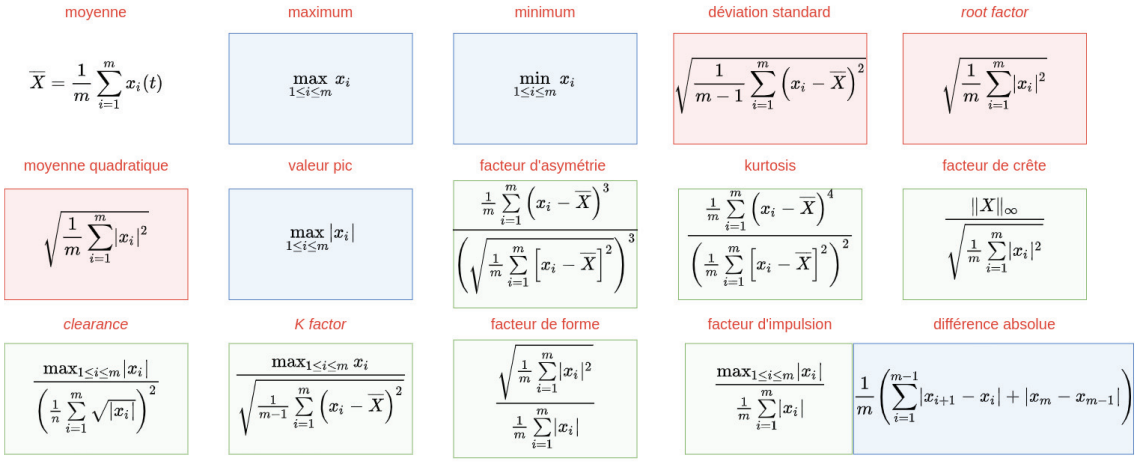


FIGURE 3.2 – Liste des grandeurs statistiques calculées (descripteurs) à partir des données brutes

La figure 3.2 présente *a minima* l'ensemble des descripteurs introduits dans [LEI, 2017, table 2.1] appliqués à un vecteur $X \in \mathcal{M}_{n,1}(\mathbb{R})$ pour généraliser l'approche. Dans la pratique, le vecteur X est remplacé par une colonne de la matrice $\mathbb{A}_{\mathcal{R}}^{\text{LLA}}$. De plus, à ces grandeurs statistiques sont rajoutés des indicateurs témoignant des propriétés statistiques de la distribution des valeurs des signaux vibratoires. Sont ensuite calculés les percentiles de chaque distribution. Ainsi chaque colonne de $\mathbb{A}_{\mathcal{R}}^{\text{LLA}}$ est considérée comme une réalisation d'une variable aléatoire. L'idée étant d'obtenir les valeurs qui séparent chaque distribution en 100 ensembles de tailles égales. Les percentiles sont un cas particulier des quantiles. D'après [GARNIER et MÉLÉARD, 2017] les quantiles sont introduits à la définition 3.2.1.

Définition 3.2.1 : Quantile

Soit une variable aléatoire à valeurs dans $\mathbb{R}$ *de fonction de répartition* F*. Pour* $r \in [0;1]$*, le* quantile *(ou* fractile*) est un nombre réel défini comme :*

$$q_r = \inf \{x \in \mathbb{R}, F(x) \geq r\}$$

q_r *est appelé le* quantile d'ordre r.

Exemple 3.2.1. *Pour* $r = \frac{1}{2}$ *la définition de la médiane est retrouvée. Le quantile divise l'échantillon de la population en deux ensembles de tailles égales. Une variable aléatoire réelle a alors tout autant de chance d'être plus petite ou plus grande que la médiane.*

Ainsi, 13 percentiles allant du 1e au 99e sont choisis pour représenter au mieux toutes les dynamiques des signaux étudiés. Au total, l'ensemble des fonctions d'agrégation $\mathcal{S}^j$ est constitué de 28 éléments. C'est pourquoi dans les sections suivantes sera pris $j \in [\![1;28]\!]$. Le système étudié est alors entièrement caractérisé par l'ensemble de ses descripteurs à chaque cycle. Le calcul de l'ensemble des fonctions $\mathcal{S}^j$ à partir des données d'exploitation pour un actionneur donné génère donc une matrice de taille (n,p). En rajoutant l'ensemble des actionneurs étudiés, une base de données $\mathfrak{M}_{n,p,q}(\mathbb{R})$ est créée contenant l'ensemble des séries d'intérêt pour ces travaux, avec q le nombre d'actionneurs surveillés. Ce sont ces données qui seront labellisées par la suite.

3.2.3 Application aux données d'exploitation

En reprenant l'exemple précédent des signaux vibratoires captés sur l'actionneur LLA en phase de déploiement, les figures 3.3 et 3.4 sont obtenues en normalisant la valeur absolue des descripteurs entre 0 et 1. Dans la pratique, la fonction `sklearn.preprocessing.MinMaxScaler` [Pedregosa *et al.*, 2011] a été utilisée. L'écart de chaque point à la valeur minimum de la distribution est calculé puis normalisé par l'étendue de la distribution. Soit X une distribution de données, la formule de la normalisation est données à l'équation 3.1.

$$X_{\text{norm}} = \frac{X - \min(X)}{\max(X) - \min(X)} \tag{3.1}$$

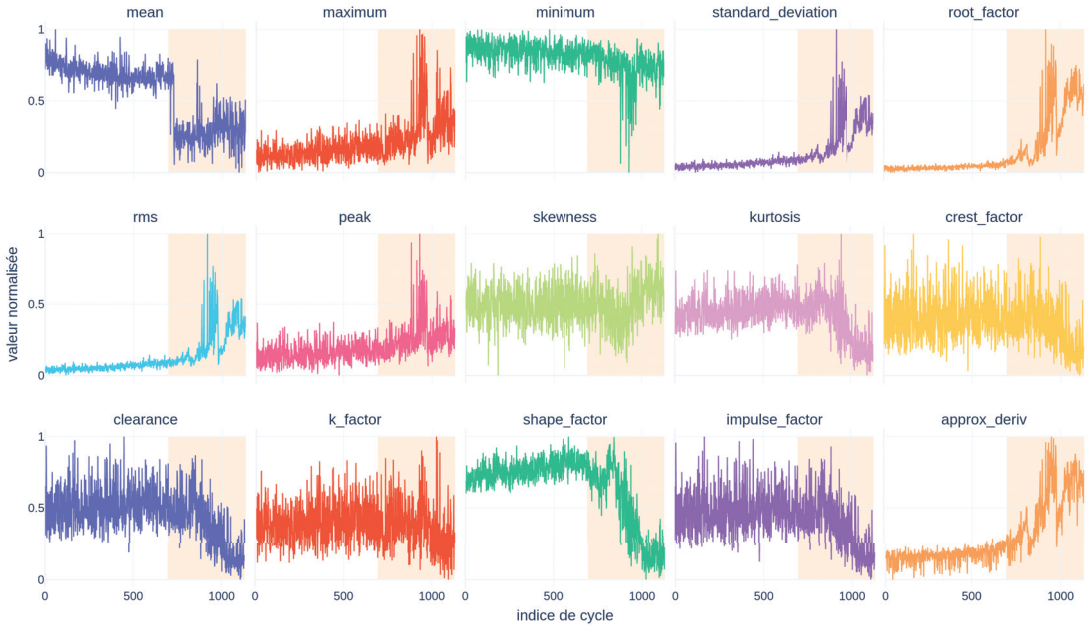


Figure 3.3 – Descripteurs statistiques pour l'accélération du vérin LLA en mode de déploiement

La figure 3.3 permet de mettre en exergue deux tendances principales dans les indicateurs. En effet, certains comme le Root Mean Square (RMS) et la déviation standard sont croissants, tandis que d'autres, comme le kurtosis ou le facteur de dissimalirité, sont décroissants et bien plus bruités que les précédents. Ainsi, par construction, certains sont plus adaptés à la surveillance d'un vieillissement continu, dans le sens où le système se dégrade de manière non brutale, tandis que d'autres sont plus adaptés à la détection de défauts brusques, comme par exemple un défaut d'écaillage dans les bagues de roulement. Les premiers sont des indicateurs énergétiques. L'énergie d'un signal étant définie comme étant la somme continue du carré de ses valeurs, cette grandeur se retrouve dans la formule de l'écart-type (au changement de variable près), du *root factor* et de la moyenne quadratique. C'est pourquoi ces trois descripteurs possèdent des dynamiques voisines avec peu de bruit. Le second type principal sont les indicateurs impulsionnels, plus adaptés à la détection de défauts du même nom. À la différence de la famille précédente, ils sont ici globalement décroissants et bien plus bruités. Un filtrage préalable de l'information est alors nécessaire pour faire ressortir l'information de dégradation subite. Au-delà de ces deux familles de descripteurs, certains comme le maximum, le minimum ou la moyenne possèdent leurs propres dynamiques. Notons que dans le cas de la moyenne, la discontinuité du signal qui apparaît n'a pas trouvé d'explication dans les informations fournies avec les données. Aujourd'hui, il est supposé que cet artefact pourrait être dû à un changement de capteur accélérométrique sur l'actionneur LLA ou à un changement de position de ce dernier pendant la campagne de test.

Notons que la partie couleur orange de chaque graphique représente la zone définie, dans une première analyse d'expert, de l'occurrence du défaut. À partir de cette zone, une dégradation avait été détectée *a posteriori*. L'objet des travaux de ce chapitre est de déterminer avec précision et mesure d'incertitude les différentes phases de dégradation de l'actionneur et ce, de manière totalement automatique.

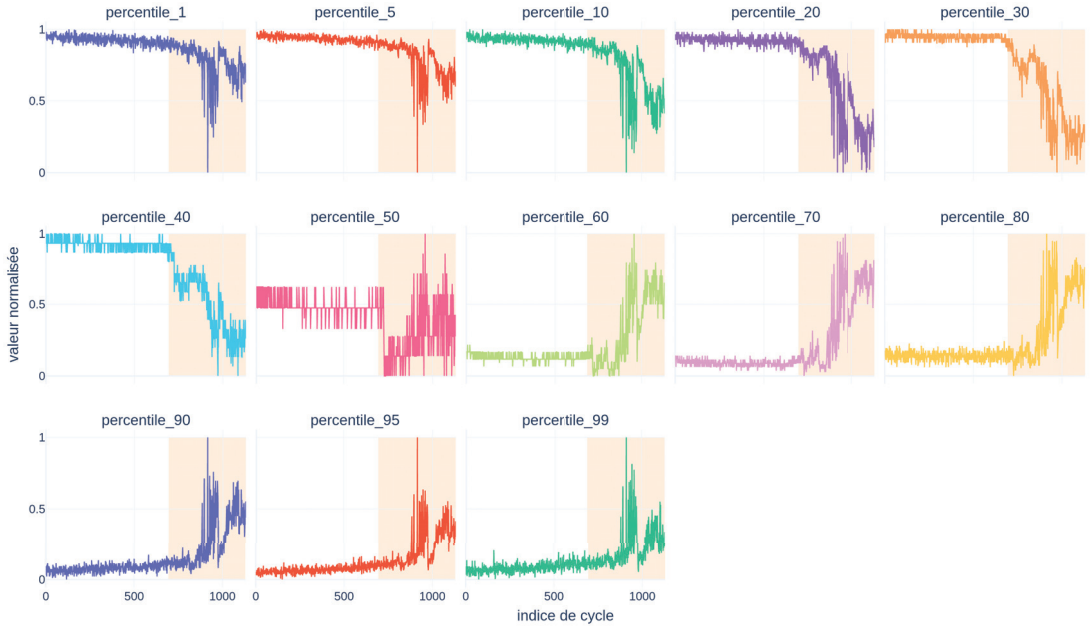


FIGURE 3.4 – Percentiles pour l'accélération du vérin LLA en mode de déploiement

La figure 3.4 permet de détecter trois tendances distinctes dans les dynamiques des descripteurs. En effet, tous les percentiles avant la médiane (le 50^{e} percentile) sont décroissants, la médiane pourrait être stationnaire et tous les percentiles suivants sont croissants. En considérant les percentiles après normalisation en conjonction avec l'identification de la zone de défaut suivant l'expert (zone couleur orange), il peut être remarqué que l'apparition d'un défaut augmente la probabilité qu'ont les données de se trouver en queue de distribution. En se représentant une distribution de données normale, l'apparition d'un défaut engendre alors une translation du centre de cette distribution.

Les figures 3.3 et 3.4 ont présenté les descripteurs qui seront utilisés dans ces travaux, néanmoins, les signaux obtenus sont extrêmement bruités. C'est pourquoi, pour la suite du processus, il convient de filtrer les valeurs de ces indicateurs de défauts. Pour ce faire, l'implémentation de [VIRTANEN *et al.*, 2020] de l'algorithme de Savitzky-Golay [SAVITZKY et GOLAY, 1964] est

utilisée. Pour lisser le signal, la méthode consiste à ajuster un polynôme de degré choisi sur une fenêtre glissante. Le signal filtré devient la valeur de ce polynôme localement. Dans ces travaux, un polynôme de degré 2 ainsi qu'une fenêtre glissante de 51 points ont été choisis pour conserver la courbure du signal. La fenêtre a été déterminée comme un compromis entre la résolution du signal résultat et le niveau de lissage souhaité. Les figures 3.5 et 3.6 représentent les résultats de ce filtrage.

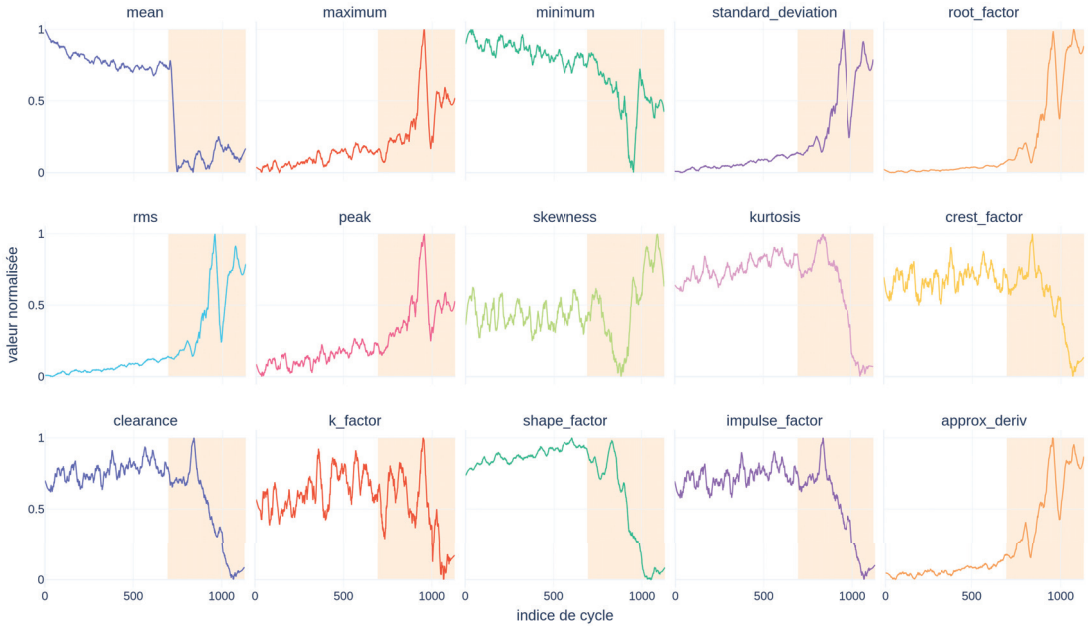


FIGURE 3.5 – Descripteurs statistiques pour l'accélération du vérin LLA en mode de déploiement, lissés

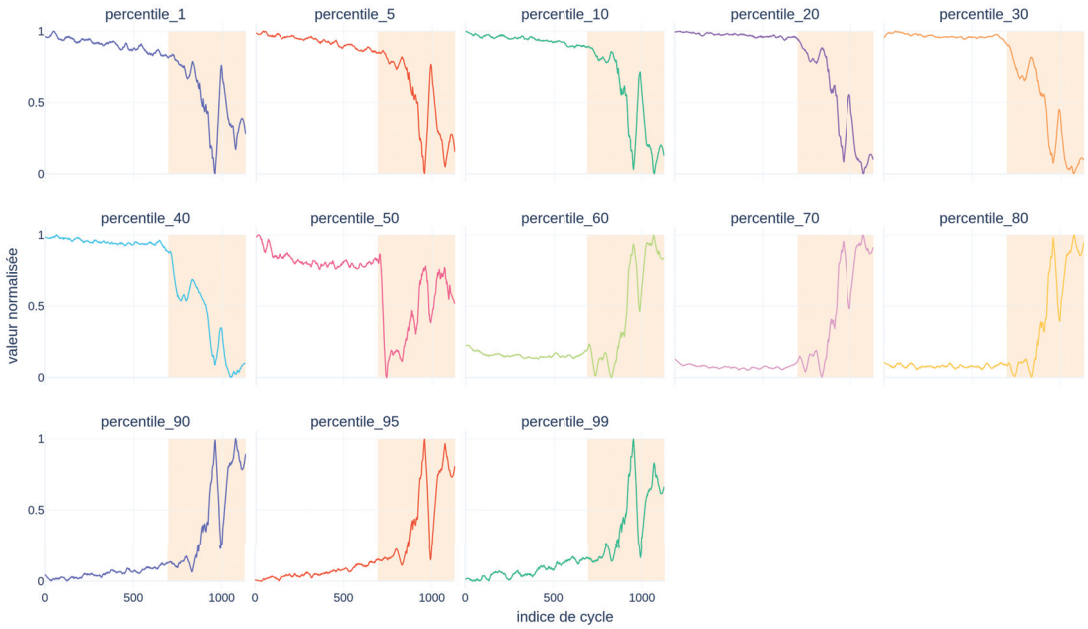


FIGURE 3.6 – Percentiles pour l'accélération du vérin LLA en mode de déploiement, lissés

Les figures 3.5 et 3.6 permettent d'inférer les tendances détectées précédemment. L'intérêt de la multiplication des descripteurs statistiques est ici de pouvoir couvrir toutes les représentations possibles de l'information de défaut présente dans les signaux vibratoires récupérés. À la différence d'un réseau de neurones profond qui extrairait cet ensemble dans ses premières couches, le parti

pris dans ces travaux a été de fournir une première représentation de l'information aux algorithmes suivants d'étiquetage de données, de diagnostic et de pronostic. Même si l'approche consistant à laisser un réseau effectuer sa propre compression de l'information possède de nombreux avantages pour les résultats de classification et régression, elle n'en demeure pas moins très coûteuse en ressources et pour l'instant non adaptée à la problématique de transfert d'informations de vol. En effet, dans l'optique d'un usage opérationnel efficient et donc pour ne pas surcharger la bande passante des canaux de transmission au sol, la valeur de chacun des descripteurs sera calculée en vol et seuls $p = 28$ points seront transférés à la place des relevés de données vibratoires brutes. Remarquons maintenant que, même après une première phase de filtrage, les descripteurs obtenus ne sont pas monotones, aussi leur corrélation avec le vieillissement de l'actionneur n'est pas immédiate. Nous nous attacherons dans ces travaux à développer des algorithmes pouvant extraire une information de diagnostic et de pronostic fiable à partir de ces grandeurs pour étudier la faisabilité de la méthode avec des données d'exploitation industrielles et non des données de modèles.

3.3 Méthode de partitionnement de séries temporelles

3.3.1 Encodage des séries temporelles

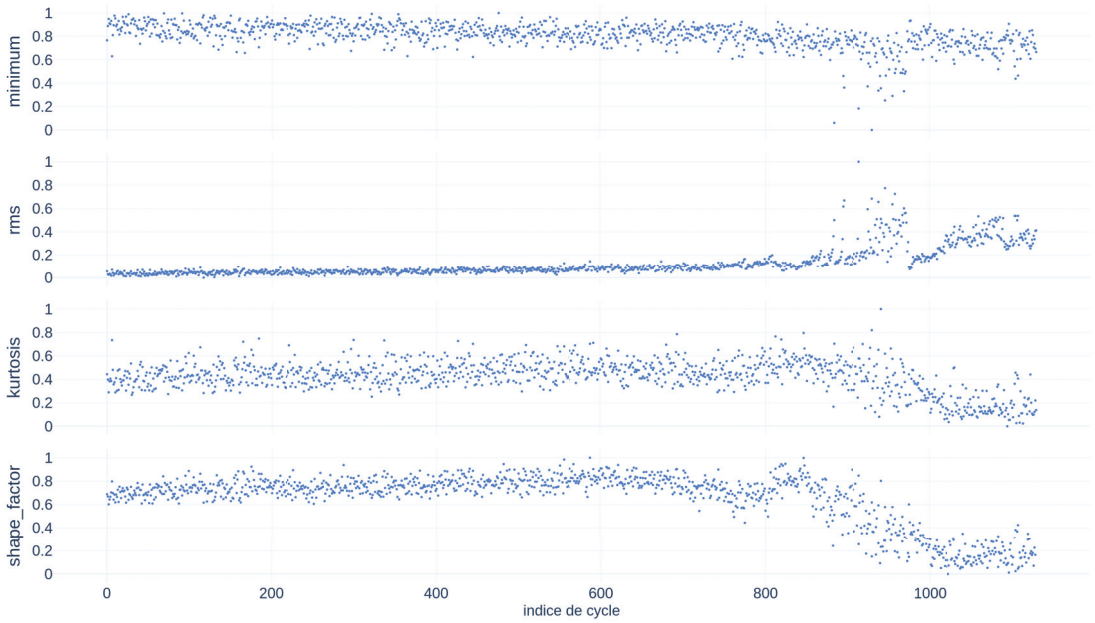


FIGURE 3.7 – $\{\mathbb{X}_1^{\text{LLA}}, \mathbb{X}_6^{\text{LLA}}, \mathbb{X}_9^{\text{LLA}}, \mathbb{X}_{13}^{\text{LLA}}\}$, l'ensemble des quatre descripteurs collectés sur l'actionneur LLA

La section 3.2 a permis d'extraire des données d'exploitation des descripteurs de défauts. Ceux-ci sont des séries temporelles indexées par un horodatage qui, dans ce cas, ne représente pas le temps absolu de captation des données mais l'indice de cycle. Au même titre qu'un horodatage standard, ces séquences obéissent à une relation de précédence commune. Dans cette section, le nombre de descripteurs sera indexé par la variable $1 \leq j \leq p$ avec $p = 28$ avec l'ensemble des descripteurs défini à la sous-section 3.2.2. De plus, l'ensemble des actionneurs étudiés $\{LLA, LRA, URA, ULA\}$ sera indexé par la variable $1 \leq l \leq q$ avec $q = 4$ et l'ensemble des cycles sera indexé par la variable $1 \leq i \leq n$ avec $n = 1130$[1]. Soit une série temporelle $X_j^l = (x_i)_{1 \leq i \leq n}$, X_j^l représente la j^{e} colonne de la matrice $\mathbb{X}_{\mathcal{D}}^l \in \mathcal{M}_{n,p}(\mathbb{R})$ de tous les descripteurs calculés pour le mode de déploiement de l'actionneur l, elle-même contenue dans la base de données $\mathfrak{M}_{n,p,q}(\mathbb{R})$. Notons que j pourra être remplacé par la suite par le nom du descripteur représenté par cet indice, pour clarifier le propos. Il en va de même pour l'indice l. Ainsi, pour $j = 6$, le descripteur de l'actionneur $l = 1$ sera référencé comme $X_{\text{RMS}}^{\text{LLA}}$. La base de données $\mathfrak{M}_{n,p,q}(\mathbb{R})$ contient l'ensemble des matrices des descripteurs calculés pour chaque actionneur pendant un même mode opérationnel de déploiement

[1] Exception faite des variables muettes dans les expressions qui suivront.

de nacelle. Le but de cette section 3.3 est de développer une méthode pour partitionner l'ensemble des séries temporelles X_j^l et, ce faisant, permettre la détermination d'une zone de dégradation dans les données pour améliorer l'estimation faite à la sous-section 3.2.3. Pour illustrer les différentes étapes de l'algorithme, l'ensemble des descripteurs $\{\mathbb{X}_1^{\text{LLA}}, \mathbb{X}_6^{\text{LLA}}, \mathbb{X}_9^{\text{LLA}}, \mathbb{X}_{13}^{\text{LLA}}\}$ présentés à la figure 3.7 seront utilisés dans toute cette section. Chacun des descripteurs est alors normalisé et standardisé entre 0 et 1. Ces descripteurs ont été sélectionnés pour évaluer la méthode sur des signaux qui présentent des dynamiques diverses mais qui restent cohérents pour l'extraction d'une information de surveillance de la santé. De toute évidence, d'après la figure 3.7, les caractéristiques sont plutôt bruyantes et ne sont pas strictement monotones. Cependant, une tendance générale ascendante ou descendante est reconnaissable.

Une méthode de partitionnement (ou *clustering*) réalise la classification de données sans connaissance préalable en minimisant une distance intra-classe et en maximisant une distance extra-classe. Cela permet alors d'agréger les objets aux propriétés similaires et d'éloigner les objets aux propriétés hétérogènes. En respectant ce principe, et dans l'optique de créer une nouvelle méthode de partitionnement, la distance euclidienne entre deux réels telle que définie à l'équation 3.2 est choisie.

$$\forall (x, y) \in \mathbb{R}^2, d(x, y) = \sqrt{x^2 - y^2} \tag{3.2}$$

Pour obtenir une première information de groupement, cette distance est appliquée à chacun des points entre eux des séries $X_{1\leq j\leq p}^l$. Une matrice de distance $\mathbb{G}_j^l \in \mathcal{M}_n(\mathbb{R}_+)$(cas particulier d'une *matrice de Gram*) carrée, positive est alors obtenue. Sa définition est présentée à l'équation 3.3.

$$\mathbb{G}_j^l\left(X_j^l\right) = \begin{bmatrix} d(x_1,x_1) & d(x_1,x_2) & \dots & d(x_1,x_n) \\ d(x_2,x_1) & d(x_2,x_2) & \dots & d(x_2,x_n) \\ \vdots & \vdots & \vdots & \vdots \\ d(x_n,x_1) & d(x_n,x_2) & \dots & d(x_n,x_n) \end{bmatrix} \tag{3.3}$$

Avec la définition de l'équation 3.3, plus les points d'une série temporelle X_j^l sont proches les uns des autres, plus la valeur de leur distance sera faible. Cette dynamique se traduit par des zones de la matrice $\mathbb{G}_j^l$ remplies de valeurs proches de zéro. La matrice de distance étant symétrique, une première information visuelle de regroupement dans les données est obtenue. Avec l'utilisation de la distance euclidienne, des formes carrées aux valeurs quasi nulles seront créées le long de la diagonale de la matrice. Pour illustrer cette méthode d'encodage de l'information, quatre matrices de distances $\{\mathcal{G}_1^{\text{LLA}}, \mathcal{G}_6^{\text{LLA}}, \mathcal{G}_9^{\text{LLA}}, \mathcal{G}_{13}^{\text{LLA}}\}$ sont calculées à partir des signaux de la figure 3.7 et présentées à la figure 3.8.

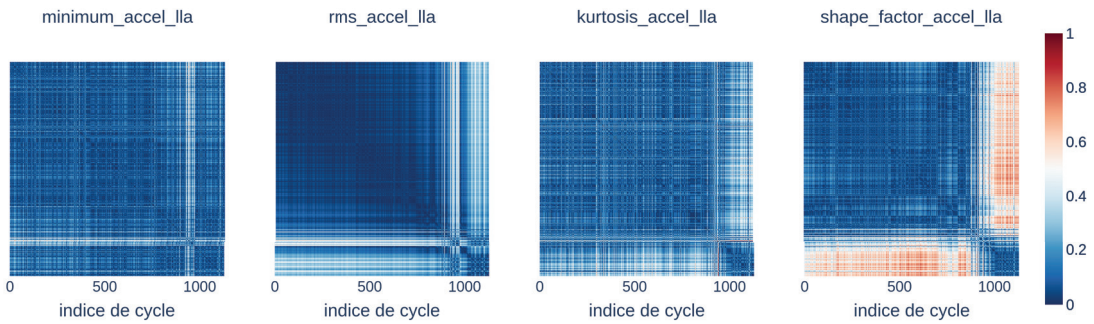


FIGURE 3.8 – Matrices de distance $\{\mathcal{G}_1^{\text{LLA}}, \mathcal{G}_6^{\text{LLA}}, \mathcal{G}_9^{\text{LLA}}, \mathcal{G}_{13}^{\text{LLA}}\}$ extraites de l'ensemble de la figure 3.7 $\{\mathbb{X}_1^{\text{LLA}}, \mathbb{X}_6^{\text{LLA}}, \mathbb{X}_9^{\text{LLA}}, \mathbb{X}_{13}^{\text{LLA}}\}$.

La figure 3.8 représente les matrices de distances associées avec une échelle colorimétrique allant de 0 à 1. Cette représentation permet de constater que les matrices en question possèdent des motifs spécifiques bruités qui pourrait dès lors être étudiés par un traitement d'image. C'est l'approche qui a été choisie dans ces travaux. Chaque matrice de distance créée sera considéré comme une image RGB à un seul canal dont chaque pixel correspond à un élément. Notons que pour la figure 3.8, les images ont été colorisées pour mieux faire ressortir les symétries pour l'œil

humain. Chaque pixel ayant une valeur comprise dans l'intervalle $[0; 1]$, les images représentées à la figure 3.8 sont dans les faits en niveaux de gris. Avec cette échelle colorimétrique, plus le pixel est bleu foncé, plus la distance représentée est faible. Comme indiqué précédemment, les groupes potentiels sont détectés comme des formes symétriques sombres le long de chaque diagonale de l'image. Ces zones représentent alors des intervalles temporels durant lesquels les points de la série ont une valeur proche. En dehors de celles-ci, la distance augmente. Les bandes horizontales et verticales dans les images proviennent, quant à elles, du bruit des données d'entrée. Ainsi, selon les motifs de la figure 3.8, il est déjà possible d'inférer des limites de groupes. La matrice du facteur de forme (`shape_factor`) $\mathcal{G}_{13}^{\text{LLA}}$ posséderait vraisemblablement au moins deux groupes.

Avant de faire le lien entre les groupements de données et notre problématique de surveillance d'état de santé, les hypothèses fondamentales 3 et 4 sont faites.

Hypothèse 3. *Les systèmes étudiés ne peuvent pas se régénérer sans maintenance extérieure.*

Hypothèse 4. *Les systèmes étudiés ne bénéficient d'aucune opération de maintenance.*

Ces deux prédicats sont couramment utilisés en PHM en raison notamment de la nature des données disponibles dans la communauté. Par exemple, le jeu de données Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) de [Saxena et Goebel, 2008] et sa récente amélioration par [Arias Chao *et al.*, 2021] contient des données de fonctionnement jusqu'à défaillance d'un moteur d'avion, sans opération de maintenance entre plusieurs cycles de vol. De plus, l'hypothèse 3 a déjà été clairement formulée dans [Breuneval, 2017, hypothèse 12]. Si l'hypothèse 4 n'est pas vérifiée, il pourrait y avoir deux configurations possibles. Dans le premier cas, si la maintenance rajeunit complètement le système, la méthode s'applique toujours après avoir déplacé l'origine de la série temporelle avant le traitement. Dans le second cas, si le rajeunissement est partiel, alors il est aussi nécessaire d'évaluer la partie résiduelle de la vie utile du produit gagnée après la maintenance. Cette dernière configuration n'entre pas dans le cadre de ces travaux.

Ainsi, avec les deux hypothèses 4 et 3, chaque groupe trouvé dans les données peut être associé à un degré de sévérité de défaut. Comme le partitionnement est effectué sur des séries temporelles, chaque nouveau groupe augmentera la valeur de sévérité affectée au défaut détecté. Les matrices de distance définies dans l'équation 3.3 permettent l'extraction d'informations de partitionnement pertinentes. Comme le montre la figure 3.8, les données industrielles traitées produisent des images très bruitées et les techniques traditionnelles de traitement d'images testées n'étaient pas efficaces pour améliorer le rapport signal sur bruit des matrices. C'est pourquoi une approche à base d'apprentissage profond couplée à des techniques de traitement du signal a été choisie. Comme énoncé précédemment, les matrices de distances sont considérées comme des images dont le contraste sera à améliorer.

3.3.2 Segmentation sémantique avec réseau de neurones profond

Dans la sous-section 3.3.1, les séries temporelles en une dimension ont été encodées sous forme d'images. Pour détecter les différents groupements de données dans les images, le principe de segmentation sémantique est utilisé. Initialement introduit par [Zucker, 1977] puis formalisé par [Fu et Mui, 1981], il consiste à associer chacun des pixels d'une image à un ensemble. Dans notre cas, cet ensemble est une suite de labels représentant un degré de sévérité de défaut. Le concept de détection de forme adapté aux données précédentes s'effectue en deux étapes :

- Premièrement, sélectionner une valeur pour chaque classe et attribuer ces dernières valeurs à chaque pixel de l'image. Par exemple, la valeur 0 a été attribuée aux formes carrées recherchées et 1 à l'arrière-plan.

- Secondement, construire une structure pour révéler les différentes formes carrées et éliminer le fond bruyant des images, afin de conserver ainsi uniquement les informations pertinentes pour le partitionnement.

Pour réaliser ce dernier point, les réseaux de neurones profonds sont utilisés. Cependant, un inconvénient majeur de ces architectures réside dans le besoin d'avoir une quantité importante de données d'entraînement. Dans l'industrie, et plus particulièrement dans l'aéronautique, les données d'entraînement pertinentes sont rares et coûteuses à obtenir. Afin de pallier cette limitation,

[Ronneberger, Fischer et Brox, 2015] ont développé une architecture, U-NET, pour la segmentation d'images cellulaires, spécialisée dans les petits échantillons de données. Notons que la dimension de l'ensemble de données doit être comparée aux ensembles habituels utilisés pour l'entraînement des architectures de réseaux neuronaux de l'état de l'art. Pour donner un ordre de grandeur, le jeu de données Common Object in Context (COCO) de Microsoft [T.-Y. Lin *et al.*, 2015] et ses 3×10^6 images pourrait être cité.

Le réseau U-NET est composé de deux parties principales. La partie gauche de la configuration compresse les données d'entrée et en extrait des caractéristiques (*features*). Ces *features* représentent, dans un sens, d'autres descripteurs de données. Les caractéristiques les plus profondes de toutes sont ensuite passées à travers un *goulot d'étranglement*. Puis le résultat de cet étage passe par un ensemble de couches qui décompresse l'information d'entrée. Enfin, en fonction du nombre de classes à segmenter, une couche *sigmoïde* ou *softmax* est utilisée pour transformer chaque valeur de pixel d'entrée traitée en probabilité d'appartenance à une classe. Chaque bloc comporte deux couches de convolution, pour la partie descendante, associées à une activation *ReLU* se terminant par une *MaxPooling*. Pour la partie ascendante, on utilise la structure symétrique faite de *ConvTranspose2d*, une opération de convolution inverse pour sur-échantillonner son entrée, comme développé dans *Pytorch* [Paszke *et al.*, 2019]. Notons qu'à chaque étage de l'architecture, une connexion résiduelle permet d'améliorer les résultats de segmentation en propageant à chaque couche de décompression le résultat de la couche lui faisant face. De ce fait, une information moins compressée est concaténée à l'information qui remonte le réseau, celle-ci, possédant moins de détails car ils ont disparu dans les convolutions successives. L'ajout de connexions résiduelles dans les architectures profondes a été introduit par [He *et al.*, 2015] pour faciliter l'apprentissage de structures avec beaucoup de couches et pallier notamment le problème de gradient évanescent (*vanishing gradient*). Une illustration de U-NET utilisé a été réalisée par [Iqbal, 2020] et présentée à la figure 3.9.

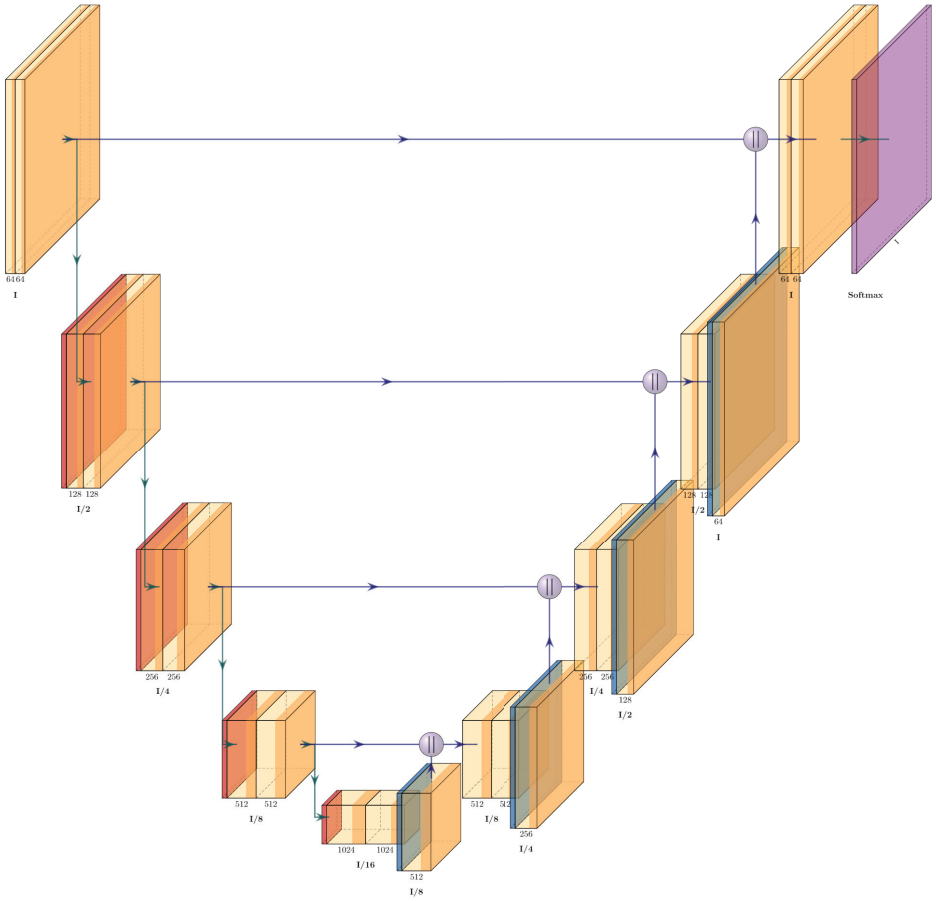


Figure 3.9 – Architecture du réseau de segmentation profond U-NET représenté à l'aide des algorithmes de [Iqbal, 2020]

Le but de cette architecture est de segmenter chaque matrice de distance $\mathcal{G}_j^l$ avec $1 \leq j \leq p$ et $1 \leq l \leq q$. Comme indiqué précédemment, le réseau doit donc apprendre à reconnaître des motifs carrés le long de la diagonale de chacune de ses entrées. C'est avec cet objectif en tête que la procédure d'apprentissage sera définie.

Comme de nombreux environnements industriels, l'aéronautique manque d'une quantité appropriée de données réparties de manière équilibrée entre les différentes classes de fonctionnement et qui pourraient servir de base d'entraînement. Pour pallier ce problème, une base de données d'apprentissage complètement artificielle a été générée, permettant ainsi l'apprentissage de U-NET pour segmenter les précédentes matrices de distances. D'après les données de la figure 3.8, entre autres, le constat peut être fait que la structure symétrique des matrices de distances peut être modélisée, en première approximation, par un ensemble de carrés le long d'une diagonale. Pour produire ce type de schéma avec des séries temporelles, il suffit de calculer la matrice de distance d'une fonction constante par morceaux ou Constant Continuous Piecewise Function (CCPF). En raffinant le modèle de génération de données petit à petit, des matrices à partir de fonctions linéaires continues par morceaux ou Linear Continuous Piecewise Function (LCPF) sont créées. Puis l'ajout d'un bruit blanc gaussien sur l'image résultante pour simuler le bruit des données est effectué. Un échantillon des différents candidats créés pour la base de données d'apprentissage est donné par la figure 3.10.

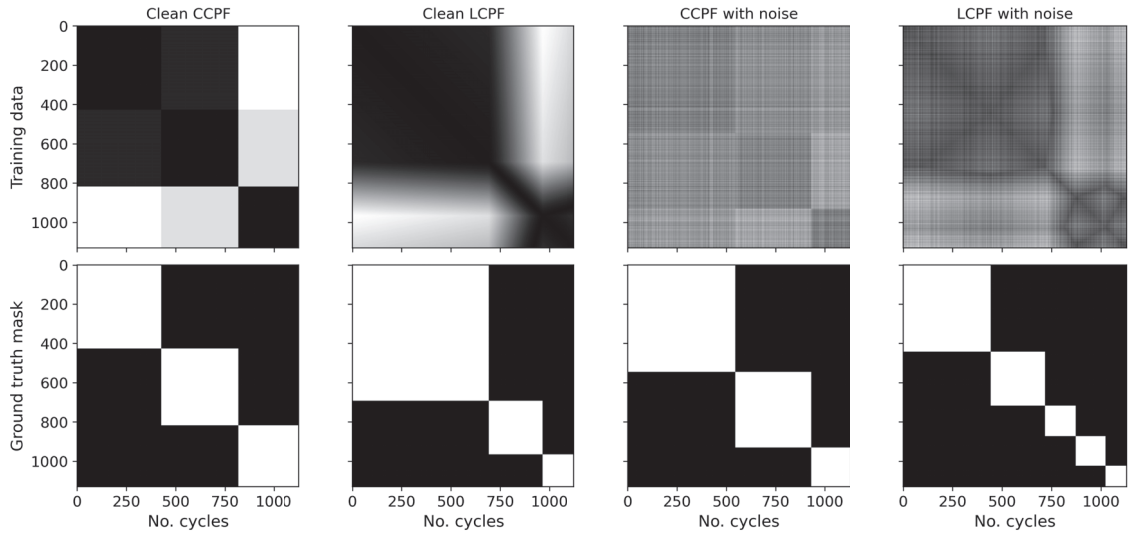


FIGURE 3.10 – Matrices de distance représentées sous forme d'images bruitées avec plusieurs configurations, utilisées pour la formation U-NET avec le gabarit cible associé, d'après [EID *et al.*, 2021, figure 2]

Les images de la figure 3.10 ont été créées de manière aléatoire en calculant les matrices de distance des fonctions CCPF, des fonctions LCPF, et en ajoutant un bruit gaussien aux derniers résultats. La première ligne de la figure 3.10 représente les données d'apprentissage à fournir en entrée du réseau, tandis que la seconde ligne représente le masque associé à chacune de ces entrées. Une fois que l'image d'entrée a traversé le réseau, le résultat est comparé au masque (*ground truth*) associé. Les poids du réseau sont alors mis à jour par rétro-propagation du gradient en fonction du résultat de comparaison. Dans ces travaux, la meilleure procédure d'apprentissage a été déterminée de manière empirique. Tout d'abord, les poids du réseau sont initialisés de manière aléatoire. Un premier apprentissage est ensuite effectué avec une base de données constituée uniquement de matrices de distances de fonctions CCPF. Une fois que la fonction de perte converge vers un minimum et se stabilise, la base d'apprentissage est modifiée pour ne contenir que des matrices aux limites floues créées à partir de fonctions LCPF. L'opération est ensuite répétée, et un bruit gaussien est ajouté à chaque entrée pour augmenter encore la capacité de généralisation du réseau neuronal. Enfin, un nouvel ensemble de données est généré avec un biais spécifique au domaine afin de favoriser la convergence de la formation vers l'objectif de l'étude. Ce biais est le corollaire de l'hypothèse 5.

Hypothèse 5. *La dégradation d'un système ne peut que s'aggraver au cours du temps car le processus de vieillissement a un effet cumulatif.*

De même que les hypothèses 3 et 4, l'hypothèse 5 est couramment faite dans la littérature. Dans [Arias Chao *et al.*, 2021], plusieurs modèles de dégradation ont été appliqués aux données Run To Failure (R2F). Le modèle anormal tend à représenter le comportement de plusieurs modèles connus de propagation des dommages, comme par exemple les lois d'Arrhenius ou de Coffin-Manson.

Par conséquent, le dernier lot de données d'entraînement est généré avec une contrainte : les carrés le long de la diagonale de la matrice doivent avoir des dimensions décroissantes. Ce qui se traduit au niveau des séries temporelles artificielles générées comme une décroissance des intervalles de rupture de pentes au cours du temps. De par le changement des données d'apprentissage au fur et à mesure du processus d'optimisation des poids du réseau, cette méthode peut être considérée comme une adaptation de domaine. Au total, ce sont quatre processus d'apprentissage différents qui ont été réalisés. À chaque nouvelle étape, l'état initial du réseau était cependant modifié et non plus déterminé de manière aléatoire. Cela a permis d'aider à la convergence de l'architecture avec des ressources limitées.

De manière plus détaillée chaque jeu de données artificielles contenait 3 000 images. Quelques échantillons des différents ensembles ont été représentés à la figure 3.10. Pour chaque nouvelle optimisation, la base de données considérée a été divisée en 2 250 images d'entraînement et 750 images pour constituer l'ensemble de validation. La répartition des éléments dans l'un ou l'autre des ensemble a été effectuée de manière aléatoire. Notons que, dans cette procédure pour trouver les poids optimaux du réseau, seuls les deux ensembles d'apprentissage et de validation sont générés à partir de données artificielles. L'ensemble de tests pour vérifier le bon comportement du réseau avec des données inconnues sera constitué des données réelles d'exploitation que sont les signaux vibratoires captés sur chacun des actionneurs. Pour ne pas confondre la définition de l'ensemble de test avec l'ensemble de validation, comme c'est souvent le cas dans la littérature [Kuhn et Johnson, 2013], les définitions de [Ripley, 1996] seront adoptées ici. Cependant, la particularité de l'ensemble de validation est qu'il n'est pas labellisé. En effet, le but de la méthode est justement d'assigner des labels sur des données inconnues. C'est pourquoi la performance du réseau sur l'ensemble de validation, et donc les données d'exploitation, ne sera pas quantifiée ici. Ainsi, en comptant les quatre bases de données différentes utilisées, cela représente un total de 12 000 images générées pour l'entraînement de U-NET. Notons qu'à partir du 3^{e} ensemble, 10 % de bruit blanc a été ajouté aux images des matrices de distances. La descente de gradient a été effectuée avec l'algorithme *Adam* [Kingma et Ba, 2017]. Pour la sélection du taux d'apprentissage, l'ordonnanceur adaptatif *ReduceLROnPlateau* du paquet *optim.lr_scheduler* [Paszke *et al.*, 2019] avec une valeur initiale de 1×10^{-4} a été sélectionné. Après avoir contrôlé la phase d'apprentissage de U-NET avec *tensorboard* [Abadi *et al.*, 2015], 200 époques ont été suffisantes pour atteindre une précision stable pour chaque ensemble de données. De plus, chaque apprentissage a été réalisé par lot. La taille de ces derniers étant limitée par la quantité de Video Random Access Memory (VRAM) des Graphics Processing Unit (GPU) utilisés. Pour des images de dimension 1130×1130 avec une quantité de VRAM totale de 48 GB, un lot était constitué de quatre images. Enfin, la fonction binaire d'entropie croisée avec fonction de coût logistique (*BCEWithLogitsLoss*) de la librairie *torch.nn* [Paszke *et al.*, 2019] a été utilisée comme fonction de coût. Notons qu'une fois cette quadruple phase d'apprentissage faite, le réseau de neurones est utilisé tel quel pour les étapes d'inférence qui permettront d'assigner un label à chaque point d'une série temporelle. Elle est effectuée une seule fois dans la méthode.

Utiliser une image de l'ensemble de test pour déterminer le comportement du réseau à l'issue des phases d'apprentissages successives permet d'obtenir la figure 3.11. Celle-ci met en exergue l'amélioration du résultat de segmentation au cours des différentes étapes d'un apprentissage *récursif*. Ainsi, sur la figure 3.11, le première image en partant de la gauche représente une matrice de distance non segmentée. La deuxième est le résultat de segmentation de notre U-NET ayant appris a segmenter des images de fonctions CCPF pendant 150 époques. Cet apprentissage a ensuite été renforcé par des fonctions LCPF de même pendant 150 époques, le résultat est fourni à la troisième figure. Puis, le quatrième motif représente l'apprentissage réseau renforcé par des données contenant 10 % de bruit gaussien sur une dimension, de même pendant 150 époques. L'avant dernière image présente, elle, le résultat de segmentation après 200 époques de plus avec des données bruités sur les deux dimensions (motifs horizontaux et verticaux). Enfin, le fait de rajouter 200 époques de plus avec d'autres données d'apprentissage bruités n'améliore pas significativement les résultats de segmentation obtenus. C'est ce qui est constaté avec la sixième image de la figure 3.11.

Notons que pendant tous ces apprentissage la taille des lots de données fournis au réseau est de 4. Au-delà, les images de taille 1130×1130 saturaient la VRAM des processeurs graphiques utilisés.

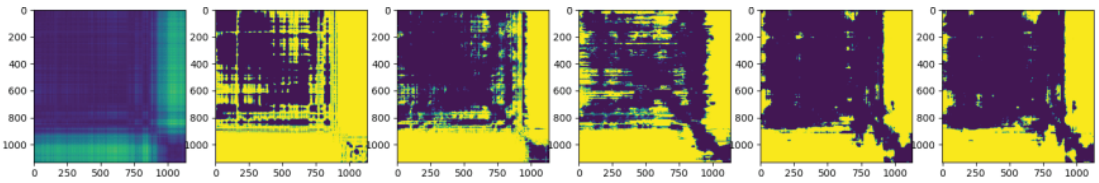


FIGURE 3.11 – Résultats d'apprentissages successifs sur une matrice de distance de l'ensemble de test

Chaque apprentissage durait environ 38 heures sur des ressources High Performance Computing (HPC) munies d'un bi-Intel Xeon Silver 4215R, une bi-NVIDIA Quadro RTX 6000 avec 24 GB de GDDR6 chacune et 192 GB de Random Access Memory (RAM). Au total ce sont alors environ 190 heures qui ont été nécessaires à la convergence du modèle, ce qui représente à peu près 8 jours de calculs cumulés. Cet apprentissage permet de produire les images binaires de la figure 3.12.

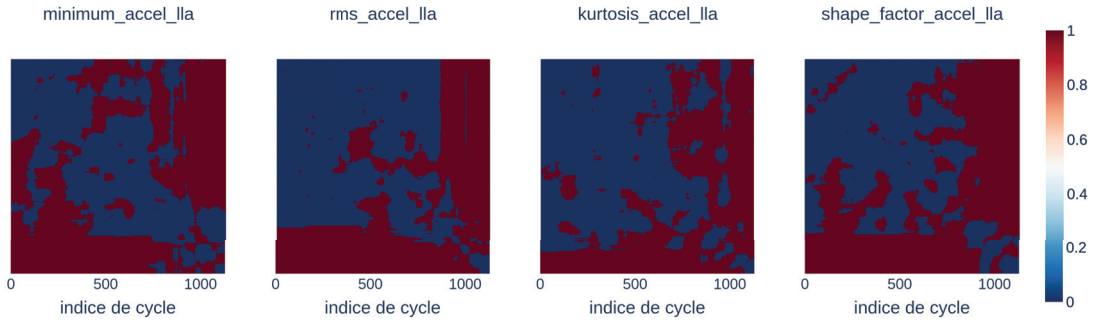


FIGURE 3.12 – Ensemble $\{\mathcal{G}_1^{\text{LLA}}, \mathcal{G}_6^{\text{LLA}}, \mathcal{G}_9^{\text{LLA}}, \mathcal{G}_{13}^{\text{LLA}}\}$ des matrices de distances segmentées par U-NET

D'après la figure 3.12, il apparaît que la phase d'apprentissage de U-NET pourrait être améliorée. Les résultats de segmentation sont toujours bruités, c'est pourquoi les étapes suivantes de la méthode s'attacheront à extraire une information pertinente de ces images.

3.3.3 Extraction d'un signal de frontières

Maintenant que U-NET a été entraîné, il peut produire des images segmentées à partir des matrices de distance. Néanmoins, comme le montre la figure 3.12, les résultats obtenus possèdent encore du bruit, ce qui ne permet pas de détecter directement les frontières de groupes de données à partir de ces derniers résultats. Par conséquent, d'autres procédures de traitement du signal doivent être appliquées pour en extraire un élément d'information pertinent pour le partitionnement.

En reprenant les masques d'apprentissage présentés à la figure 3.10, le motif parfait à détecter dans les données est symétrique par rapport à l'anti-diagonale de la matrice considérée. Aussi, les algorithmes de traitement d'image permettant l'extraction de frontières de groupes a pour objectif de détecter les discontinuités entre motifs (en blanc sur les masques de la figure 3.10). Les résultats de segmentation obtenus (voir figure 3.12) étant bruités, l'extraction d'un signal de frontière se fera en plusieurs étapes. La première étape consiste à extraire une première information à partir de laquelle pourra être construit le signal de frontière discret final. Pour ce faire, les images résultantes sont parcourues selon leur anti-diagonale, ce qui permet de créer une séries de valeurs. La moyenne de cette série est alors calculée. La moyenne a été choisie ici en raison des valeurs des pixels de sortie. En effet les images produites par le réseau sont binaires. Si un pixel a été jugé comme appartenant à un groupe alors sa valeur est nulle. Rappelons alors qu'un groupe est représenté par un intervalle dans les données d'entrée dans lequel les ordonnées de points sont proches au sens d'une mesure de distance. Dans le cas contraire, sa valeur est unitaire et le pixel n'appartient pas à un motif mais au fond de l'image. Ainsi, utiliser la valeur moyenne sur les anti-diagonales de chaque matrice résultante de la segmentation permet de détecter des groupements de données. Lorsqu'un groupe est détecté, la moyenne tendra vers la valeur unitaire tandis que le cas échéant,

elle tendra vers la valeur nulle. La figure 3.13 illustre visuellement l'algorithme utilisé pour extraire une première information des frontières de groupes. Notons que le terme "anti-diagonale" utilisé précédemment fait référence à la direction de chaque flèche rouge.

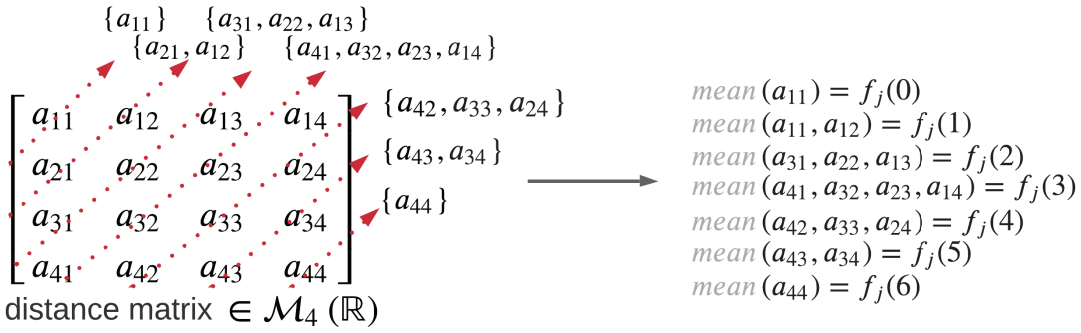


FIGURE 3.13 – Schéma de principe pour l'extraction de la frontière des groupes — phase 1 : moyenne sur les anti-diagonales

Ce schéma de principe illustre l'algorithme pour une matrice carrée de taille 4. La série de points récupérés est alors de taille croissante et le calcul de la moyenne sur chaque ligne permet d'obtenir $2 \times 4 - 1 = 7$ points différents pour former un premier signal intermédiaire. La généralisation de cette procédure est donnée par l'algorithme 1.

Algorithme 1 Extraction de la frontière du groupe

Entrées :
M est une matrice carrée de taille n
Sortie :
frontierSig est le premier signal de frontière extrait

```
1:  function Indices Matrix Elements(n)
2:      /* Initialiser un tableau de taille 2n − 1 */
3:      declare matrixIndices of [1; 2n − 1] arrays
4:      /* Récupérer les indices triangulaire supérieures gauche*/
5:      for i ← 1 to n do
6:          for j ← 1 to i do
7:              matrixIndices[i].append([i − j + 1; j])
8:          end for
9:      end for
10:     /* Récupérer les indices triangulaire inférieure droit*/
11:     for i ← n − 1 to 1 do
12:         for j ← i to 1 do
13:             matrixIndices[2n − i].append([j + 1; i − j + 2])
14:         end for
15:     end for
16: return matrixIndices
17: end function
18:
19: function Intermediate Frontier Signal(M)
20:     declare frontierSig de [1; 2n − 1] floats
21:     /* Initialiser une matrice carrée*/
22:     n ← size(M)
23:     matrixIndices ← Indices Matrix Elements(n)
24:     for i ← 1 to 2n − 1 do
25:         frontierSig[i] ← mean(M[matrixIndices[i]])
26:     end for
27: return frontierSig
28: end function
```

L'algorithme 1 est constitué de deux fonctions principales. La première, INTERMEDIATE FRONTIER SIGNAL, prend en entrée une matrice de taille n et produit le signal résultant, représentant une première approximation des frontières de groupes. Pour ce faire, elle fait appel à une seconde fonction : INDICES MATRIX ELEMENTS. Cette dernière fonction, générique, construit un objet contenant les couples d'indices (i, j) des éléments d'une matrice carré quelconque de taille n. Ici, i représente l'indice des lignes tandis que j représente l'indice des colonnes. L'objet `matriceIndices` est de taille $2n-1$ parce que, comme intuité avec la figure 3.13, le fait de parcourir une matrice sur l'anti-diagonale génère deux fois plus de lignes que sa dimension, soit $2n$ lignes. Pour ne pas compter la série de taille maximale deux fois, il est alors nécessaire de soustraire une unité à la dimension finale. C'est pourquoi aux lignes 5 à 9 le parcours de la partie supérieure gauche est effectué en n itérations alors qu'aux lignes 11 à 15 le parcours de la partie inférieure droite est effectuée en $n-1$ itérations. Cet ensemble d'indices est utilisé aux lignes 24 à 26 pour calculer la moyenne sur l'ensemble des points obtenus, ce qui permet la création du signal résultant `frontierSig`. Les résultats de l'algorithme sont présentés à la figure 3.14 à partir des signaux utilisés précédemment.

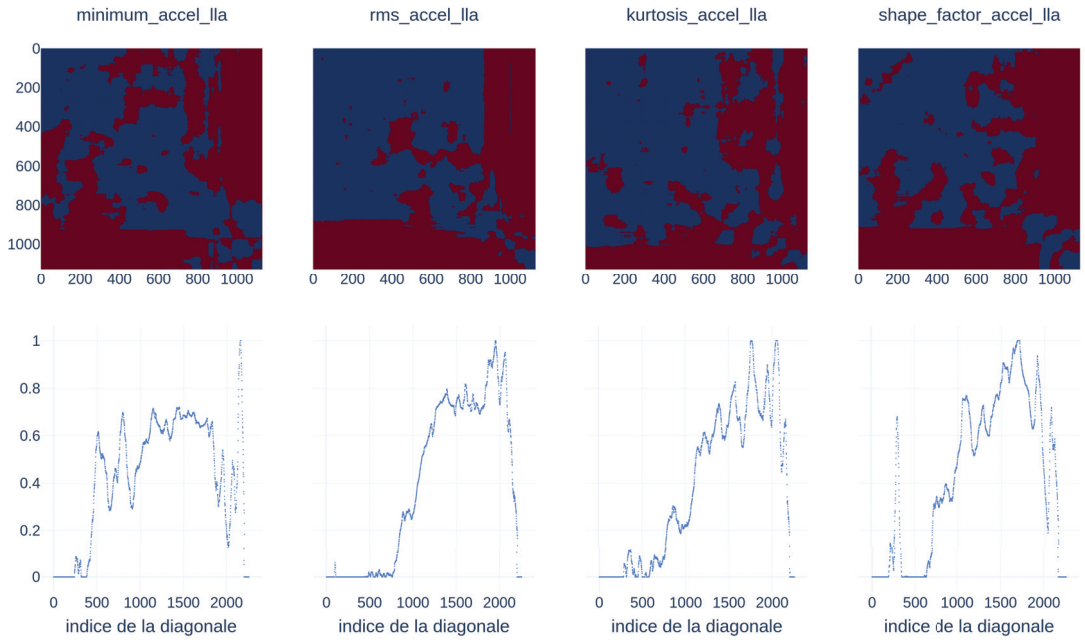


FIGURE 3.14 – Moyenne des valeurs (seconde ligne) des matrices de distance segmentées de la figure 3.12 (première ligne) sur chaque anti-diagonale

Les deux séries d'images ont été représentées en vis-à-vis pour faciliter l'interprétation. Par exemple, en considérant le facteur de forme `shape_factor_accel_lla`, deux pics dans le signal apparaissent aux 218^{e} et 299^{e} cycles. Cette dynamique signifie que dans un groupe, la présence de fond de l'image a été détectée. En effet, il correspondent aux deux motifs bordeaux situés en haut à gauche de l'image correspondante. Enfin, comme vu avec l'algorithme 1, les images segmentées étant de taille 1130×1130, le signal de frontière temporaire obtenu est de taille $2 \times 1130 - 1$, soit composé de 2 259 points.

À partir de ces signaux, la détermination d'une frontière de groupe n'est pas immédiate. D'après le choix de l'algorithme précédent, une frontière sera détectée lorsque le signal passera d'une valeur proche de 0 à une valeur proche de 1 et donc que l'image correspondante n'aura que des pixels bordeaux sur une de ses anti-diagonales. Cependant, cette condition n'est pas suffisante pour extraire une information pertinente. En effet, pour de petits groupes de données qui représentent l'accélération du vieillissement de l'actionneur comme supposé à l'hypothèse 5, la moyenne obtenue sera fortement déviée vers la valeur unitaire. C'est ce qui est remarqué avec la figure 3.14 où les signaux intermédiaires obtenus sont, en grande approximation, plutôt croissants. C'est pourquoi faire ressortir une information de frontière requiert une autre procédure non triviale pour détecter les *extrema* locaux de chaque signal. Chaque pic de fonction créé représente alors le contour probable d'un motif carré dans l'image segmentée. Pour extraire des *extrema* d'un signal, les méthodes de filtrage standard nécessitent néanmoins un réglage non trivial de paramètres. Entre autres choses, la fenêtre glissante qui représente le nombre de points auxquels l'algorithme de filtrage sera appliqué

simultanément est un paramètre important à fixer pour ajuster le comportement de l'algorithme de filtrage. Elle contrôle la sensibilité de la procédure de filtrage aux dynamiques locales et globales. Pour éviter d'introduire dans la méthode un hyperparamètre de plus, un algorithme itératif a été développé. Il consiste à parcourir en boucle le signal à filtrer en calculant une statistique sur une fenêtre de points. Cette dernière taille de fenêtre est, quant à elle, accrue à chaque itération de la boucle. La figure 3.15 représente visuellement cet l'algorithme pour un signal avec 7 échantillons.

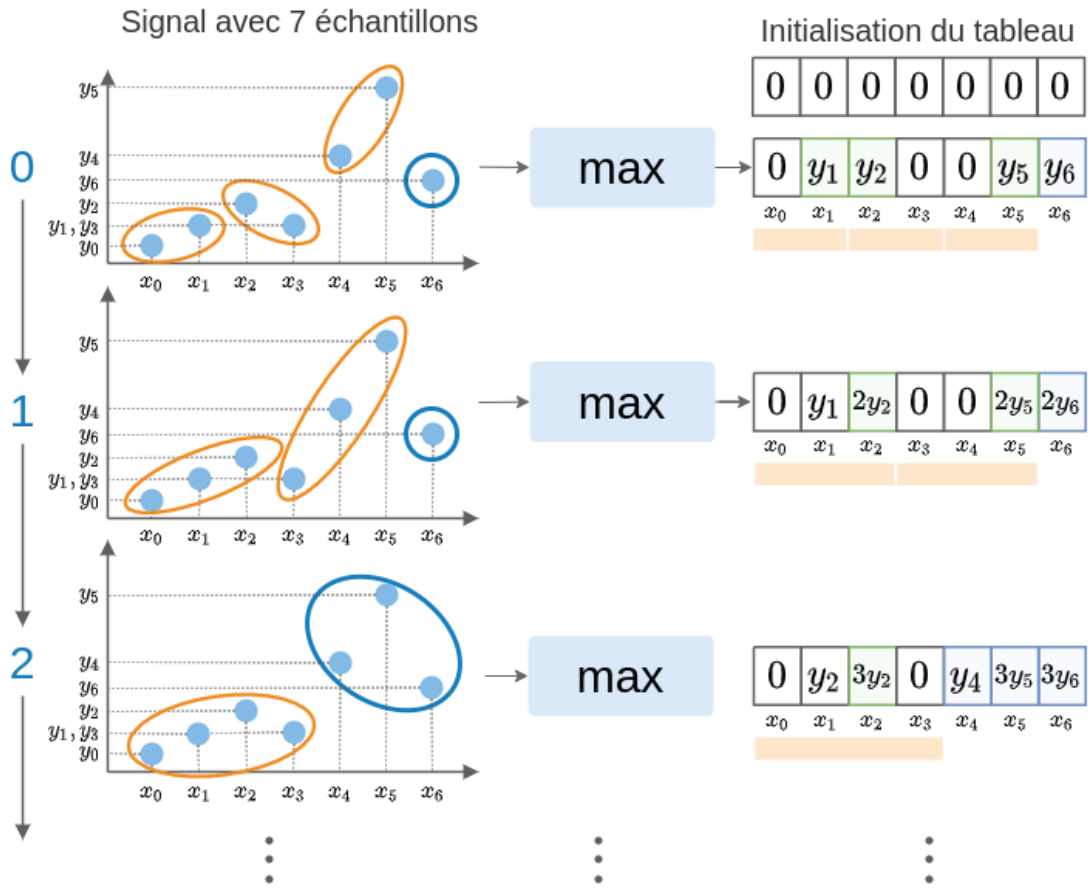


FIGURE 3.15 – Schéma de principe pour l'extraction de la frontière des groupes — phase 2 : détection des pics par lots

.

L'idée de cette procédure est de diviser le signal d'entrée en plusieurs ensembles de taille croissante tout au long du processus. L'algorithme commence par choisir une fenêtre de taille minimale, donc de taille 2. Au fur et à mesure des itérations, la taille de la fenêtre est incrémentée jusqu'à atteindre sa taille maximale n. Puisque les limites des groupes sont détectées pour des valeurs approchant 1, le maximum est calculé sur chaque fenêtre locale. Pour recueillir ces valeurs, un tableau de taille $2n-1$ est initialisé à 0 avant le début de la boucle. Ce dernier stocke la valeur maximale de chaque point à l'endroit de son index respectif dans le signal d'entrée. Chaque fois qu'un maximum est trouvé localement, il est ajouté au tableau. Par conséquent, pendant les itérations initiales, l'algorithme amplifie les *maxima* locaux, alors qu'à la fin, il a tendance à amplifier les *maxima* plus globaux. Pour plus de détails sur la façon de réaliser cette opération, l'algorithme 2 est donné.

L'algorithme 2 est constitué de la fonction BATCH PEAK DETECTION qui prend en entrée un signal quelconque. Dans un premier temps, la structure `resArray` est créée comme un tableau de même taille que le signal d'entrée et initialisée avec des valeurs nulles. Ensuite, une structure `noWindowList` contenant toutes les tailles de fenêtres glissantes est créée aux lignes 8 à 15. Notons qu'une autre structure, `remainingSizeList` contient pour chaque fenêtre glissante les points restants du signal d'entrée lorsque la taille de la fenêtre n'est pas multiple de la taille du signal. La convention adoptée dans cet algorithme est d'ajouter aux tableaux de valeurs finales l'ensemble des points restants du signal d'entrée. Les lignes 16 à 43 permettent alors de vectoriser les calculs

pour optimiser la complexité de l'algorithme. Finalement, le tableau résultant `resArray` est mis à jour aux lignes 45 à 47.

Algorithme 2 Extraction de la frontière du groupe - phase 2 : Détection des pics par lots

```
1:  function Batch Peak Detection(signalArray)
2:      n ← size(signalArray)
3:      declare resArray de n floats
4:      /* Initialiser le tableau contenant le signal résultant à 0 */
5:      for i ← 1 to n do
6:          resArray[i] ← 0
7:      end for
8:      declare noWindowList de n entiers
9:      declare remainingSizeList de n entiers
10:     /* Créer un tableau avec toutes les fenêtres et les tailles restantes */
11:     for i ← 2 to n do
12:         noWindowList[i] ← i
13:         /* % représente l'opérateur modulo */
14:         remainingSizeList[i] ← n % i
15:     end for
16:     for i ← 0 to n − 1 do
17:         noWindow ← n // i
18:         remainElmt ← remainingSizeList[i]
19:         if remainElmt ≠ 0 then
20:             tmpArray ← signalArray[: −remainElmt].reshape(noWindow, i)
21:         else
22:             tmpArray ← signalArray.reshape(noWindow, i)
23:         end if
24:         declare indexStatValueList un tableau d'objets de taille inconnue
25:         declare indexMaxArray un tableau de flottants de taille inconnue
26:         for j ← 0 to noWindow do
27:             indexMaxArray ← arg( tmpArray[i, :] == max(tmpArray[i, :]))
28:             for k ← 0 to size(indexMaxArray) do
29:                 /* Conserver l'indice des maxima trouvés cf. principe d'une liste chaînée */
30:                 indexStatValueList.append([j, indexMaxArray[k])
31:             end for
32:         end for
33:         declare idxMaskArray de taille m × l entiers
34:         for j ← 1 to m × l do
35:             idxMaskArray[j] ← 0
36:         end for
37:         idxMaskArray.reshape(m, l)
38:         /* Créer un masque */
39:         for k, p ← [1;noWindow] × [1; i] do
40:             idxMaskArray[k, p] ← 1
41:         end for
42:         Apply idxMaskArray to tmpArray
43:         Unfold tmpArray de taille (noWindow, i) to (1, n)
44:         /* Accumuler les valeurs maximales à leur indice d'apparition pour le résultat */
45:         for j ← [1; m] do
46:             resArray+ = tmpArray
47:         end for
48:     end for
49:     return resArray
50: end function
```

Notons que l'algorithme 2 ne modifie pas la taille du signal d'entrée : aussi le signal filtré est toujours de taille $2n - 1$ à l'issue du filtrage. Pour pallier ce problème et revenir à une échelle de temps représentative du cycle, l'échelle des abscisses du signal résultant est divisée par un facteur

deux. Cela permet de compenser la dilatation générée par la méthode expliquée dans l'algorithme 1. Ce faisant, nous supposons qu'une incertitude de deux cycles est tolérée sur les frontières de groupes dans le cadre de ces travaux de surveillance d'état de santé d'actionneurs électromécaniques. Les signaux résultants sont notés $F_j^l \in \mathcal{M}_{n,1}(\mathbb{R})$.

Remarque. *Jusqu'ici, l'introduction des deux procédures 1 et 2 n'a nécessité le choix d'aucun seuil ou d'hyperparamètre.*

En appliquant l'algorithme 2 de la section à l'ensemble $\{\mathcal{G}_1^{\text{LLA}}, \mathcal{G}_6^{\text{LLA}}, \mathcal{G}_9^{\text{LLA}}, \mathcal{G}_{13}^{\text{LLA}}\}$, les premières frontières discrètes potentielles de groupes sont obtenues à la figure 3.14.

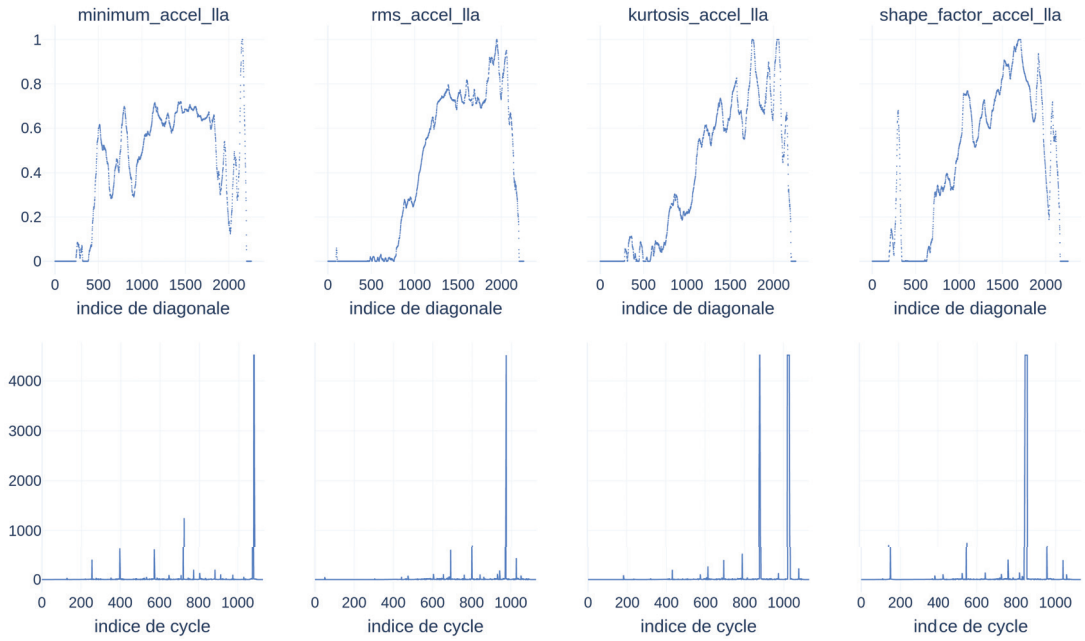


FIGURE 3.16 – Extraction de frontières de groupes préliminaires (seconde ligne) à partir des signaux calculés à la figure 3.14

La figure 3.16 met en exergue différentes frontières de groupes pour chacun des descripteurs étudiés. Pour le `minimum_accel_lla`, le `rms_accel_lla` et le `shape_factor_accel_lla`, une frontière est clairement définie respectivement aux 1078$^{\text{e}}$, 974$^{\text{e}}$ et entre le 842 et 855$^{\text{e}}$ cycles. Tandis que pour le `kurtosis_accel_lla`, deux frontières probables sont définies au 877$^{\text{e}}$ et entre le 1020$^{\text{e}}$ et 1030$^{\text{e}}$ cycles. Ces résultats sont attendus et cohérents au regard des motifs de la figure 3.8 assez bruités. En effet, sur ces derniers, la variation de groupes peut visuellement être détectée essentiellement en bas à droite de chaque image et, donc, vers des numéros de cycles plus importants. Ce résultat est de plus cohérent avec la physique du vieillissement sans régénération des actionneurs.

Maintenant qu'un ensemble de signaux de frontières intermédiaires $(F_j^l)_{1\leq j\leq p}^{1\leq l\leq q}$ a été déterminé, il reste à fusionner l'ensemble de ces informations pour obtenir un profil de frontières de groupes avec une mesure d'incertitude. Cette dernière partie est l'objet de la sous-section 3.3.4 suivante.

3.3.4 Obtention de frontières de groupes avec mesure d'incertitude

À ce stade de la méthode, $p \times q$ séries temporelles ont été transformées en $p \times q$ matrices de distance $\mathbb{G}_j^l$ qui ont été segmentées par le réseau neuronal de la sous-section 3.3.2. À partir de chaque image de sortie, $p \times q$ premiers candidats de frontières de groupes sont obtenus et chacun d'entre eux est filtré par l'algorithme de détection de pic par lot 2. Ainsi, $p \times q$ nouvelles séries temporelles F_j^l sont créées. C'est à partir de ces séries temporelles que le signal de frontières de groupes final sera calculé. Dans un premier temps, celles-ci sont concaténées dans une matrice $\mathbb{F}$ comme représenté à l'équation 3.4.

$$\mathbb{F} = \begin{matrix} F_1^{\text{LLA}} & F_2^{\text{LLA}} & \cdots & F_p^{\text{ULA}} \end{matrix} \begin{bmatrix} f_{11} & f_{12} & \cdots & f_{1(p\times q)} \\ f_{21} & f_{22} & \cdots & f_{2(p\times q)} \\ \vdots & \vdots & \vdots & \vdots \\ f_{n1} & f_{n2} & \cdots & f_{n(p\times q)} \end{bmatrix} \in \mathcal{M}_{n,p\times q}(\mathbb{R}) \tag{3.4}$$

La méthode développée suppose donc que l'information contenue dans chaque descripteur contribue de manière égale au résultat du partitionnement. Par conséquent, toutes les colonnes de la matrice $\mathbb{F}$ doivent être fusionnées pour obtenir un signal de frontière final global. Pour réaliser cela, le principe d'estimation de densité par noyaux - Kernel Density Estimation (KDE) [Parzen, 1962] est utilisé. Chaque colonne de $\mathbb{F}$ est alors considérée comme la réalisation d'un processus aléatoire, le prédicat de départ étant que le signal de frontières de groupes est une variable aléatoire. Aussi, les descripteurs initiaux p sont eux-mêmes considérés comme des variables aléatoires. Une fois ces hypothèses faites, la distribution de chaque F_j^l est une discrétisation de leur densité de probabilité théorique respective. Avec les hypothèses faites sur les fonctions « noyau », telles que présentées par [Scott, 2015, théorème 6.7], la KDE permet de construire une densité de probabilité empirique qui converge vers son résultat théorique. En considérant les $p \times q$ segments temporels comme des réalisations de plusieurs processus aléatoires indépendants et identiquement distribués, il est cohérent d'additionner toutes les colonnes de $\mathbb{F}$ définies à l'équation 3.4 pour obtenir un vecteur total $\overline{F}$ défini à l'équation 3.5.

$$\overline{F} = \begin{bmatrix} \sum_{j=1}^{p\times q} f_{1j} & \sum_{j=1}^{p\times q} f_{2j} & \cdots & \sum_{j=1}^{p\times q} f_{nj} \end{bmatrix}^\top \tag{3.5}$$

Maintenant que le vecteur d'intérêt a été créé à partir de la somme des signaux de frontières intermédiaires de tous les descripteurs et de tous les actionneurs, la densité de probabilité empirique peut être calculée. D'après [Scott, 2015, chapitre 6, équation 6.1], la densité empirique peut être exprimée par l'équation 3.6.

$$\hat{p}(x) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x - \overline{f}_i}{h}\right) \tag{3.6}$$

Dans l'équation 3.6, K est une fonction noyau appliquée à la variable centrée de la série temporelle de l'échantillon et mise à l'échelle par un facteur h. Bien que le noyau gaussien ne soit pas optimisé au niveau des coûts de calculs, il est choisi dans ces travaux pour ses propriétés. Il est défini à l'équation 3.7.

$$K(x) = \frac{1}{\sqrt{2\pi}} \exp^{-\frac{1}{2}x^2} \tag{3.7}$$

L'inconvénient de la méthode d'estimation de densité par noyau telle qu'exprimée à l'équation 3.6 réside dans le choix non trivial du facteur h. Le facteur h représente la *bande passante* de l'estimation et contrôle la taille du support des gaussiennes qui seront centrées sur chaque élément f_i. L'intérêt d'avoir choisi un noyau gaussien est alors de pouvoir déterminer la valeur de cette bande passante à partir d'une formule simple de l'état de l'art. En effet, comme détaillé dans [Silverman, 1986, équation 3.31], la largeur de bande h d'un noyau gaussien peut être exprimée comme $h = 0.9An^{-1/5}$, avec la constante $A = \min(\sigma, \text{inter-quartile range}/1.34)$. L'écart inter-quartile étant une mesure de dispersion définie comme la différence entre le premier et le troisième quartile des données. Notons que d'autres méthodes existent comme les méthodes développées par [Turlach, 1993], [Bashtannyk et Hyndman, 2001] ou encore [Botev, Grotowski et Kroese, 2010].

L'objectif de la KDE est d'obtenir un signal « continu » représentant une mesure de probabilité des frontières discrètes de groupes. Dans ces travaux, la fonction *scipy.stats.gaussian_kde* [Virtanen *et al.*, 2020] a été utilisée et, comme énoncé précédemment, l'estimation de la largeur de bande a été obtenue par la méthode de Silverman [Silverman, 1986]. Afin de comprendre la KDE, dans un premier temps la procédure est appliquée aux quatre signaux utilisés pour illustrer cette section. La figure 3.17 donne l'estimation de la densité par noyau gaussien pour ces quatre signaux.

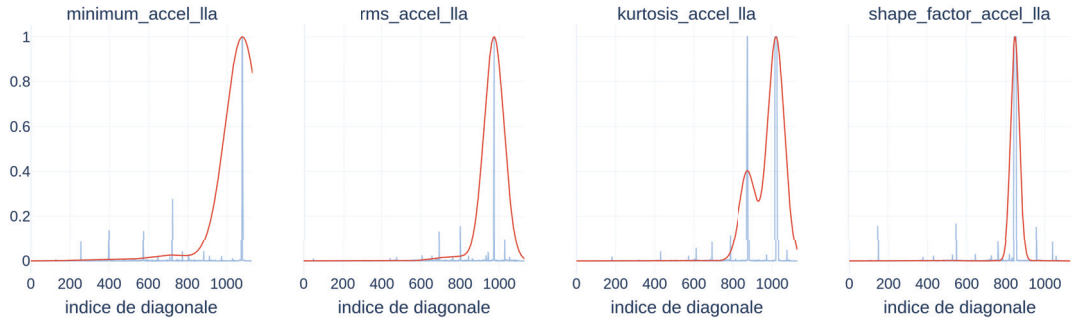


FIGURE 3.17 – Estimation de densité par noyau gaussien appliquée au signaux de la figure 3.16

L'ensemble des courbes de la figure 3.17 montre que l'étape d'estimation de densité permet de filtrer une grande partie des frontières de groupes intermédiaires obtenues à cause du bruit dans la segmentation des matrices de distances de l'étape 3.3.2. Cette capacité de filtrage est contrôlée par la bande passante h précédemment choisie. Enfin, la taille du support de chaque distribution permet de rendre compte de l'incertitude de la frontière considérée. À partir de cette étape, les signaux considérés ne sont plus limités à l'ensemble choisi pour l'exemple filé dans cette section. Afin d'obtenir un signal global de frontière de groupes, ce sont les ensembles des $p \times q$ séries temporelles qui sont utilisés. Ainsi, en appliquant l'estimation de densité par noyau gaussien au vecteur $\overline{F}$, la figure 3.18 est obtenue.

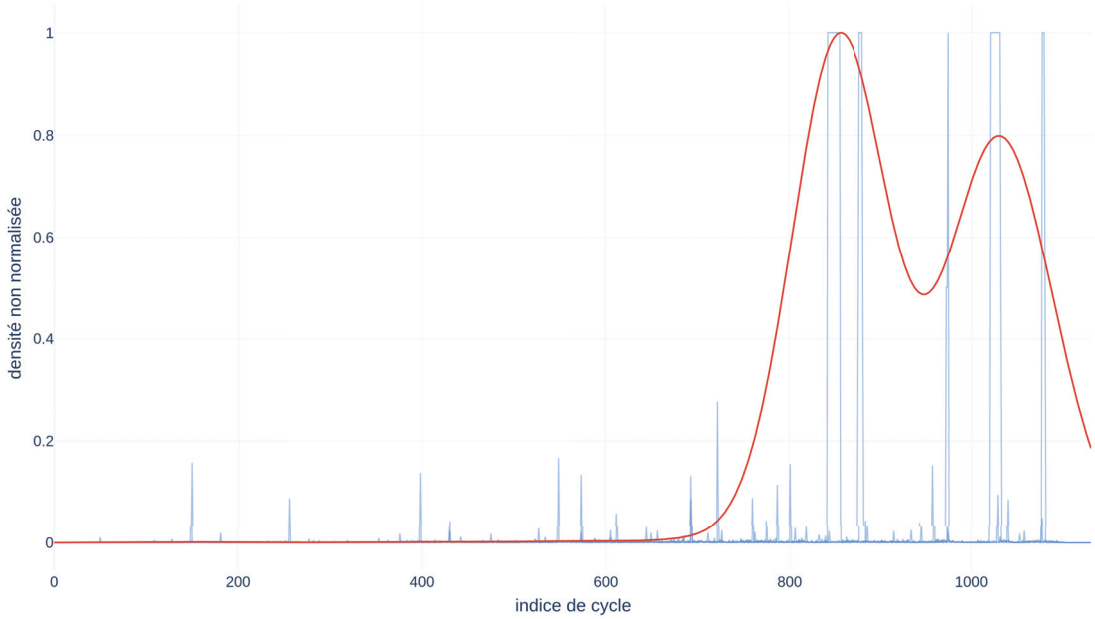


FIGURE 3.18 – Estimation de densité par noyau gaussien calculée à partir des $p \times q$ signaux rassemblés dans la matrice $\mathbb{F}$

Par principe, chaque maximum local de la fonction de densité est susceptible d'être une frontière de groupe. Ainsi, deux frontières de groupe sont détectées dans la figure 3.18 : une première au 857^{e} cycle ainsi qu'une seconde au 1029^{e} cycle. Cela signifie que les données mettent en exergue trois groupes aux degrés de sévérité différents. Le premier groupe des cycles 0 à 855 qui correspond au premier degré de sévérité où le système est sain ; le deuxième groupe allant du cycle 856 au cycle 1028 correspond au deuxième degré de gravité des défauts du système et enfin le dernier groupe commençant au cycle 1029 correspond au troisième et pire stade de gravité des défauts.

La distribution obtenue à la figure 3.18 est un mélange de gaussiennes. L'objectif de la méthode de partitionnement est d'obtenir une distribution pour chaque frontière de groupe. Aussi, il est nécessaire d'extraire les composantes indépendantes de la densité empirique obtenue, résultat de l'estimation de densité par noyau. C'est pourquoi la distribution de données obtenues sert à générer un échantillon aléatoire de données. Cet échantillon représente alors la distribution obtenue à la

figure 3.18 et la fonction *sklearn.mixture.GaussianMixture* [Pedregosa *et al.*, 2011] permet d'estimer les deux composantes indépendantes du modèle de mélange. Le résultat de cette estimation est donné à la figure 3.19.

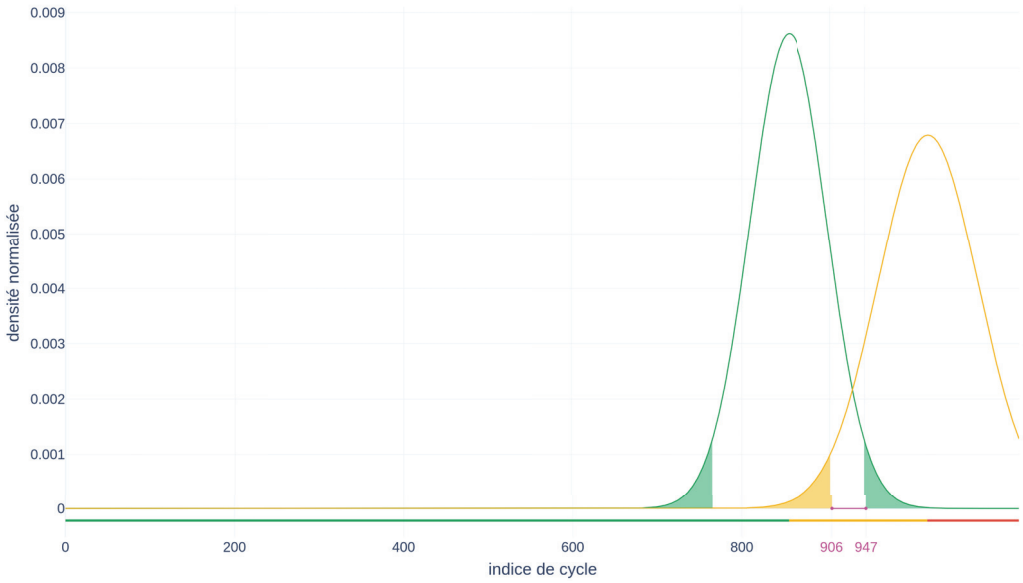


Figure 3.19 – Frontières de groupe avec intervalle de confiance de 95 %

Notons que la moyenne de chaque composante indépendante varie de quelques cycles par rapport à la distribution mélangée de la figure 3.18. Avec la figure 3.19, les frontières de groupes sont localisées aux 865^{e} et 1040^{e} cycles, ce qui équivaut à une erreur relative de 0 % et 1 % par rapport aux valeurs initiales. Ce décalage est dû au caractère aléatoire de l'échantillonnage d'une population pour identifier les composantes indépendantes. En effet, la dimension d'intérêt dans cette section est le cycle. L'intervalle de confiance à 95 % de chaque distribution est compris entre chaque surface colorée en queue de gaussienne, en vert et jaune. En plus d'obtenir la moyenne des composantes indépendantes, cette décomposition permet aussi de rendre compte d'un intervalle d'incertitude entre le 906^{e} et le 947^{e} cycle. Cette zone identifie le support commun des intervalles de confiance à 95 % des distributions liées aux frontières de groupes. Elle est représentée en violet. Dans cette zone, il est alors impossible en l'état de discerner dans les données si un point appartient plus à un degré de sévérité qu'à un autre. Il y a donc 41 cycles d'incertitude durant lesquels l'information sur l'intensité d'un défaut n'est pas donnée. La barre de couleur horizontale, quant à elle, située sous l'axe des abscisses de la figure 3.19, représente l'état de santé (ou le degré de sévérité de défaut) déduit à partir des calculs précédents pour l'actionneur électromécanique complet E-TRAS à partir des mesures vibratoires de ses vérins constitutifs. En vert aucun défaut n'est détecté dans les données, puis, entre le 865^{e} et le 1040^{e} cycle, un premier degré de sévérité est détecté. Par-delà le degré de sévérité final, le dernier défaut avant fin de vie de l'actionneur est alors reproduit en rouge.

Ainsi des frontières de groupes de données ont été déterminées avec leur mesure d'incertitude associée dans les données d'exploitation d'E-TRAS pour la campagne 2017. Au-delà des hyperparamètres déterminés pour l'apprentissage du réseau de segmentation, cette méthode a l'avantage de ne pas reposer sur le choix d'hyperparamètres par savoir expert. En cela, il est possible de l'utiliser telle quelle pour obtenir un profil de partitionnement de données temporelles allant jusqu'à fin de vie (R2F). De plus, en comparant le résultat de la figure 3.19 avec la première assignation de label effectuée par expert, notamment aux figure 3.5 et 3.6, les résultats sont cohérents. Notre méthode permet de rajouter de la précision ainsi que la quantification de l'incertitude au regard des données étudiées.

3.3.5 Assignation d'un degré de sévérité à chaque point

Maintenant que le profil des frontières de groupes a été déterminé, il est possible de partitionner les séries temporelles des données d'exploitations. La sous-section 3.3.3 précédente a permis la

création de frontières sous forme de distributions (voir figure 3.19) mais pour ne pas surcharger les représentations, le partitionnement des séries s'effectuera ici en considérant les moyennes de chaque frontière comme étant un changement de label. Ainsi, chaque maximum local de fonction de densité estimée est considéré comme une frontière de groupe. En appliquant ce schéma de partitionnement aux signaux de la figure 3.7, la figure 3.20 est obtenue.

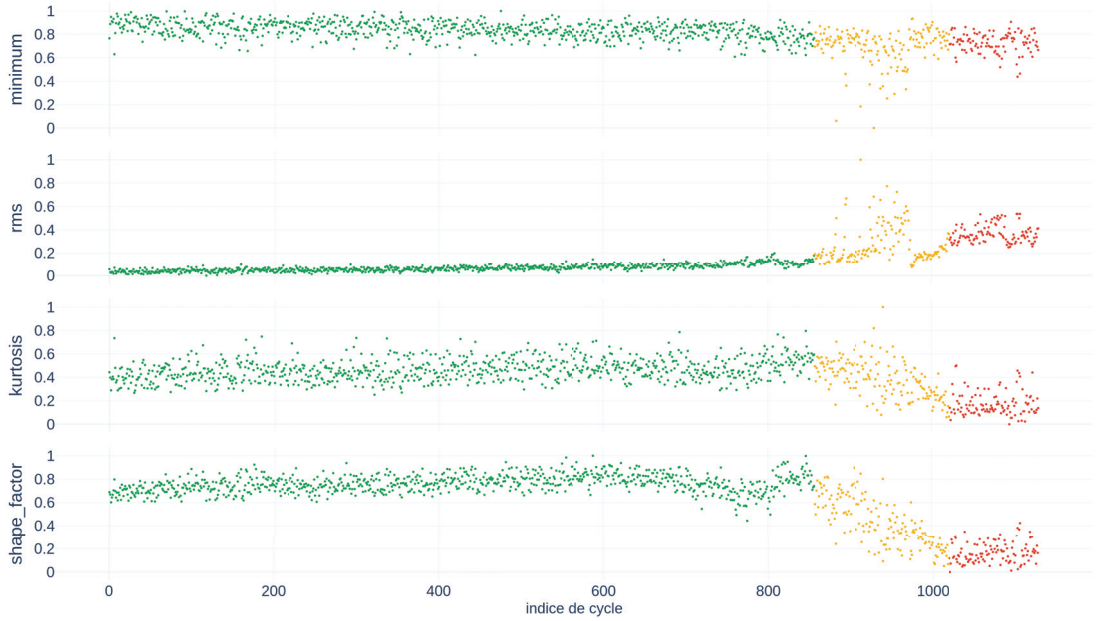


FIGURE 3.20 – Résultats de partitionnement pour les descripteurs introduits à la figure 3.7

Dans la figure 3.20, chaque groupe possède une couleur qui correspond à l'état de santé de l'actionneur. Le premier groupe vert est associé à l'état sain, le deuxième groupe jaune est associé au premier degré de sévérité tandis que le dernier groupe rouge est associé au dernier degré de sévérité avant fin de vie de l'actionneur.

En appliquant ce partitionnement aux données d'exploitation, remarquons que le résultat est cohérent avec leurs dynamiques. En effet, pour s'en convaincre, considérons le descripteur `minimum`. Jusqu'au 856^{e} cycle, il présente une dynamique linéaire légèrement décroissante puis, à partir du 857^{e} jusqu'au 1021^{e} cycle, la variance de l'ensemble augmente fortement pour revenir à une pseudo-stabilité à partir du 1022^{e} cycle. Même si le schéma de partitionnement obtenu dans cette section 3.3 est commun à tous les descripteurs de tous les actionneurs étudiés, il n'en demeure pas moins cohérent avec les dynamiques de chacun des signaux. Néanmoins, cette interprétation ne reste qu'approximative sans indicateur mathématique permettant de quantifier la qualité de partitionnement.

3.4 Nouvel indicateur de qualité interne

Afin d'évaluer la méthode de partitionnement qui a été développée, il est nécessaire de pouvoir quantifier sa performance par rapport aux autres méthodes de partitionnement de l'état de l'art. Seulement, les indicateurs usuels ne sont pas adaptés à la qualification de résultats de partitionnement temporels interdisant, notamment, la régénération des actionneurs et donc, un retour en arrière des labels. Par ailleurs, il est difficile de trouver un indicateur général car les indices de qualité du partitionnement sont spécifiques aux hypothèses sur lesquelles reposent celui-ci [JAVED, LEE et RIZZO, 2020].

En effet, alors que les indicateurs pour le partitionnement spatial sont bien connus, les indicateurs spécifiques existants pour déduire la qualité d'un regroupement pour le partitionnement des séries temporelles internes sont rares. Parmi les indicateur spatiaux, le *S_Dbw* [HALKIDI et VAZIRGIANNIS, 2001] peut être cité pour mesurer la compacité et la bonne séparation des groupes ou encore la revue exhaustive de [AGHABOZORGI, SEYED SHIRKHORSHIDI et YING WAH, 2015] sur le partitionnement de séries temporelles. De plus, récemment, un critère guidé par l'invariance a

été créé par [Forest *et al.*, 2021]. Malgré leurs comportements spécifiques, l'ensemble des indices de l'état de l'art peuvent être classés en deux groupes principaux : les externes et les internes [Yanchi Liu *et al.*, 2010]. Les indicateurs *externes* évaluent la qualité d'un regroupement en comparant ce dernier à une vérité connue. Les indicateurs de qualité interne, quant à eux, évaluent la qualité du regroupement uniquement en vérifiant des hypothèses faites au préalable par l'utilisateur sur la forme des groupes obtenus. Ne possédant pas de données précédemment labellisées dans ces travaux, ou encore de labels pour les données d'exploitation, nous ne pouvons compter que sur les informations contenues dans les données étudiées, comme indiqué dans [Yanchi Liu *et al.*, 2010].

Avant toute chose, onze indicateurs de qualité interne, connus de l'état de l'art, ont été comparés. Dans ce travail, il s'agit de l'indicateur *Silhouette* [Rousseeuw, 1987], de la mesure de séparation Davies-Bouldin [Davies et Bouldin, 1979], de l'indicateur Calinski Harabasz [Caliński et Harabasz, 1974], de la conjonction d'une mesure d'inertie et de cohérence temporelle de [Breuneval, 2017], des indices de validité SD [Halkidi, Vazirgiannis et Batistakis, 2000] et de *S_dbw* [Halkidi et Vazirgiannis, 2001] ou encore de la mesure de similarité de [Mori, Mendiburu et Lozano, 2016]. Mais, comme ce sera présenté à la section 3.5, aucun d'entre eux n'a réussi à sélectionner le bon résultat de partitionnement. Partant de cet état de fait, il est nécessaire de développer un nouvel indicateur. Cette nouvelle mesure devrait être capable de quantifier les informations pertinentes en fonction de la physique du système. D'une part, comme indiqué dans l'hypothèse fondamentale 3, le système étudié ne peut pas être réparé pendant son cycle de vie. Une fois que le degré de sévérité de défaut a augmenté, c'est-à-dire qu'une frontière de groupe a été franchie, le système ne peut pas revenir à un stade antérieur plus sain. En d'autres termes, chaque classe doit être continue dans le temps. Les sauts temporels dans les groupes sont pas permis avec les hypothèses de modélisation des défauts choisies dans ces travaux. Par conséquent, l'indicateur doit pénaliser toute incohérence temporelle. D'autre part, l'hypothèse peut être faite qu'un bon partitionnement est réalisé lorsque deux groupes consécutifs ne partagent pas les mêmes propriétés de densité. Cette information de forme doit également être prise en compte.

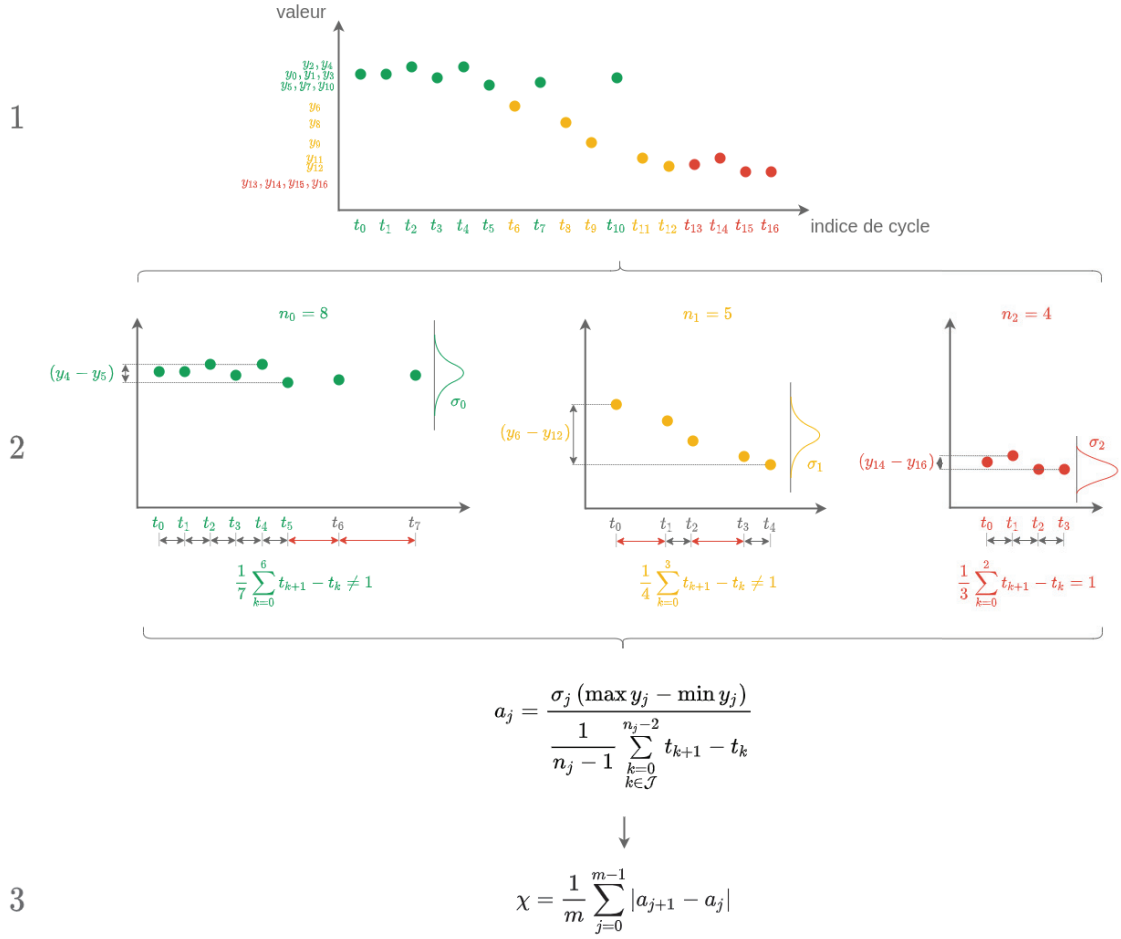


Figure 3.21 – Schéma de principe pour le calcul de χ

L'équation 3.8 présente le nouvel indicateur de qualité interne développé pour le partitionnement des séries temporelles, répondant à ces précédentes caractéristiques.

$$\chi = \frac{1}{m}\sum_{j=0}^{m-1} |a_{j+1} - a_j| \text{, avec}$$
$$a_j = \frac{\sigma_j \cdot (\max y_j - \min y_j)}{\frac{1}{n_j - 2}\sum\limits_{\substack{k=0 \\ k\in\mathcal{J}}}^{n_j-1} t_{k+1} - t_k} \tag{3.8}$$

Dans l'équation 3.8, m représente le nombre de groupes trouvés dans les données, σ_j l'écart-type du j^{e} groupe, $\max y_j - \min y_j$ rend compte de l'étendue du j^{e} groupe, y_j ses éléments, et t_k ses indices respectifs et n_j son nombre de points. Remarquons que $\mathcal{J}$ est l'ensemble des indices de points dans le groupe j. Ainsi, alors que le dénominateur de a_j mesure la cohérence temporelle, le numérateur mesure la variation maximale des valeurs d'un groupe. Enfin, l'indice de qualité χ traite de la variation entre deux groupes consécutifs et non de a_j en lui-même. Notons que le dénominateur de a_j ne peut jamais être nul pour les séries temporelles. En effet, la figure 3.21 montre pour le nouvel algorithme que les première et troisième partitions de données présentent une dynamique similaire. Cependant, comme le système ne peut pas rajeunir, celles-ci doivent être considérées comme deux groupes différents. Pour reconnaître le meilleur résultat de partitionnement, la valeur $\mathcal{I}$ définie dans l'équation 3.8 doit être maximisée.

Maintenant qu'un indicateur mathématique permet de quantifier la bonne qualité des résultats de partitionnement, il est possible de comparer les résultats de la méthode de ce chapitre à l'état de l'art.

3.5 Comparaison aux méthodes de l'état de l'art

À présent que la méthode de partitionnement a été détaillée et qu'un indicateur de performance permet de quantifier la qualité de son résultat, il reste à choisir des algorithmes de l'état de l'art mesurant la qualité des partitionnements pour pouvoir comparer leurs résultats à ceux de la méthode précédemment décrite. Le choix s'est porté sur des algorithmes classiques et l'ensemble de ceux-ci avec leurs paramètres pour l'étude sont détaillés dans le tableau 3.1. Ils sont issus de la bibliothèque Python *scikit-learn* [Pedregosa *et al.*, 2011] à l'exception de *Kmeans* qui provient de la librairie *tslearn* [Tavenard *et al.*, 2020].

Méthodes	Catégorie	Métrique	Nom de fonction	Paramètres
K-Means	Partitional	*euclidean*	KMeans	max_iter=200 n_jobs=-1 init='k-means++' n_init=10
		dtw		
K-medoids	Partitional	metric='euclidean'	KMedoids	max_iter=200 init='k-medoids++'
Mean Shift	Partitional	/	MeanShift	max_iter=200 n_jobs=-1
OPTICS	Density-based	metric='euclidean'	OPTICS	n_jobs=-1 min_samples=3
		metric='manhattan'		
Agglomerative	Hierarchical	metric='euclidean'	Agglomerative Clustering	min_cluster_size=50
		metric='manhattan'		
Gaussian Mixture Model	Model-based	cov='spherical'	GaussianMixture	max_iter=1000 n_init=100
		cov='diag'		
		cov='full'		
		cov='tied'		

Table 3.1 – Méthodes de partitionnements utilisées dans ces travaux, d'après [Eid *et al.*, 2021, table 1]

Les méthodes de la table 3.1 ont été choisies en fonction de deux critères. Le premier, très pragmatique, s'intéressait à la facilité d'implémentation dans le cadre de notre *framework* de PHM. Ainsi, les procédures retenues sont contenues dans les bibliothèques standard d'apprentissage automatique en Python. Le second critère concernait, lui, la représentativité de la diversité des

algorithmes de partitionnement. En effet, selon [Aghabozorgi, Seyed Shirkhorshidi et Ying Wah, 2015, figure 6], ces algorithmes peuvent être séparés en six classes différentes. Au moins trois d'entre elles ont été étudiées dans ces travaux : les méthodes de partitionnement, les méthodes hiérarchiques et celles basées sur des modèles. Pour effectuer la comparaison, les signaux de la figure 3.7 ont été utilisés. Ainsi, chaque méthode du tableau 3.1 a été alimentée par une structure de données comportant cinq colonnes et $n = 1130$ lignes. Quatre des colonnes sont la concaténation des vecteurs $(X_1^{\text{LLA}}, X_6^{\text{LLA}}, X_9^{\text{LLA}}, X_1 3^{\text{LLA}})$ tandis que la cinquième colonne est constituée d'un tableau d'index T allant de 0 à 1 129 représentant les cycles, autrement dit le temps. Inclure le numéro de cycle dans les données d'entrée permet à chaque algorithme de prendre en compte la relation de précédence temporelle imposée entre les points. Dans la procédure de comparaison, chaque méthode de partitionnement utilisée s'adapte aux données et prédit les étiquettes de chaque point. Ces labels sont ensuite stockés dans un vecteur pour représenter les résultats du partitionnement. Pour ne pas favoriser notre méthode vis-à-vis de l'état de l'art, chaque algorithme de regroupement produira un résultat à partir de tous les signaux d'entrée, ici l'ensemble précédent. C'est pourquoi le résultat de partitionnement sera commun à tous les signaux d'entrées.

Avant de présenter les résultats finaux de partitionnement, l'indicateur développé à l'équation 3.8 est testé pour l'ensemble des méthodes de regroupement appliquées à la matrice de données $\mathbb{X} = \begin{bmatrix} T & X_1^{\text{LLA}} & X_6^{\text{LLA}} & X_9^{\text{LLA}} & X_1 3^{\text{LLA}} \end{bmatrix}^\top$. L'ensemble des algorithmes de la table 3.1 sont alors testés en imposant la recherche de plusieurs groupes dans les données. En effet, la plupart des algorithmes de partitionnement nécessitent la connaissance de l'information préalable du nombre de groupes présents à trouver. C'est un hyperparamètre à fixer *a priori*. Notons que la méthode développée dans ces travaux ne nécessite pas une telle information. Ainsi, pour chaque groupe, le résultat de partitionnement est évalué par l'indicateur χ, le but étant de le maximiser.

	Nombre de groupes							
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
KMeans_euclidean		0.074	0.051	0.034	0.025	0.026	0.035	0.032
KMeans_dtw		0.074	0.051	0.036	0.025	0.026	0.035	0.031
MeanShift				0.040				
KMedoids		0.074	0.051	0.034	0.026	0.027	0.037	0.032
OPTICS_euclidean	0.008							
OPTICS_manhattan	0.008							
Agglomerative_euclidean		0.073	0.050	0.024	0.020	0.018	0.019	0.017
Agglomerative_manhattan		0.074	0.051	0.038	0.032	0.027	0.030	0.027
GMM_spherical		0.074	0.051	0.034	0.025	0.026	0.034	0.032
GMM_tied		0.068	0.021	0.025	0.021	0.012	0.011	0.009
GMM_diag		0.041	0.027	0.027	0.023	0.013	0.011	0.011
GMM_full		0.049	0.034	0.026	0.023	0.028	0.023	0.021
SUNRISE			0.083					

Table 3.2 – Valeurs du nouvel indicateur de qualité χ pour chaque algorithme et pour chaque nombre de groupes

Dans le tableau 3.2, les cellules grisées correspondent aux méthodes pour lesquelles le nombre de groupe n'est pas nécessaire pour inférer un partitionnement. Ce sont en général des méthodes à base de mesure de densité comme `MeanShift`, `OPTICS` ou encore la nôtre que nous noterons Soft clUsteriNg foR tIme SEries (SUNRISE). Deux informations principales peuvent être extraites de ce tableau. La première est que tous les algorithmes, à l'exception de `MeanShift` et de `SUNRISE`, considèrent que les données contiennent uniquement deux groupes et donc, deux degrés de sévérité. Rappelons que ce résultat est à étudier à l'aune de l'indicateur χ défini à l'équation 3.8. Autrement dit, les partitionnements qui vérifient les propriétés physiques imposées par l'indicateur χ sont, pour la majorité, constitués de deux groupes pour les algorithmes de l'état de l'art. La seconde information est que la méthode `SUNRISE` possède le maximum global du tableau. Elle produit donc un schéma de partitionnement *ad hoc* vis-à-vis des hypothèses faites dans ces travaux. De plus, certaines valeurs sont comprises dans le voisinage du maximum global $0,083$. En effet, celles de `KMeans_euclidean`, `KMeans_dtw`, `KMedoids`, `Agglomerative_euclidean`, `Agglomerative_manhattan` et `GMM_spherical` sont quasiment toutes égales à 0.074. Ce résultat se traduit, à la figure 3.22, par des schémas de partitionnements identiques. De manière plus générale, pour un nombre de groupes

donné, les algorithmes ont tendance à regrouper les données de manière similaire. Enfin, de par le caractère identique des valeurs des indicateurs obtenues par les méthodes à base de centroïdes, il semble que la distance choisie n'ait pas d'impact significatif sur les résultats de partitionnement.

	Nombre de groupes							
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
KMeans_euclidean		11 %	39 %	59 %	70 %	69 %	58 %	61 %
KMeans_dtw		11 %	39 %	57 %	70 %	69 %	58 %	61 %
MeanShift				52 %				
KMedoids		11 %	39 %	59 %	69 %	67 %	55 %	61 %
OPTICS_euclidean	90 %							
OPTICS_manhattan	90 %							
Agglomerative_euclidean		12 %	40 %	71 %	76 %	78 %	77 %	80 %
Agglomerative_manhattan		11 %	39 %	54 %	61 %	67 %	64 %	67 %
GMM_spherical		11 %	39 %	59 %	70 %	69 %	59 %	61 %
GMM_tied		18 %	75 %	70 %	70 %	86 %	87 %	89 %
GMM_diag		51 %	67 %	67 %	72 %	84 %	87 %	87 %
GMM_full		41 %	59 %	69 %	72 %	66 %	72 %	70 %

TABLE 3.3 – Erreurs relatives des mesures de χ par rapport à la valeur obtenue par notre méthode : 0.083

Afin de mieux rendre compte des différences relatives entre la méthode SUNRISE et les méthodes de l'état de l'art testées ici, la table 3.3 rassemble la quantification de l'écart des valeurs obtenues de l'indice χ. Cela permet de faire ressortir le fait que certaines méthodes produisent des résultats de regroupement de bien moindre qualité que les autres. Les paragraphes suivant s'attacheront à expliquer la cause d'une telle différence.

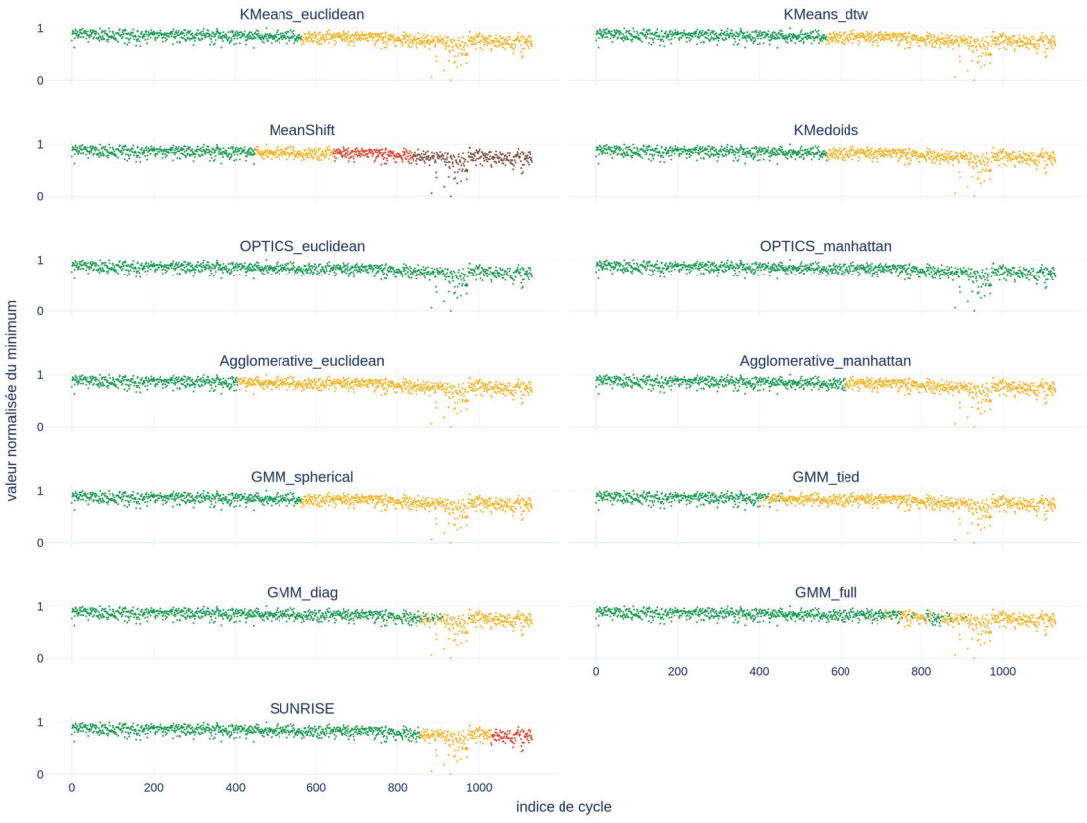


FIGURE 3.22 – Résultats de partitionnement du signal X_1^{LLA} avec les algorithmes de la table 3.1 et la nouvelle méthode développée

En utilisant les données d'exploitation $\mathbb{X}$ avec le nombre de groupes optimal déterminé à la table 3.2, le partitionnement avec les algorithmes du tableau 3.1 et la méthode SUNRISE est donné à la figure 3.22. Seul le descripteur `minimum` a été sélectionné pour servir d'exemple car, par construction du protocole expérimental, le schéma de partitionnement est identique pour tous les descripteurs *in fine*.

Comme intuité à partir des résultats de la table 3.2, les partitionnements de `KMeans_euclidean`, `KMeans_dtw`, `KMedoids`, `Agglomerative_manhattan` sont identiques. Notons que la méthode `OPTICS` ne détecte qu'une classe dans les données, d'où la valeur de l'indicateur qui est environ 10 fois plus basse que le maximum global du tableau. C'est d'ailleurs le minimum global de ce dernier. En cela, l'indicateur χ prouve sa performance. De même pour `MeanShift` qui détecte quatre classes de défauts. La différence entre les trois premiers groupes (en vert, jaune et rouge) n'étant pas évidente, ce partitionnement est pénalisé et n'atteint qu'une valeur de 0.040, soit deux fois moins que le maximum global de 0.083. Concernant les autres méthodes, `GMM_diag`, `GMM_tied` et `GMM_full`, en plus de ne trouver que deux classes dans les données, ces dernières mélangent les assignations de degrés de sévérités entre eux produisant ainsi un exemple d'incohérence temporelle. Enfin, `Agglomerative_euclidean` et `GMM_spherical` qui possèdent dans la table 3.2 des valeurs très proches, présentent un schéma de partitionnement différent. Néanmoins, la limite entre le premier et le second degré de sévérité apparaît dans les deux cas dans deux zones aux dynamiques similaires.

Afin d'expliquer les différentes disparités de valeurs de χ entre les différents algorithmes, divisons le en deux parties distinctes pour étudier son comportement. Une première partie est constituée du numérateur de a_j défini à l'équation 3.8. La quantité $\sigma_j\,(\max y_j - \min y_j)$ produit une image de l'étendue verticale d'un groupe. Son dénominateur $\frac{1}{n_j-1}\sum_{k=0}^{n_j-1} t_{k+1} - t_k$, quant à lui, permet de mesurer la cohérence temporelle locale des groupements effectués. Dans le but de mieux comprendre l'apport de chaque partie dans la quantification du résultat de regroupement, les tableaux 3.4 et 3.5 représentent ces deux composantes relatives aux résultats exposés dans la table 3.2 qui a produit les résultats de la figure 3.22.

	Nombre de groupes							
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
KMeans_euclidean		0.074	0.051	0.034	0.025	0.026	0.035	0.032
KMeans_dtw		0.074	0.051	0.036	0.025	0.026	0.035	0.031
MeanShift				0.040				
KMedoids		0.074	0.051	0.034	0.026	0.027	0.037	0.032
OPTICS_euclidean	0.055							
OPTICS_manhattan	0.055							
Agglomerative_euclidean		0.073	0.050	0.024	0.020	0.018	0.019	0.017
Agglomerative_manhattan		0.074	0.051	0.038	0.032	0.027	0.030	0.027
GMM_spherical		0.074	0.051	0.034	0.025	0.026	0.034	0.032
GMM_tied		0.072	0.037	0.028	0.023	0.018	0.016	0.015
GMM_diag		0.061	0.042	0.030	0.026	0.039	0.035	0.031
GMM_full		0.068	0.036	0.028	0.025	0.040	0.037	0.033
SUNRISE			0.083					

TABLE 3.4 – Valeur de la composante de forme $\sigma_j\,(\max y_j - \min y_j)$ appliquée aux différents algorithmes de partitionnement pour un nombre de groupes différents

Exception faite de `OPTICS_euclidean`, `OPTICS_manhattan`, `GMM_tied`, `GMM_diag` et `GMM_full` les valeurs respectives de l'indicateur de forme obtenues à la table 3.4 par les autres algorithmes sont identiques à l'indicateur complet χ de la table 3.2. Remarquons que ces résultats sont cohérents avec les partitionnements obtenus à la figure 3.22. En effet, les algorithmes précédemment cités présentaient tous des incohérences temporelles dans le schéma de regroupement de données. Par définition, la cohérence temporelle est unitaire lorsqu'elle est parfaite. Aussi, apparaissant au dénominateur de χ, cette dernière divisera la valeur de l'indicateur des algorithmes de partitionnement incohérents, autrement dit, les algorithmes qui ne détectent pas de degrés de sévérités continus dans le temps. Cette hypothèse est corroborée par les résultats du tableau 3.5 où les deux méthodes `OPTICS` obtiennent, par exemple, une valeur de 2.785. Cette pénalisation est du coup

propagée dans le calcul de χ.

	Nombre de groupes							
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
KMeans_euclidean		1.000	1.000	1.000	1.000	1.000	1.000	1.000
KMeans_dtw		1.000	1.000	1.000	1.000	1.000	1.000	1.000
MeanShift				1.000				
KMedoids		1.000	1.000	1.000	1.000	1.000	1.000	1.000
OPTICS_euclidean	2.785							
OPTICS_manhattan	2.785							
Agglomerative_euclidean		1.000	1.000	1.000	1.000	1.000	1.000	1.000
Agglomerative_manhattan		1.000	1.000	1.000	1.000	1.000	1.000	1.000
GMM_spherical		1.000	1.000	1.000	1.000	1.000	1.000	1.000
GMM_tied		1.075	1.310	1.197	1.268	1.919	1.804	1.789
GMM_diag		1.259	1.758	1.903	2.503	2.700	3.002	2.980
GMM_full		1.247	1.472	1.633	1.746	1.609	1.889	1.761
SUNRISE			1.000					

TABLE 3.5 – Valeur de la composante de cohérence temporelle $\frac{1}{n_j-1}\sum t_{k+1} - t_k$ pour chaque méthode de partitionnement et chaque groupe

Dans cette section, la méthode de partitionnement de séries temporelles développée, SUNRISE, a été comparée aux méthodes de regroupement classiques de l'état de l'art. La comparaison a été permise par la création d'un nouvel indicateur χ contenant dans sa forme l'ensemble des hypothèses faites pour déterminer des degrés de sévérités dans des données ne présentant pas de phénomènes de régénération. SUNRISE permet d'affecter une classe *a posteriori* à chaque point d'une série temporelle. En cela elle pourrait être considérée comme un classifieur.

Conclusion

Les travaux antérieurs sur le partitionnement des séries temporelles pour la surveillance de l'état des systèmes aéronautiques ne tenaient jusque-là pas compte de la dimension temporelle des données, comme l'illustrent [ZAPOROWSKA *et al.*, 2020]. Ils utilisent entre autres choses des algorithmes tels que les K-plus proches voisins ou les Support Vector Machine (SVM) pour regrouper les données. Ce faisant, ils n'incorporent pas l'information de précédence temporelle dans les résultats. Le travail de [BREUNEVAL, 2017] est, à notre connaissance, la tentative la plus récente prenant en compte cette contrainte. Néanmoins, ses travaux exigeaient de l'utilisateur une connaissance préalable des données. Ce faisant, l'utilisateur pouvait risquer d'ajouter ses propres biais dans la procédure. De plus, comme développé dans [JAVED, LEE et RIZZO, 2020], il est difficile de trouver un indicateur permettant de quantifier la qualité d'un partitionnement car une telle grandeur dépend fortement des hypothèses faites dans la recherche et du jeu de données utilisé.

Ces travaux ont développé une nouvelle méthode de partitionnement pour les séries temporelles avec, en son noyau, un réseau neuronal profond. L'ensemble de la procédure se décompose en trois étapes principales. Tout d'abord, les données temporelles sont encodées en images pour être transmises au réseau de neurones, images à partir desquelles le réseau fournit un résultat segmenté. Ensuite, les images segmentées sont traitées avec des algorithmes de traitement du signal pour extraire un signal de frontière de groupes intermédiaire. Enfin, les informations de tous les descripteurs sont fusionnées pour obtenir le profil général de partitionnement des données. L'ensemble de l'algorithme permet alors de trouver les frontières des groupes avec une mesure de vraisemblance à partir d'un ensemble de séries temporelles. Il a été conçu spécifiquement pour prendre en compte la relation de précédence des éléments et pour éviter les incohérences temporelles dans le partitionnement. Enfin, il a été appliqué à un ensemble de données industrielles pour évaluer sa pertinence. Notez que l'ensemble de l'algorithme pourrait être utilisé avec des séries temporelles multivariées car la nature des données n'a aucune incidence sur ses résultats. Ainsi, le partitionnement automatique de séries temporelles, sans connaissance préalable, pour assigner un label à des données brutes, est possible. Cela permet de déterminer, notamment en analyse *post mortem* comme c'est le cas ici, l'ensemble des degrés de sévérité par lesquels est passé l'actionneur. L'étiquetage des séries temporelles pourrait servir aussi à développer une base de données qui servirait

de base d'apprentissage pour des algorithmes d'apprentissage statistiques futurs. Malgré ses nombreux avantages, cette méthode n'en est pas moins coûteuse en calculs du fait de l'entraînement initial d'une structure profonde. Cependant, la phase d'apprentissage n'est censé être effectuée qu'une seule fois pour l'utilisation de cette méthode. *In fine*, cette nouvelle approche constitue une réelle amélioration par rapport aux techniques traditionnelles puisque le réseau neuronal et les opérations de traitement du signal qui suivent la segmentation de l'image peuvent partitionner des données très bruitées. Notons que plus le réseau de segmentation est performant, meilleurs seront les résultats du partitionnement.

De surcroît, pour quantifier les résultats de cette nouvelle méthode par rapport à l'état de l'art, un nouvel indicateur a été créé, atténuant les inconvénients des indicateurs habituels pour le partitionnement de séries temporelles. Alors que l'état de l'art se concentre sur la mesure des formes relatives des groupes constitués par l'algorithme de partitionnement, le nôtre peut évaluer deux propriétés : la cohérence temporelle du partitionnement ainsi que la forme du groupe. Il a été spécialement conçu pour tenir compte des hypothèses physiques liées au vieillissement, *de facto* il présente de meilleurs résultats que ses homologues de l'état de l'art. Une variante des indicateurs habituels se trouve dans [Forest *et al.*, 2021], où l'invariance du résultat du partitionnement est quantifiée.

Enfin, l'ensemble de cette méthode n'est que la première étape d'une méthode visant à déterminer la durée de vie utile restante des actionneurs dans le cadre du PHM. Cette méthode sera développée dans le chapitre suivant. Les travaux futurs pourraient adapter cette méthode sans tenir compte des hypothèses 4 et 5 afin d'utiliser le processus dans un cadre plus large.

Chapitre 4

Pronostic d'actionneurs électromécaniques

TABLE 4.1 – Notations du chapitre

$n \in \mathbb{N}$	nombre de points dans une série temporelle opérationnelle
$m \in \mathbb{N}$	nombre de points dans une série temporelle en base de données
$N_d = 28$	nombre de descripteurs étudiés
$N_a = 4$	nombre d'actionneurs étudiés
$p \in [\![1; N_d]\!]$	indice du descripteur
$q \in [\![1; N_a]\!]$	indice de l'actionneur, équivalent au nombre de réalisations en base de données
$X \in \mathbb{R}^n$	un vecteur de réels de dimension n
$X \in \mathcal{M}_{n,1}(\mathbb{R})$	autre notation pour un vecteur réel colonne avec n lignes
$\mathbb{X} \in \mathcal{M}_{n,p}(\mathbb{R})$	une matrice de réels de dimension (n, p)
$K \in \mathbb{N}$	nombre d'itérations
$\mathfrak{M}_{m,p,q,\ldots}(\mathbb{R})$	tenseur de réels de dimension $(m, p, q, \ldots)$ vu comme la généralisation d'une matrice à n dimensions
$(\mathcal{I}, \mathcal{J})^2$	Ensembles des indices des séries X et Y
$\mathbb{R}_+$	Ensemble des réels positifs ou nuls
$\mathbb{R}^*$	Ensemble des réels non nuls
$\mathbb{N}$	Ensemble des entiers naturels
$\mathbb{Q}$	Ensemble des nombres rationnels

Dans le chapitre précédent, la méthode de partitionnement de séries temporelles a été développée pour pouvoir définir l'état de santé des données d'exploitations utilisées ici. Les résultats obtenus serviront à déduire l'état de santé probable de l'actionneur étudié pour ensuite extrapoler la valeur de sa durée de vie restante. Pour ce faire, une méthode à partir de mesures de similarité est développée. Pour la replacer dans le contexte de la littérature, la première section présente un état de l'art des méthodes de Prognostics and Health Management (PHM) à base de mesures de similitudes, principalement axé sur ces trois dernières années. Puis, la méthode développée pendant ces travaux de thèse est détaillée et étayée d'exemples à partir de données d'exploitation. Enfin, la Remaining Useful Life (RUL) est obtenue et présentée.

4.1 État de l'art des méthodes de pronostic par alignement

Pour réussir à diagnostiquer et à pronostiquer l'état de santé d'un actionneur avec peu de données, une méthode à partir d'alignements de séries temporelles a été développée. Esquissons les contours de ce type de méthode pour expliquer en quoi elle pourra effectuer à la fois l'étape de diagnostic et l'étape de pronostic dans cet ensemble de méthodes de PHM.

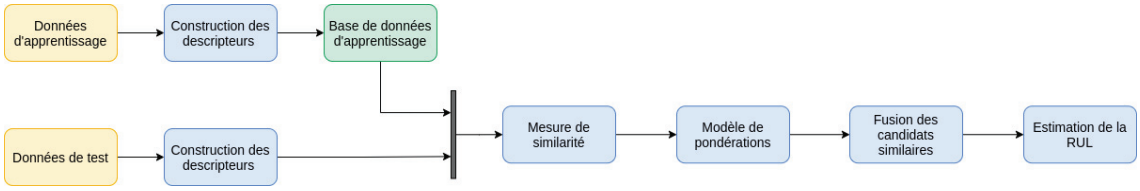


FIGURE 4.1 – Schéma de principe pour toutes les méthodes Trajectory Similarity-Based Prediction (TSBP)

La première mention d'une méthode à base de similarité pour la prédiction de RUL a été faite par [T. WANG *et al.*, 2008] pour l'adapter à l'ensemble de données Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) [SAXENA et GOEBEL, 2008]. Plus tard, [YOU et MENG, 2011] décrira un *framework* détaillé appliqué à la surveillance de joints de soudures Ball Grid Array (BGA) en environnements vibrants ou encore sur un modèle stochastique de données simulées [Ming-Yi YOU et Guang MENG, 2013a] sous la dénomination Similarity-based Residual Life Prediction (SbRLP). Les avantages d'une telle approche sont multiples. Elle permet la réalisation d'un pronostic avec peu de données disponibles, ce qui est souvent le cas dans un milieu industriel. De plus, au fur et à mesure de la durée de vie du produit, la base de données de comparaison pourra être enrichie et de ce fait, sans changement de méthode, la détermination de la RUL sera d'autant plus précise. Cependant, si la base de données n'est pas suffisamment riche,

les prédictions seraient entachées de fortes incertitude ; ceci présente l'inconvénient majeur de la méthode.

La littérature sur la prédiction de RUL à partir de mesures de similitudes ou d'alignement de séries temporelles fait appel à des méthodes uniquement à base de traitement de données ou des méthodes hybrides. L'ensemble de données le plus utilisé dans la littérature est l'ensemble de données C-MAPSS créé par [Saxena et Goebel, 2008], qui contient des signaux multivariés pour étudier la dégradation d'un turboréacteur simulé. Comme le montre la figure 4.2, il représente 63 % des données utilisées sur les quatre dernières années.

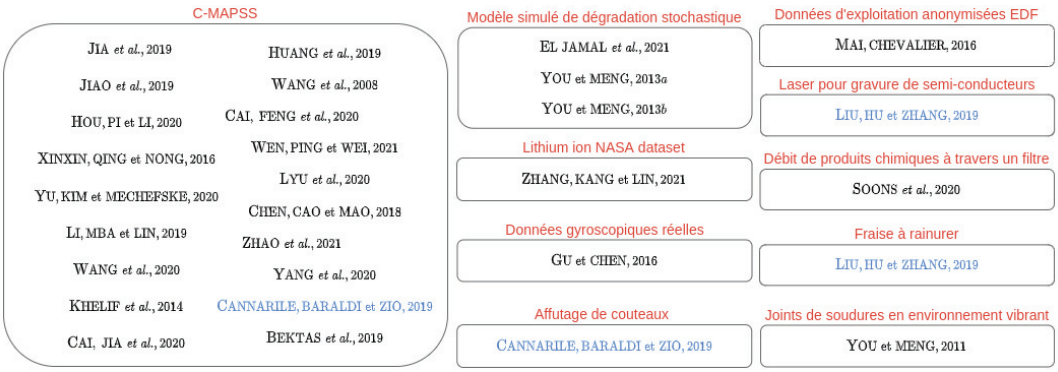


Figure 4.2 – Principaux ensembles de données utilisés pour tester les algorithmes de l'état de l'art

L'avantage d'un ensemble commun de données est de pouvoir fournir des résultats comparables entre les travaux des différentes équipes de recherches. Cependant, l'utilisation massive de C-MAPSS [Saxena et Goebel, 2008] témoigne du problème récurrent de manque de données industrielles exploitables dans le domaine du PHM. En effet, les quelques ensembles de données industrielles utilisés sont pour la plupart confidentiels, de sorte que la reproductibilité des résultats entre études n'est pas garantie. Malgré cela, et malgré la qualité démontrée de l'ensemble de [Saxena et Goebel, 2008], cet ensemble académique de données permet de couvrir uniquement l'étude de plusieurs réalisations d'un seul système en aéronautique. Pour pallier cela, [El Jamal *et al.*, 2021], [Ming-Yi You et Guang Meng, 2013a] et [Ming-Yi You et Guang Meng, 2013b], utilisent des modèles stochastiques simulés représentant la tendance d'une dégradation. Ces travaux s'intéressant au diagnostic et au pronostic d'actionneurs électromécaniques, comme présenté à la sous-section 3.2.3, l'ensemble des signaux vibratoires anonymisés propriétés de Safran Electronics & Defense sera utilisé.

Après la sélection des données d'exploitation et l'extraction de descripteurs pour l'étude, il faut quantifier la similarité entre deux séries temporelles. Pour ce faire, une grande partie de l'état de l'art utilise la distance euclidienne évaluée sur une fenêtre glissante pour élaborer une RUL, somme pondérée de toutes les RUL intermédiaires obtenues pour chaque comparaison entre un descripteur test et un descripteur d'apprentissage. C'est le cas de [Jiao *et al.*, 2019] qui évalue la RUL à partir de descripteurs —autrement appelés Health Indicator (HI) —extraits par la méthode des moindres carrés partiels couplée à une régression à vecteurs de supports ou Partial Least Square Support Vector Regression (PLS-SVR). Une fenêtre glissante est appliquée simultanément sur le descripteur test et le descripteur d'intérêt de la base d'entraînement. La mesure de similitude consiste à calculer la distance euclidienne entre chaque couple de points de la fenêtre pour ensuite sommer le tout. À chacun des intervalles ainsi créé un poids, proportionnel à l'indice temporel de l'intervalle, est affecté. Les échantillons les plus récents sont alors considérés comme plus pertinents pour la dégradation. La RUL est obtenue en réalisant une somme pondérée des k descripteurs de la base de données ressemblant le plus aux données testées. Notons qu'en plus de la taille de la précédente fenêtre glissante, le nombre de descripteurs choisi pour le modèle de fusion dépend de l'application. De même, [Bektas *et al.*, 2019] extrait d'une matrice de signaux multivariés des HI normalisés, filtrés et standardisés pour augmenter la performance de la prédiction de la RUL. Ce pré-traitement est effectué avec un ensemble de réseaux de neurones. La trajectoire de dégradation de test est alors comparée à celles qui sont présentes en base de données, ce qui permet de sélectionner dix candidats similaires. Les données utilisées allant jusqu'à fin de vie du système (Run To Failure (R2F)), la RUL est alors la moyenne du comportement de ces dix candidats. À partir des signaux bruts, sans

filtrage, la même mesure de similarité est adoptée par [Wen, Ping et Wei, 2021]. Pour [H. Zhao *et al.*, 2021], une analyse en composantes principales (Principal Component Analysis (PCA)) permet d'extraire un HI global pour l'ensemble des données capteurs. Le signal obtenu est modifié par un filtre de Kalman puis normalisé. Ensuite l'algorithme Particle Swarm Optimization (PSO) est utilisé en conjonction avec une Support Vector Regression (SVR) pour prédire le comportement futur de l'indicateur HI. Les particules créées sont comparées aux points d'apprentissages pour obtenir un score de similarité dont dérive les poids du modèle de fusion. Les 5 premiers scores de similarités sont alors sélectionnés comme pondération pour obtenir la RUL finale. Ces derniers travaux étant menés pour des données d'apprentissage allant jusqu'à la fin de vie (R2F), [Z. Chen, S. Cao et Mao, 2018] ont développé une méthode qui pourrait fournir des résultats pertinents à partir de trajectoires de dégradation incomplètes. Lorsque les données R2F sont disponibles, la mesure de similarité est la distance euclidienne sur fenêtre glissante. Dans le modèle de fusion, comme pour [Jiao *et al.*, 2019], pour favoriser le comportement des points les plus récents dans les données récoltées, une pondération proportionnelle à la nouveauté de la donnée est affectée à la mesure. Dans le cas où les données qui vont jusqu'à fin de vie ne sont pas disponibles, une machine à vecteurs de supports —Support Vector Machine (SVM) —est entraînée à partir des données R2F pour pouvoir détecter le changement de schéma de dégradation dans les données incomplètes et ainsi permettre le calcul de la RUL. Une autre approche consiste à apprendre les coefficients d'un modèle de régression pour créer des HI monotones, sans bruits, à partir des données capteurs multi-dimensionnelles. Cette approche est adoptée par [Bektas *et al.*, 2019] pour construire une librairie d'indicateurs d'états de santé. Lorsque les données tests lui sont comparées, si le résultat de la mesure de similarité est supérieur à un seuil (l'indicateur montre une similarité parfaite lorsqu'il est unitaire, sinon il est nul), alors la fenêtre glissante est conservée. De même, plus les données sont récentes, plus la pondération dans le modèle de fusion est importante. Cette procédure crée un indicateur pour chaque trajectoire qui est proportionnel au nombre de fenêtres similaires trouvées sur le nombre de fenêtres glissantes totales calculées dans la comparaison entre le signal test et celui de référence. Dans ces travaux la taille de la fenêtre était de 30 points et 7 voisins étaient utilisés pour l'obtention de la RUL. Dans [X. Yang *et al.*, 2020], tous les descripteurs sont centrés, réduits puis agrégés en un seul signal par la méthode Multiple Linear Regression (MLR) développée par [C.-G. Huang *et al.*, 2019]. Cet algorithme permettrait de conserver la tendance de dégradation originale des données d'entrée. En plus de la taille de la fenêtre glissante, un seuil de distance est fixé. Au-delà de ce seuil, les échantillons ne seront pas considérés comme assez similaires pour pouvoir permettre la prédiction d'une RUL. Les mesures de similarités sont effectuées et les résultats récupérés servent de *coefficients de développement* pour un modèle dont le comportement est régi par une équation différentielle du premier ordre. Il permet de calculer les instants suivants des trajectoires de dégradation. Le modèle doit être entraîné avec la base de références pour obtenir un ensemble de paramètres pertinents. Enfin, le temps de vie restant est déterminé comme la fusion du dernier instant récupéré et de la valeur du modèle. Le fait d'avoir un modèle ayant appris à partir de données d'apprentissage permet de s'affranchir de la dépendance des données R2F pour le pronostic. Toujours avec le même principe de mesure de similitude, [Hou, Pi et B. Li, 2020], [Yu, I. Y. Kim et Mechefske, 2020] et [M. Wang *et al.*, 2020] utilisent de l'apprentissage profond pour obtenir la prédiction du temps de vie restant. En effet, [Hou, Pi et B. Li, 2020] développe une machine de Boltzman restreinte —Restricted Boltzmann Machine (RBM) —pour extraire des HI à la manière d'un auto-encodeur —Auto-Encoder (AE) —, la relation entre l'espace d'entrée et l'espace latent étant surveillée pour la détection de défauts. Avec une RBM qui a appris à encoder un état sain, toute déviation entre l'entrée et la sortie témoigne de l'apparition d'un défaut sur le système. Un bi-Long Short Term Memory (LSTM) est ensuite utilisé pour prédire les instants suivants des séries temporelles tests. Les descripteurs obtenus par la RBM étant bruités, ils sont lissés par une moyenne glissante avant la mesure de similarité. La RUL finale est obtenue par moyenne pondérée de toutes les RUL similaires, la RUL étant cette fois continue par morceaux [Hou, Pi et B. Li, 2020, figure 5, section 3.3]. Dans cette modélisation, il est important de noter que le point de rupture de pente est déterminant pour les résultats d'extrapolation de la RUL. Dans le cas de [Yu, I. Y. Kim et Mechefske, 2020], un Recurrent Neural Network (RNN) couplé avec un AE extrait des descripteurs à partir des données brutes puis un modèle de régression exponentielle leur est appliqué pour les rendre monotones. Toujours appliqué aux données de [Saxena et Goebel, 2008], [X. Li, Mba et T. Lin, 2019] construisent à partir des données multi-dimensionnelles des HI par la méthode Canonical Variate Analysis (CVA). Ces descripteurs peuvent alors être utilisés dans la régression d'un modèle exponentiel couplé à un filtre à particules - Particle Filter (PF). Intrinsèquement, d'après la structure d'un PF, une mesure

d'incertitude est obtenue pour la RUL. Toujours sur le même ensemble de données, [Xinxin, Qing et Nong, 2016] étudient les performances de trois méthodes différentes pour la prédiction du temps de vie restant : Hidden Semi-Markov Model (HSMM), SVM et une méthode à base de similarité. Dans un premier temps, plusieurs HI sont extraits avec un modèle de régression linéaire à partir des données brutes. Un modèle exponentiel paramétré est alors adapté à chacun des HI, après filtrage par moyenne glissante, dans la méthode par similarité : ce sont les trajectoires qui sont comparées afin d'obtenir la RUL. Les 7 meilleurs candidats, au regard de la mesure de similarité euclidienne, sont sélectionnés pour donner la RUL finale. Chacune des méthodes permet d'obtenir un temps de vie restant différent, l'idée étant que chacune puisse apporter une information indépendante. La méthode de fusion d'entropie commune de [J. Xu, Yusheng Wang et L. Xu, 2014] permet d'obtenir des pondérations spécifiques à chaque RUL intermédiaire pour obtenir la RUL finale. Une méthode différente de toutes les précédentes est développée par [Cannarile, Baraldi et Zio, 2019] à base de théorie de la preuve —Evidence Theory (EvT) —pour obtenir la RUL avec un intervalle de confiance sur la prédiction.

D'autres travaux de la littérature ont développé une approche de mesure de similarité qui n'est pas fondée sur la distance euclidienne. C'est le cas de [Jia, Cai *et al.*, 2019] qui, dans un premier temps, sélectionne les descripteurs par leur monotonie avec le test de Mann-Kendall. Ensuite la Maximum Mean Discrepancies (MMD) [Gretton *et al.*, 2012] calculée sur une fenêtre glissante est utilisée comme mesure de similarité puis le Kernel Two Sample Test (KTST) [Gretton *et al.*, 2012] permet de définir les candidats les plus ressemblants à la séquence de test. La RUL est calculée en ajustant les paramètres d'une distribution de Weibull, ce qui permet d'obtenir le temps de vie restant. Le même principe est utilisé par [Cai, Jia *et al.*, 2020] qui utilisent la MMD pour calculer la divergence entre deux distributions de données. Ensuite la KTST permet de sélectionner un ensemble de distributions similaires. Les calculs sont ici toujours effectués sur une fenêtre glissante dont la taille est un hyper-paramètre sélectionné par l'utilisateur en fonction des résultats obtenus. Les premières étapes de la mesure de similarité sont identiques dans les travaux de [Cai, Feng *et al.*, 2020]. Un filtre à particules PF est rajouté et permet d'échantillonner une population dont la moyenne et l'écart-type de leur distribution sont calculés à partir des résultats de similarité précédents. La médiane et la mesure d'incertitude sont données pour chaque itération à partir de l'état du nuage de particules. Pour mettre à jour les résultats du modèle, une fonction de transition est utilisée pour donner le Rao-Blackwellized Particle Filter (RBPF), partant du principe que la dégradation est exponentielle. En résumé, l'état initial du nuage est donné par la MMD et la mise à jour par RBPF. Dans les travaux de [M. Wang *et al.*, 2020], les descripteurs sont extraits à partir d'un auto-encodeur. Ils utilisent la distance Complexity Invariant Distance (CID) [Batista *et al.*, 2014] qui mesure la complexité d'une série par sa taille. Un facteur de normalisation est ensuite ajouté au résultat. Une fenêtre glissante de la taille du segment à aligner (le test) est initialisée et la mesure de similarité est appliquée à tous les segments avec un pas unitaire (pour le parcours de la fenêtre glissante). Une moyenne pondérée avec les RUL intermédiaires est effectuée. Seules les RUL qui possèdent un score de similarité au moins égal à 70 % du maximum de similarité de l'ensemble participent à la construction de la RUL finale.

Pour [C.-G. Huang *et al.*, 2019] un problème majeur des méthodes de prédiction de RUL par TSBP provient du fait que le résultat obtenu est présenté sans intervalle de confiance, ce qui rend les méthodes difficile à appliquer dans des domaines où la représentation et le management de l'incertitude est critique, notamment dans le domaine de ces travaux : l'aéronautique. Certains travaux le prennent donc en compte, comme [Jia, Cai *et al.*, 2019], [Cai, Jia *et al.*, 2020] ou encore [Cai, Feng *et al.*, 2020] où l'ajustement des données à une courbe de Weibull permet d'obtenir une mesure de l'incertitude sur la prédiction de la RUL ou encore les caractéristiques de la population de particules d'un filtre. D'autres méthodes utilisent, elles, l'estimation de densité par noyau [C.-G. Huang *et al.*, 2019][Lyu *et al.*, 2020]. Dans [C.-G. Huang *et al.*, 2019], à partir des signaux normalisés, des trajectoires de dégradation sont générées à partir d'un modèle exponentiel. Le minimum de similitude pour chaque trajectoire est sélectionné et une fonction de pondération permet de fusionner toute l'information pour obtenir le temps de vie restant. Enfin une dernière moyenne pondérée est effectuée pour obtenir le temps de vie restant final. La Kernel Density Estimation (KDE) permet de calculer l'incertitude due aux calculs de cycles avec une méthode de sélection de bande passante spécifique à un noyau gaussien (voir méthode de [Botev, Grotowski et Kroese, 2010]). Enfin le risque de deuxième espèce, β, est utilisé pour définir un intervalle de confiance sur le résultat de la RUL finale. Pour [Lyu *et al.*, 2020], la mesure de similarité utilisée est la Dynamic Time Wrapping (DTW) pour mesurer la similarité entre deux segments de tailles différentes. Une matrice rectangulaire contenant la distance euclidienne entre

chaque paire de points est créée pour les deux segments. Le chemin qui traverse la matrice en minimisant les coûts (éléments de la matrice) sur son passage permet d'obtenir un ensemble de valeurs dont la somme représente la mesure de similarité. Pour trouver ce chemin, un algorithme de recherche spécifique ayant une complexité algorithmique linéaire est utilisé. Ainsi, il est possible de définir la ressemblance entre deux segments de données. Le problème des vitesses de dégradation et sauts de temps sont aussi abordés en définissant une fonction d'ajustement pour les données. Les poids pour le modèle de fusion final, pour l'obtention de la RUL, sont choisis par un modèle dit *auto-adaptatif*. Deux séries de poids sont générées, chacune à partir d'un modèle d'exponentielles décroissantes en fonction de la similitude d'un segment et de l'erreur absolue entre la séquence test et la référence. Enfin, une estimation de densité par noyau permet d'obtenir la RUL finale pour un modèle de RUL linéaire par morceaux [Lyu *et al.*, 2020, figure 7]. Au-delà de l'aéronautique, dans le domaine des batteries lithium-ion avec l'ensemble des données de [Saha et Goebel, 2007], le travail de [S.-J. Zhang, R. Kang et Y.-H. Lin, 2021] s'intéresse à déterminer leurs trajectoires de dégradation. L'image de la dégradation est modélisée par mesure de la capacité de la batterie. Dans les batteries, un phénomène de régénération apparaît lors du vieillissement, ce qui pourrait modéliser les opérations de maintenance dans le domaine des actionneurs électromécaniques. C'est pourquoi la dégradation est modélisée par un processus incertain. Initialement les paramètres de ce processus sont déterminés par la résolution du principe incertain des moindres carrés [S.-J. Zhang, R. Kang et Y.-H. Lin, 2021, section 3.3]. Il s'agit alors de calculer la distance entre le *degré de croyance obtenu* et *la distribution supposée de l'incertitude*. Le premier est calculé à partir de *la médiane de la fonction approximative de rang* [S.-J. Zhang, R. Kang et Y.-H. Lin, 2021, Figure 2]. Pour ce faire, les points d'une série sont triés par ordre croissant et un degré de confiance est attribué à chacun en fonction de sa place dans la série triée. Puis, ils sont affectés aux points de la série initiale. La distribution supposée, quant à elle, est de la forme $1/(1+\exp x)$. C'est pour la mise à jour des paramètres du processus que la mesure de similarité est utilisée. La différence au carré qui était originalement calculée entre une distribution incertaine et un ensemble de degrés de confiance [S.-J. Zhang, R. Kang et Y.-H. Lin, 2021, équation 8] est alors pondérée par un ensemble de poids [S.-J. Zhang, R. Kang et Y.-H. Lin, 2021, équation 9]. Pour calculer ces pondérations, deux distances sont calculées sur fenêtre glissante entre une référence et une séquence test : la distance euclidienne et la distance cosinusoïdale [S.-J. Zhang, R. Kang et Y.-H. Lin, 2021, équation 10]. Le maximum de leur inverse respectif permet de créer deux variables intermédiaires, qui, une fois normalisées entre 0 et 1 sont sommées puis divisées par deux [S.-J. Zhang, R. Kang et Y.-H. Lin, 2021, équation 14]. Cette opération est répétée pour chaque fenêtre glissante, de ce fait, l'ensemble de poids de la fonction de coût à optimiser est obtenu. Pour le pronostic d'état de santé de filtres traversés par un débit de produits chimiques, [Soons *et al.*, 2020] modélisent les trajectoires de dégradation par des fonctions exponentielles paramétrées. La mesure de similarité utilisée est alors la distance euclidienne et la distance de Manhattan évaluées sur une fenêtre glissante. Par une fusion classique, la RUL est obtenue. La mesure d'incertitude de la RUL est obtenue par modélisation bayésienne. Notons que les travaux antérieurs de [Yingchao Liu, X. Hu et W. Zhang, 2019] présentent une méthode de mesure de similarité pour trois ensembles différents de données : des données industrielles de fraise à rainurer, des trajectoires de dégradation de lasers pour la gravure de semi-conducteurs [Z. Liu *et al.*, 2017] ainsi qu'un modèle de dégradation stochastique simulé. Enfin, les travaux de [Ming-Yi You et Guang Meng, 2013b] présentent une méthode pour mesurer la robustesse de la prédiction (la capacité du modèle prédictif à être invariant au bruit de mesure du processus de dégradation) ainsi qu'une méthode pour la mesure d'incertitude à partir de validation croisée. De plus [Ming-Yi You et Guang Meng, 2013b] étudient l'importance du schéma des pondérations des mesures de similarité intervenant aussi dans la prédiction d'une RUL globale à partir des RUL intermédiaires. Les données utilisées sont les trajectoires de dégradation d'un modèle stochastique simulé.

Le travail de [El Jamal *et al.*, 2021] souligne le fait que les données d'exploitation ne sont quasiment jamais obtenues jusqu'à la fin de vie pour des raisons soit financières, soit de sûreté. En effet, il serait difficile d'imaginer un actionneur de commande de vol en opération jusqu'à sa fin de vie complète, en sachant que toute panne de ce dernier s'avèrerait fatale pour l'aéronef. C'est pourquoi il présente une approche à base de mouvement brownien adaptatif —Brownian Motion (BM) —permettant ainsi de pallier le problème de manque de données R2F. Les données ont été captées pour des conditions opérationnelles identiques. Les HI sont filtrés puis le BM est utilisé pour extrapoler les HI obtenus. La mesure de similarité est appliquée à un HI test et pour un descripteur de référence. La distance euclidienne est utilisée pour une mesure de similarité sur une fenêtre glissante de position fixe à taille croissante. La référence minimisant la mesure de similarité

est sélectionnée. Les travaux de [Gu et Y. Chen, 2016] sur des données gyroscopiques réelles se concentrent sur l'apport de deux améliorations aux méthodes de similarité de l'état de l'art : la prise en compte des différentes maintenances et des changements de conditions opérationnelles. Ces informations héritées de l'environnement sont alors incluses dans les poids calculés pour obtenir la RUL finale.

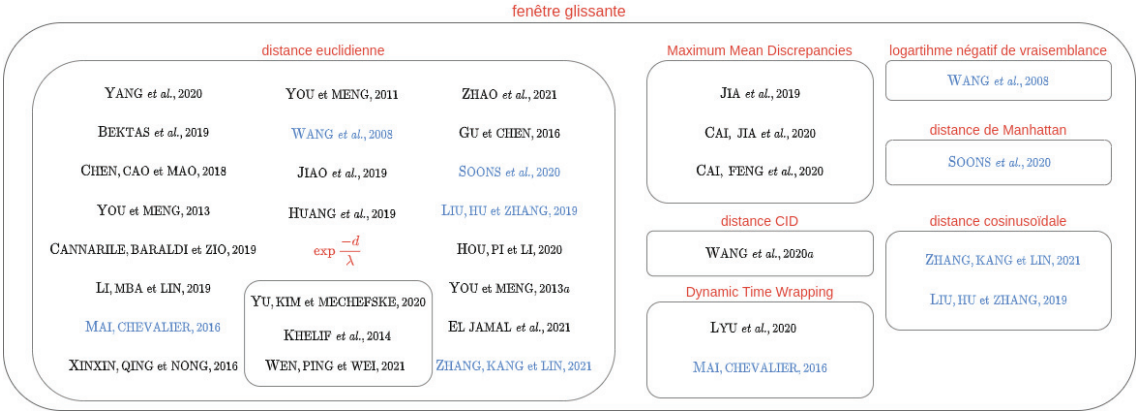


Figure 4.3 – Distances utilisées pour mesurer la similarité de deux séries temporelles dans l'état de l'art

Enfin, pour sortir du domaine purement académique et faire le lien avec le domaine industriel qui nous intéresse dans ces travaux, à partir de données d'exploitation anonymisées, [Mai et Chevalier, 2016] présentent la manière dont pourrait être adaptée la méthode de pronostic à base de similarités dans un environnement *big data* appliqué, dédié à l'étude des séries temporelles. Dans un premier temps, ces dernières sont centrées. Ensuite deux types de normalisations sont effectuées sur les données : une normalisation en amplitude et une normalisation en temps. La normalisation en temps est particulièrement intéressante pour notre problématique. En effet, soit deux signaux $x_1(t)$ et $x_2(t)$ alors la normalisation en temps est définie comme : $x_1(t\frac{\delta t_1}{\delta t_2}) = x_2(t)$, avec δt_1 et Δt_2 les différences respectives entre deux éléments des séries t_1 et t_2. Cela permet de synchroniser les indices temporels d'un signal compressé sur un signal qui ne le serait pas. En l'état, la formule donnée prend en compte un facteur de compression uniforme sur toute la séquence. Les sections suivantes montreront que notre méthode de pronostic permet de réaliser cette normalisation temporelle et ce même avec un facteur de compression variable. C'est un point d'intérêt car dans la pratique, avec les potentielles variations d'environnements, il est peu probable que deux produits se dégradent de manière uniforme l'un par rapport à l'autre. Après la double normalisation, [Mai et Chevalier, 2016] utilisent une fenêtre glissante avec une distance euclidienne pour synchroniser deux séries temporelles entre elles. Comme dans tous les travaux précédents, la synchronisation consiste à aligner le premier point du signal $x_1(t)$ sur son plus proche voisin dans le signal $x_2(t)$. L'algorithme DTW est aussi mentionné pour contribuer à la construction d'un indice de similarité global obtenu avec un schéma spécifique de pondérations.

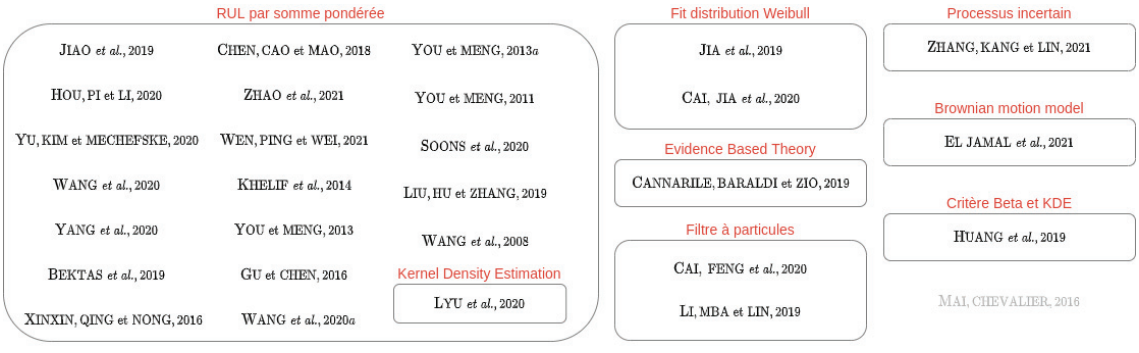


Figure 4.4 – Méthodes utilisées pour la détermination de la durée de vie restante dans l'état de l'art

Enfin, pour donner une vision globale des méthodes utilisées dans l'état de l'art, la figure 4.3 et la figure 4.4 représentent l'ensemble des mesures de similarité utilisées par la littérature rapportée ici ainsi que l'ensemble des méthode utilisées pour le pronostic du temps de vie restant. Remarquons que 79 % des travaux de recherche utilisent la distance euclidienne tandis que seulement 9 % utilisent la divergence statistique MMD.

Concernant l'obtention de la RUL, 69 % de la littérature la construit en tant que somme, pondérée par le score de similarité, de RUL intermédiaires. Dans ce contexte, une RUL intermédiaire représente une estimation de durée de vie restante à partir d'un descripteur uniquement. La RUL globale est obtenue en réalisant la fusion de toutes ces précédentes estimations. La proportion d'usage de 69 %, proche de celle de la distance euclidienne, permet de mettre en exergue une tendance classique dans l'état de l'art, l'utilisation de la distance euclidienne sur une fenêtre glissante permettant d'obtenir un schéma pertinent de pondérations pour la fusion de RUL finale, selon l'état de l'art.

Au-delà de cette tendance commune, [Jia, Cai *et al.*, 2019] font le constat que la plupart des méthodes développées ne prennent pas en compte la mesure d'incertitude du résultat obtenu. En effet, en rassemblant ces méthodes dans la figure 4.5, seule 34 % de la littérature étudiée propose une telle méthode.

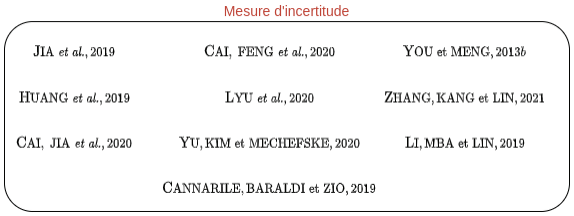


Figure 4.5 – Références de la littérature ayant implémenté une mesure d'incertitude pour la détermination de durée de vie restante

Le pronostic à base de méthodes reposant sur une mesure de similarité est encore peu utilisé dans la littérature [Jia, Cai *et al.*, 2019]. Cela pourrait être dû au fait que le nombre de critères d'évaluation de similarité est limité, ce qui se traduit par une utilisation massive de la distance euclidienne sur une fenêtre glissante par exemple. De par cette construction, il est alors difficile de définir un seuil au-delà duquel la ressemblance de deux segments serait actée. Enfin les résultats de ce type de méthodes sont fortement dépendants de la richesse de la base de données de comparaison, ce qui, dans un milieu industriel, est compliqué à obtenir et coûteux.

La méthode développée dans les sections suivantes permet de se démarquer par rapport à l'état de l'art sur plusieurs points. Tout d'abord, contrairement à la plupart des travaux où un travail de sélection et de filtrage des descripteurs est nécessaire pour obtenir des résultats de pronostic cohérents, notre méthode a été appliquée sur un ensemble de descripteurs (voir chapitre 3.2) simplement filtrés par un polynôme de degré 2 avec l'algorithme de [Savitzky et Golay, 1964]. La sélection de descripteurs pertinents est, quant à elle, induite dans la création de l'ensemble de pondérations. De plus, la mesure de similarité entre deux séries temporelles ne s'effectue pas sur une fenêtre glissante et n'utilise pas une simple distance euclidienne entre les points tests et les points d'apprentissage. De ce fait, cette mesure est valable pour toute taille de série. À la différence de l'algorithme DTW qui assigne tous les points du test à un intervalle donné par l'utilisateur dans les séries en base d'apprentissage, notre procédure d'alignement déduit automatiquement l'intervalle sur lequel effectuer l'assignation. Ensuite, la fonction d'évaluation de similarité est une nouvelle fonction de coût développée spécialement pour quantifier les résultats d'alignements de séries. De cette fonction de coût sera dérivée des pondérations pour obtenir la RUL finale, non pas à partir d'une somme pondérée, mais comme solution d'un problème de transport optimal. De surcroît, elle permet la quantification de l'incertitude du résultat de pronostic obtenu. Cette méthode sera appliquée sur le jeu de données anonymisées de vieillissements d'actionneurs électromécaniques en notre possession. Bien que ce dernier ne soit pas conséquent, nous verrons que les résultats de la méthode sont cohérents et de par sa structure, la méthode de pronostic permet une mise à l'échelle rapide des résultats de pronostic lorsque, en production, plus de données pourront être récupérées sur des systèmes en vols. Enfin, bien qu'utilisée avec des profils R2F ici, ce dernier point n'est pas limitant car la méthode s'applique pour tous signaux.

Dans les section suivantes, sera développée une nouvelle méthode d'alignement de séries temporelle s'appliquant à des séquences de tailles variables 4.2.2 ainsi que sa fonction de coût associée pour quantifier chaque alignement 4.2.3. La conjonction de ces deux derniers algorithmes permet alors la création d'une nouvelle mesure de similarité robuste entre séries temporelles. Une fois que les candidats en base de données ont été triés par pertinence pour la série inconnue, une nouvelle mesure d'incertitude 4.2.5 est développée pour obtenir une distribution de l'estimation de l'état de santé des données inconnues. Enfin, en comparant ce dernier résultat au profil de degrés de sévérité obtenus au chapitre 3.3, le degré de sévérité associé à l'état actuel et le temps de vie restant de l'actionneur est déterminé avec une mesure d'incertitude 4.3 et la généralisation de la méthode à plusieurs environnement opérationnels est abordée.

4.2 Méthode de diagnostic et de pronostic par alignement de séries temporelles

4.2.1 Principe général de la méthode

La méthode de maintenance prédictive développée repose sur l'alignement de séries temporelles. Cela consiste à associer les points d'une série à une autre. L'association est alors faite par minimisation d'un critère entre les points associés et les points à associer. Pour faire cela, les données étudiées sont séparées en deux ensembles disjoints :

- Les points de la série à aligner qui seront appelés les *points sources*.
- Les points de la série modèle sur lesquels seront alignés les points de la série source. Ils seront appelés les *points cibles*.

L'objectif est donc d'aligner les points *sources* sur les points *cibles*.

Dans la pratique, soit une série temporelle $X = (x_i)_{1\leq i\leq n}$ quelconque de taille $n \in \mathbb{N}$ appelée série *source* et une série temporelle $Y = (y_j)_{1\leq j\leq m}$ de taille $m \in \mathbb{N}$ appelée série *cible*. Dans le cas étudié où $n \leq m$, notons $\mathcal{I}$ (resp. $\mathcal{J}$) l'ensemble des valeurs des indices de X (resp. Y).

Hypothèse 6. *Le nombre de points des séries* sources *inconnues est inférieur au nombre de points des séries* cibles *connues qui constituent la base de données.*

L'objectif de l'alignement est de trouver une fonction injective $\mathcal{T} : \mathcal{I} \hookrightarrow \mathcal{J}$ qui associe à chaque indice d'un point de X, l'indice de son point voisin dans Y. Cette fonction $\mathcal{T}$ est définie comme étant injective car tous les points de Y ne sont pas nécessairement associés à des points de X. Dans le cas étudié où la série *source* possède moins de points que la série *cible*, la propriété est alors vérifiée. Dans le cas contraire, un ré-échantillonnage de la série cible serait nécessaire. La fonction $\mathcal{T}$ caractérise alors l'alignement de la série X sur Y. De manière plus précise pour notre étude, des signaux vibratoires captés sur $q = 4$ actionneurs supposés indépendants (voir hypothèse 2) sont extraits $p = 28$ descripteurs statistiques de m points (voir sous-section 3.2.3). L'ensemble de ces signaux forme une base de données notée $\mathfrak{M}_{m,p,q}(\mathbb{R})$. Les indices $1 \leq i \leq n$ (ou m), $1 \leq j \leq p$, $1 \leq l \leq q$ et $1 \leq h \leq q-1$ sont introduits pour permettre un schéma de notation homogène dans les sections suivantes. Pour les besoins de la thèse, un actionneur sera sélectionné pour simuler une série de données qui auraient été captées en fonctionnement opérationnel. Ainsi, ces données seront contenues dans une matrice $\mathbb{X}^l \in \mathcal{M}_{n,p}(\mathbb{R})$ qu'il faudra comparer aux données contenues dans un tenseur $\mathfrak{Y} \in \mathfrak{M}_{m,p,(q-1)}(\mathbb{R})$. Il est possible de déterminer l'état de santé de l'actionneur dont proviennent les signaux inconnus en les comparant à ceux présents en base de données. Cette base de données peut être issue d'essais accélérés dans un environnement donné ou encore de profils de fonctionnement réels enregistrés en vol. La série source X précédemment définie sera alors une colonne indexée par j, donc un descripteur, de la matrice $\mathbb{X}^l$. D'où $X_j^l = \mathbb{X}^l[:, j] \in \mathcal{M}_{n,1}(\mathbb{R})$. La cible quant à elle sera notée $Y_j^h = \mathfrak{Y}[:, j, h] \in \mathcal{M}_{m,1}(\mathbb{R})$. L'objectif de ces travaux est alors d'aligner la série supposée inconnue X_j^l sur l'ensemble des séries $Y_j^h, \forall h \in [\![1; q-1]\!]$. En d'autres termes, un descripteur sera aligné sur le même descripteur de chaque actionneur. L'idée est ensuite d'ordonner les candidats Y_j par ordre de ressemblance à X_j^l. Pour ce faire, une fonction de coût $\mathcal{L}$ est alors définie pour associer à chaque fonction d'assignation $\mathcal{T}$ une valeur réelle. Une étape de fusion d'information pour chacun des paramètres classé est alors réalisée pour permettre l'estimation de la durée de vie consommée et la donnée d'une distribution représentative de sa qualité.

Cette méthode d'alignement permet alors d'extraire des données une information à la fois sur le diagnostic et le pronostic du système. En effet, pour le diagnostic, les séries en base de données étant labellisées par degré de sévérité (voir méthode section 3.3), une fois que la source X_j^l sera alignée sur Y_j^h pour les différents h, le label de chaque point de Y_j^h sera assigné aux points de X_j^l. Enfin, pour le pronostic, l'abscisse du dernier point aligné de la source dans la cible permet d'obtenir le temps de vie restant de l'actionneur dont les descripteurs ont été calculés. La RUL est alors le temps restant entre le dernier point aligné de la source et le dernier point de la cible. La figure 4.6 illustre le principe par un schéma simplifié pour uniquement deux séries temporelles différentes.

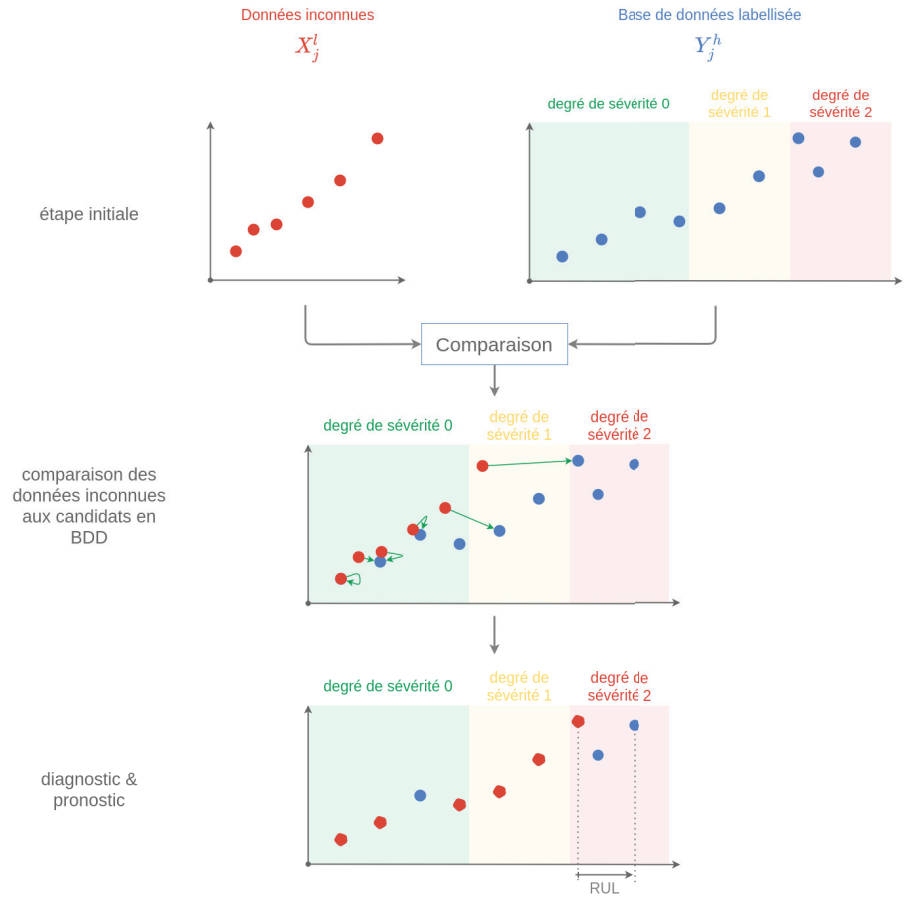


FIGURE 4.6 – Schéma de principe pour le diagnostic et le pronostic par alignement de séries temporelles

Maintenant que les grandes lignes de la méthode ont été présentées, détaillons l'alignement de séries temporelles par déplacements itératifs.

4.2.2 Alignement de séries temporelles par déplacements itératifs

Dans l'état de l'art, l'assignation partielle d'un nuage de points sources à un nuage de points cibles en n dimensions (n est ici différent du nombre de points de la source X) par des techniques de transport optimal a été présentée dans les travaux de [BONNEEL et COEURJOLLY, 2019]. La méthode développée, Sliced Partial Optimal Transport (SPOT), a aussi été appliquée au transport d'histogramme pour transférer l'espace colorimétrique d'une image à une autre. L'avantage de cette dernière méthode est sa complexité algorithmique maîtrisée qui lui permet une résolution rapide du problème pour des nuages composés d'un nombre important de points. Ainsi, dans leur article, [BONNEEL et COEURJOLLY, 2019, section 6.4] utilisent leur méthode pour un nuage de points sources avec 8000 points et 10000 points pour le nuage cible. Sur une machine de 16 cœurs, l'alignement est réalisé en une fraction de secondes. La figure 4.7 illustre le principe de l'algorithme.

Les deux ensembles source X_j^l et cible Y_j^h définis à la sous-section 4.2.1 sont représentés sous la forme de nuages de points à l'étape 0 de la figure 4.7. Dans les travaux originaux de [BONNEEL

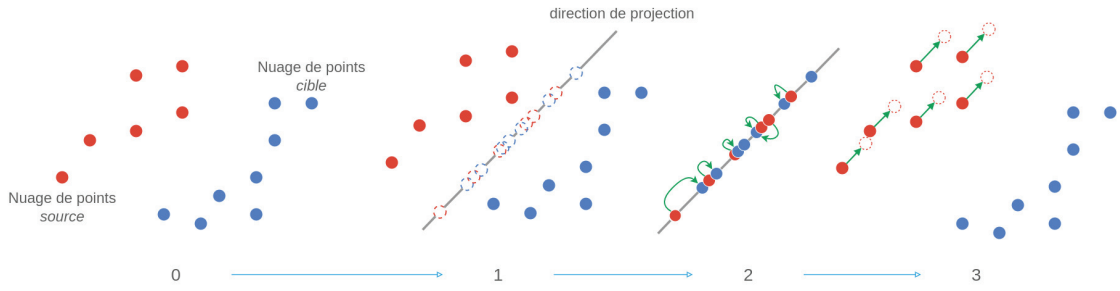


FIGURE 4.7 – Schéma de principe pour une itération de la méthode itérative d'alignement de séries temporelles de [BONNEEL et COEURJOLLY, 2019]

et COEURJOLLY, 2019], la méthode SPOT est appliquée, entre autres, à des nuages de points représentant des formes en 3 dimensions. Dans notre cas d'étude, il convient alors de trouver une représentation de séries temporelles qui pourrait convenir à l'usage de SPOT. Représentée dans un plan, il est alors naturel de penser la série temporelle comme un objet en deux dimensions : une dimension pour l'axe des temps et une dimension pour les valeurs associées à chaque instant. Ainsi, du vecteur X_j^l une matrice $\tilde{\mathbb{X}}_j^l \in \mathcal{M}_{n,2}(\mathbb{R})$ peut être créée. Chaque colonne de $\tilde{\mathbb{X}}_j^l$ contient la concaténation du vecteur $X_j^l = (x_i)_{1\leq i\leq n,j}^l$ et d'un vecteur $T_j^l = (t_i)_{1\leq i\leq n,j}^l \in \mathcal{M}_{n,1}(\mathbb{R})$. À chaque élément $(x_i)_j^l$ est alors associé un indice temporel $(t_i)_j^l$. Dans notre cas d'étude, $(t_i)_j^l$ ne représente pas la valeur de l'horodatage absolu des points $(x_i)_j^l$ mais un numéro de cycle c'est à dire un incrément allant de 0 à n (le nombre de points de la source). Ainsi, ce nouveau vecteur T_j^l rend compte uniquement de la relation de précédence entre les éléments de X_j^l. Le nuage cible $\tilde{\mathbb{Y}}_j^l \in \mathcal{M}_{m,2}(\mathbb{R})$ est défini de manière similaire à la différence près qu'il possède plus de points que la source donc $m > n$. Une série temporelle est alors considérée comme un nuage de points de dimension 2. La première étape de l'algorithme est donc réalisée. À l'étape suivante, une direction est choisie pour permettre la projection des nuages de points sur une seule et unique dimension. Le problème d'assignation ne se résout alors plus en 2 dimensions mais dans ces espaces projetés en 1 dimension. L'étape 2 permet ensuite d'obtenir la fonction $\mathcal{T}$ pour l'ensemble des points considérés. Cette fonction, définie plus haut à la sous-section 4.2.1, assigne à chacun des points des sources projetées leurs points voisins dans les cibles projetées. Enfin, l'étape 3 définie comment les points de la source se déplaceront vers leurs voisins dans la cible. Dans les travaux de [BONNEEL et COEURJOLLY, 2019], cette étape est résolue par descente de gradient suite à la résolution d'un problème de transport optimal dans chaque espace projeté dont la solution permet d'assigner à chaque point de la source X_j^l une quantité de déplacement. Ce déplacement s'effectuant dans l'espace en 2 dimensions, les quantités de mouvement calculées précédemment pour chaque point en 1 dimension sont ajoutées aux coordonnées des points de la source. Les points du nuage source X_j^l (en rouge sur la figure 4.7) se déplacent selon la direction de projection choisie à l'étape 1 (le déplacement étant représenté par les vecteurs en vert sur la figure 4.7). Notons que sur la figure 4.7, les cercles en pointillés représentent la position après projection, ou après déplacement, des points pleins correspondants.

Testée avec les profils de nos séries temporelles, notre implémentation de la méthode de [BONNEEL et COEURJOLLY, 2019] ne fournissait cependant pas de résultats pertinents. Certains points de la source convergeaient vers des intervalles dans la cible qui ne présentaient aucune similarités avec les intervalles d'origine. La dépendance temporelle des points entre eux n'était parfois pas conservée lors du déplacement du nuage. Cette dernière caractéristique est cependant indispensable pour l'étude de séries temporelles et l'interprétation des résultats, notamment pour diminuer l'incertitude lors de l'estimation de la durée de vie restante. Ce dernier point sera étudié dans les sections suivantes. Aussi, l'algorithme original de [BONNEEL et COEURJOLLY, 2019] a été modifié dans le cadre de ces travaux. La méthode résultant de cette modification permet d'aligner des séries temporelles de tailles quelconques, moyennant la prise en compte de l'hypothèse 6, hypothèse également faite dans les travaux originaux de [BONNEEL et COEURJOLLY, 2019]. Le schéma schéma de principe de la nouvelle méthode développée est présenté à la figure 4.8.

Contrairement à la figure 4.7, les étapes 1 et 3 ont été modifiées et les points de la source ne sont plus séparés en ensembles de points disjoints. À l'étape 1 de la méthode, tous les points sources sont projetés sur une direction non plus aléatoire mais « optimale » puis à la dernière étape, le

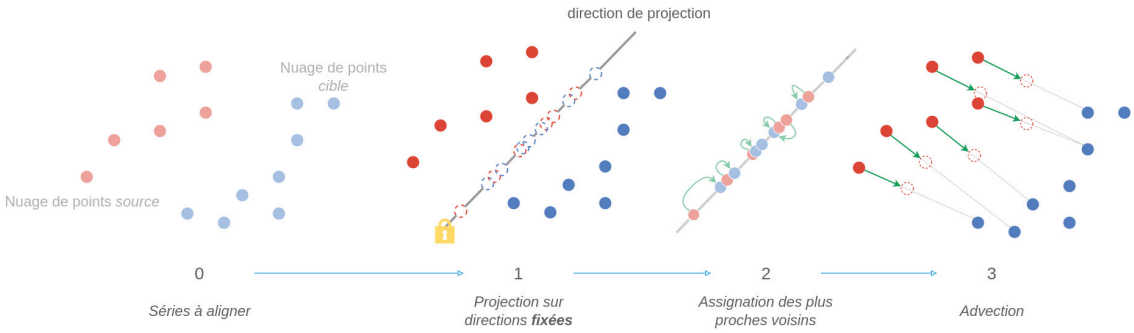


FIGURE 4.8 – Schéma de principe pour une itération de la méthode itérative développée dans ce manuscrit d'alignement de séries temporelles (les étapes inchangées ont été masquées)

déplacement des points sources n'est plus calculé dans l'espace de projection mais directement dans l'espace final en deux dimensions. Le transport optimal a été remplacé par un calcul de quantité de déplacement dans l'espace euclidien et les points de la source se déplacent vers les points voisins de la cible qui leurs ont été assignés. Notons que l'étape 2, bien que représentée différemment de son homologue dans la figure 4.7, consiste uniquement en la recherche d'une fonction d'assignation. Elle s'adapte juste ici à des données d'une structure différente mais l'algorithme original de recherche des plus proches voisins de [BONNEEL et COEURJOLLY, 2019] n'est pas modifié.

Étape 1 - La recherche de directions de projections optimales Détaillons maintenant l'étape 1 modifiée, consistant à choisir les directions de projection pour résoudre le problème d'assignation des plus proches voisins dans un espace en 1 dimension. Soit Θ l'ensemble des directions de projection. Cet ensemble est alors un échantillonnage de la sphère unitaire des directions $\mathcal{S}^{d-1}$, avec d le nombre de dimensions du problème. Pour toute dimension considérée, et plus particulièrement pour notre espace en 2 dimensions d'intérêt pour l'alignement de séries temporelles, Θ est constitué de droites. Dans les travaux de [BONNEEL et COEURJOLLY, 2019], Θ était un échantillonnage aléatoire de S^{d-1}. Ce caractère aléatoire n'était pas pertinent pour nos travaux. En effet, les signaux étudiés possèdent des dynamiques bien spécifiques que l'étape de projection fait parfois disparaître lors de la projection sur une direction spécifique. L'assignation des plus proches voisins de l'étape 2 était alors réalisée avec des signaux aux comportements trop éloignés des signaux originaux avant projection. De ce fait, l'alignement final n'était pas pertinent. Par conséquent, il convient de trouver un ensemble $\Theta^* \subset \Theta$ contenant un ensemble de droites fournissant un bon résultat d'alignement final. La méthode d'alignement étant itérative, une direction θ_k est alors affectée pour chaque itération. C'est pourquoi l'ensemble Θ^* constitué de plusieurs directions $(\theta_0^*, \theta_1^*, \ldots, \theta_K^*)$ doit être la solution minimisant un critère de qualité. Ce critère de qualité sera la fonction de coût $\mathfrak{L}$, une variante de la fonction de coût $\mathcal{L}$ construite plus loin à la sous-section 4.2.3. Ce problème d'optimisation est formalisé à l'équation 4.1.

$$\begin{aligned}
&\Theta_j^{h,*} = \min_{(\theta_1,\theta_2,\ldots\theta_K)_j^l} \sum_{k=0}^{K} \mathfrak{L}\left(\{\mathbb{X}_{\mathcal{T}}|_k\}_j^l\right) \\
&\text{avec } \mathbb{X}_{\mathcal{T}}|_k \in \mathcal{M}_{n,k}\left(\mathbb{R}\right) \text{ tel que : } \mathbb{X}_{\mathcal{T}}|_k = \begin{bmatrix} X_{\mathcal{T}}|_{k=0} & X_{\mathcal{T}}|_{k=1} & \ldots & X_{\mathcal{T}}|_{k=k} \end{bmatrix}^\top \\
&\text{avec } X_{\mathcal{T}}|_k \in \mathcal{M}_{n,1}\left(\mathbb{R}\right) \text{ tel que : } X_{\mathcal{T}}|_k = \mathrm{T}\left(\tilde{X}_j^l \theta_{k,j}^h, \tilde{Y}_j^h \theta_{k,j}^h\right)
\end{aligned} \tag{4.1}$$

Dans l'équation 4.1, $\Theta_j^{h,*}$ représente l'ensemble des directions optimales déterminées pour le descripteur j et l'actionneur h. L'ensemble est obtenu en minimisant la somme des coûts de déplacement, sur une dimension, des différents états du nuage source à chaque itération. $\{\mathbb{X}_{\mathcal{T}}|_k\}_j^h$ rassemble ces coûts de déplacement. Pour la première itération $k = 0$, $\mathbb{X}_{\mathcal{T}}|0 = X_{\mathcal{T}}|_{k=0}$, pour la deuxième itération $k = 1$, $\mathbb{X}_{\mathcal{T}}|_1 = \begin{bmatrix} X_{\mathcal{T}}|_{k=0} & X_{\mathcal{T}}|_{k=1} \end{bmatrix}^\top$ et pour la dernière itération K $\mathbb{X}_{\mathcal{T}}|_K = \begin{bmatrix} X_{\mathcal{T}}|_{k=0} & X_{\mathcal{T}}|_{k=1} & \ldots & X_{\mathcal{T}}|_{k=K} \end{bmatrix}^\top$. L'ensemble $\{\mathbb{X}_{\mathcal{T}}|_k\}_j^l$ sur lequel est calculé la fonction de coût $\mathfrak{L}$ est donc de taille K. Chaque état intermédiaire de la source le constituant $X_{\mathcal{T}}|_k$ est le résultat à la k^{e} itération du transport du nuage source $\tilde{X}_j^l$ sur la cible $\tilde{Y}_j^h$. La fonction permettant ce transport, et par conséquent la création de la fonction $\mathcal{T}$ qui assigne chaque point de la source aux points de la cible, est la fonction T appliquée aux deux nuages projetés $\tilde{X}_j^l \theta_{k,j}^h$ et $\tilde{Y}_j^h \theta_{k,j}^h$. Ne

pouvant trouver de solution théorique simple à l'équation 4.1, l'ensemble $\Theta_j^{h,*}$ sera déterminé par une procédure d'optimisation itérative. Le principe est alors d'échantillonner le domaine de définition de l'ensemble Θ pour y trouver un candidat optimal au regard de l'équation 4.1. Notons que le résultat de l'optimisation n'est pas ici une direction mais un ensemble de directions associées chacune à une itération de la méthode d'alignement, c'est pourquoi $\Theta_j^h = (\theta_1, \theta_2, \dots \theta_K)_j^h$. Au total ce sont alors $p \times (q-1) = 28 \times 3 = 84$ ensembles de directions à trouver, soit $p \times (q-1) \times K = 84K$ directions à trouver pour réaliser l'alignement en base de données. L'idée sous-jacente étant qu'après le déplacement du nuage source modifie sa forme et la direction de projection précédemment choisie n'est peut-être plus optimale. La même fonction de coût est utilisée pour chaque itération. L'implémentation de la méthode Tree-structured Parzen Estimator Sampler (TPESampler) [Bergstra *et al.*, 2011] [Ozaki *et al.*, 2020] de la librairie *Optuna* [Akiba *et al.*, 2019] est utilisée pour permettre cette recherche. Afin de réduire les coûts de calculs, seulement deux directions $(\theta_0, \theta_1)_j^h$ avec $1 \leq j \leq p$ et $1 \leq h \leq q-1$ sont calculées par descripteur. Les directions de projections deviennent alors indépendantes de l'itération courante de la méthode et le changement de direction dans cette dernière sera réalisé après évaluation d'une condition sur l'alignement en cours. Plus précisément la direction θ_0 est utilisée tant que le déplacement de la source est important, dès que l'accélération de ce dernier devient moindre, la direction θ_1 est alors choisie afin de limiter la convergence de la source vers un minimum local. Le nombre de directions de projection est ainsi grandement réduit. Il reste à trouver 56 directions au total. La série source est alors alignée sur la série cible et dès lors qu'une certaine valeur de coût est atteinte, la direction de projection θ_1 est remplacée par la direction θ_2 et l'alignement se poursuit. L'optimisation 4.1 étant coûteuse en ressources, la librairie [Liaw *et al.*, 2018] est utilisée pour distribuer les calculs sur l'infrastructure d'un serveur de calculs [1] constitué entre autres, d'un bi-processeur Intel® Xeon® Silver 4215R (8 cœurs) et 192 GB de Random Access Memory (RAM). Pour l'alignement étudié dans les sections suivantes, le calcul a été réalisé en 15 heures, 39 minutes et 3 secondes. Maintenant qu'un ensemble de directions optimales $\Theta_j^{h,*}$ a été défini pour chaque descripteur et pour chaque actionneur, décrivons l'étape 2 de la méthode d'alignement.

Étape 2 - La fonction d'assignation source - cible Cette étape consiste à trouver la fonction d'assignation $\mathcal{T}_k$, recalculée à chaque itération k, à partir de l'assignation des plus proches voisins dans l'espace projeté entre les points du nuage source $\tilde{X}_j^l$ et les points du nuage cible $\tilde{Y}_j^h$. Pour ce faire, [Bonneel et Coeurjolly, 2019] ont développé un algorithme permettant de résoudre le problème d'alignement complet en temps quadratique, donc en $O(n^2)$. L'algorithme 3 permet cette assignation quadratique et représente l'étape 2 de la figure 4.7. Il peut être séparé en trois parties principales dont une première partie, des lignes 2 à 15, permet la construction d'une première version de la fonction d'assignation $\mathcal{T}_k$. Une deuxième partie des lignes 17 à 20 permet de décomposer l'intervalle initial en sous-intervalles de manière linéaire et une troisième partie résout le problème d'assignation sur les précédents sous-intervalles. Le tableau `nnAssignArray` représentant la fonction d'assignation $\mathcal{T}_k$ est alors obtenu. Comme illustré à la figure 4.9, le tableau contient les indices des points du nuage $\tilde{Y}_j^l$. Les indices de `nnAssignArray` représentent, quant à eux, les indices des points du nuage $\tilde{X}_j^l$. L'assignation des points du nuage source au nuage cible est ainsi faite. L'ensemble $\mathcal{I}$ est constitué des n indices des points du nuage source $\tilde{X}_j^l$, tandis que $\mathcal{J}$ est constitué des m indices des points du nuage cible $\tilde{Y}_j^h$.

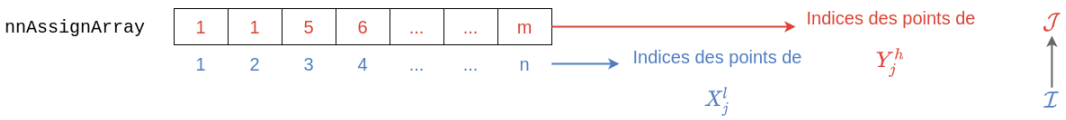


Figure 4.9 – Implémentation algorithmique de la fonction d'assignation $\mathcal{T}_k : \mathcal{I} \hookrightarrow \mathcal{J}$

La fonction d'assignation $\mathcal{T}_k$ est obtenue par la méthode Quadratic Assignment de l'algorithme 3. La méthode produit alors une assignation à partir des deux nuages projetés sur une direction $\theta_{d,j}^h \in \Theta_j^h$, du nombre n de points à aligner dans la source et du nombre m de points sur

[1] Ces travaux ont eu accès à des ressources de calculs de type High Performance Computing (HPC) de la Fédérationh Lyonnaise de Modélisation et Sciences du Numérique (FLMSN), partenaire de l'EQUIPEX EQUIP@MESO.

lesquels aligner dans la cible . Les tailles de segments à aligner sont alors stockées dans une structure `param`. Notons qu'en l'état, les segments à aligner possèdent nécessairement n et m points, car l'index du point de départ est automatiquement fixé à 1. Il est cependant possible d'étendre la méthode à des segments ne commençant pas par le premier point de la série. Pour l'alignement d'un segment de 900 points sur un segment de 1130 points, la structure sera, par exemple, initialisée comme suit : `param` $= \{\text{start}_0 = 0, \text{end}_0 = 900, \text{start}_1 = 0, \text{end}_1 = 1130\}$ dans l'algorithme 3. Après cette initialisation, une première affectation des plus proches voisins est faite par la fonction Nearest Neighbor Match définie par l'algorithme 4. Lors de cette première procédure, différents points de la source peuvent être assignés au même point dans la cible. Ces dernières répétitions sont alors comptées et contenues dans la variable `nonInjMatch` déclarée à la ligne 5 et mise à jour à la ligne 9 de l'algorithme 3. Cette information permet de décomposer la série en intervalles et ainsi de réduire la taille de l'intervalle considéré pour l'alignement. À cet effet, la fonction Reduce Range reproduisant le comportement de [Bonneel et Coeurjolly, 2019, figure 2], est utilisée. Ces nouveaux intervalles créés et représentant une fonction d'assignation bijective doivent repasser par l'application de la fonction Nearest Neighbor Match 4 à la ligne 17. Après cette deuxième assignation, les intervalles sont décomposés en un ensemble de sous-intervalles par la fonction Linear Time Decomposition décrite par [Bonneel et Coeurjolly, 2019, algorithme 2]. Cette opération représente une troisième décomposition d'une série en intervalles distincts. Une liste de structures contenant le début et la fin de chaque intervalle est alors stockée dans la variable `paramListArray`. Sur chacun de ces intervalles, une assignation des plus proches voisins est calculée à la ligne 23, puis la fonction Handle Simple Case, qui reproduit le comportement décrit dans [Bonneel et Coeurjolly, 2019, section 3.2], permet de résoudre certaines configurations en un temps linéaire. Enfin, après une dernière séquence de procédures, la fonction Simple Solve décrite par [Bonneel et Coeurjolly, 2019, algorithme 1] permet l'obtention de l'assignation finale. La fonction $\mathcal{T}_k$ a donc été générée.

Algorithme 3 Assignation quadratique, d'après [Bonneel et Coeurjolly, 2019]

Entrées :
$\tilde{X}_j^l\theta_{d,j}^h$, $\tilde{Y}_j^h\theta_{d,j}^h$ sont les nuages projetés sur la direction $\theta_{d,j}^h$
n et m sont les tailles des segments sources et cibles
Sortie :
Tableau d'assignation représentant la fonction $\mathcal{T}_k$

1: **function** Quadratic Assignment($\tilde{X}_j^l\theta_{d,j}^h$, $\tilde{Y}\Theta$, n_0, m_0)
2: **declare** globalAssignArray de length($\tilde{X}_j^l\theta_{d,j}^h$) entiers
3: **declare** param $= \{\text{start}_0 = 0, \text{end}_0 = n_0, \text{start}_1 = 0, \text{end}_1 = m_0\}$ de 4 entiers
4: nnAssignArray = Nearest Neighbor Match($\tilde{X}_j^l\theta_{d,j}^h, \tilde{Y}_j^h\theta_{d,j}^h$, param)
5: **declare** nonInjMatch = 0
6: **for** $i \leftarrow$ param.start$_0 + 1$ **to** param.end$_0$ **do**
7: **if** nnAssignArray$[i]$ = nnAssignArray$[i-1]$ **then**
8: /* *Compte le nombre de répétitions* ie. *assignation non injectives* */
9: nonInjMatch + = 1
10: **end if**
11: **end for**
12: subpbSolvedFlag, newParam = Reduce Range($\tilde{X}_j^l\theta_{d,j}^h, \tilde{Y}_j^h\theta_{d,j}^h$, param, nonInjMatch)
13: **if** subpbSolvedFlag **then**
14: **return** nnAssignArray
15: **end if**
16: /* *Maintenant que les intervalles ont été modifiés, l'assignation des plus proches voisins doit être calculée à nouveau* */
17: nnAssignArray $\leftarrow$ Nearest Neighbor Match($\tilde{X}_j^l\theta_{d,j}^h, \tilde{Y}_j^h\theta_{d,j}^h$, newParam)
18: **declare** paramListArray as a list of parameter structure
19: /* *Décompose les intervalles en sous intervalles de manière linéaire* */
20: paramListArray $\leftarrow$ Linear Time Decomposition($\tilde{X}_j^l\theta_{d,j}^h, \tilde{Y}_j^h\theta_{d,j}^h$, newParam)
21: /* *Résout l'alignement pour chaque sous intervalle trouvé* */
22: **for** $i \leftarrow 0$ **to** length(paramListArray) **do**
23: nnAssignArray $\leftarrow$ Nearest Neighbor Match($\tilde{X}_j^l\theta_{d,j}^h, \tilde{Y}_j^h\theta_{d,j}^h$, paramListArray[i])

```
        subpbSolvedFlag ← HANDLE SIMPLE CASE(X̃_j^l θ_{d,j}^h, Ỹ_j^h θ_{d,j}^h, paramListArray[i])
        if subpbSolvedFlag then
            return nnAssignArray
        end if
        for i ← param.start_0 + 1 to paramListArray[i].end_0 do
            if nnAssignArray[i] ← nnAssignArray[i − 1] then nonInjMatch ← nonInjMatch +1
            end if
        end for
        subpbSolvedFlag, newParam ← REDUCE RANGE(X̃_j^l θ_{d,j}^h, Ỹ_j^h θ_{d,j}^h, paramListArray[i], nonInjMatch)
    end for
    if subpbSolvedFlag then
        return nnAssignArray
    end if
    subpbSolvedFlag ← HANDLE SIMPLE CASE(X̃_j^l θ_{d,j}^h, Ỹ_j^h θ_{d,j}^h, paramListArray[i])
    if subpbSolvedFlag then
        return nnAssignArray
    end if
    nnAssignArray ← NEAREST NEIGHBOR MATCH(X̃_j^l θ_{d,j}^h, Ỹ_j^h θ_{d,j}^h, paramListArray[i])
    nnAssignArray ← SIMPLE SOLVE(X̃_j^l θ_{d,j}^h, Ỹ_j^h θ_{d,j}^h, paramListArray[i])
    return nnAssignArray
end function
```

Les algorithmes 3 et 4 permettent donc d'obtenir, pour une itération, la fonction d'assignation $\mathcal{T}_k$. L'étape 2 de la figure 4.7 est alors terminée. Cette fonction sera donc utilisée dans le modèle d'advection de l'étape finale, l'étape 3.

Étape 3 - Le modèle d'advection Comme schématisé à la figure 4.8, le modèle d'advection développé diffère de celui de la méthode SPOT de [BONNEEL et COEURJOLLY, 2019] où les points de la source se déplacent vers leurs voisins dans la cible d'une quantité calculée par résolution d'un problème de transport en une dimension. Ici, une quantité de déplacement est aussi calculée pour chaque point de la source mais les points se déplaceront alors d'une quantité donnée sur la direction créée entre eux et leurs plus proches voisins respectifs dans la cible. Ce principe est illustré par la figure 4.10 pour un point de la source vers deux points différents de la cible ayant été définis comme son voisin du point n pour chaque itération. Notons que sur la figure 4.10, les cercles en pointillés rouges représentent les positions précédents l'itération courante du point source.

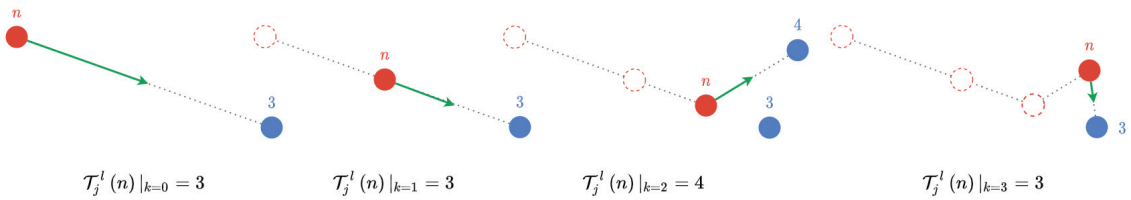


FIGURE 4.10 – Exemple du déplacement du point source n vers son voisin 3 dans la cible avec le modèle d'advection pour 4 itérations

La fonction $\mathcal{T}_j^l(n)|_{k=0}$ de la figure 4.10 est la fonction d'assignation évaluée au n^{e} point source pour la première itération de l'alignement du descripteur j de l'actionneur l. À chaque nouvelle itération de l'algorithme de transport, le nuage source se rapprochera du nuage cible jusqu'à atteindre complètement ce dernier. Sur la figure 4.10, les vecteurs de déplacement des points sources sont représentés par une flèche verte. Soit $n\,(k)$ le point source considéré à l'itération k et $m(k)$ le point cible correspondant à son plus proche voisin à l'itération k. Alors pour chaque point n et m le vecteur de déplacement a comme direction la droite passant par $n(k)$ et $m(k)$, est dirigé vers le point $m(k)$ et possède un norme, ou *quantité de mouvement*, la moitié de la distance euclidienne séparant $n(k)$ de $m(k)$. Ainsi, la quantité de mouvement (représentée par la norme de la flèche vert) est calculée à la figure 4.10 par dichotomie pour simplifier l'illustration du concept.

La quantité de mouvement du i^{e} point de la source à l'itération k avec la distance euclidienne d_E est donnée par l'équation 4.2.

$$\zeta_i(k) = \frac{d_E\left(n(k), m(k)\right)}{2} \tag{4.2}$$

Dans le cas de la méthode effectivement développée à l'algorithme 5, la quantité de mouvement est calculée en divisant la distance restante entre $n(k)$ et $m(k)$ par un facteur $\alpha \in]1; \infty)$. Soit d la distance parcourue par un point de la source, à chaque itération k une distance d_k peut être associée à un point. La suite des distances $(d_k)_{0\leq k\leq K}$ est alors créée. C'est de ce fait une suite géométrique de raison $\frac{1}{\alpha}$. Sa somme partielle a donc une expression simple $\sum_{k=0}^{K} \frac{d_0}{\alpha_k}$ qui converge pour un nombre d'itérations infinis, $K \to \infty$, vers la grandeur $\frac{\alpha d_0}{\alpha - 1}$. Le modèle d'advection atteint donc un point de stabilité. Par extension, l'algorithme d'alignement converge nécessairement vers la cible mais, en l'état, il n'est pas encore possible d'évaluer la qualité d'une telle convergence. Une fonction de coût $\mathcal{L}$ sera développée à cette intention dans la section 4.2.3 suivante.

De manière plus détaillée, la fonction ADVECTION de l'algorithme 5 produit un nuage source $\tilde{X}_{\mathcal{T}}$ à partir des deux nuages à aligner $\tilde{X}_j^l$ et $\tilde{Y}_j^h$, le tableau d'assignation `nnAssignmentArray` qui représente la fonction $\mathcal{T}_k$ ainsi que `sigmoid`, un tableau contenant la variable précédente α pour chaque direction. Les lignes 2 à 4 initialisent les structures pour calculer la distance totale parcourue par le nuage source. Les lignes 6 à 9 définissent x_{target} et y_{target} qui représentent les coordonnées des points à rejoindre dans la cible $\tilde{Y}_j^h$ tandis que x_{src} et y_{src} représentent les coordonnées des points de départ dans la source. La distance entre ces deux points est alors calculée à la ligne 12 et les quantités de déplacement $\text{x}_{\text{displacement}}$ et $\text{y}_{\text{displacement}}$ sont calculées aux lignes 15 et 16 comme proportion de la distance entre la source et la cible. Les coordonnées du nuage de points source $\tilde{X}_j^l$ sont alors mises à jour aux lignes 18 et 19. Enfin, un coût d'advection est calculé comme étant la somme normalisée des distances parcourues par chaque point source et le nuage source déplacé $\tilde{X}_{\mathcal{T}}$. Notons ici que le coût d'advection est différent de la fonction de coût $\mathcal{L}$ développée dans la section 4.2.3 suivante.

Algorithme 4 Assignation des plus proches voisins, d'après [BONNEEL et COEURJOLLY, 2019]

Entrées :
X_j^l, Y_j^h deux séries de points
param une structure contenant les paramètres des segments source et cibles à aligner
$param = \{\text{start}_0, \text{end}_0, \text{start}_1, \text{end}_1\}$
Sortie :
nnAssignArray est un tableau d'assignation de plus proches voisins de X_j^l sur Y_j^h

```
1:  function NEAREST NEIGHBOR MATCH(X_j^l, Y_j^h, param)
2:      declare nnAssignArray de length(X_j^l) entiers
3:      /* Initialise le cursor au premier élément de la cible */
4:      cursor ← param.start_1
5:      /* Parcours de la source */
6:      for i ← param.start_0 to param.end_0 do
7:          /* Initialise la variable à FLOAT_MAX_VALUE */
8:          minDist ← 3.40 × 10^38
9:          minTargetIndex ← −1
10:         lockIndex ← max(param.start_1, cursor)
11:         /* Parcours de la cible à partir du point courant de la source */
12:         for k ← lockIndex to param.end_1 do
13:             dist ← (X_j^l[i] − Y_j^h[k])^2
14:             /*Conserve l'indice du dernier point voisin*/
15:             cursor ← k − 1
16:             if dist ≤ minDist then
17:                 minDist ← dist
18:                 minTargetIndex ← k
19:             else
```

```
            if dist > minDist then break
            end if
        end if
    end for
    nnAssignArray[i] ← minTargetIndex
  end for
  return nnAssignArray
end function
```

Algorithme 5 Advection de la source sur la cible

Entrées :
$\tilde{X}_j^l$ et $\tilde{Y}_j^h$ sont les nuages de points sources et cibles
nnAssignmentArray est le tableau des plus proches voisins de X vers Y
projSrcWIndex est le tableau de la source projetée avec les indices associés
projTargetWIndex est le tableau de la cible projetée avec les indices associés
sigmoid contient deux hyperparamètres pour le déplacement, un pour chaque dimension

Sortie :
$\tilde{X}_{\mathcal{T}}$ est le nuage des points sources transportés sur la cible
advectCost représente le coût de mouvement ie. mouvement de la source vers la cible

function ADVECTION($\tilde{X}_j^l$, $\tilde{Y}_j^h$, nnAssignmentArray, projSrcWIndex, projTargetWIndex, sigmoid)
  advectCost ← 0
  cumXDisplacement ← 0
  distArray of size length($\tilde{X}_j^l$) of integers
  for $i \leftarrow 0$ **to** length(projSrcWIndex) **do**
    $\mathrm{x}_{\mathrm{target}} \leftarrow \tilde{Y}_j^h[\text{projTargetWIndex}[\text{nnAssignmentArray}[i]]]_x$
    $\mathrm{y}_{\mathrm{target}} \leftarrow \tilde{Y}_j^h[\text{projTargetWIndex}[\text{nnAssignmentArray}[i]]]_y$
    $\mathrm{x}_{\mathrm{src}} \leftarrow \tilde{X}_j^l[\text{projSrcWIndex}[i]]_x$
    $\mathrm{y}_{\mathrm{src}} \leftarrow \tilde{X}_j^l[\text{projSrcWIndex}[i]]_y$

    /* *Calcul de la distance euclidienne séparant les points sources des points cibles* */
    $\text{distArray}[i] \leftarrow \sqrt{(\mathrm{y}_{\mathrm{target}} - \mathrm{y}_{\mathrm{src}})^2 + (\mathrm{x}_{\mathrm{target}} - \mathrm{x}_{\mathrm{src}})^2}$

    /* *La sigmoïde permet de conserver uniquement une proportion de distance* */
    $\mathrm{x}_{\mathrm{displacement}} \leftarrow \text{sigmoid}[0] \times (\mathrm{x}_{\mathrm{target}} - \mathrm{x}_{\mathrm{src}})$
    $\mathrm{y}_{\mathrm{displacement}} \leftarrow \text{sigmoid}[1] \times (\mathrm{y}_{\mathrm{target}} - \mathrm{y}_{\mathrm{src}})$
    $\text{cumXDisplacement} \leftarrow \text{cumXDisplacement} + \mathrm{x}_{\mathrm{displacement}}$
    $\tilde{X}_j^l[\text{projSrcWIndex}[i]]_x \leftarrow \tilde{X}_j^l[\text{projSrcWIndex}[i]]_x + \mathrm{x}_{\mathrm{displacement}}$
    $\tilde{X}_j^l[\text{projSrcWIndex}[i]]_y \leftarrow \tilde{X}_j^l[\text{projSrcWIndex}[i]]_y + \mathrm{y}_{\mathrm{displacement}}$
  end for
  $\text{advectCost} \leftarrow \text{advectCost} + \left(\sum_i \text{distArray}[i]\right) / \text{length}(\tilde{X}_j^l)$
  $\tilde{X}_{\mathcal{T}} \leftarrow \tilde{X}_j^l$
  return advectCost, $\tilde{X}_{\mathcal{T}}$
end function

Maintenant que toutes les étapes des figures 4.7 et 4.8 ont été détaillées, l'algorithme global 6 a été développé pour rassembler toutes ces fonctions et obtenir l'alignement final d'une série temporelle X sur une série Y.

La fonction TIME SERIES ALIGNMENT produit le nuage de points sources déplacé $\tilde{X}_{\mathcal{T}}$ à partir des deux nuages de points $\tilde{X}_j^l$ et $\tilde{Y}_j^h$, de l'ensemble des directions optimales trouvées à l'équation 4.1 $\theta_j^{h,*}$ et d'un tableau contenant une suite d'hyperparamètres `weightArray`. La fonction TIME SERIES ALIGNMENT réalise alors K itérations des étapes 1, 2 et 3 de la figure 4.8 tant que le déplacement cumulé des points de la source vers la cible n'est pas nul. La variable `cost` est déclarée

à la ligne 7 par une valeur non nulle, et mise à jour à la ligne 56 par la valeur absolue d'un coût calculé par la fonction $\mathfrak{L}$. Notons que le coût sera considéré nul lorsque qu'il sera inférieur à une valeur limite 10^{-16}. Cela permet d'éviter que l'algorithme continue les itérations alors que le coût de déplacement est de l'ordre de la précision machine et donc potentiellement une erreur numérique. Maintenant que le critère d'arrêt automatique a été présenté, abordons le fonctionnement de l'algorithme 6. Les premières lignes de l'algorithme 6 permettent l'initialisation de variables et la ligne 6 permet de conserver les valeurs initiales du nuage de points sources $\tilde{X}_j^l$. Pour rappel, ce dernier est défini comme $\tilde{X}_j^l = (t_i, x_i)^l_{1\leq i\leq n,j}$, la notation $\tilde{X}_j^l[.,:]$ ne conserve que les $(x_i)_{1\leq i\leq n}$ soit la série initiale X_j^l c'est à dire $\tilde{X}_j^l[.,:] = (x_i)_{1\leq i\leq n}$. La boucle itérative commence alors par tester, à la ligne 8, si le coût calculé `cost`, différent du coût de déplacement du nuage, n'est pas négatif. Tel est le cas lorsque les points du nuage source ne convergent plus que très lentement vers la cible. C'est pourquoi, pour conserver une vitesse de convergence optimale, l'index de la direction de projection est modifié. *De facto* les points du nuage source déplacé seront projetés sur la seconde et dernière direction de l'ensemble $\Theta_j^{h,*}$. Comme dans l'algorithme 3, la projection d'une matrice sur une droite est traduite par une multiplication matricielle. En effet, avec $\tilde{X}_j^l \in \mathcal{M}_{n,2}(\mathbb{R})$ et $\Theta_j^{h,*}[i] \in \mathcal{M}_{2,1}(\mathbb{R})$, l'opération de la ligne 22 crée un vecteur colonne $\tilde{X}_j^l\Theta_j^{h,*}[i] \in \mathcal{M}_{n,1}(\mathbb{R})$ stocké dans `projSrcWIndex` avec son vecteur d'indices respectifs. La même opération est effectuée dans l'algorithme 6 à la ligne 26 pour le nuage cible. Les tableaux `projSrcWIndex` et `projTargetWIndex` sont ensuite classés par ordre croissant de leurs valeurs contenues dans la première colonne de ces deux structures. Cette étape permet alors de classer les points projetés tout en conservant leurs indices respectifs, et donc leur place, dans le nuage initial. Deux nouveaux vecteurs colonnes `projSrc` et `projTarget` sont alors créés aux lignes 35 à 38 avec les points ordonnés. La fonction Quadratic Assignment décrite à l'algorithme 3 est alors appliquée à ces vecteurs pour obtenir la valeur de la fonction d'assignation $\mathcal{T}_k$ via le tableau `nnAssignArray`. Par la suite, des hyperparamètres, déterminés manuellement pour l'instant, sont choisis pour créer deux valeurs d'une fonction sigmoïde. Ces deux valeurs pondéreront le déplacement sur les deux dimensions du nuage source vers le nuage cible. Comprises entre 0 et 1 elles permettent de modérer le déplacement de la source pour éviter une convergence trop rapide vers un alignement non optimal. La fonction Advection de l'algorithme 5 produit alors le nuage déplacé $\tilde{X}_\mathcal{T}$ à la ligne 49. Ce dernier est ajouté aux états de la source $\mathbb{X}_j^l$ et le coût local est mis à jour. La répétition de ces étapes jusqu'à atteindre un déplacement nul de la source vers la cible permet d'aligner deux séries temporelles X et Y de tailles quelconques l'une sur l'autre.

Algorithme 6 Alignement de séries temporelles (1ère partie)

Entrées :
$\tilde{X}_j^l$ et $\tilde{Y}_j^h$ sont les nuages de points sources et cibles
$\Theta_j^{h,*}$ est une liste contenant les deux directions de projection optimales
weightArray est un tableau contenant deux poids

Sortie :
$\tilde{X}_\mathcal{T}$ est le nuage de points sources transporté sur la cible

```
1:  function Time Series Alignment(X̃_j^l, Ỹ_j^h, Θ_j^{h,*}, weightArray)
2:      dirIndex ← 0
3:      flag ← 1
4:      noIter ← 1
5:      countIter ← 1
6:      𝕏_j^l ← X̃_j^l[, :]
7:      cost ← 42
8:      while cost > 0 do
9:          if localCost < 0 then
10:             if flag then
11:                 dirIndex ← dirIndex +1
12:                 flag ← 0
13:             else
14:                 break
15:             end if
16:         end if
```

```
        /* Projection et indexation des nuages de points */
        dirPoint ← Θ_j^{h,*}[dirIndex]
        projSrcWIndex ← tableau de (length(X̃_j^l), 2) entiers
        projTargetWIndex ← tableau de (length(Ỹ_j^h), 2) entiers
        for i ← 0 to length(X̃_j^l) do
            projSrcWIndex ← (X̃_j^l Θ_j^{h,*}[i], i)
        end for

        for j ← 0 to length(Ỹ_j^h) do
            projTargetWIndex ← (Ỹ_j^h Θ_j^{h,*}[i], i)
        end for
        projSrc ← tableau de length(X̃_j^l) entiers
        projTarget ← tableau de length(Ỹ_j^h) entiers
        /* Tri des valeurs projetées par valeurs */
        projSrcWIndex ← projSrcWIndex.sort(by= value)
        projTargetWIndex ← projTargetWIndex.sort(by= value)
        /* Création d'un tableau avec uniquement les valeurs projetées */
        for i ← 0 to length(X̃_j^l) do
            projSrc ← projSrcWIndex[i, 0]
        end for
        for j ← 0 to length(Ỹ_j^h) do
            projTarget ← projTargetWIndex[j, 0]
        end for

        /* Assignation des plus proches voisins en temps quadratique */
        nnAssignArray ← QUADRATIC ASSIGNMENT(projSrc, projTarget, length(X̃_j^l), length(Ỹ_j^h))

        /* Hyperparamètres pour le modèle d'advection */
        α ← 0.1
        weight_0 ← weightArray[0] × α
        weight_1 ← weightArray[1] × α
        sigmoid ← 1/(1 + exp(−α× weightArray ))
        X̃_T, alignCost ← ADVECTION(projSrc, projTarget, nnAssignmentArray, projSrcWIndex, projTargetWIndex, sigmoid)

        /* Structure contenant les valeurs intermédiaires d'une dimension du nuage source */
        𝕏_j^l ← 𝕏_j^l.append(X̃_T)

        /* Calcul d'un coût d'alignement intermédiaire */
        localCost ← 𝔏(𝕏_j^l)
        cost ← |localCost|
        if |localcost| < 1 × 10^{−16} then
            cost ← 0
            countIter ← countIter + 1
        end if
        noIter ← noIter + 1
    end while
    return X̃_T
end function
```

Maintenant que l'algorithme global d'alignement de séries temporelles a été détaillé, il est testé sur des données expérimentales. Pour ce faire, la source sera représentée par le descripteur Root Mean Square (RMS) de l'actionneur Lower Left Actuator (LLA) et la cible sera représentée par le même descripteur RMS de l'actionneur Lower Right Actuator (LRA). Le nuage de points qui est associé à chacun d'eux, est noté respectivement $\tilde{X}^{\text{lla}}_{\text{rms}}$ et $\tilde{X}^{\text{lra}}_{\text{rms}}$. Pour montrer que la méthode fonctionne avec des séries de toutes tailles, la source sera tronquée à 900 cycles et, du segment

résultant, seront tirés au sort 10 % des points pour constituer $\tilde{X}^{\text{lla}}_{\text{rms}}[0:900:0.10]$ tandis que la cible de 1130 points sera inchangée $\tilde{Y}^{\text{lra}}_{\text{rms}}[0:1130:1]$. Les résultats de l'alignement sont données par la figure 4.11.

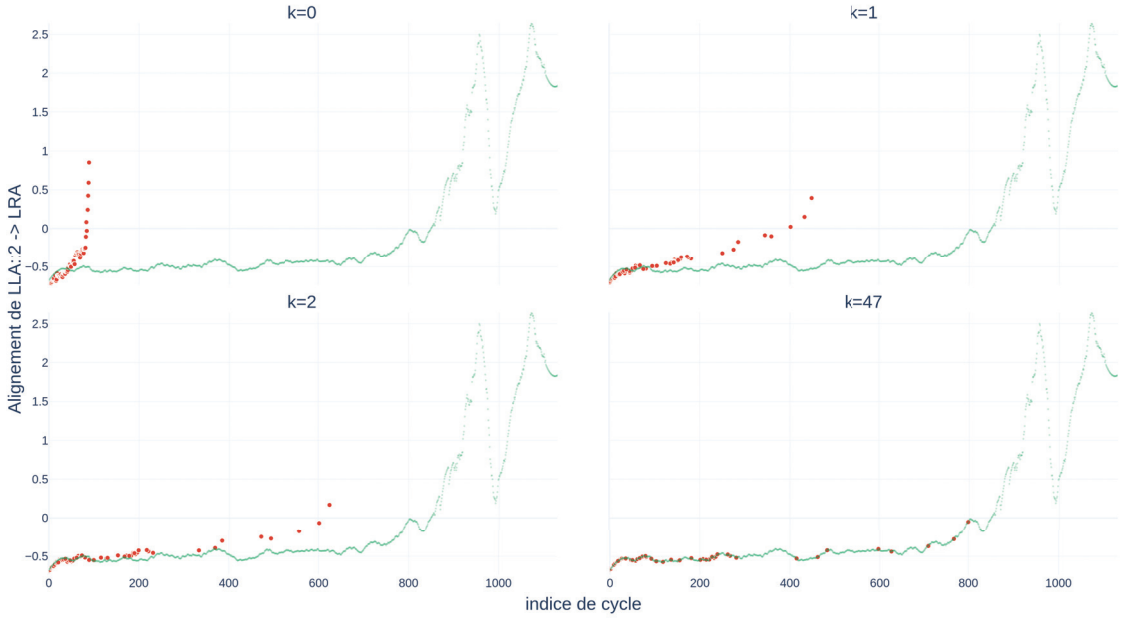


FIGURE 4.11 – Alignement de 10 % des points sélectionnés au hasard du descripteur RMS de LLA sur le descripteur RMS de LRA

La figure 4.11 représente 4 états différents du nuage source pendant la procédure d'alignement : $\tilde{X}_{\mathcal{T}}|_{k=0}$, $\tilde{X}_{\mathcal{T}}|_{k=1}$, $\tilde{X}_{\mathcal{T}}|_{k=2}$ et $\tilde{X}_{\mathcal{T}}|_{k=47}$. L'espacement des itérations k n'est pas constant pour mieux visualiser le déplacement de la source vers la cible. En effet, le déplacement est plus important lors des premières itérations de l'algorithme d'alignement. Ce résultat permet de montrer que la méthode fonctionne même pour des séries incomplètes tant que l'hypothèse 6 est vérifiée. Cependant, à cette étape, rien ne permet de quantifier la qualité d'un tel alignement. C'est pourquoi la sous-section 4.2.3 détaille le développement d'une nouvelle fonction de coût $\mathcal{L}$.

4.2.3 Quantification d'un alignement de séries

Pour pouvoir déterminer dans la base de données les candidats les plus ressemblants, il faudrait associer une valeur à chaque alignement, comme présenté en premier dans la sous-section 4.2.1. Pour ce faire, il faudrait préciser ce qu'est un alignement de série pertinent. Cependant, dans notre problème, les données récupérées formant la série X^l_j opérationnelle sont inconnues. Aussi, une fois l'alignement de X^l_j sur Y^h_j réalisé, le calcul d'une déviation par rapport à une mesure réelle connue est impossible. Il faut alors créer un indicateur $\mathcal{L}$ de qualité intrinsèque qui permettrait cette quantification sans connaissance *à priori* sur l'état des données.

Résumons la méthode développée dans la sous-section 4.2.2. Une série temporelle $X^l_j \in \mathcal{M}_{n,1}(\mathbb{R})$ doit être alignée sur une série temporelle $Y^h_j \in \mathcal{M}_{m,1}(\mathbb{R})$. Pour ce faire, chacune des deux séries est d'abord transformée en nuages de points $\tilde{X}^l_j \in \mathcal{M}_{n,2}(\mathbb{R})$ et $\tilde{Y}^h_j \in \mathcal{M}_{m,2}(\mathbb{R})$. Ces deux nuages sont alors projetés sur un même ensemble de directions $\Theta^{h,*}_j \in \mathcal{M}_{2,1}(\mathbb{R})$. Pour une direction donnée sont alors obtenues deux séries $X^l_{\perp,j}$ et $Y^h_{\perp,j}$ desquelles une fonction d'assignation injective $\mathcal{T} : \mathcal{I} \hookrightarrow \mathcal{J}$ sera construite. Les valeurs de cette fonction serviront au modèle d'advection qui engendrera le déplacement des points de $\tilde{X}^l_j$ vers ceux de $\tilde{Y}^h_j$. Le déplacement se faisant entre nuages de points, il s'effectue dans un espace en deux dimensions. Le modèle d'advection calcule alors un vecteur de déplacement pour chaque point du nuage source dans la base canonique de $\mathbb{R}^2$. Notons $\mathcal{B}_2 = (e_x, e_y)$ cette base orthonormée avec ses vecteurs constitutifs $e_x = (0,1)$ et $e_y = (1,0)$. Les étapes de projection, d'élaboration de la fonction d'assignation et de mouvement du nuage source seront alors répétées K fois pour atteindre la convergence des points de la source vers les points de la cible.

Dans cette méthode, quelle serait alors l'étape qui permettrait l'extraction d'une information judicieuse pour quantifier la qualité d'un alignement ? Étudions dans cette optique la fonction d'assignation $\mathcal{T}$ dont le comportement est illustré à la figure 4.9. Elle associe à chaque élément de la source son voisin dans la cible. Pour un alignement s'étant effectué en K itérations, définissons maintenant la matrice $\mathbb{T} \in \mathcal{M}_{n,K}(\mathbb{N})$ contenant sur chacune de ses colonnes les valeurs de la fonction $\mathcal{T}|_k$ déterminée à chaque itération tel que $\mathbb{T} = \begin{bmatrix} \mathcal{T}|_0 & \mathcal{T}|_1 & \dots & \mathcal{T}|_K \end{bmatrix}^\top$. Caractérisons un peu plus une telle fonction. $\mathcal{T}|_k$ est définie dans $\mathcal{I}_k = \{1, 2, \dots, n\}$ à valeurs dans $\mathcal{J}_k = \{1, 2, \dots, m\}$. Soit $\mathbb{T}[:, k]$ la colonne k représentant l'ensemble des valeurs prises par $\mathcal{T}_k([\![1; n]\!])$ à une itération donnée. Si la source partage des similarités avec la série cible, après avoir détaillé la méthode d'alignement à la sous-section 4.2.2, il est raisonnable de penser que les images de la fonction $\mathcal{T}$ devraient être presque identiques entre chaque itération. *A contrario*, un alignement totalement aléatoire pourrait se traduire par une variété importante parmi les ensembles images de chaque fonction $\mathcal{T}_k$. L'ensemble image d'une fonction d'assignation $\mathcal{T}_k$ est un arrangement avec répétitions à n éléments (les indices des points sources) dans un ensemble à m éléments (les indices des points cibles). Il y a alors un nombre $\mathcal{A}_m^n = m^n$, d'ensembles images possibles. Dans la matrice $\mathbb{T}$ à K colonnes, il y aurait tout de même $\binom{m^n}{K}$ combinaisons de colonnes différentes. *De facto*, pour un alignement totalement aléatoire, si la convergence de la source vers la cible n'avait pas lieu, les colonnes de $\mathbb{T}$ seraient alors toutes différentes. Autrement dit, $\forall (i, j) \in [\![1; K]\!]^2, i \neq j \Rightarrow \mathbb{T}[:, i] \neq \mathbb{T}[:, j]$. À l'inverse, dans le cas limite où l'alignement est parfaitement consistant, toutes les colonnes de $\mathbb{T}$ seraient identiques. Autrement dit, $(i, j) \in [\![1; K]\!]^2, \mathbb{T}[:, i] = \mathbb{T}[:, j]$. Selon la définition de la fonction $\mathcal{T}$ cela voudrait dire que les points de la source $\tilde{X}_j^l$ se dirigeraient tous, au fur et à mesure de l'alignement, vers les mêmes points de la cible $\tilde{Y}_j^h$. Ainsi, dans le modèle d'advection, la direction du vecteur de déplacement de chaque point $(t_i, x_i)_j^l$ de la source serait constante au fur et à mesure des itérations. Ce qui conduit à la création de l'hypothèse 7.

Hypothèse 7. *Dans le cas d'un alignement parfaitement consistant, une direction de déplacement unique est associée à chaque point source lors de tout le processus de convergence.*

L'hypothèse 7 n'est cependant pas suffisante pour construire une fonction $\mathcal{L}$ qui pourrait quantifier la qualité d'un alignement. L'hypothèse seule autorise des cas d'alignements pathologiques comme l'inversion de tous les points de X_j^l dans son alignement sur Y_j^h. Le premier point $X_j^l[1]$ serait aligné sur le dernier point $Y_j^h[m]$ tandis que le dernier point $X_j^l[n]$ serait aligné sur le premier point de $Y_j^h[1]$. Un tel alignement serait naturellement défini comme mauvais. Avec le comportement du modèle d'advection décrit à l'algorithme 5, et le fait que nos descripteurs sont, au bruit près, monotones dans le cas idéal, ce dernier exemple aurait pour conséquence de modifier grandement la valeur initiale du point du nuage source. Autrement formulé, pour K le nombre d'itérations de l'algorithme et i l'indice du point considéré, $X_j^l[i]|_{k=0} \neq X_j^l[i]|_{k=K}$. Éviter ces configurations reviendrait à limiter la variation angulaire des directions des points de $\tilde{X}_j^l$ dans le modèle d'advection. Le cas limite serait dès lors des déplacements uniquement horizontaux. De manière plus formelle, soit $\mathcal{V}_k = \begin{bmatrix} v_1 & v_2 & \dots & v_n \end{bmatrix}^\top|_k$ l'ensemble des vecteurs de déplacement créés par le modèle d'advection à une itération donnée. Pour chaque point peut être trouvé un vecteur de déplacement idéal pour l'alignement, v_i^* tel que :

$$\forall i \in [\![1; n]\!], \exists v_i^* \text{ tq. } < v_i^*, e_x > \geq < v_i, e_x > \tag{4.3}$$

Avec $< ., . >$ le produit scalaire usuel défini sur $\mathbb{R}^2$ à valeurs dans $\mathbb{R}$. Cela se traduit donc par la colinéarité du vecteur idéal v_i^* avec le vecteur de base e_x. Pour la construction de $\mathcal{L}$, l'hypothèse 8 doit donc être rajoutée.

Hypothèse 8. *Dans le cas d'un alignement de bonne qualité, tout vecteur de mouvement v_i avec $i \in [\![1; n]\!]$ doit être colinéaire au premier vecteur e_x de la base canonique $\mathcal{B}$.*

Résumons les deux objectifs précédents :

- L'hypothèse 7 imposerait à la fonction $\mathcal{T}_k$ de converger vers une unique valeur.
- L'hypothèse 8 imposerait à l'alignement de peu modifier les éléments de la source X_j^l. Celle-ci deviendrait la consigne à atteindre lors de l'alignement.

Enfin, la méthode d'alignement étant itérative, une fonction de coût construite uniquement à partir des hypothèses 7 et 8 verrait sa valeur biaisée par le nombre d'itérations pour arriver à convergence. En effet, le critère d'arrêt de la méthode ne reposant pas sur le nombre d'itérations, deux alignements différents pourraient être pénalisés par leur rapidité de convergence, critère qui ne sera pas pris en compte dans ces travaux. En effet, l'accent est porté sur la justesse estimée du résultat et non sur les performances de la méthode. Ainsi, cette dynamique pourrait se traduire par une troisième et dernière hypothèse pour la construction de $\mathcal{L}$. Un nombre d'itérations important se traduit, dans les faits, par un chemin parcouru par les points de la source plus important. Ce qui reviendrait à considérer que le signal source X_j^l serait une compression, non nécessairement uniforme, du signal cible Y_j^h. Avec l'hypothèse 9, il a été décidé de ne pas tenir compte de ces dynamiques dans le choix de la pénalité.

Hypothèse 9. *Les dilatations temporelles de signaux ne sont pas pénalisées dans la méthode d'alignement.*

Maintenant que les trois hypothèses fondamentales 7, 8 et 9 ont été énoncées pour déterminer les critères à prendre en compte dans la fonction de coût $\mathcal{L}$, intéressons-nous à construire l'objet sur lequel elle pourra s'appliquer.

Pour ce faire, définissons $\mathbb{S}_j^l \in \mathcal{M}_{n,K}(\mathbb{R})$, une matrice contenant tous les états intermédiaires de la deuxième dimension du nuage de la source $\tilde{X}_j^l[,:]_{\mathcal{T}}$ lors de la phase d'alignement. Seule la colonne du nuage source contenant les valeurs $(x_i)_{1\leq i\leq n}$ est considérée compte tenu du fait de notre volonté de ne pas pénaliser les dilatations temporelles (voir hypothèse 9). De manière un peu plus détaillée, la matrice d'états de la source sera définie à l'équation 4.4.

$$\mathbb{S}_j^l = \begin{array}{c} \begin{matrix} \tilde{X}_j^l[,:]_{\mathcal{T}}|_{k=1} & \tilde{X}_j^l[,:]_{\mathcal{T}}|_{k=2} & \dots & \tilde{X}_j^l[,:]_{\mathcal{T}}|_{k=K} \end{matrix} \\ \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1K} \\ x_{21} & x_{22} & \dots & x_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{nK} \end{bmatrix} \end{array} \tag{4.4}$$

La fonction de coût développée s'appliquera alors à la matrice $\mathbb{S}_j^l$ et produira un réel positif. Elle est définie à l'équation 4.5.

$$\begin{aligned} \mathcal{L}\colon \mathcal{M}_{n,K}(\mathbb{R}) &\to \mathbb{R}_+ \\ \mathbb{S}_j^l &\mapsto \frac{1}{nK}\sum_{i=1}^{n}\sum_{k=1}^{K}(x_{i1}-x_{ik})^2 \end{aligned} \tag{4.5}$$

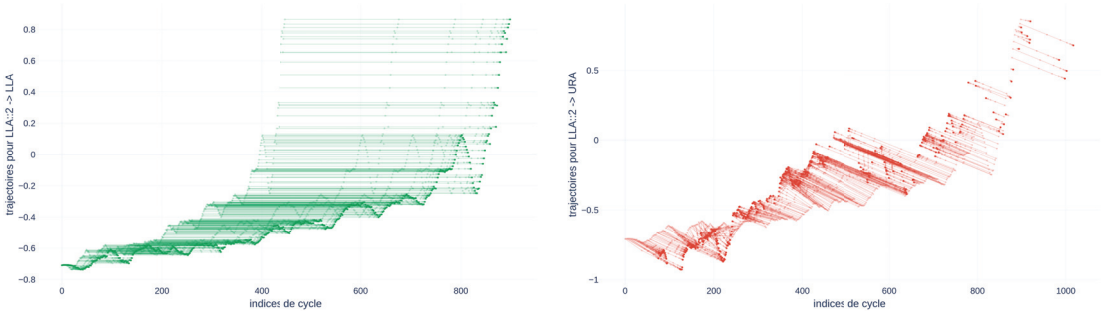
Les hypothèses 7 et 8 sont modélisées par le terme $\sum_{i=1}^{n}(x_{i1}-x_{ik})^2$ qui représente la somme des écarts quadratiques des états intermédiaires de la série à son état initial. L'hypothèse 9, quant à elle, est modélisée par le facteur de normalisation K, représentant le nombre d'itérations de l'algorithme d'alignement présent au dénominateur de l'expression 4.5. En effet, plus le nombre d'itérations pour arriver à la convergence de la source sur la cible est important, plus la dilatation de la série source est importante. En normalisant l'expression ainsi, cet effet n'impacte plus la valeur de la fonction de coût. Enfin, la somme sur les itérations de l'algorithme d'alignement $\sum_{k=1}^{K}$ permet d'agréger l'ensemble des écarts sur le nombre total d'itérations. Pour rappel, pour chaque itération k est calculée une fonction $\mathcal{T}_k$ qui assigne chaque point source à son voisin dans la cible. La valeur de la fonction $\mathcal{L}$ est donc susceptible de changer à chaque itération. L'expression de $\mathcal{L}$ étant donnée, le fait d'évaluer cette dernière nous permet de répondre aux différentes hypothèses faites pour quantifier la valeur d'un alignement. Dans le cas où les hypothèses 7, 8 et 9 sont vérifiées, la fonction de coût $\mathcal{L}$ définie à l'équation 4.5 est nulle. En effet, $\forall(i,k)\in[\![1;n]\!]\times[\![1;K]\!], x_{i1}=x_{iK}$ donc $(x_{i1}-x_{ik})^2=0$. Dans le cas où l'une des hypothèses ne serait pas vérifiée, $\mathcal{L}$ étant une double somme de carrés, on aura : $\forall\mathbb{X}, \mathcal{L}(\mathbb{X})\geq 0$.

Une fonction de coût $\mathcal{L}$ a été définie de manière théorique à l'équation 4.5, testons alors sa pertinence avec des données expérimentales. Pour ce faire, la source à aligner $X_{\text{RMS}}^{\text{LLA}}$ sera le descripteur RMS de l'actionneur LLA et la cible sera alors le même descripteur choisi dans l'ensemble des actionneurs $\{LRA, UpperRightActuator(URA), UpperLeftActuator(ULA)\}$.

Nos validations suivront toujours la même approche. **Les descripteurs de LLA seront pris comme source et les descripteurs correspondants des autres actionneurs seront pris**

comme cible potentiel. En effet nous supposerons qu'un comportement voisin existe en base de données sans être totalement identique.

Les données utilisées dans ces travaux proviennent d'essais de vieillissement accélérés des différents actionneurs électromécaniques. Aussi, les descripteurs calculés possèdent un nombre d'échantillons ainsi qu'une fréquence d'échantillonnage commune. Souhaitant montrer le caractère général de la fonction de coût $\mathcal{L}$, simulons une source avec un facteur de compression uniforme par rapport aux cibles. Un point sur deux est alors sélectionné dans la source $X^{\text{LLA}}_{\text{RMS}}$. De plus, cette même source est tronquée à 900 points. Un nuage de points est alors créé avec la méthode décrite à la section 4.2.2 pour obtenir $\tilde{X}^{\text{LLA}}_{\text{RMS}}[0:900:2]$. Le nuage ainsi créé possède 450 échantillons. Cette série de données en deux dimensions représente le comportement du descripteur d'un actionneur évoluant de 0 à 900 cycles. Ainsi, le but de la méthode d'alignement est d'aligner le dernier point de la source $\tilde{X}^{\text{LLA}}_{\text{RMS}}[0:900:2]$ sur, *a minima*, le 900$^{\text{e}}$ point de la cible $\tilde{Y}^{\text{LLA}}_{\text{RMS}}$ non compressée. La figure 4.12 représente les trajectoires des points de la source au fur et à mesure de la convergence de l'algorithme d'alignement dans deux configurations différentes : un alignement de la source compressée sur elle-même, non compressée à la figure 4.12a, et un alignement de la source compressée sur le descripteur d'un autre actionneur, non compressé à la figure 4.12b.



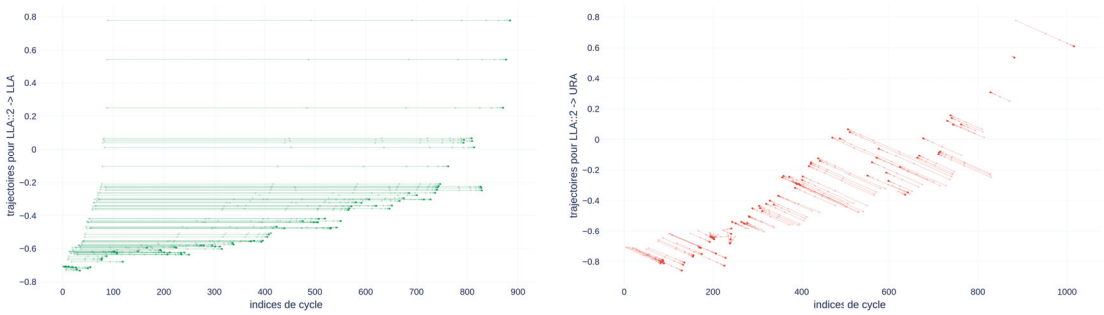
(a) Alignement $\tilde{X}^{\text{LLA}}_{\text{RMS}}[0:900:2] \rightarrow \tilde{Y}^{\text{LLA}}_{\text{RMS}}$
Valeur de la fonction de coût : $\mathcal{L}(\mathbb{S}^{\text{LLA}}_{\text{RMS}}) = 0$

(b) Alignement $\tilde{X}^{\text{LLA}}_{\text{RMS}}[0:900:2] \rightarrow \tilde{Y}^{\text{URA}}_{\text{RMS}}$
Valeur de la fonction de coût : $\mathcal{L}(\mathbb{S}^{\text{LLA}}_{\text{RMS}}) = 0.54$

FIGURE 4.12 – Trajectoires des points pour l'alignement du descripteur compressé d'un facteur 2 RMS de LLA sur différentes cibles et valeurs de la fonction de coût associée

Les deux figures 4.12 illustrent des comportements bien différents. En effet, la première figure 4.12a représente des trajectoires de points sources toutes parallèles entre elles et le dernier point de la source compressée atteint le 898$^{\text{e}}$ cycle lors de son alignement dans la cible, ce qui diffère de seulement deux cycles de la réalité, alors que la figure 4.12b représente des trajectoires de points sources plus disparates. De plus, le 448$^{\text{e}}$ point - autrement dit l'antépénultième point aligné - atteint le 1016$^{\text{e}}$ point dans la cible. Cette différence s'explique par le fait que les données cibles sont bien différentes des données sources alors que la source est alignée sur son descripteur homologue dans l'actionneur *URA*. Toutefois, cette information permet de conforter l'idée qu'avec cette méthode, la bonne qualité d'un alignement de séries peut être quantifiée par le comportement des directions de déplacement de chaque point. De ce fait, le comportement de chaque point pourrait être quantifié par la fonction $\mathcal{L}$ construite à l'équation 4.5. Les valeurs de coût obtenues à partir des deux alignements permettent alors de vérifier le bon comportement de $\mathcal{L}$. Ainsi, à la figure 4.12a, la fonction ne pénalise pas la dilatation d'une série lors de l'alignement et l'hypothèse 7 est vérifiée. L'hypothèse 8 est aussi vérifiée car les directions de chaque point sont horizontales, et donc colinéaires au premier vecteur de base $\mathcal{B}$, e_x. Enfin, l'hypothèse 9 est validée car la valeur de la fonction de coût est bien nulle. Pour la figure 4.12b, les 3 hypothèses fondamentales de $\mathcal{L}$ n'étant pas vérifiées, la valeur de la fonction de coût est positive non nulle : $\mathcal{L} = 0.54$.

Testée précédemment avec un facteur de compression uniforme, la méthode d'alignement de séries temporelle développée ici montre des résultats pertinents pour des facteurs de compression non uniformes ; la fonction de coût permet de les quantifier. Mais pourquoi s'intéresser à un facteur de compression non uniforme ? Cette configuration pourrait modéliser une variation des conditions environnementales d'utilisation d'un actionneur. En effet, dans ce cas, l'actionneur ne vieillirait pas de la manière régulière comme représenté par les données utilisées ici, à savoir avec une dynamique pseudo-exponentielle durant 1130 cycles d'un usage constant à conditions invariantes. C'est pourquoi un facteur de compression non uniforme reste un point d'étude intéressant. Pour cette modélisation, sera sélectionné un point sur n, avec $n \in \mathbb{N}$ tiré de manière aléatoire à chaque



(a) Alignement $\tilde{X}^{\mathrm{LLA}}_{\mathrm{RMS}}\,[0:900:0.1] \rightarrow \tilde{Y}^{\mathrm{LLA}}_{\mathrm{RMS}}$
Valeur de la fonction de coût : $\mathcal{L}(\mathbb{S}^{\mathrm{LLA}}_{\mathrm{RMS}}) = 0.0$

(b) Alignement $\tilde{X}^{\mathrm{LLA}}_{\mathrm{RMS}}\,[0:900:0.1] \rightarrow \tilde{Y}^{\mathrm{URA}}_{\mathrm{RMS}}$
Valeur de la fonction de coût : $\mathcal{L}(\mathbb{S}^{\mathrm{URA}}_{\mathrm{RMS}}) = 0.66$

FIGURE 4.13 – Trajectoires des points pour l'alignement du descripteur RMS de LLA avec 10 % des points sélectionnés de manière aléatoire et valeurs de la fonction de coût associée

nouvelle étape de sélection de la source précédente $X^{\mathrm{LLA}}_{\mathrm{RMS}}$. En choisissant seulement 10 % des points de la source, le nuage $X^{\mathrm{LLA}}_{\mathrm{RMS}}\,[0:900:0.1]$ est obtenu. Le protocole expérimental est alors identique à celui qui est utilisé pour obtenir la figure 4.12. La figure 4.13 est alors réalisée.

La figure 4.13 est similaire au comportement mis en évidence par la figure 4.12 précédente. Comme le montre la figure 4.13a, le dernier point aligné de la source converge toujours vers le 898$^{\mathrm{e}}$ point de la cible. De même, dans le cas d'un alignement d'une série sur elle-même la valeur de la fonction de coût est nulle alors que lors de l'alignement sur une série provenant d'un autre actionneur, $\tilde{Y}^{\mathrm{URA}}_{\mathrm{RMS}}$, la valeur est positive non nulle. Notons que cette dernière est voisine de la valeur de coût obtenue avec l'alignement de la série compressée d'un facteur deux de la figure 4.12. Sans pour autant quantifier avec rigueur la sensibilité de la méthode d'alignement itérative à ces différences de fréquence d'échantillonnage, ces quatre derniers alignements permettent de montrer la robustesse de la méthode aux données manquantes. Une étude plus approfondie pourra être menée en perspectives de cette thèse.

Les précédents paragraphes ont permis de rendre compte du bon comportement de la fonction de coût pour l'alignement d'un descripteur RMS, sur deux actionneurs différents. Testons maintenant la fonction $\mathcal{L}$ sur l'ensemble de la base de données $\mathfrak{M}_{n,p,q}(\mathbb{R})$ avec $(n,p,q) = (1130, 28, 4)$. Pour chacun des $p = 28$ descripteurs peut alors être créée une matrice de coût $\mathbb{L}_j \in \mathcal{M}_q(\mathbb{R})$ contenant $q^2 = 16$ valeurs différentes, chaque valeur représentant le coût de l'alignement du descripteur d'un actionneur l sur celui d'un autre actionneur h. Toutes les sources considérées sont alors compressées d'un facteur deux. Soit $\mathcal{L}_{r\rightarrow s}$ la fonction de coût liée à l'alignement de l'actionneur r sur l'actionneur s, avec les notations de l'équation 4.5, un élément de la matrice de coût est défini comme : $\forall\,(r,s) \in [\![1;4]\!]^2, (l_{rs})_j = \mathcal{L}_{r\rightarrow s}\left(\mathbb{S}^j_r\right)$. La matrice de coût $\mathbb{L}_j$ peut alors être explicitée par l'équation 4.6.

$$\mathbb{L}_j = \begin{array}{c} \\ \mathrm{LLA} \\ \mathrm{LRA} \\ \mathrm{URA} \\ \mathrm{ULA} \end{array} \begin{array}{c} \begin{array}{cccc} \mathrm{LLA} & \mathrm{LRA} & \mathrm{URA} & \mathrm{ULA} \end{array} \\ \begin{bmatrix} \mathcal{L}_{1\rightarrow1}\left(\mathbb{S}^j_1\right) & \mathcal{L}_{1\rightarrow2}\left(\mathbb{S}^j_1\right) & \mathcal{L}_{1\rightarrow3}\left(\mathbb{S}^j_1\right) & \mathcal{L}_{1\rightarrow4}\left(\mathbb{S}^j_1\right) \\ \mathcal{L}_{2\rightarrow1}\left(\mathbb{S}^j_2\right) & \mathcal{L}_{2\rightarrow2}\left(\mathbb{S}^j_2\right) & \mathcal{L}_{2\rightarrow3}\left(\mathbb{S}^j_2\right) & \mathcal{L}_{2\rightarrow4}\left(\mathbb{S}^j_2\right) \\ \mathcal{L}_{3\rightarrow1}\left(\mathbb{S}^j_3\right) & \mathcal{L}_{3\rightarrow2}\left(\mathbb{S}^j_3\right) & \mathcal{L}_{3\rightarrow3}\left(\mathbb{S}^j_3\right) & \mathcal{L}_{3\rightarrow4}\left(\mathbb{S}^j_3\right) \\ \mathcal{L}_{4\rightarrow1}\left(\mathbb{S}^j_4\right) & \mathcal{L}_{4\rightarrow2}\left(\mathbb{S}^j_4\right) & \mathcal{L}_{4\rightarrow3}\left(\mathbb{S}^j_4\right) & \mathcal{L}_{4\rightarrow4}\left(\mathbb{S}^j_4\right) \end{bmatrix} \end{array} \tag{4.6}$$

Avec l'ensemble des actionneurs $A = \{\mathrm{LLA}, \mathrm{LRA}, \mathrm{URA}, \mathrm{ULA}\}$ indexés respectivement par l'intervalle $[\![1;4]\!]$, la matrice $\mathbb{L}_j$ représente la valeur de coût de tous les arrangements à deux éléments avec répétition possible de l'ensemble A. Elle possède donc $\mathcal{A}^4_2 = 16$ éléments. Le choix de la méthode d'alignement et de sa fonction de coût associée sera pertinent si lors de l'alignement du descripteur d'un actionneur sur lui-même, la fonction de coût reste nulle et ce, même pour un taux de compression positif. Ainsi, tous les éléments diagonaux de $\mathbb{L}_j$ doivent être nuls, cette matrice doit donc vérifier la propriété suivante :

Propriété 4.2.1. $\forall (r,s) \in [\![1;q]\!], \quad r = s \Rightarrow (l_{rs})_j = 0$

Pour pouvoir vérifier cette propriété sur l'ensemble des descripteurs j donnés, construisons le nouvel élément $\mathbb{L}_\Sigma = \sum_{j=1}^{p} \mathbb{L}_j$. Elle devra aussi vérifier la propriété 4.2.1. Après l'alignement de

chaque descripteur et de chaque actionneur de la base $\mathfrak{M}_{n,p,q}(\mathbb{R})$ entre eux, la fonction de coût $\mathcal{L}$ est évaluée $28 \times 4^2 = 448$ fois pour donner la figure 4.14.

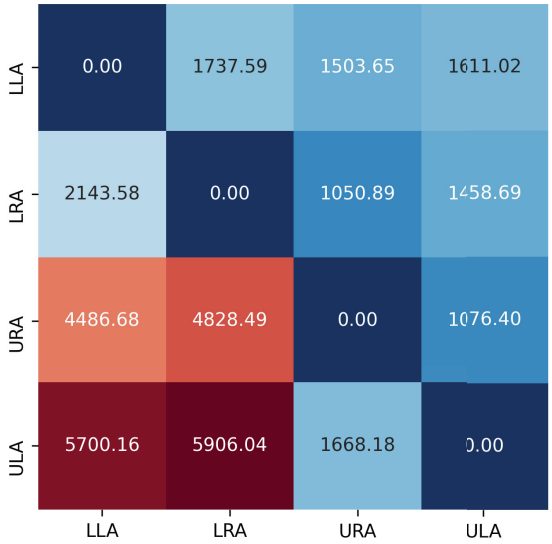


FIGURE 4.14 – $\mathbb{L}_\Sigma$ somme des matrices de coûts $\mathbb{L}_i$ pour la base de données $\mathfrak{M}_{n,p,q}(\mathbb{R})$

Conformément aux résultats précédents, la figure 4.14 montre que lorsqu'une série compressée est alignée sur sa propre dilatation temporelle, la fonction de coût est nulle. Pour l'ensemble des descripteurs p contenus dans la base de données $\mathfrak{M}_{n,p,q}(\mathbb{R})$, la propriété 4.2.1 reste vérifiée. De fait, la fonction de coût $\mathcal{L}$ construite dans cette sous-section est bien pertinente pour quantifier la qualité d'un alignement. La matrice $\mathbb{L}_\Sigma$ représentée par la carte thermique de la figure 4.14 permet aussi de mettre en évidence le caractère non symétrique de la méthode d'alignement. Cette carte thermique peut alors être divisée en quatre zones principales, chacune de ces zones représentant l'alignement d'un groupe d'actionneurs sur un autre. Définissons les quatre ensembles par les indices de leurs éléments dans la figure 4.14 en notant $(0,0)$ le coin supérieur gauche.

$$\begin{aligned} L \to L &= \{(0,0),(0,1),(1,0),(1,1)\} \\ L \to U &= \{(0,2),(0,3),(1,2),(1,3)\} \\ U \to L &= \{(2,0),(2,1),(3,0),(3,1)\} \\ U \to U &= \{(2,2),(2,3),(3,2),(3,3)\} \end{aligned} \tag{4.7}$$

Pour rappel, les données étudiées proviennent de quatre actionneurs faisant partie d'un inverseur de poussée. Aussi, ils sont repérés dans ces travaux par leur position sur la nacelle L'ensemble $L \to L$ correspond à l'alignement sur eux-mêmes des descripteurs des actionneurs situés au bas de la nacelle, tandis que l'ensemble $U \to U$ correspond à l'alignement sur eux mêmes des descripteurs des actionneurs situés en haut de la nacelle. Les deux autres ensembles sont des combinaisons des deux précédents. Ces ensembles possèdent alors une valeur médiane voisine, à l'exception de l'ensemble $U \to L$. Dans l'ordre de l'équation 4.7, l'ensemble numérique $\{4487, 4828, 5700, 5906\}$ est obtenu. Ce dernier possède une valeur médiane à peu près quatre fois supérieure aux autres ensembles. Cette disparité de valeur de coût témoigne des différences de comportement entre les deux étages de la nacelle. Non soumis exactement aux mêmes contraintes, ils présenteraient des comportements hétérogènes dans le vieillissement. Cette disparité de comportement est en partie expliquée par le fait que sur l'inverseur de poussée, les actionneurs ne subissent par les mêmes charges aérodynamiques en fonctionnement.

La fonction de coût a été développée et la pertinence de son comportement sur la base de données d'intérêt $\mathfrak{M}_{n,p,q}(\mathbb{R})$ a été montrée. La fonction $\mathcal{L}$ permet alors d'associer à un alignement donné une valeur réelle représentative de sa qualité, autrement dit représentative de la ressemblance d'une série temporelle source à sa cible. Il est alors possible de changer d'échelle pour considérer non pas la qualité globale d'un alignement mais la variation de chaque point de la série source par rapport à son état initial. Cela représente alors la pénalité associée à chaque point dans l'alignement, pénalité qui était agrégée par la fonction $\mathcal{L}$. Ce nouveau terme peut être formalisé

par : $\mathcal{L}_{1\leq i\leq n} = (1/K)\sum_{j=1}^{K}[x_i(0) - x_i(j)]^2$. Adaptée à l'alignement de la source $\tilde{X}^{\text{LLA}}_{\text{RMS}}[0:900:2]$ sur la cible $\tilde{Y}^{\text{LLA}}_{\text{RMS}}$ de la figure 4.12, la figure 4.15 est réalisée.

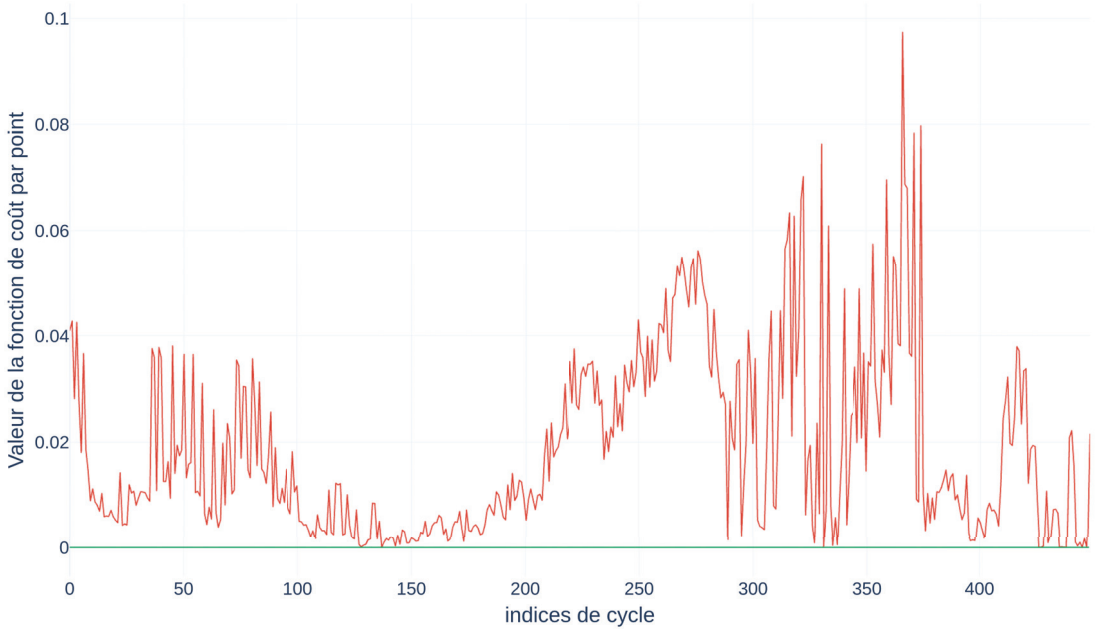


FIGURE 4.15 – Visualisation de la variabilité des trajectoires en fonction des points pour l'alignement $\tilde{X}^{\text{LLA}}_{\text{RMS}}[0:900:2] \rightarrow \tilde{Y}^{\text{URA}}_{\text{RMS}}$

La figure 4.15 possède en vert la variation de $\mathcal{L}_i$ pour l'alignement $\tilde{X}^{\text{LLA}}_{\text{RMS}}[0:900:2] \rightarrow \tilde{Y}^{\text{LLA}}_{\text{RMS}}$ et en rouge la variation de $\mathcal{L}_i$ pour l'alignement $\tilde{X}^{\text{LLA}}_{\text{RMS}}[0:900:2] \rightarrow \tilde{Y}^{\text{URA}}_{\text{RMS}}$. Cette figure permet de détecter les zones de faibles mouvements de points, autrement dit les portions de courbe source qui ressemblent le plus à la cible : une première zone entre $[0;100]$ points, une deuxième entre $]100;200]$, une troisième entre $]200;376]$ points et une dernière à partir du 376^e point. La deuxième et la dernière zone, de faibles valeurs, coïncident avec les endroits de faible déplacement des points de la figure 4.12b. Cette information pourrait alors être intégrée dans les étapes suivantes de l'algorithme pour améliorer la quantification de l'incertitude dans la méthode d'alignement.

Dans cette partie, une fonction de coût $\mathcal{L}$ a été créée pour permettre d'évaluer la qualité des résultats rendus par la méthode d'alignement itérative. Les hypothèses fondamentales pour permettre sa construction ont été faites et la fonction a été testée sur l'ensemble des données utilisées lors de ces travaux. Elle est pertinente pour la méthode développée. Enfin, les composantes $\mathcal{L}_i$ de la fonction $\mathcal{L}$ peuvent être étudiées pour obtenir une information sur la qualité d'alignement locale.

4.2.4 Sélection des meilleurs candidats

Le développement de la fonction de coût $\mathcal{L}$ à la sous-section 4.2.3 a rendu possible la comparaison des alignements de séries entre eux. Comme présenté à la section 3.2, des signaux bruts vibratoires de chaque actionneur sont extraits $p = 28$ descripteurs statistiques. Les actionneurs étant considérés, pour les besoins de cette démonstration, comme indépendants (voir hypothèse 2), un actionneur sera sélectionné pour modéliser le comportement de données inconnues récupérées sur un avion en vol. À titre d'exemple, dans ces travaux, comme dans la sous-section 4.2.3 précédente, l'actionneur LLA est choisi. Soit $\mathbb{X}^{\text{LLA}} \in \mathcal{M}_{n,p}(\mathbb{R})$ la matrice définie comme concaténation des p descripteurs à n points sur les colonnes. La matrice est alors définie comme $\mathbb{X}^{\text{LLA}} = \begin{bmatrix} X_1^{\text{LLA}} & X_2^{\text{LLA}} & \dots & X_p^{\text{LLA}} \end{bmatrix}$ et il faut l'aligner sur les données restantes contenues en base de données $\mathfrak{M}_{n,p,q-1}(\mathbb{R})$. Introduisons de plus la fonction T qui réalise toutes les étapes de la méthode d'alignement itérative. La figure 4.16 représente alors un schéma de principe pour expliquer l'alignement de chaque descripteur de $\mathbb{X}^{\text{LLA}}$ sur le descripteur correspondant dans $\{\mathbb{Y}^{\text{LRA}}, \mathbb{Y}^{\text{URA}}, \mathbb{Y}^{\text{ULA}}\} \in \mathfrak{M}_{n,p,q-1}(\mathbb{R})$.

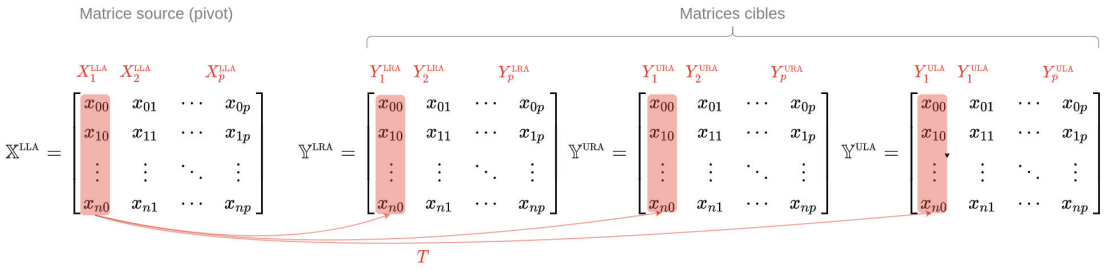


FIGURE 4.16 – Schéma de principe pour l'alignement d'un descripteur de LLA sur les données restantes

Chaque colonne de $\mathbb{X}^{\text{LLA}}$ sera donc alignée avec sa colonne correspondante dans $\mathbb{Y}^h$ avec $h \in [\![1;3]\!]$ auquel on associe les éléments respectifs de l'ensemble $\{LRA, URA, ULA\}$. Pour chaque alignement la fonction de coût $\mathcal{L}$ est évaluée ce qui permet d'obtenir une matrice de coût $\mathbb{L}_{\cdot}^{\text{LLA}} \in \mathcal{M}_{p,3}(\mathbb{R})$ dont l'indice représente la configuration de la série source à aligner. Par la suite, la source est alors tronquée à 900 points mais aucun facteur de compression ne lui est appliqué contrairement à la section 4.2.3 précédente. La matrice source se notera donc $\mathbb{X}^{\text{LLA}}_{[0:900:1]}$ et la matrice de coût associée lors de l'alignement à la base de données $\mathbb{L}^{\text{LLA}}_{[0:900:1]}$. L'alignement pour chaque descripteur est réalisé avec la méthode de la sous-section 4.2.2 pour toutes les données présentes en base $\mathfrak{M}_{n,p,q-1}(\mathbb{R})$. Au total cela représente 28 descripteurs et 3 actionneurs. Ainsi la matrice $\mathbb{L}^{\text{LLA}}_{[0:900:1]}$ est représentée par la carte de chaleur de la figure 4.17.

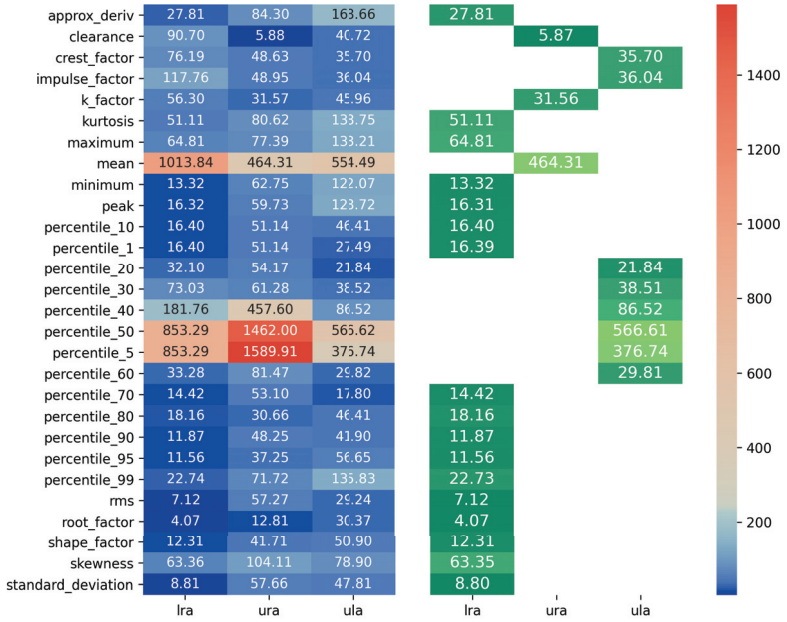


FIGURE 4.17 – Matrice des coûts d'alignement $\mathbb{L}^{\text{LLA}}_{[0:900:1]}$

La figure 4.17 présente deux cartes de chaleur représentant la matrice $\mathbb{L}$: une carte complète à gauche et une carte partielle à droite partageant les mêmes axes. Ainsi, l'ensemble des descripteurs statistiques vus à la section 3.2 forment l'ordonnée tandis que les différents actionneurs forment l'abscisse. La carte complète possède $28 \times 3 = 84$ valeurs correspondant à l'évaluation de $\mathcal{L}$ pour chaque alignement entre un descripteur de la source $\mathbb{X}^{\text{LLA}}_{[0:900:1]}$ et son descripteur associé pour l'actionneur cible. Dans cette matrice sont exploités les *minima* sur les lignes par lecture de l'abscisse. Cela permet de localiser l'actionneur le plus ressemblant à la source alignée parmi ceux contenus en base de données, c'est pourquoi la carte thermique de droite est incomplète. Elle permet de rendre compte uniquement des *minima* de coût associés à chaque descripteur, dans le but de faciliter la lecture des données de la matrice de coût $\mathbb{L}^{\text{LLA}}_{[0:900:1]}$. Les actionneurs les plus semblables à la source pour un descripteur donné sont dès lors mis en évidence. En indexant les descripteurs par ordre

alphabétique sur l'intervalle $[\![1;28]\!]$ avec la première correspondance `approx_deriv` $\rightarrow 1$, une série d'actionneurs idéaux est obtenue au tableau 4.2.

1	2	3	4	5	6	7	8	9	10	11	12
LRA	URA	ULA	ULA	URA	LRA	LRA	URA	LRA	LRA	LRA	LRA
13	14	15	16	17	18	19	20	21	22	23	24
ULA	ULA	ULA	ULA	ULA	ULA	LRA	LRA	LRA	LRA	LRA	LRA
25	26	27	28								
LRA	LRA	LRA	LRA								

TABLE 4.2 – Séquence des actionneurs dont l'évaluation de l'alignement par la fonction $\mathcal{L}$ est minimale pour chaque descripteur

La méthode de sélection des meilleurs candidats a donc été développée ici avec l'intégralité des données présentes dans la base $\mathfrak{M}_{n,p,q-1}(\mathbb{R})$. Ainsi, à chaque descripteur de la source $\mathbb{X}^{\text{LLA}}_{[0:900:1]}$ a été associé le descripteur de l'actionneur qui lui est le plus ressemblant, le plus similaire. Cependant nous avons supposé que les données de la source LLA n'étaient pas présent dans la base, comme présenté précédemment. Rappelons que l'objectif final de la méthode de pronostic est de déterminer l'instant à partir duquel le système étudié passerait à un degré de sévérité supérieur dans son vieillissement ou attendrait sa fin de vie. C'est à partir de cette information que la RUL du système pourrait être extrapolée. Une fois l'alignement fait et les candidats actionneurs les plus ressemblants déterminés à la table 4.2, il convient d'agréger toutes ces informations pour en extraire un point représentant l'index de cycle du système, autrement dit son âge, à l'instant de sa captation. Seulement la figure 4.17 met en évidence la piètre qualité de certains alignements par rapport à leurs voisins, comme par exemple le 50^e, le 40^e, le 5^e percentile ainsi que la moyenne. Ces grandeurs ressemblent donc peu à leur équivalent dans la source $\mathbb{X}^{\text{LLA}}_{[0:900:1]}$ au sens de la pénalité imposée par la fonction de coût $\mathcal{L}$. Ainsi, pour ne pas biaiser le modèle d'agrégation (ou modèle de fusion) par les résultats d'actionneurs peu pertinents, il convient de pondérer l'apport de chaque descripteur en fonction des valeurs présentes dans $\mathbb{L}^{\text{LLA}}_{[0:900:1]}$. À partir des valeurs de la fonction de coût, construisons alors un vecteur de pondérations $\mathcal{W}$.

Pour ce faire, introduisons le vecteur de coût minimaux $\mathcal{L}^*$ comme étant :

$$\mathcal{L}^* = \left[\, \mathcal{L}\left(T\left(X_1^{\text{LLA}} \rightarrow Y_1^{\text{LRA}}\right)\right)\ \mathcal{L}\left(T\left(X_2^{\text{LLA}} \rightarrow Y_2^{\text{URA}}\right)\right)\ \mathcal{L}\left(T\left(X_3^{\text{LLA}} \rightarrow Y_3^{\text{ULA}}\right)\right)\ \ldots\ \mathcal{L}\left(T\left(X_3^{\text{LLA}} \rightarrow Y_3^{\text{ULA}}\right)\right) \,\right]^\top \in \mathcal{M}_{p,1}(\mathbb{R})$$

$\mathcal{L}^*$ contient toutes les évaluations minimales de la fonction de coût par alignement de descripteur. Dans le cas étudié, la matrice possède $p = 28$ lignes. $\mathcal{L}^*$ devient la base à partir de laquelle les pondérations seront calculées. Dans un premier temps, $\mathcal{L}^*$ est normalisée en utilisant la fonction `RobustScaler` [PEDREGOSA *et al.*, 2011] ce qui permet de soustraire à une distribution sa médiane puis de la diviser par son écart inter-quartile :

$$\mathcal{L}_N^* = \frac{\mathcal{L}^* - \text{médiane}\,(\mathcal{L}^*)}{Q_3 - Q_1} \tag{4.8}$$

Dans l'expression 4.8, la fonction `médiane` utilisée correspond, de manière usuelle, à la valeur de $\mathcal{L}^*$ qui sépare la matrice en deux ensembles de cardinaux égaux. L'écart inter-quartile est quant à lui défini comme l'étendue entre le premier et le troisième quartile Q. Autrement dit, il représente l'étendue entre le 25^e quantile et le 75^e quantile. Par conséquent, après avoir ordonné $\mathcal{L}^*$, il représente l'écart après élimination des 25 % d'éléments les plus faibles et des 25 % d'éléments les plus forts. La figure 4.17 mettant en évidence des coûts d'alignement fortement dispersés, cette méthode de normalisation a été choisie pour minimiser l'impact des valeurs aberrantes. Une fois cette mise à l'échelle robuste appliquée, la fonction d'exponentielle normalisée est appliquée au vecteur des poids calculés avec l'équation 4.8. Cela permet d'obtenir un vecteur dont la somme des éléments sera unitaire. Ainsi, le vecteur de poids $\mathcal{W}$ est obtenu à l'équation 4.9 à partir du vecteur $\mathcal{L}_N^*$.

$$\mathcal{W} = (w_j)_{1 \leq j \leq p} \quad \text{tq.} \quad w_j = \frac{\exp \mathcal{L}_j^N}{\sum_{i=1}^{p} \exp \mathcal{L}_i^N} \tag{4.9}$$

En transformant les données du vecteur de coût $\mathcal{L}_N^*$, lui même extrait de la matrice de coût $\mathbb{L}_{[0:900:1]}^{\text{LLA}}$, le vecteur de pondération $\mathcal{W}$ est obtenu. Il affecte donc un poids à chacun des descripteurs des actionneurs sélectionnés pour leur ressemblance avec la source. Ses valeurs sont représentées par la figure 4.18.

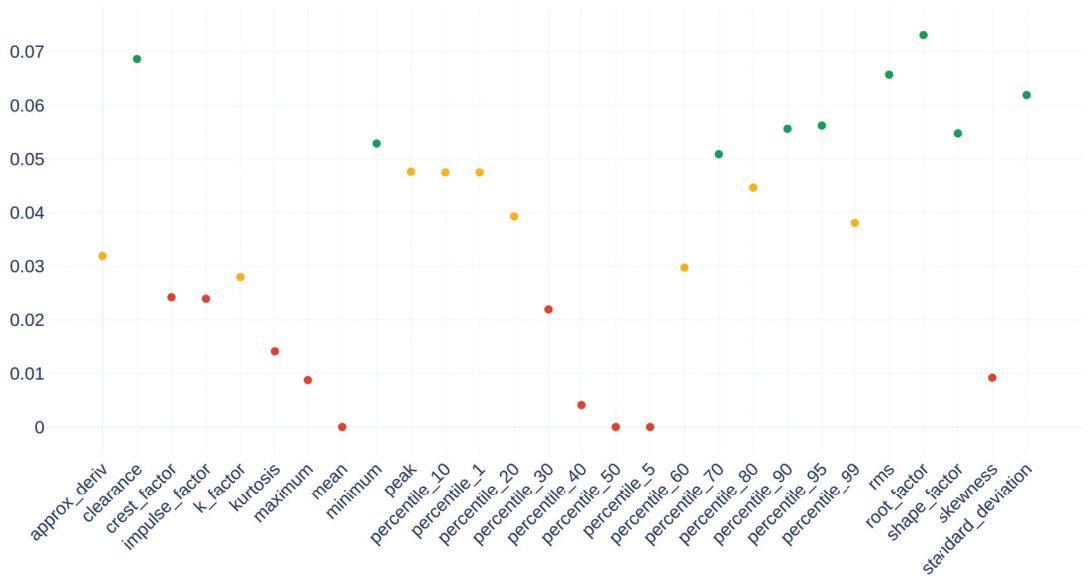


FIGURE 4.18 – Vecteur $\mathcal{W}$, valeurs normées des poids pour l'ensemble des descripteurs sélectionnés

Chaque point de la figure 4.18 est un élément de $\mathcal{W}$, w_j. La couleur des points de la figure 4.18 permet de mieux segmenter visuellement les résultats obtenus. Ainsi pour un poids inférieur à 0.025 le point sera rouge, pour un poids supérieur à 0.05 le point sera vert et pour le reste de l'intervalle le point sera de couleur jaune. Remarquons que les poids w_j ainsi obtenus possèdent de faibles valeurs. En effet, en considérant 28 grandeurs, la propriété imposée de sommation à 1 réduit l'impact absolu de chaque descripteur. Notons alors que cette pondération est relative à un ensemble de descripteurs. De plus, alors que certains poids sont proches de la valeur maximale 0.07, certains autres sont nuls, ce qui est le cas pour la moyenne (`mean`), le 50^e et le 5^e percentiles (`percentile_50` et `percentile_5`). En croisant les informations de la figure 4.17, il est possible de leur faire correspondre leurs coûts respectifs par la fonction $\mathcal{L}$, représentée à la figure 4.17. Ainsi ils possèdent un coût respectif de 464.31, 566.61 et 376.74. La médiane de $\mathcal{L}^*$ étant de 22.29, les valeurs prises par la quantification de l'alignement de ces descripteurs sont 20, 25 et 16 fois plus importantes que leurs voisins dans le vecteur de coûts optimaux $\mathcal{L}^*$. Le schéma de détermination des poids est donc cohérent et représente bien la dispersion des valeurs de $\mathcal{L}^*$.

Ainsi, dans cette sous-section, l'étape de sélection des meilleurs candidats a été développée. Elle consiste dans un premier temps à aligner les séries sources sur l'ensemble des séries cibles présentes en base de données puis à quantifier chaque alignement avec la fonction de coût créée à la sous-section 4.2.3. Les actionneurs possédant le coût minimal pour chaque descripteur permettent la création d'un vecteur de coûts optimaux. Après normalisation, un vecteur de pondération est obtenu. Ce dernier permet de pondérer l'apport de l'information extraite de chaque alignement dans le modèle de fusion d'information qui sera développé dans les sections suivantes. Les opérations peuvent être résumées par la séquence suivante :

$$\mathbb{L}_{[0:900:1]}^{\text{LLA}} \rightarrow \mathcal{L}^* \rightarrow \mathcal{L}_N^* \rightarrow \mathcal{W}$$

4.2.5 Mesure d'incertitude de chaque alignement

À ce stade de la méthode, il est possible d'aligner l'ensemble des descripteurs de données sources inconnues sur l'ensemble des descripteurs contenus en base de données (voir sous-section 4.2.2). De plus, la fonction de coût développée $\mathcal{L}$ à la sous-section 4.2.3 permet de quantifier la qualité de tels alignements, qu'il y ait eu ou non compression temporelle (comme dans le cas d'essais accélérés). Étant donné qu'à chaque ensemble de descripteurs est affecté un actionneur, il est désormais possible de déterminer l'ensemble des actionneurs les plus ressemblants aux données sources inconnues et de leur affecter chacun une pondération, autrement dit un score normalisé de ressemblance (voir sous-section 4.2.4). Le but de la méthode complète de pronostic est de pouvoir situer les données inconnues dans les durées de vies et les comportements connus de la base de données. Aussi, s'intéresser aux indices des derniers points alignés pour chaque descripteur permettrait la détermination du temps de vie restant de l'actionneur source étudié en fonction des données connues. Dans le cas de LLA utilisé pour cette démonstration, les résultats d'alignement permettent d'obtenir un vecteur $\mathcal{C}^{\text{LLA}} = \begin{bmatrix} t_{n,1} & t_{n,2} & \dots & t_{n,p} \end{bmatrix}^\top \in \mathcal{M}_{p,1}(\mathbb{N})$ contenant l'indice du dernier point du nuage source $(t_n, x_n)_j^l$ aligné pour chacun des descripteurs j. Rappelons que le nuage source a été déplacé sur le nuage cible et donc que les indices des deux nuages partagent la même échelle temporelle. Le vecteur $\mathcal{C}^{\text{LLA}}$ sert alors de base pour la création d'une matrice $\mathbb{I}^l \in \mathcal{M}_{n,p}(\llbracket 0;1 \rrbracket)$ remplie de 0 avec un 1 par colonne, ce qui revient créer une matrice par la concaténation de p Diracs centrés en $t_{n,j}$. Avec les variables $1 \leq i \leq n$, $1 \leq j \leq p$ et $1 \leq l \leq q$, la matrice $\mathbb{I}^l$ est définie à l'équation 4.10.

$$\mathbb{I}^l = (\iota_{i,j})^l \text{ tq. } \mathbb{I}^l = \begin{cases} \iota_{t_{n,j},j} = 1 \text{ si } t_{n,j} = j \\ 0, \text{ sinon} \end{cases} \tag{4.10}$$

Pour incorporer l'information des pondérations dans la matrice de l'équation 4.10, il faut d'abord transformer le vecteur $\mathcal{W}$ en matrice de carré avec les pondérations réparties sur sa diagonale. Pour ce faire, introduisons $(e_j)_{1 \leq j \leq p}$ les vecteurs de la base canonique de $\mathbb{R}^p$. Avec w_j les éléments du vecteur de pondération $\mathcal{W} \in \mathcal{M}_{p,1}(\mathbb{R})$, la matrice carré $\sum_{j=1}^{p} e_j^\top \mathcal{W} e_j e_j^\top$ est obtenue. L'équation 4.11 permet alors de pondérer les données de la matrice $\mathbb{I}^l$.

$$\mathbb{J}^l = \mathbb{I}^l \left(\sum_{j=1}^{p} e_j^\top \mathcal{W} e_j e_j^\top \right) \in \mathcal{M}_{n,p}(\mathbb{R})$$

$$\mathbb{J}^l = \begin{array}{c} \\ \\ \text{i}^{\text{ième}} \rightarrow \\ \\ \text{j}^{\text{ième}} \rightarrow \\ \\ \end{array} \begin{array}{c} \begin{matrix} \delta_1^{\text{LLA}} & \delta_2^{\text{LLA}} & \dots & \delta_p^{\text{LLA}} \end{matrix} \\ \begin{bmatrix} 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ w_1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & w_2 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & 0 \end{bmatrix} \end{array} \tag{4.11}$$

La matrice $\mathbb{J}^l \in \mathcal{M}_{n,p}(\mathbb{R})$ est ainsi créée à l'équation 4.11 avec autant de lignes que de points dans la série cible. Ici l'ensemble des séries cibles $\{Y^{\text{LRA}}, Y^{\text{URA}}, Y^{\text{ULA}}\}$ possèdent le même nombre d'échantillons. La méthode peut facilement être étendue à des séries cibles de tailles différentes. La dimension conservée pour la matrice $\mathbb{J}^l$ sera le nombre de points maximum d'une des séries cibles. De plus, la matrice $\mathbb{J}^l$ possède autant de colonnes que de descripteurs. Ici elle en possède $p = 28$.

De manière concrète, $\mathbb{J}^l$ est une matrice contenant la concaténation de Diracs pondérés représentant chacun l'index du dernier point aligné d'un descripteur de la source. Pour obtenir un numéro de cycle final et ainsi situer l'actionneur inconnu dans la distribution probable de degrés de sévérités - résultat de la méthode de partitionnement de la section 3.3 - il convient de fusionner l'information de toutes les colonnes de $\mathbb{J}^l$. En l'état, il serait alors obtenu uniquement un numéro d'index correspondant au dernier cycle supposé. Pour réaliser un pronostic, il convient aussi de quantifier l'incertitude associée à ce résultat. Ainsi, un numéro de cycle unique n'est plus suffisant, c'est pourquoi les Diracs seront remplacés par des densités de probabilités. Le résultat obtenu

ne sera plus un entier compris dans l'intervalle $[\![1;1130]\!]$ mais une fonction densité f de support $[\![1;1130]\!]$ à valeurs dans $\mathbb{R}$.

À ce stade plusieurs questions se posent. Comment utiliser l'information apportée par les Diracs pondérés pour obtenir une densité de probabilité ? Comment transformer la matrice creuse $\mathbb{J}^l$ en une matrice contenant des distributions continues sur chaque colonne, et donc pour chaque descripteur ? Sur quelle grandeur quantifier une incertitude ? Rappelons que c'est la fusion de tous les indices temporels des derniers points de chaque descripteur aligné provenant d'une même source qui permet d'estimer pour cette source la durée de vie à partir de celles des candidats en base de données. C'est donc sur cette grandeur qu'une mesure d'incertitude serait pertinente. Soit Z_j^h la variable aléatoire qui représente l'indice de cycle du dernier point aligné à chaque itération k. Le descripteur d'un actionneur donné, par exemple $\tilde{X}_j^l$, est aligné sur le descripteur d'une cible donnée, $\tilde{Y}_j^h$. L'algorithme d'alignement défini à la sous-section 4.2.2 est itératif. Et donc, comme cela a été détaillé à l'algorithme 6, à chaque itération k est calculé pour le nuage source une nouvelle assignation des points sources aux points cibles par la méthode 3. Une fonction d'assignation $\mathcal{T}_k$ étant calculée à chaque itération, durant l'alignement, est alors construit la suite $\left(\mathcal{T}_j^l|_k\right)_{1\leq k\leq K}$. L'assignation représentée par cette fonction, dont le comportement est illustré à la figure 4.9, est indépendante de la distance des points du nuage source aux points du nuage cible dans $\mathbb{R}^2$ car elle est calculée dans l'espace projeté, donc dans $\mathbb{R}$. La valeur d'intérêt pour la mesure d'incertitude est par conséquent $\mathcal{T}_j^l(n)|_k$ qui représente le plus proche voisin du dernier point du nuage source $\tilde{X}_j^l \in \mathcal{M}_{n,2}(\mathbb{R})$ dans le nuage cible $\tilde{Y}_j^h \in \mathcal{M}_{m,2}(\mathbb{R})$. À la fin du processus itératif est alors créé un vecteur $\mathbb{T}_j^h = \left[\mathcal{T}_j^l(n)|_{k=0} \quad \mathcal{T}_j^l(n)|_{k=1} \quad \dots \quad \mathcal{T}_j^l(n)|_{k=K}\right]^\top \in \mathcal{M}_{K,1}(\mathbb{R})$, avec n le dernier indice du nuage cible. L'incertitude mesurée est donc une image du nombre de valeurs différentes contenues dans le vecteur $\mathbb{T}_j^h$. Cet histogramme permet, en quelque sorte, d'estimer la « confiance » que l'on peut avoir dans le résultat obtenu ainsi que sa « stabilité ».

Maintenant qu'un vecteur de valeurs discrètes a été obtenu, il peut être considéré comme l'échantillon d'une population dont les individus représenteraient les valeurs de la variable aléatoire Z_j^h. Afin d'estimer l'incertitude de l'alignement global à partir de cet échantillon $\mathbb{T}_j^h$, il convient de déterminer la loi de probabilité que suivra Z_j^h. Pour cela, on émet l'hypothèse 10.

Hypothèse 10. *La variable aléatoire Z_j^h suit une loi normale paramétrée par la moyenne et l'écart-type de l'échantillon $\mathbb{T}_j^h$. D'où : $Z_j^h \sim \mathcal{N}\left(\mu\left(\mathbb{T}_j^h\right), \sigma\left(\mathbb{T}_j^h\right)\right)$.*

Notons qu'avec l'hypothèse 10, toute asymétrie présente dans l'échantillon $\mathbb{T}_j^h$ sera perdue dans le calcul de la moyenne et de l'écart-type pour la loi Normale considérée. Par nature, la loi Normale ne peut représenter ces dynamiques. De surcroît, l'échantillon $\mathbb{T}_j^h$ de petite taille K permet difficilement de conclure sur la normalité des données. Dans ces travaux, nous avons fait le choix d'utiliser cette hypothèse simplificatrice pour ne pas rajouter une étape d'identification de distribution. Les calculs qui suivront pourront être réalisés plus tard avec des densités de probabilités non gaussiennes. Pour l'instant, dans notre cas, la densité de probabilité de la loi Normale [2] dont les paramètres sont introduits à l'hypothèse 10 sera une gaussienne de support $z \in [0;n]$ comme présenté à l'équation 4.12.

$$f_j^h(z) = \frac{1}{\sigma(\mathbb{T}_j^h)\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{z-\mu(\mathbb{T}_j^h)}{\sigma(\mathbb{T}_j^h)}\right)^2} \tag{4.12}$$

Il est désormais possible de créer une matrice $\mathbb{F} = \left[f_0^l \quad f_1^l \quad \dots \quad f_p^l\right]^\top \in \mathcal{M}_{n,p}(\mathbb{R})$ qui serait la concaténation sur les colonnes d'une distribution de la variable Z_j^h pour chaque descripteur, à la différence de $\mathbb{J}_j^l$ qui contenait des Diracs pondérés. Cette méthode est alors illustrée à la figure 4.19 pour $K = 4$ itérations de l'algorithme d'alignement pour le descripteur d'un actionneur donné.

À la première étape de la figure 4.19 (à lire de haut en bas), le vecteur $\mathbb{T}_j^h$ est construit en suivant la procédure décrite trois paragraphes plus haut. Cet exemple montre la trajectoire prise par le dernier point du nuage source (en rouge) vers les points du nuage cible (en bleu) pendant 4

[2] Quitte à simplifier un peu trop le problème et générer un biais de normalité dans la méthode. La taille des échantillons étudiés ne nous permet pas de conclure de manière sereine quant à la nature de la loi de probabilité ayant généré ces échantillons.

itérations. Pour illustrer un cas usuel dans l'alignement, le point source considéré ne converge pas vers un unique point cible au fur et à mesure des itérations. Aussi :

$$\begin{aligned}\mathbb{T}_j^h &= \left[\mathcal{T}_j^l(n)|_{k=1} \quad \mathcal{T}_j^l(n)|_{k=2} \quad \mathcal{T}_j^l(n)|_{k=3} \quad \mathcal{T}_j^l(n)|_{k=4}\right] \\ &= \begin{bmatrix}3 & 2 & 1 & 3\end{bmatrix}\end{aligned}$$

Ce vecteur est alors représenté sous forme d'histogramme et à partir de l'échantillon $\mathbb{T}_j^h$. La moyenne $\mu\left(\mathbb{T}_j^h\right) = 2.25$ ainsi que l'écart-type $\sigma\left(\mathbb{T}_j^h\right) \approx 0.95$ de l'échantillon sont calculés. Ces résultats permettent de paramétrer la gaussienne représentant la densité de la loi de probabilité suivie par l'échantillon $\mathbb{T}_j^h$.

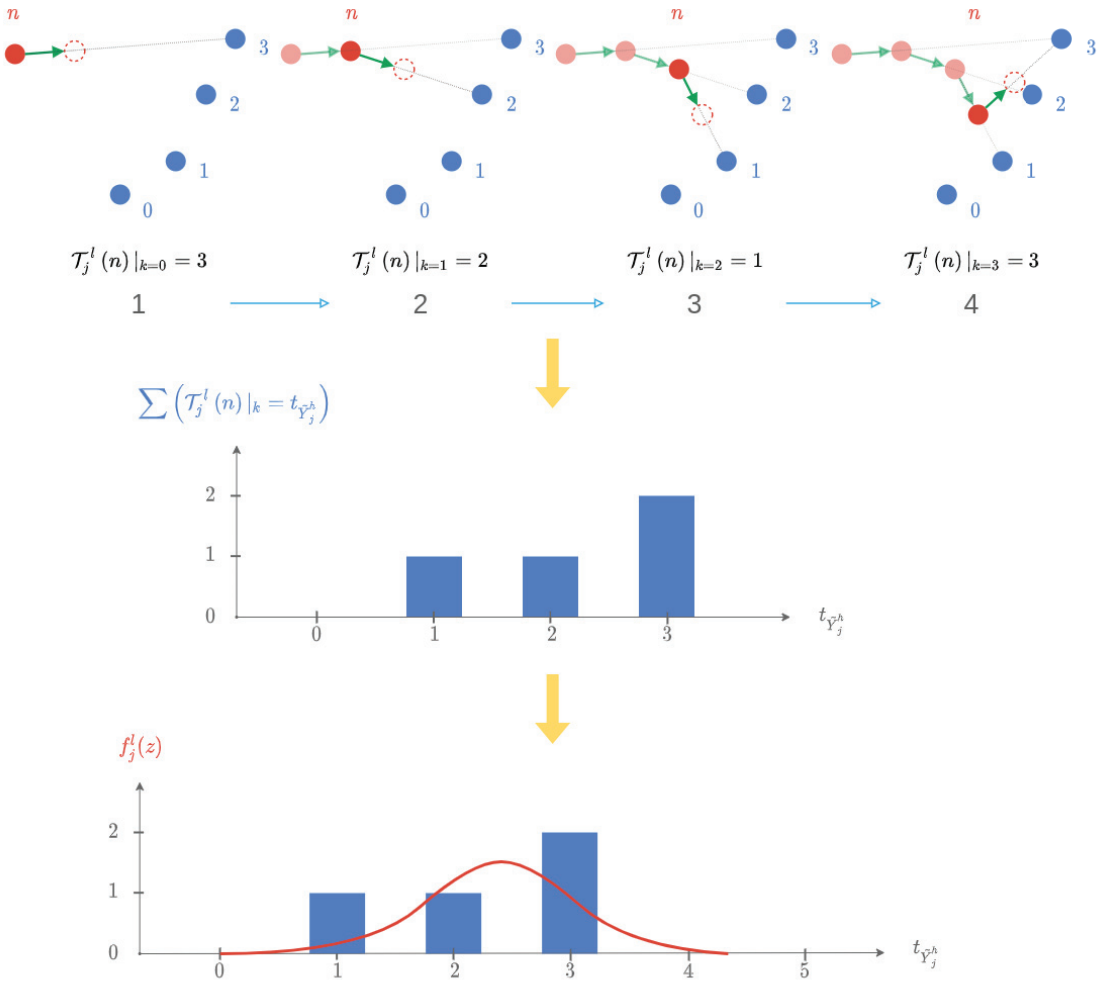


FIGURE 4.19 – Illustration de la méthode de calcul d'incertitude pour $K = 4$ itérations sur l'assignation du plus proche voisin du dernier point de $\tilde{X}_j^l$ sur $\tilde{Y}_j^h$ (restriction à 4 points)

Maintenant qu'une matrice représentant l'incertitude par une densité de probabilité de l'index temporel du dernier point aligné pour chaque descripteur a été obtenue, il reste à pondérer chacune de ses colonnes par les éléments du vecteur $\mathcal{W}$. Du vecteur $\mathcal{W}$ il est possible construire une matrice $\mathbb{W} \in \mathcal{M}_{n,p}(\mathbb{R})$ créée par n concaténations de vecteurs $\mathcal{W}^\top$ sur les lignes. La matrice est alors définie comme : $\mathbb{W} = \sum_{i=1}^{n} e_i \mathcal{W}^\top$. En définissant le produit de Hadamard pour deux matrices, qui permet la multiplication des éléments de deux matrices termes à termes, par le symbole $\odot$, la matrice $\mathbb{F}^l$ est modifiée pour créer la matrice $\mathbb{D}^l$. Cette dernière est donc définie par l'équation 4.13.

$$\begin{aligned}\mathbb{D}^l &= \mathbb{F}^l \odot \mathbb{W}^l \\ &= \mathbb{F}^l \odot \left(\sum_{i=1}^{n} e_i \mathcal{W}^\top\right)\end{aligned} \tag{4.13}$$

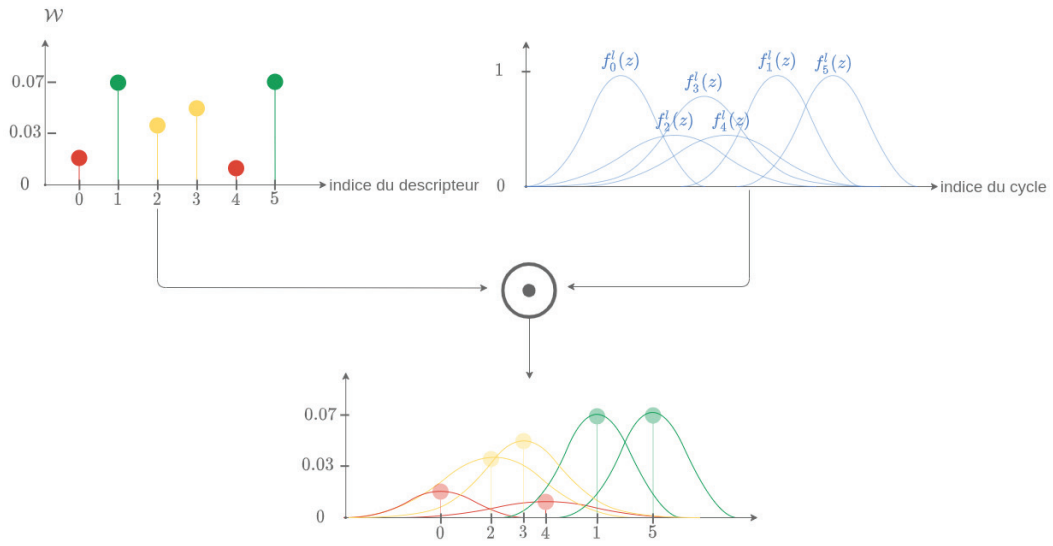


FIGURE 4.20 – Schéma de principe pour la détermination des densités de probabilités intermédiaires avant fusion

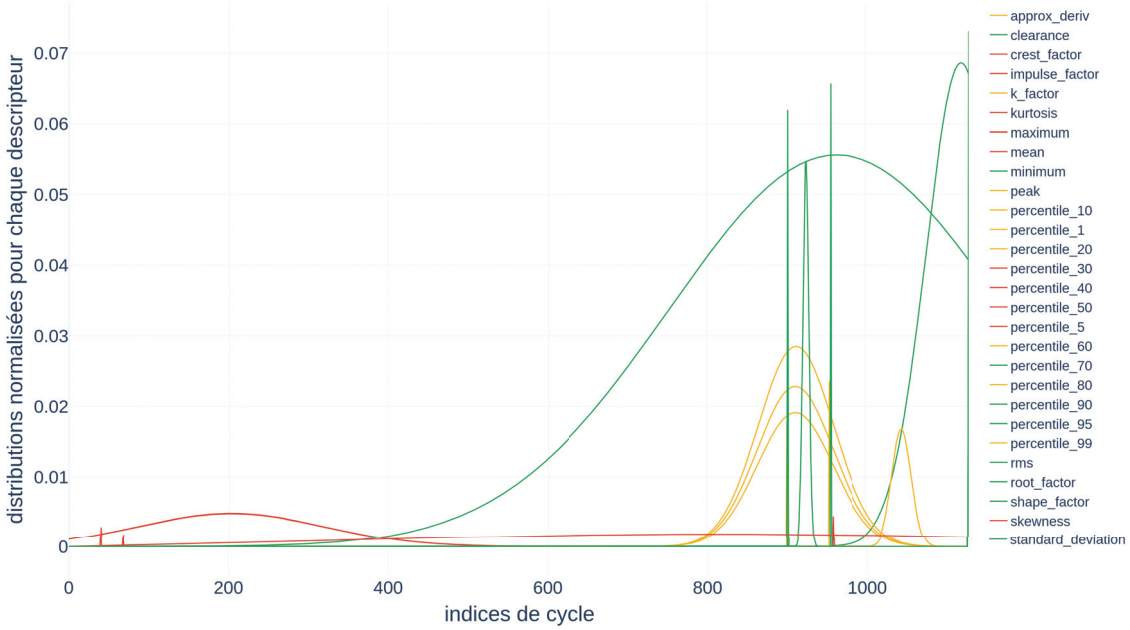


FIGURE 4.21 – Densités de probabilités obtenues pour chaque descripteur lors de l'alignement $X^{\text{LLA}}_{[0:900:1]} \rightarrow \{Y^{\text{LRA}}, Y^{\text{URA}}, Y^{\text{ULA}}\}$

Cette dernière étape est illustrée par le schéma de principe de la figure 4.20.

La matrice $\mathbb{F}^l$ regroupe alors deux informations pour mesurer l'incertitude de l'alignement : l'incertitude liée à la variation de la valeur Z_j^h du dernier point aligné de la source et la qualité de l'alignement apportée par le vecteur de pondérations $\mathcal{W}$. En appliquant cette méthode aux données expérimentales dans l'alignement du nuage source $X^{\text{LLA}}_{[0:900:1]}$ sur l'ensemble des cibles $\{Y^{\text{LRA}}, Y^{\text{URA}}, Y^{\text{ULA}}\}$, avec le vecteur de poids $\mathcal{W}$ calculé à la section 4.2.4 et représenté à la figure 4.18, on obtient la figure 4.21.

La figure 4.21 présente 28 gaussiennes aux supports disjoints. Sur la figure 4.21, le code couleur des poids de la figure 4.18 a été repris pour faciliter la compréhension du graphique. Cette représentation met exergue la qualité hétérogène de l'alignement des différents descripteurs sur la base de données. Le résultat d'alignement de certains descripteurs est pénalisé à juste titre. C'est ainsi le cas pour les deux pics extrêmes pour le `kurtosis` et le `maximum` respectivement localisés

au 40^{e} et 68^{e} cycles ce qui représente une erreur de 95 % et 92 % par rapport à l'index cible réel 900. De même pour `impulse_factor` avec un écart-type plus grand, le pic de densité n'en demeure pas moins à 78 % d'erreur au 195^{e} cycle et `skewness` qui représente presque une distribution de cycles uniformes sur l'intervalle $[0; 1130]$. Cependant, remarquons le descripteur `percentile_30` localisé au 958^{e} cycle qui, malgré 6 % d'erreur par rapport à l'objectif, pâtit de sa pondération. Outre cette singularité, remarquons que les descripteurs, qui possèdent les pondérations les plus importantes, sont tous autour du 900^{e} cycle. Notons que bien que proche de l'objectif, les descripteurs percentile_10, percentile_99 et approx_deriv tous les trois à 1 % d'erreur, n'ont pas la pondération maximale du fait de l'oscillation des valeurs de $Z^{\text{LLA}}_{\text{percentile\_10}}$, $Z^{\text{LLA}}_{\text{percentile\_99}}$ et $Z^{\text{LLA}}_{\text{approx\_deriv}}$ lors de l'alignement, alors que leurs distributions paraissent pertinentes. Ainsi, depuis le début de cette méthode, l'étape de sélection des descripteurs qui n'avait pas été abordée, est réalisée par la conjonction de la mesure d'incertitude et de la pondération de l'alignement. Plus le nombre de descripteurs est important à cette étape, plus le résultat de pronostic final sera fiable au détriment du coût de calculs de la méthode complète.

Dans cette partie, une mesure de l'incertitude globale pertinente de l'alignement d'une série source sur une série cible a été définie. Une suite de gaussiennes à supports disjoints est alors créée pour rendre compte de la variation de l'index du dernier point source aligné. Cela permet alors de préparer les données pour le modèle de fusion d'information qui sera développé dans la section suivante.

4.2.6 Barycentre dans l'espace de Wasserstein

Dans les sections précédentes, les données sources inconnues ont été alignées sur une base de données (voir section 4.2.2) et les actionneurs les plus ressemblants ont été sélectionnés suite aux résultats de quantification de chaque alignement (voir section 4.2.4). Par la suite (voir section 4.2.5) un ensemble de gaussiennes a été obtenu. Chaque gaussienne représente la densité de probabilité empirique associée à l'indice du dernier point aligné d'un descripteur de la source inconnue sur le même descripteur dans la cible. Dès lors, nous souhaitons fusionner l'information de ces $p = 28$ fonctions pour obtenir une distribution centrée sur l'instant probable, et sur l'indice de cycle, correspondant à l'état actuel estimé de l'actionneur source inconnu étudié. La détermination de sa durée de vie restante sera possible en croisant cette estimation avec les résultats de la méthode de partitionnement obtenus au chapitre 3.3.

Ce paragraphe présente la méthode suivie pour fusionner ces informations.

Pour cela, dans un premier temps, définissons plus formellement les objets d'intérêt obtenus ici à la section 4.2.5. Les calculs ont été réalisés jusqu'à maintenant avec des séries temporelles définies sur $\mathcal{I} = [\![0; n]\!]$ un intervalle d'entiers. Pour les besoins des méthodes qui suivront et sans invalider les calculs des sous-sections précédentes, $\mathcal{I}$ sera désormais redéfini comme un intervalle borné $[0; n]$ du corps des réels $\mathbb{R}$. Dans le cas où, au fur et à mesure de la vie du produit, d'autres séries modèles seraient récoltées pour enrichir la base de données, les calculs seraient alors effectués sur un intervalle de taille au moins égale à $\mathcal{I}$. De manière plus générale, l'intervalle d'intérêt peut être considéré non borné dans ces travaux. Soit $\mathcal{X}$ cet intervalle de $\mathbb{R}$. Dans la section 4.2.5, l'hypothèse 10 permet la création de densités de probabilités de différentes lois normales paramétrées f_j^h. Rappelons que $1 \leq j \leq p$ avec p le nombre de descripteurs utilisés dans ces travaux, et $1 \leq h \leq q-1$ avec q le nombre d'actionneurs considérés. Ici h n'atteindra jamais la valeur q car les données d'un actionneur ont été sorties de la base de données $\mathfrak{Y}$ pour servir de test. Les f_j^h sont donc des mesures de probabilités absolument continues. Notons $\mathcal{P}(\mathcal{X})$ l'ensemble auquel elles appartiennent. Travaillant avec des données échantillonnées, les mesures $f_j^h \in \mathcal{P}(\mathcal{X})$ ont comme support un nombre fini de points. En introduisant la mesure de Dirac δ_{x_i} en $x_i \in \mathcal{X}$, les précédentes mesures peuvent être reformulées comme une somme finie de Diracs pondérés : $f_j^h = \sum_{i=1}^{n} \mu_i \delta_{x_i}$ avec $\mu = \begin{bmatrix} \mu_1 & \mu_2 & \dots & \mu_n \end{bmatrix}^\top \in \mathbb{R}_+^n$ un vecteur de poids positifs de taille n. Ce dernier vecteur appartient au simplexe, dont la définition est donnée l'équation 4.14.

$$\Sigma_n = \{\mu = (\mu_i)_{1 \leq i \leq n} \in \mathbb{R}_+^n \text{ tq. } \sum_{i=1}^{n} \mu_i = 1\} \tag{4.14}$$

Par la suite, l'espace $\mathcal{X}$ étant fixé, une mesure discrète f_j^h sera complètement identifiée par son vecteur de poids μ. Par abus de notation, nous considérerons que $f_j^h = \mu$. De la même manière introduisons $\nu = f_i^l$ la mesure du descripteur d'un autre actionneur en base de données. Considérons

enfin $\mathbb{R}$ comme un espace métrique muni de la 1-distance usuelle, autrement appelée la distance de Manhattan, qui calcule la valeur absolue entre deux points.

On souhaite alors calculer un centroïde généralisé, concept proche de la moyenne de Fréchet [Fréchet, 1948] qui sera appelé par la suite *barycentre*. Le résultat serait un barycentre généralisé de distributions de probabilités. Comme présenté par [Agueh et Carlier, 2011], dans un espace euclidien (ici défini sur le corps des réels précédents) muni d'une distance d, le barycentre x d'un ensemble de points $(x_0, x_1, \ldots, x_p)$ auquel les poids $(w_0, w_1, \ldots, w_p)$ sont respectivement associés est défini par $\inf_x \sum_{i=0}^{p} w_i \left(x - x_i\right)^2$. En considérant chaque x_i comme étant une distribution μ, le résultat permettrait de calculer la distribution globale recherchée. Il convient de définir la distance d. Les travaux de [Csiszár, 1972] ont introduit la notion de divergence entre deux mesures de probabilités. L'opérateur est présenté à la définition 4.2.1.

Définition 4.2.1 : φ-divergence

Soit $\varphi : [0; \infty) \to \mathbb{R}$ une fonction convexe définie telle que $\varphi(1) = 0$, soit M et N deux distributions de probabilités définies sur $\mathcal{X}$. Supposons que $M(dx) = \mu(x)\rho(dx)$ et $N(dx) = \nu(x)\rho(dx)$ pour une mesure commune dominante ρ. Alors :

$$D_\varphi\left(M \| N\right) = \int_{N>0} \varphi\left(\frac{\mu(x)}{\nu(x)}\right) \nu(x) d\rho$$

La divergence permet alors de produire un réel représentant la somme des variations d'une mesure par rapport à une autre. Cette forme générale de mesure de dissimilarité est la base de la définition de la divergence de Kullback-Leibler [Kullback et Leibler, 1951]. En prenant $\varphi : x \to x \log\left(x\right)$ une fonction convexe, cela permet de calculer le logarithme des ratios de deux distributions. Nous aurions alors trouvé la notion de distance utile pour obtenir le barycentre des distributions si la divergence de Kullback-Leibler ne souffrait pas d'un problème structurel dans notre cas d'étude. En effet, dans le cas de distributions à supports disjoints, comme c'est le cas pour μ et ν la divergence est égale à $+\infty$ et ce tant que les deux distributions ne possèdent pas de support commun. De même pour l'ensemble des distances $\mathbb{L}_p$ étendues aux distributions sur la droite réelle, dès lors qu'elles ont un support disjoint, la valeur de distance sera constante quelle que soit leur proximité sur la droite réelle [Cazelles, 2018, A.2]. En utilisant la norme L_1 (valeur absolue) avec l'ensemble des densités de probabilités discrètes de la matrice $\mathbb{F}^l$, le résultat de la figure 4.21 est obtenu.

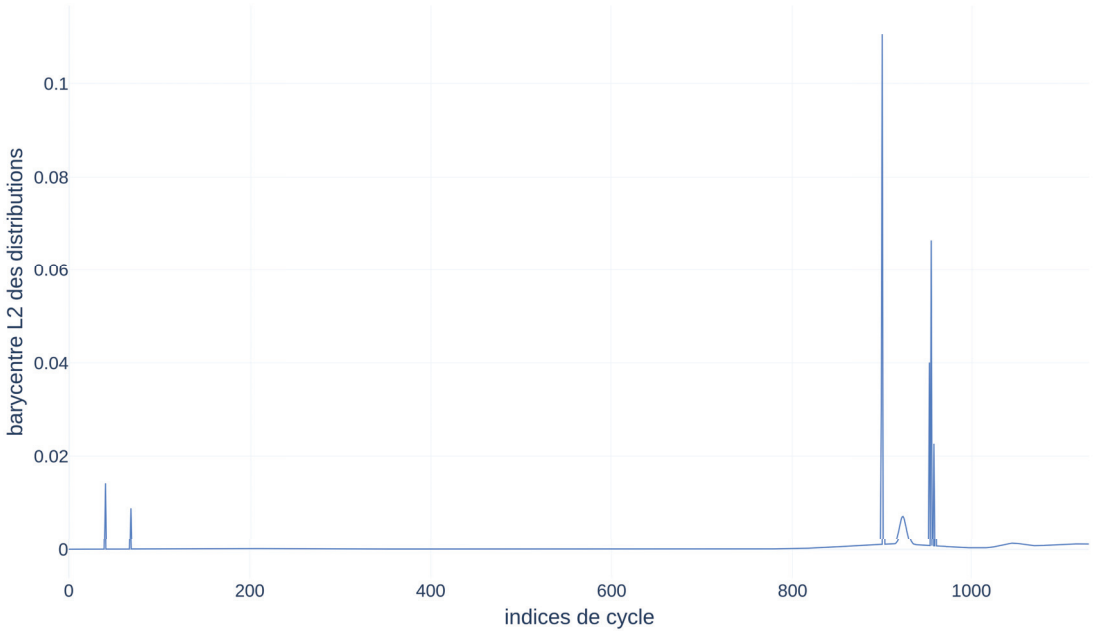


Figure 4.22 – Barycentre des distributions de la figure 4.21 dans l'espace euclidien avec norme l_1

Ce résultat montre clairement que l'interpolation dans l'espace euclidien ne donne pas d'information pertinente pour l'obtention d'une futur RUL. Le résultat en tant que mélange de gaussiennes pondérées ne permet pas d'obtenir un numéro de cycle final avec une mesure d'incertitude associée. De plus, les deux pics situés aux cycles 40 et 68 qui correspondaient respectivement aux descripteurs `kurtosis` et `maximum` ont été conservés malgré leur faible pondération. Dans la figure 4.22 les distributions discrètes de Diracs sont sur-représentées dans le résultat final et la distribution finale est multi-modale.

D'autres familles de distances pourraient être explorées, comme par exemple la MMD [Gretton *et al.*, 2012] utilisée dans les travaux de [Jia, Cai *et al.*, 2019] pour mesurer la similitude entre deux séries de données. Par la suite, le problème sera modélisé en utilisant la théorie du transport optimal dans le but d'obtenir un résultat pertinent pour la détermination de la RUL. Présentons alors ces concepts, ainsi que la distance de Wasserstein qui leurs est associée.

Le transport optimal a été initialement introduit par Monge [Monge, 1781] dans son *Mémoire sur la théorie des déblais et des remblais*. L'idée centrale était de trouver comment déplacer un tas de terre, un déblai, dans un trou, un remblai, et ce en minimisant une certaine fonction de coût. Avec la figure 4.23, le lien entre le problème de transfert de masse et les densités de probabilités est alors immédiat.

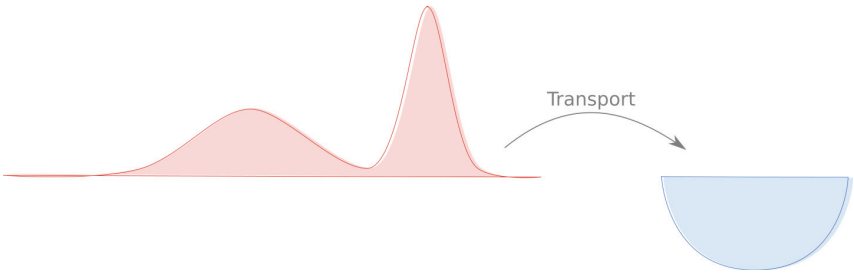


Figure 4.23 – Schéma de principe du problème de déblai (rouge) et remblai (bleu) de Monge

Mais, la modélisation de Monge [Monge, 1781] ne permet pas de prendre en compte le cas où la masse μ pourrait se diviser en plusieurs morceaux. C'est l'apport de Kantorovich [Kantorovich, 1960] dans son article de 1939 qui définit alors le comportement du transfert de masse entre la mesure de départ (source) et la mesure cible, toutes deux scindables. Pour mettre au point sa théorie, il avait étudié le problème de l'optimisation de l'organisation de la production industrielle sur le territoire Russe. Pour deux masses respectives μ et ν, il introduit alors le plan de transport π appartenant à l'ensemble $\Pi(\mu,\nu)$, l'ensemble des polytopes de transport [Brualdi, 2006, section 8.1]. Pour rappel, un polytope est une notion introduite dès 1882 par le mathématicien allemand Hoppe qui représente une figure géométrique limitée par des portions de droites, des plans ou des hyperplans [Coxeter, 1947]. Un polytope peut être représenté par une matrice de taille (n,m). Un polytope de transport est par conséquent, une matrice $\pi = (\pi_{i,j})_{1\leq i\leq n, 1\leq j\leq m}$ non négative définie non nulle telle que $\sum_{i=1}^{n}\pi_{ij} = \sum_{j=1}^{m}\pi_{ij}$ [Brualdi, 2006, section 8.1], autrement appelée matrice stochastique double [Brualdi, 2006, section 1.7] dans le cas où $\sum_{i=1}^{n}\pi_{ij} = \sum_{j=1}^{m}\pi_{ij} = 1$. Avec $\mathbb{1}_n \in \mathbb{R}^n$ le vecteur constitué uniquement de 1, l'ensemble des plans de transports est défini à l'expression 4.15. Notons que les matrices π le constituant sont aussi appelées des *polytopes de couplages* pour les mesures μ et ν.

$$\Pi(\mu,\nu) = \left\{\pi \in \mathcal{M}_{n,n}(\mathbb{R}_+), (\mu,\nu) \in (\Sigma_n)^2 ; \pi\mathbb{1}_n = \mu, \pi^\top \mathbb{1}_n = \nu\right\} \tag{4.15}$$

Soit une fonction de coût c à valeurs dans $\mathbb{R}_+$. Soit $\mathcal{X}$ et $\mathcal{Y}$ les supports respectifs des mesures de départ et d'arrivée. Alors la formulation de Kantorovich [Peyré et Cuturi, 2020, équation 2.15] est donnée par l'expression 4.16.

$$\mathcal{L}_\kappa(\mu,\nu) = \min_{\pi\in\Pi(\mu,\nu)} \iint_{\mathcal{X}\times\mathcal{Y}} c(x,y)d\pi(x,y) \tag{4.16}$$

Dans l'équation 4.16, π est un plan de transport défini continuement sur $\mathcal{X}\times\mathcal{Y}$. Dans notre cas les mesures considérées sont à support dans $\mathbb{R}$ et donc $\mathcal{X} = \mathcal{Y} = \mathbb{R}$. Pour deux mesures μ et ν à support réel, la figure 4.24 représente une solution du problème de transport posé par 4.16.

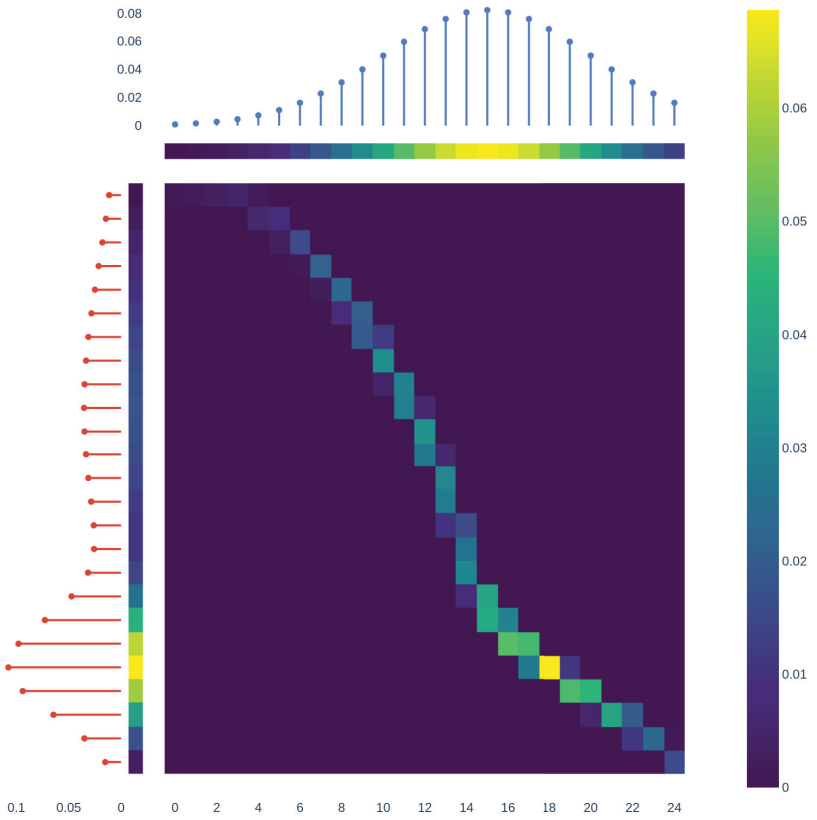


FIGURE 4.24 – Exemple d'un plan de transport π de marginales respectives μ (rouge) et ν (bleu)

Dans l'exemple de la figure 4.24, les deux mesures *source* et *cible* possèdent le même nombre de points, $n = 25$. Le plan de transport π de l'une à l'autre est représenté par la matrice centrale creuse dont chaque élément peut être interprété comme la proportion de masse de la source qui doit être envoyée à un certain endroit dans la cible. De manière concrète, l'élément de π indexé par le couple $(x = 18, y = 20)$, valeur maximale de $\pi(x, y)$, indique que 7 % de la masse source à $x = 18$ doit être affecté à la masse cible à l'endroit $y = 20$. La projection de chaque mesure sur son support est représentée par les cartes de chaleurs qui leurs sont directement adjacentes.

Dans ce cas d'étude où les mesures considérées appartiennent au même espace et si cet espace est un espace polonais, alors le problème de transfert de masse 4.16, introduit par Kantorovich, définit une distance pour un coût $c = d^p$ avec $p \in [1; +\infty)$ [CAZELLES, 2018]. Précisons la notion d'espace polonais [BOURBAKI, 2007b, définition 1, §6] à la définition 4.2.2.

Définition 4.2.2 : Espace polonais

Un espace dit polonais *est un espace métrique complet et séparable.*

Un espace métrique (E, d) est un ensemble E non vide muni d'une distance d qui vérifie les trois propriétés fondamentales de symétrie, séparation et inégalité triangulaire. Un espace métrique est complet quand toute suite de Cauchy d'éléments de cet espace converge. Intuitivement cela correspond à un ensemble qui n'aurait pas de points manquants, pas de trous. Enfin un espace séparable est défini comme un espace topologique contenant un sous-ensemble dense et au plus dénombrable. Les notions de topologie, de densité et de dénombrement sont définies respectivement dans [BOURBAKI, 2007a, chapitre 1, définition 2], [BOURBAKI, 2007a, chapitre 1, p.8] et [BOREL, 1898, chapitre 1, p.6]. De manière plus concrète, le support de nos mesures $\mathcal{X} = \mathbb{R}$ muni de la 1-distance usuelle (valeur absolue) est un espace complet. En effet, une suite de Cauchy est bornée dans $\mathbb{R}$ et d'après le théorème de Bolzano-Weierstrass, de toute suite de réels bornés, il est possible d'extraire une sous-suite convergente. Enfin, $\mathbb{R}$ muni de sa topologie usuelle est séparable car l'ensemble des rationnels $\mathbb{Q}$ est dense dans $\mathbb{R}$. Ainsi, le support des mesures sources et cibles, ici, est un espace polonais. La formulation du problème de transport 4.16 définit alors une distance

dans $\mathcal{X} \times \mathcal{X}$. Cette distance d^p est appelée *distance de Wasserstein d'ordre p* (ou distance de Kantorovich). Elle est définie notamment par [Villani, 2009, Définition 6.1] :.

Définition 4.2.3 : Distance de Wasserstein d'ordre p

Soit $(\mathcal{X}, d)$ et $(\mathcal{Y}, d)$ deux espaces métriques polonais munis de la distance d et $p \in [1; \infty)$. Soient deux mesures de probabilités μ et ν définies sur l'espace $\mathcal{X}$ (resp. $\mathcal{Y}$), soit $\Pi(\mu, \nu)$ l'ensemble des polytopes de couplages associés aux mesures μ et ν, la distance de Wasserstein d'ordre p entre ces deux mesures est définie comme :

$$W_p(\mu, \nu) = \left(\inf_{\pi \in \Pi(\mu,\nu)} \iint_{\mathcal{X}\times\mathcal{Y}} d(x, y)^p \, d\pi(x, y) \right)^{\frac{1}{p}}$$

Cette dernière vérifie les trois propriétés fondamentales d'une distance [Peyré et Cuturi, 2020, Proposition 2.2] [Villani, 2003, théorème 7.3].

La distance de Wasserstein vient d'être présentée et interprétée et elle s'applique aux données étudiées dans ces travaux qui sont les mesures sur $\mathbb{R}$ contenues dans la matrice $\mathbb{D}^l$ définie à l'équation 4.13. Pour rappel, les normes usuelles dans $\mathbb{R}$ ne permettaient pas de rendre compte de la distance parcourue lors d'un déplacement de masse comme illustré par la figure 4.25. De ce fait, la fusion des différentes densités de probabilités contenues dans $\mathbb{D}^l$ n'était pas pertinente. Le barycentre obtenu à la figure 4.22 avait plusieurs modes et donc n'était pas utilisable pour la prédiction d'une RUL. C'est pourquoi la notion de barycentre dans un espace métrique muni de la distance de Wasserstein a été utilisée.

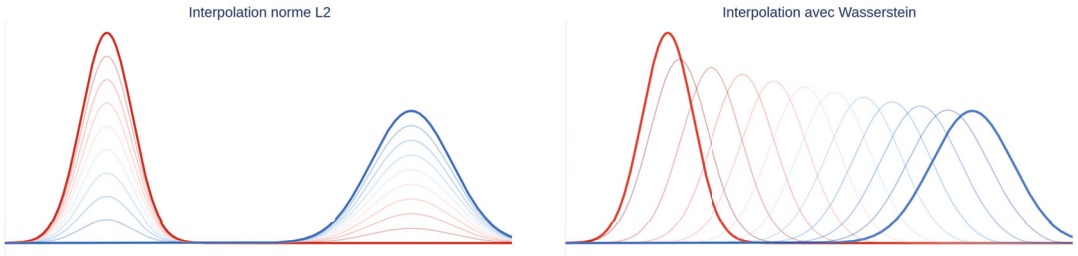


Figure 4.25 – Schéma de principe pour l'interpolation entre une source (rouge) et une cible (bleu)

Cette notion a été définie par [Agueh et Carlier, 2011] comme une généralisation à plus de deux distributions des interpolations de [McCann, 1995]. Par analogie avec la formulation du barycentre dans l'espace euclidien muni de la distance usuelle, le même problème d'optimisation peut être résolu dans l'espace de Wasserstein en remplaçant la distance usuelle par une distance de Wasserstein d'ordre 2 au carré définie d'après [Agueh et Carlier, 2011, équation 2.1] par l'équation 4.17.

$$W_2^2(\mu, \nu) = \inf_{\pi \in \Pi(\mu,\nu)} \iint_{\mathbb{R}\times\mathbb{R}} |x - y|^2 d\pi(x, y) \tag{4.17}$$

Travaillant avec des séries échantillonnées, il convient alors de transposer cette équation du domaine continu dans le domaine discret, et plus précisément dans l'espace $\mathbb{R}^n$. Pour ne pas surcharger les notations, il est admis que les mesures utilisés jusqu'à maintenant sont équivalentes à leurs distributions associées à chaque échantillon représentant une colonne de la matrice $\mathbb{D}^l$. Soient $\mathcal{P}(\mathbb{R}^n)$ l'ensemble de ces mesures et $(\mu, \nu) \in \mathcal{P}(\mathbb{R}^n)^2$ un couple sélectionné dans l'ensemble $\mathcal{P}(\mathbb{R}^n)$. La distance de Wasserstein d'ordre 2 au carré peut alors être exprimée par l'équation 4.18.

$$W_2^2(\mu, \nu) = \min_{\pi \in \Pi(\mu,\nu)} < \pi, C > \tag{4.18}$$

Avec C une matrice de distance définie par [Brickell *et al.*, 2008] comme appartenant à l'ensemble $\mathcal{C}$ caractérisé à l'équation 4.19.

$$\mathcal{C} = \{C \in \mathcal{M}_{n,n}(\mathbb{R}) \text{ tq. } \forall (i,j) \in \mathbb{N} \times \mathbb{N},\, c_{ij} \geq 0,\, c_{ii} = 0 \text{ et } \forall (i,j,k)\, c_{ij} \leq c_{ik} + c_{kj}\} \tag{4.19}$$

Notons que la matrice C est fréquemment construite à partir de la distance euclidienne. Par exemple, avec $(x_i)_{1 \leq i \leq n}$ une suite d'éléments allant de 0 à $n-1$ les éléments de la matrice de coût C seront de la forme $c_{ij} = \|x_i - x_j\|^2$ avec $(i,j) \in [\![1;n]\!]^2$. À l'équation 4.18, le symbole $< .,. >$ représente le produit scalaire matriciel usuellement défini tel que :

$$\forall (A,B) \in \mathcal{M}_{n,m}(\mathbb{R}) \times \mathcal{M}_{n,m}(\mathbb{R}), < A,B >= \mathrm{Tr}\left(A^\top B\right) \tag{4.20}$$

Avec Tr l'opérateur *trace* défini tel que $\forall A \in \mathcal{M}_{n,n}(\mathbb{R}), \mathrm{Tr}(A) = \sum_{i=1}^{n} a_{ii}$. En introduisant $\mathcal{W} = (w_j)_{1 \leq j \leq p} \in \Sigma_p$ un vecteur de pondérations, $(\nu_1, \nu_2, \ldots, \nu_p) \in \mathcal{P}(\mathbb{R}^n)^p$ avec $\mathcal{P}$ un ensemble de p mesures de probabilités, d'après [Agueh et Carlier, 2011], le barycentre dans l'espace de Wasserstein est la solution $\hat{\mu}$ du problème d'optimisation 4.21.

$$\hat{\mu} = \arg\min_{\mu \in \mathcal{P}(\mathbb{R}^n)} \sum_{j=1}^{p} w_j W_2^2(\mu, \nu_j) \tag{4.21}$$

Pour trouver le barycentre d'un ensemble de distributions, l'équation 4.21 indique qu'il faut résoudre un problème d'optimisation, autrement dit calculer la distance de Wasserstein, pour chaque couple de marginales $(\mu, \nu_j)_{1 \leq j \leq p}$. Il faut alors résoudre p fois le problème de transport 4.18. Ce calcul peut être mené par des algorithmes de programmation linéaire (*network flow solver* ou méthode du simplex par exemple). Seulement, la détermination de la distance de Wasserstein nécessitant la minimisation d'un problème d'optimisation à deux contraintes, possède un coût de calcul en $\mathcal{O}\left(n^3 \log(n)\right)$. Pour des distributions avec de larges supports, telles que les nôtres, qui possèdent $n = 1130$ points, cette optimisation ne peut être résolue avec des ressources de calculs raisonnables. Pour pallier ce problème, [Cuturi, 2013] a introduit un terme de régularisation entropique réduisant ainsi la complexité du problème linéaire de transfert de masse. Il introduit alors la notion de transport optimal régularisé par l'entropie, également nommée divergence de Sinkhorn duale [Cuturi, 2013, section 4] à l'équation 4.22. Notons que le terme de divergence est utilisé ici parce que cette pseudo-distance ne vérifie pas la propriété de séparation.

$$W_{2,\varepsilon}^2(\mu,\nu) = \min_{\pi \in \Pi(\mu,\nu)} \langle \pi, C \rangle - \varepsilon E(\pi) \tag{4.22}$$

Avec $\varepsilon \in \mathbb{R}_+$ un facteur de régularisation et E la fonction d'entropie définie d'après [Benamou *et al.*, 2015] pour tout plan de transport $\pi \in \mathcal{M}_{n,n}(\mathbb{R})$ par l'équation 4.23 avec les conventions suivantes : $0 \lim_{x \to 0} \log(x) = 0$ et si $\exists \pi_{i,j} \notin \mathbb{R}_+$ alors $E(\pi) = -\infty$.

$$E(\pi) = -\sum_{i,j=0}^{n} \pi_{i,j} \left(\log(\pi_{i,j}) - 1\right) - \iota_{\mathbb{R}_+}(\pi_{i,j}) \tag{4.23}$$

Avec la fonction indicatrice $\iota_{\mathcal{C}}$, construction formelle qui permet de s'assurer que les polytopes de transport sont bien définis à valeurs dans $\mathbb{R}_+$, définie telle que :

$$\forall x,\ \iota_{\mathcal{C}}(x) : \begin{cases} 0 \text{ si } x \in \mathcal{C} \\ +\infty \text{ sinon} \end{cases}$$

Remarquons que la définition de l'entropie 4.22 est différente de la définition de l'entropie usuelle en théorie de l'information [Shannon, 1948, théorème 2], $H = -\sum_{i=1}^{n} p_i \log(p_i)$, en cela qu'un 1 est rajouté dans le logarithme de l'expression. En effet, l'entropie définie ici n'a pas de signification physique comme dans la théorie de l'information ou la mécanique statistique. C'est, ici, essentiellement un terme de pénalité qui apparaît dans la formulation duale du problème d'optimisation linéaire initial posé par l'équation 4.18. Ce dernier problème peut alors être reformulé dans une version sans contrainte à partir d'une fonction de pénalité exponentielle. Enfin, sa version duale transforme la variable a optimiser et de ce fait transforme la pénalité de l'équation en une

pénalité de la forme $x(\log(x)-1)$ [Cominetti et Martín, 1994, section 2]. Une telle formulation assure alors que la suite des solutions minimisant le problème, trouvées par des méthodes de descentes de gradient, est bornée [Cominetti et Martín, 1994, proposition 2.1]. Dans le cas de l'équation 4.22, le problème linéaire à résoudre devient alors ε-convexe [Benamou *et al.*, 2015].

Intéressons-nous maintenant aux propriétés des solutions de l'équation 4.22. Soit π_ε^* la solution exacte du problème 4.22, pour $\varepsilon \in \mathbb{R}_+$. D'après [Cuturi, 2013, lemme 2], π_ε^* est unique et peut s'écrire comme le produit de trois matrices $\pi_\varepsilon^* = \text{diag}(u)\xi\,\text{diag}(v)$. Cette solution π_ε^*, définie comme la transformation d'une matrice de coûts exponentiels, a déjà été proposée par [Erlander et Stewart, 1990, équation 3.5a] en tant que solution d'un problème de transport à double contraintes. Pour rappel, les contraintes sont imposées ici par les deux distributions marginales μ et ν. Sous réserve que $(u,v) \in \mathbb{R}_+^n \times \mathbb{R}_+^n$ permettent de construire deux matrices diagonales positives et que la matrice ξ soit aussi positive, cette décomposition est unique (à un facteur multiplicatif près) et génère une matrice stochastique double [Sinkhorn, 1964, Théorème 1]. Notons que la matrice $\xi = \exp\left(-\frac{C}{\varepsilon}\right)$ représente l'exponentiation par élément de la matrice de coût C appelée parfois le *noyau de Gibbs associé à la matrice de coût C* [Peyré et Cuturi, 2020, section 4.1]. Enfin, en ce qui concerne la solution π^* du problème primal d'optimisation 4.18, d'après [Cominetti et Martín, 1994, proposition 5.7], la solution du problème régularisé π_ε^* converge à la vitesse d'une exponentielle décroissante vers π^* lorsque $\varepsilon \to 0$.

Le problème 4.22 de transport entre deux distributions peut alors être réécrit sous la forme d'une projection [Benamou *et al.*, 2015] exprimée à l'équation 4.24.

$$W_{2,\varepsilon}^2(\mu,\nu) = \min_{\pi \in \Pi(\mu,\nu)} \{\text{KL}(\pi|\xi)\,;\pi \in \mathcal{C}_1 \cap \mathcal{C}_2\} \tag{4.24}$$

Avec KL un cas particulier d'une $\varphi-$*divergence* (voir définition 4.2.1) appelée la divergence de Kullback-Leibler [Kullback et Leibler, 1951] définie pour deux polytopes de transport π et ξ dans [Benamou *et al.*, 2015] par l'équation 4.25.

$$\text{KL}(\pi|\xi) = \sum_{i,j=1}^{n} \pi_{i,j} \log\left(\frac{\pi_{i,j}}{\xi_{i,j}}\right) - \pi_{i,j} + \xi_{i,j} \tag{4.25}$$

De plus, les ensembles sur lesquels sont échantillonnés le plan de transport sont définis comme :

$$\mathcal{C}_1 = \{\pi \in \mathcal{M}_{n,n}(\mathbb{R}_+)\,;\pi\mathbb{1} = \mu\}$$
$$\mathcal{C}_2 = \{\pi \in \mathcal{M}_{n,n}(\mathbb{R}_+)\,;\pi^\top\mathbb{1} = \nu\}$$

De même que pour la formulation 4.22, lorsque le facteur de régularisation ε inclut dans l'expression de ξ tend vers 0, alors la solution du problème d'optimisation 4.24 est unique. Cette solution vérifie le principe d'entropie maximale [Peyré et Cuturi, 2020, remarque 4.2]. Dans le cas du transport de deux distributions μ et ν par projection de Kullback-Leibler, le problème de transport optimal peut alors être résolu par la méthode itérative de Sinkhorn [Sinkhorn, 1967] ou par l'algorithme des projections itératives de Bregman [Bregman, 1967]. L'expression du plan de transport à l'itération K est alors donnée par l'équation 4.26.

$$\begin{aligned}\pi^{(K)} &= \text{diag}\left(u^{(K)}\right)\xi\,\text{diag}\left(v^{(K)}\right)\\ \pi^{(K)} &= \left(\sum_{i=1}^{n} e_i^\top u^{(K)} e_i e_i^\top\right)\xi\left(\sum_{i=1}^{n} e_i^\top v^{(K)} e_i e_i^\top\right)\end{aligned} \tag{4.26}$$

Avec :

$$u^{(K)} = \frac{\mu}{\xi v^{(K)}} \text{ et } \begin{cases} v^{(K+1)} & = \dfrac{\nu}{\xi^\top u^{(K)}} \\ \text{tq. } v^{(0)} & = \mathbb{1} \end{cases}$$

La formulation de l'équation 4.24 permet de trouver un plan de transport optimal π qui minimise la distance de Wasserstein entre les deux distributions marginales μ et ν. La méthode de calcul utilisée ajuste les valeurs du polytope de transport sur les lignes et les colonnes de manière itérative. Le calcul d'un barycentre de distribution est alors effectué en généralisant l'expression 4.24 à des problèmes faisant appel à de multiples distributions marginales. Dans notre cas, le barycentre doit être calculé à partir de toutes les colonnes de la matrice $\mathbb{D}$ définie à l'équation 4.13. Afin de

rester cohérent avec les notations du chapitre, notons $\mathcal{D}_{.,j}$ avec $j \in [\![1;p]\!]$ chacune de ces colonnes. Cette généralisation est donnée à l'équation 4.27.

$$\min_{\boldsymbol{\pi}\in(\Pi(\mu,\nu))^p} \left\{ \mathrm{KL}_{\mathcal{W}}(\boldsymbol{\pi}|\xi) = \sum_{j=1}^{p} w_j \,\mathrm{KL}(\pi_j|\xi) \;;\; \boldsymbol{\pi} \in \mathcal{C}_1 \cap \mathcal{C}_2 \right\} \tag{4.27}$$

Avec :

$$\mathcal{C}_1 = \{\boldsymbol{\pi} = (\pi_j)_{1\leq j\leq p} \in (\mathcal{M}_{n,n}(\mathbb{R}_+))^p; \forall j \in [\![1;p]\!], \pi_j^\top \mathbb{1} = \mathcal{D}_{.,j}\}$$
$$\mathcal{C}_2 = \{\boldsymbol{\pi} = (\pi_j)_{1\leq j\leq p} \in (\mathcal{M}_{n,n}(\mathbb{R}_+))^p; \exists \delta \in \mathcal{M}_{n,1}(\mathbb{R}), \forall j \in [\![1;p]\!], \pi_j^\top \mathbb{1} = \delta\}$$

Dans l'équation 4.27, $\boldsymbol{\pi}$ représente une suite de plans de transports calculés à partir des marginales $\mathcal{D}_{.,j}$. En généralisant la méthode de résolution de l'équation 4.25 à plusieurs distributions, on peut alors obtenir l'expression de la suite des plans de transports optimaux $\boldsymbol{\pi}$ à l'itération K de la méthode de Sinkhorn généralisée [Benamou *et al.*, 2015, remarque 3] :

$$\pi_j^{(K)} = \left(\sum_{i=1}^{n} e_i^\top u_j^{(K)} e_i e_i^\top\right) \xi \left(\sum_{i=1}^{n} e_i^\top v_j^{(K)} e_i e_i^\top\right)$$

Avec la forme des variables intermédiaires définies telles que :

$$u_j^{(K)} = \frac{\delta^{(K)}}{\xi v_j^{(K)}} \text{ et } \begin{cases} v_j^{(K+1)} & = \dfrac{\mathcal{D}_{.,j}}{\xi^\top u_j^{(K)}} \\ \text{tq. } v_j^{(0)} & = \mathbb{1} \end{cases}$$

À partir de ces expression il est alors possible d'obtenir l'expression du barycentre des distributions recherché. Il est exprimé à l'équation 4.28.

$$\delta^{(K)} = \prod_{j=1}^{p} \left(u_j^{(K)} \odot \left(\xi v_j^{(K)}\right)\right)^{w_j} \tag{4.28}$$

Une implémentation de cette procédure de résolution a été développée dans la librairie dédiée au transport optimal de [Flamary *et al.*, 2021]. La fonction `ot.bregman.barycenter` est alors utilisée avec une matrice de coût quadratique et avec les contraintes du problème d'optimisation définies par l'ensemble des distributions obtenues à la figure 4.21. Cette fonction implémente l'algorithme de résolution par méthode itérative de Sinkhorn [Sinkhorn et Knopp, 1967]. Cependant, bien que la solution du problème régularisé converge vers la solution du problème idéal lorsque $\varepsilon \to 0$, des problèmes de stabilités numériques apparaissent pour un facteur de régularisation trop petit. En effet, ce dernier est utilisé pour calculer la valeur de la fonction $\xi = \exp\frac{-C}{\varepsilon}$. Lorsque ε devient trop petit, alors les éléments de ξ peuvent atteindre la précision machine. Ce faisant, ils deviennent négligeables et sont stockés en mémoire en tant qu'éléments nuls. Au fur et à mesure des étapes de l'algorithme de Sinkhorn, cela se traduit par une division nulle lors de la normalisation finale des deux marginales du plan de transport calculé. À défaut de pouvoir trouver un moyen algorithmique de prévenir ce comportement, nous avons décidé de déterminer le facteur de régularisation idéal pour notre problème par dichotomie, comme illustré à la figure 4.26.

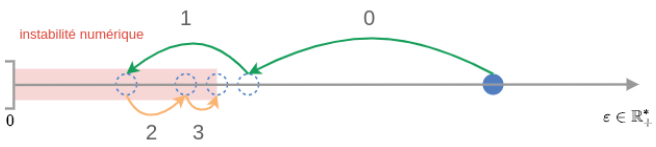


Figure 4.26 – Schéma de principe pour l'obtention du facteur de régularisation ε pour le transport avec entropie

Algorithme 7 Obtention du facteur de régularisation ε par dichotomie

Entrées :
reg_{init} la valeur initiale de ε (doit être assez grande)
max_{iter} le nombre d'itérations maximales pour l'algorithme
Sortie :
ε le facteur de régularisation minimum trouvé

function FIND REGULARIZATION FACTOR(reg_{init}, max_{iter})
  /* *Initialisation* */
  reg ← reg_{init}
  lastGoodReg ← reg_{init}
  previousReg ← 0
  declare regList d'un nombre indéfini de *floats*
  for $i \leftarrow 1$ **to** max_{iter} **do**
    /* *Mise à jour du coefficient de régularisation* */
    reg ← (lastGoodReg − previousReg)/2
    /* *Résolution du problème de transport d'après [*FLAMARY *et al., 2021]* */
    baryWass ← `ot.bregman.barycenter`(..., reg, ...)
    /* *Les instabilités numériques produisent un résultat qui n'est pas un nombre* */
    if $\sum_i$ baryWass = NaN **then**
      previous_reg ← reg
    else
      last_good_reg ← reg
      reg_list.append(reg)
    end if
  end for
  ε ← reg
  return ε
end function

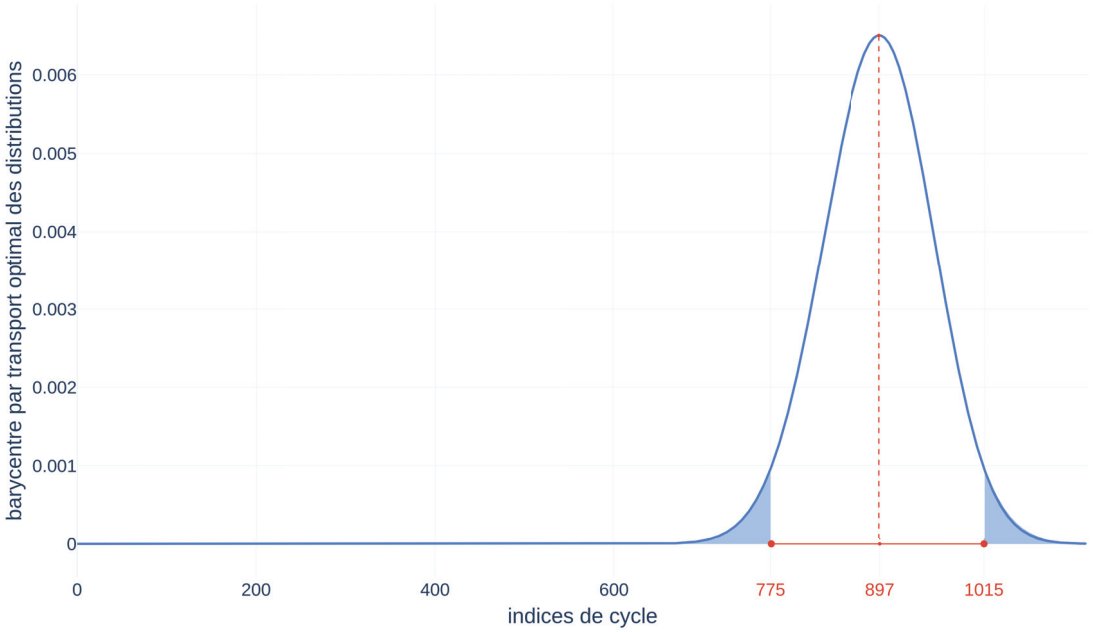


FIGURE 4.27 – Barycentre dans l'espace de Wasserstein à partir de la matrice $\mathbb{D}^l$ pour l'alignement de la matrice $\mathbb{X}^{\text{LLA}}[0:900:1]$ sur le tenseur $\mathfrak{Y} \in \mathfrak{M}_{m,p,(q-1)}(\mathbb{R})$ (voir sous-section 4.2.1 pour définitions)

L'idée de la procédure est de trouver le facteur de régularisation le plus petit possible, pour se rapprocher de la solution optimale, à la limite de l'erreur numérique. Dans un premier temps, une valeur initiale permet de calculer une première instance de la fonction de transport. Si aucune erreur de stabilité n'est détectée, alors le facteur de régularisation est divisé par deux, la procédure est à nouveau lancée. Ces étapes se répètent jusqu'à atteindre une instabilité. Alors, l'intervalle entre l'avant-dernière valeur de régularisation et la valeur courante est divisée par deux pour obtenir le nouveau ε. La procédure est répétée pour un certain nombre d'itérations. Elle peut être décrite plus en détails par l'algorithme 7.

Avec l'algorithme 7, le facteur de régularisation trouvé est $\varepsilon = 3.91 \times 10^{-3}$. Il influe alors, dans la pratique, sur la largeur du support du barycentre calculé à l'équation 4.27, qui est une gaussienne comme présenté à la figure 4.27.

Pour rappel, le but de l'exposé des sections précédentes était de déterminer, à partir de données de test inconnues, l'indice temporel de leur dernier point. Dans la pratique, l'ensemble des descripteurs, autrement dit des séries temporelles de l'actionneur LLA ont été sélectionnés et comparés à la base de données $\mathfrak{Y}$. Cette dernière base de données contient l'ensemble des descripteurs, ici ce sont des profils allant jusqu'à la fin de vie (R2F) des actionneurs restants : $\{LRA, URA, ULA\}$. En tronquant les données tests à 900 points, le but était de retrouver la valeur de cet index, une variable aléatoire Z^{LLA}, par la méthode d'alignement sur la base de données complète. La figure 4.27 permet alors de montrer que la méthode d'alignement fournit des résultats pertinents à partir de données bruitées et non nécessairement monotones. En effet, la moyenne de la distribution barycentrique est évaluée à 897 soit 3 cycles de différence avec les résultats théoriques escomptés. L'intervalle de confiance à 95 % (représenté en rouge sur la figure 4.27) définit un intervalle de 240 cycles d'incertitude centré autour de la moyenne tel que $I_{95\ \%} = [775; 1015]$ cycles. Ce qui représente, au plus, 13 % d'erreur sur l'estimation d'un cycle dans cette configuration.

Remarque. *Le barycentre obtenu par résolution du problème de transport optimal représente la densité de probabilité de la variable aléatoire Z^{LLA}. Aussi pour en estimer les paramètres de cette distribution, un échantillon de* 1 000 *individus est généré par cette même distribution avec une méthode de génération de données aléatoire [*Virtanen et al.*, 2020]. Ensuite, la moyenne ainsi que l'écart-type empirique sont évalués par la méthode d'estimation bayésienne de [*Oliphant*, 2006] implémentée dans SciPy [*Virtanen et al.*, 2020].*

4.3 Application de la méthode aux données expérimentales

4.3.1 Obtention du temps de vie restant (RUL) avec mesure d'incertitude

Maintenant que la méthode développée permet d'aligner des séries temporelles entre elles, il est possible de la tester sur l'ensemble des données contenues en base $\mathfrak{Y}$ pour obtenir le temps de vie restant —RUL de l'actionneur LLA. Pour ce faire, l'ensemble des descripteurs de LLA sont tronqués à plusieurs reprises pour simuler des données opérationnelles que l'utilisateur récupérerait au fur et à mesure de la vie du produit. Ainsi à partir de la matrice originale $\mathbb{X}^{\text{LLA}} \in \mathcal{M}_{1130,28}(\mathbb{R})$ sont créées 17 matrices, qui chacune possède un nombre de lignes multiples de 50. L'idée est alors d'appliquer la méthode développée dans ce chapitre pour estimer l'indice temporel du dernier point de chaque matrice $\mathbb{X}^{\text{LLA}}[0 : \ldots : 1]$. Rappelons que dans la notation $\mathbb{X}[x : y : n]$, x représente l'indice de la première ligne considérée de la matrice $\mathbb{X}$, y l'indice de la dernière ligne et n est, dans les faits, un facteur de compression ou dilatation représentant la sélection d'une ligne sur n dans la matrice $\mathbb{X}$. La représentation d'un taux de compression permet de tester la robustesse de la méthode d'alignement mais avec les données utilisées dans ces travaux, pour la prédiction de la RUL de cette sous-section 4.3.1, les données ne sont ni compressées, ni dilatées c'est pourquoi $n = 1$ dans la notation $\mathbb{X}^{\text{LLA}}[0 : \ldots : 1]$. De plus, $y = \ldots$ indique que la valeur de y sera variable et représente la valeur du cycle courant, inconnu, à estimer dans la prédiction de la RUL en fonction des profils contenus dans la base $\mathfrak{Y} \in \mathfrak{M}_{1130,28,3}(\mathbb{R})$. Notons que l'actionneur LLA a été retiré de la base de données afin de servir de sujet de test. Ce protocole expérimental a été construit en fonction de la définition de la RUL. En effet, la RUL est définie dans la littérature, et notamment dans [Jia, Cai *et al.*, 2019], comme étant la différence entre le temps de fin de vie connu du système d'étude t_{EOL} et l'instant estimé des données de test t_{match}. En théorie, la relation entre le temps de vie restant et le temps actuel est linéaire décroissante. Certains travaux utilisent cependant une RUL linéaire par morceaux, comme [Hou, Pi et B. Li, 2020] ou [Kefalas *et al.*, 2021] en justifiant qu'en début de vie, la dégradation de certains systèmes n'est que très faible. Aussi, la RUL

peut être représentée dans cette zone par un pallier constant. C'est ainsi qu'à partir d'un certain seuil temporel, le vieillissement est notable et, par effet cumulatif, occasionne le caractère linéaire décroissant bien connu de la littérature. Dans ces travaux, nous ne ferons pas cette hypothèse et nous considérerons que le système commence à vieillir dès ses premiers instants. La RUL obtenue est alors donnée à la figure 4.28.

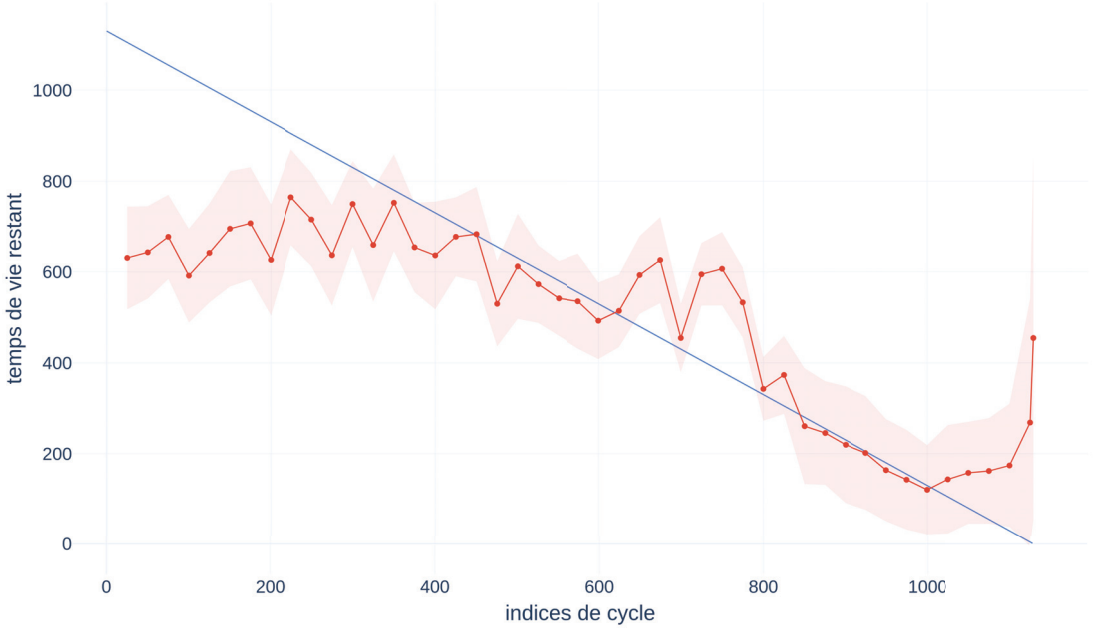


FIGURE 4.28 – Temps de vie restant calculé en alignant la source LLA sur l'ensemble de la base de données $\{LRA, URA, ULA\}$

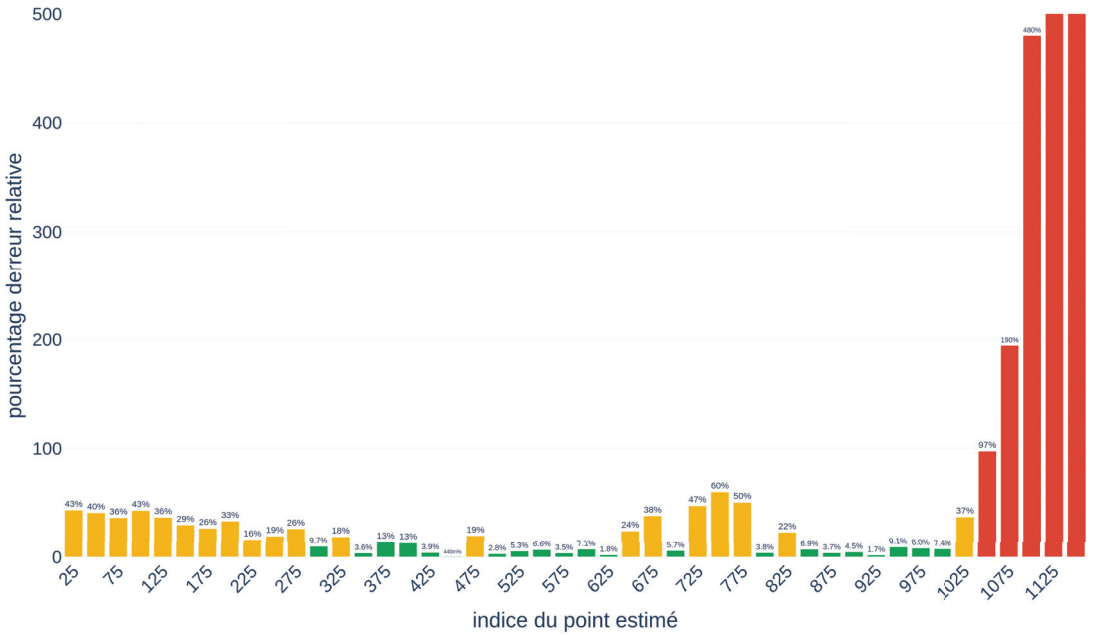


FIGURE 4.29 – Taux d'erreur relative du temps de vie restant estimé par rapport au temps de vie restant théorique

Sur la figure 4.28, la droite bleue modélise la durée de vie restante théorique précédemment présentée. Une méthode de pronostic parfaite, en considérant la modélisation linéaire de la RUL pertinente, générerait des points sur cette même droite. L'ensemble des points rouges reliés représente les estimations de durée de vie calculées par la méthode développée avec leurs intervalles de

confiance à 95 %. Il y a au total 46 points estimés dont 45 qui représentent le résultat de l'alignement $\mathbb{X}^{\text{LLA}}[0:25k:1] \xrightarrow{\mathcal{T}} \mathfrak{Y}, \forall k \in [\![1;45]\!]$ et un dernier point qui correspond à l'alignement de $\mathbb{X}^{\text{LLA}}[0:1129:1] \xrightarrow{\mathcal{T}} \mathfrak{Y}$. Ces résultats peuvent alors se décomposer en quatre parties distinctes. La première concerne l'alignement de $\mathbb{X}^{\text{LLA}}[0:25:1]$ jusqu'à $\mathbb{X}^{\text{LLA}}[0:275:1]$ inclus où les résultats de pronostic ne sont pas pertinents. L'erreur relative moyenne par rapport à la valeur théorique de la RUL est alors de 32 % d'après les données de la figure 4.29. Notons que l'erreur relative est ici utilisée pour augmenter la pénalisation des erreurs d'estimation plus l'estimation se rapproche de la fin de vie effective qui a été fixée au 1130$^{\text{e}}$ cycle dans ces travaux. Cette erreur d'estimation dans les 275 premiers cycles de vis du système s'explique par la dynamique des descripteurs utilisés. En effet, les signaux extraits à la sous-section 3.2.3 pourraient être divisés en deux phases, une première phase avec un taux d'accroissement faible et une seconde phase, témoin de l'accélération du vieillissement, qui aurait la dynamique d'une exponentielle paramétrée croissante. Il est alors difficile pour la méthode de pronostic développée ici de fournir un résultat pertinent lors de cette première phase de dégradation. Le cœur de la méthode étant un alignement de séries temporelles par projection, il est, par nature, compliqué de synchroniser deux segments entre eux si ces derniers possèdent un taux d'accroissement quasi nul. La variation des indices temporels des données n'étant pas prise en compte pour permettre la détection de dilatations et compressions, si les deux segments possèdent les mêmes valeurs, rien ne permet de discerner les points entre eux. Ainsi, dans ce cas, notre algorithme aura besoin d'un minimum de 275 cycles pour fournir des résultats de prédiction pertinents avec la base de données utilisée $\mathfrak{Y}$. La seconde phase est limitée par les points $\mathbb{X}^{\text{LLA}}[0:300:1]$ jusqu'à $\mathbb{X}^{\text{LLA}}[0:625:1]$ inclus. Pendant cette phase les résultats de prédiction sont optimaux avec en moyenne 8 % d'erreur relative par rapport à la durée de vie théorique. Notons que, comme dans les travaux de [Breuneval, 2017], une tolérance de 15 % d'erreur relative par rapport à la RUL théorique a été admise. C'est en cela que le résultat peut être qualifié d'« optimal » et c'est la raison du code couleur utilisé à la figure 4.29. De même, au-delà de 75 % d'erreur relative, les données de la figure 4.29 sont représentées en rouge. Le point $\mathbb{X}^{\text{LLA}}[0:650:1]$ marque le début d'une zone de forte variance qui se termine au point $\mathbb{X}^{\text{LLA}}[0:825:1]$ inclus sur la figure 4.28. Dans cette zone, la prédiction commence avec une erreur relative de 24 % pour atteindre 60 % au 750$^{\text{e}}$ cycle puis redescendre à 22 % au 825$^{\text{e}}$ cycle d'après la figure 4.29. Cette dynamique d'erreur qui pourrait s'apparenter à une gaussienne, témoigne d'une zone pathologique pour l'alignement localisée dans les descripteurs utilisés pour réaliser la prédiction. Une zone de fort bruit présente dans les descripteurs utilisés dans l'étude est donc détectée. Pour rappel, ces derniers ne sont pas strictement monotones. Le troisième phase comprise entre les points $\mathbb{X}^{\text{LLA}}[0:850:1]$ et $\mathbb{X}^{\text{LLA}}[0:1000:1]$ inclus est, quant à elle, pertinente car, au delà de l'illustration visuelle de la figure 4.28 l'erreur relative est quantitativement très faible car atteignant 6 % en moyenne. Enfin la dernière phase commençant au point $\mathbb{X}^{\text{LLA}}[0:1025:1]$ marque le début d'une phase d'explosion quant à l'erreur d'estimation. Au regard des données de la figure 4.29, il n'est même plus pertinent de considérer la moyenne de l'erreur relative et pour des raisons de visualisation, le graphique a été tronqué au-delà de 500 % d'erreur d'estimation. Pour information l'index du dernier point du dernier segment aligné $\mathbb{X}^{\text{LLA}}[0:1129:1]$ a été estimé au 454$^{\text{e}}$ cycle, ce qui représente 45 350 % d'erreur relative. Cela s'explique de même par les dynamiques des descripteurs utilisés. En considérant $X^{\text{LLA}}_{\text{RMS}}$ par exemple, on peut voir que son profil est croissant, au bruit près, jusqu'au 992$^{\text{e}}$ cycle où la valeur du descripteur chute de 97 % par rapport à sa précédente valeur au 955$^{\text{e}}$ cycle. Elle revient ensuite à sa valeur initiale au 1072$^{\text{e}}$ cycle. Le grippage de l'actionneur a débuté d'où un comportement erratique des descripteurs. Évaluer une RUL devient alors problématique. Le temps de vie restant est surestimé dans cette zone parce que les points projetés sont attirés par le premier front du descripteur, à peu près 100 cycles avant la fin de vie. Cette dynamique étant partagée par tous les descripteurs, à des taux de variations différents, le résultat de pronostic est alors impacté. Dans les faits, cette situation qui peut apparaître pendant des essais de vieillissement ne serait que peu probable en opérationnel où au moindre doute sur la capacité du système à remplir sa fonction, ce dernier passera en maintenance.

Maintenant que la RUL a été obtenue, intéressons-nous à la robustesse de la méthode de pronostic aux points manquants dans les séries temporelles à aligner.

4.3.2 Adaptation aux changements de conditions opérationnelles

Dans la sous-section 4.3.1 précédente, la méthode de pronostic a été appliquée à la série $X^{\text{LLA}}_{\text{RMS}}[0:900:1]$. Cette série temporelle était alors une version tronquée à 900 points du descripteur original

$X_{\text{RMS}}^{\text{LLA}}$. Aucun facteur de compression n'avait été appliqué aux données. Cependant, un facteur de compression pourrait permettre de modéliser simplement différentes conditions opérationnelles lors de l'utilisation du système surveillé. En effet, dans le cas d'un aéronef, lorsque ce dernier passe par des environnements à basses températures, le vieillissement des actionneurs peut être accéléré par rapport à d'autres vols qui auraient lieu en climat tempéré. Typiquement, si ce dernier possède de la graisse comme moyen de lubrification, elle serait plus visqueuse pour des basses températures ce qui peut accélérer la dégradation de l'actionneur en question. C'est pourquoi il est important d'étudier l'influence d'un tel facteur sur la méthode de pronostic. L'alignement de $X_{\text{RMS}}^{\text{LLA}}[0:900:k]$ avec $k \in [\![1;10]\!]$ sur la base de données $\mathfrak{N}$ sera donc effectué. Le but de l'opération est alors de vérifier que la méthode, malgré le taux de points manquant, permet de toujours détecter le cycle actuel de la série temporelle, ici, le 900$^{\text{e}}$. Les résultats de l'étude pour différents taux de compression sont donnés à la figure 4.30.

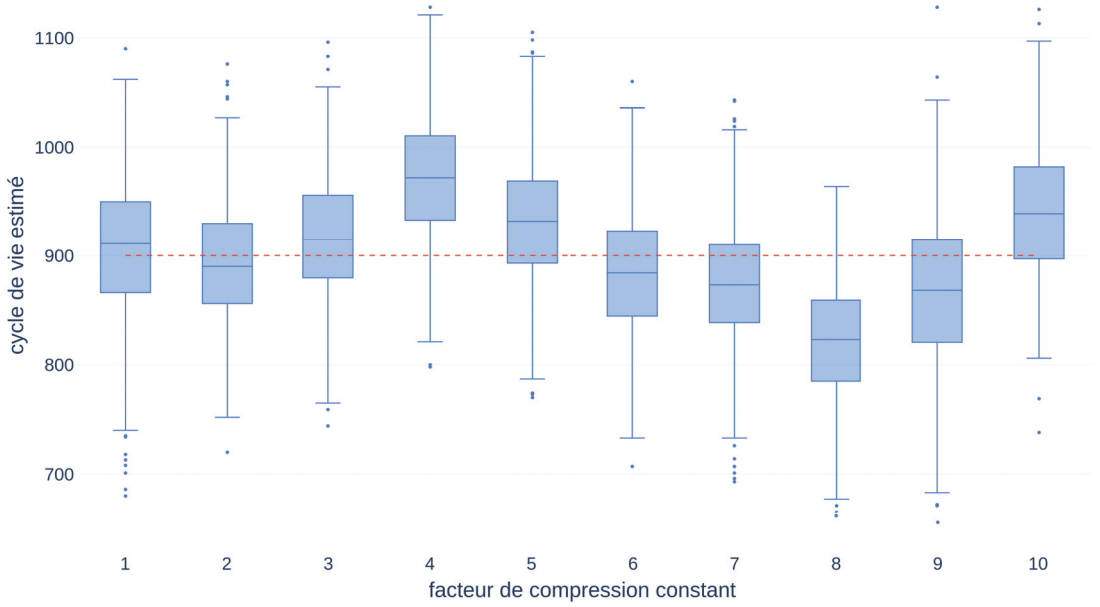


FIGURE 4.30 – Influence du facteur de compression sur les résultats de pronostic

Pour chaque résultat de pronostic, comme expliqué à la remarque de la sous-section 4.2.6, une population d'indices possibles est générée à partir de la distribution barycentrique de Wasserstein. Les caractéristiques de chaque échantillon ainsi créé sont alors représentées par la figure 4.30 sous la forme de boîtes à moustaches. Notons que la distribution de la solution de l'équation 4.27 est une gaussienne symétrique, si bien que la médiane est, aux erreurs d'échantillonnage près, confondue avec la moyenne de la distribution. Elle est représentée par un trait bleu horizontal pour chaque boîte et la valeur théorique à atteindre, le 900$^{\text{e}}$ cycle, est représenté sur le graphique par la droite pointillée rouge horizontale, l'objectif de la méthode étant de faire correspondre la moyenne de la valeur du dernier indice du dernier point de la série temporelle $X_{\text{RMS}}^{\text{LLA}}[0:900:k]$ avec la valeur théorique. Remarquons alors qu'une relation évidente entre le cycle de vie estimé et le facteur de compression ne peut pas être extraite de la figure 4.30. Tout au plus, l'on pourrait deviner une augmentation de la variance de la valeur du cycle estimée à partir d'un point sur 7 sélectionné dans la série $X_{\text{RMS}}^{\text{LLA}}[0:900:k] \in \mathcal{M}_{\lfloor \frac{900}{k} \rfloor,28}(\mathbb{R})$. Cela pourrait s'expliquer par le fait que, par nature, la méthode est sensible aux dynamiques des signaux à aligner. Donc, si le sous-échantillonnage ne permet plus de représenter certains pics ou motifs reconnaissables dans le signal, la méthode ne pourra pas en tenir compte dans la phase d'alignement. *De facto*, le résultat de prédiction s'en voit affecté. Notons tout de même que la limitation est ici théorique et les problèmes de sous-échantillonnages sont assez bien décrits dans la littérature. Ce n'est donc pas une limitation de la méthode développée ici. Par conséquent, la méthode de pronostic permet de réaliser automatiquement la normalisation temporelle introduite par [MAI et CHEVALIER, 2016] et il est donc possible de conclure quant à sa robustesse aux variations d'échantillonnages des séries étudiées.

Pour permettre le pronostic à partir de données captées dans des environnements hétérogènes, il est aussi important que la méthode puisse fournir des résultats pertinents en considérant non pas des séries temporelles complètes, mais des segments de séries temporelles. Si cette modélisation fait sens après étude des différentes conditions opérationnelles sur le terrain, une série de données pourrait être segmentée en fonction des disparités de conditions d'usage du système surveillé. Dans la pratique, les données à notre disposition dans ces travaux ont été captées dans des conditions opérationnelles identiques. Aussi, afin de simuler un tel comportement, la série de test X^{LLA} sera tronquée pour commencer au 500$^{\text{e}}$ cycle supposant de fait, que les points captés avant l'avaient été dans des conditions de fonctionnement autres. Nous faisons alors l'hypothèse que notre méthode de pronostic est assez robuste pour tenir compte de ce type de configurations. Pour le montrer, la matrice des descripteurs sources $X^{\text{LLA}}[500:900:1] \in \mathcal{M}_{400,28}(\mathbb{R})$ est utilisée. Ainsi le segment testé commence au 500$^{\text{e}}$ cycle et se termine au 900$^{\text{e}}$ cycle, ce dernier devant être retrouvé par l'estimation de la RUL. L'objectif reste identique à celui de la sous-section 4.3.1. Le résultat est donné à la figure 4.31.

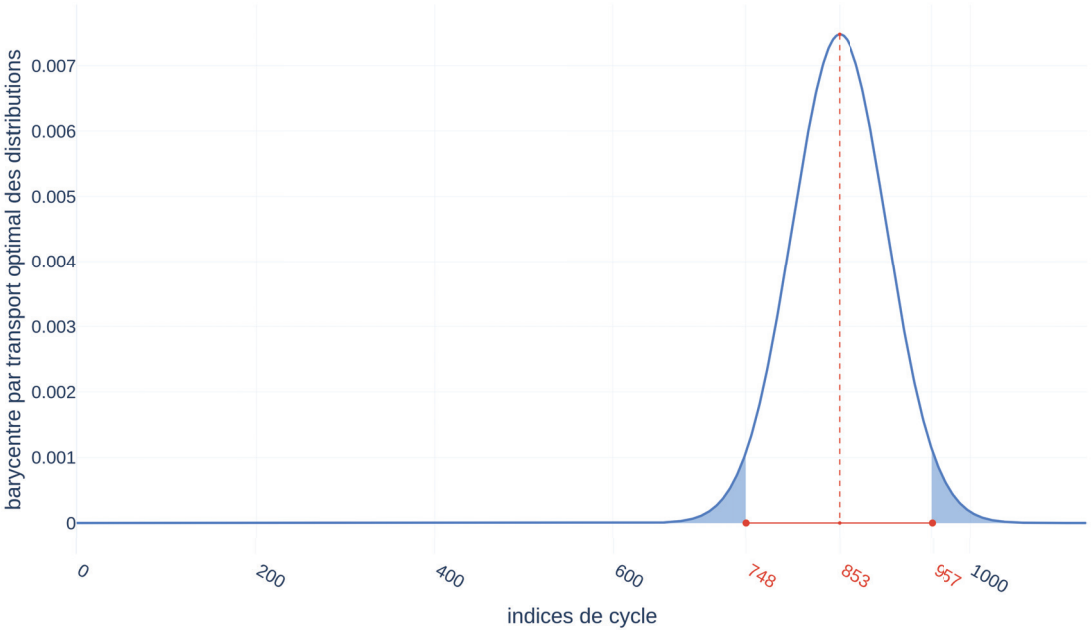


FIGURE 4.31 – Barycentre dans l'espace de Wasserstein pour l'alignement de la matrice $\mathbb{X}^{\text{LLA}}[500:900:1]$ sur la base d'apprentissage $\mathfrak{Y} \in \mathfrak{M}_{m,p,(q-1)}(\mathbb{R})$

Bien que le résultat de la figure 4.27 estimait le cycle actuel moyen au 897$^{\text{e}}$ cycle, le fait d'enlever 500 points décale la moyenne du cycle estimé à 853 d'après la figure 4.31, soit une erreur relative de 5 % par rapport à l'objectif d'une moyenne de cycle à 900. Avec ces résultats, et au regard de la marge d'erreur choisie de 15 %, les points supprimés semblent ne pas influer sur l'alignement. Notons que nous avons délibérément choisi de supprimer plus de 300 points au début de la série, compte tenu de nos remarques précédentes. Cette zone correspondait à une zone de faible variance détectée à la figure 4.28. Avec ce résultat, il est possible de conclure que la méthode de pronostic développée reste pertinente pour l'étude de segments incomplets de données. De manière plus générale, avec les résultats de cette section, nous avons montré que la méthode de pronostic est adaptable et fournit des résultats pertinents pour des séries temporelles incomplètes. L'atout de cet méthode peut alors être utilisé pour le pronostic d'un système qui passerait par plusieurs conditions opérationnelles différentes. Le schéma de principe est représenté à la figure 4.32.

La figure 4.32 décrit une procédure qui permettrait d'adapter notre méthode de pronostic au changements de conditions opérationnelles d'un système. S'il elle s'avère pertinente, ce dernier point permettrait de faire un pas de plus vers une méthode fonctionnelle qui aurait sa place sur un théâtre d'opérations réelles. De fait, l'adoption du processus PHM n'en serait que facilité. Considérons un premier enregistrement. Sa série temporelle obtenue est comparée aux séries temporelles stockées en base de données. Après alignement, une première estimation du cycle de vie courant et donc, de la consommation de durée de vie, est obtenue. À l'issue de cette première phase, un nouvel enregistrement est effectué en vol avec de nouvelles conditions opérationnelles. La nouvelle

série temporelle est alignée sur les séries de la base en prenant comme origine le même taux de vieillissement que celui obtenu à la phase précédente. Jusqu'à la fin de vie du composant, cette procédure se répète. Durant chaque enregistrement, les conditions opérationnelles peuvent changer.

Remarque. *La fin de vie d'un actionneur n'est pas nécessairement l'instant à partir duquel son intégrité physique est mise à mal mais représente l'instant à partir duquel l'actionneur perd sa fonction.*

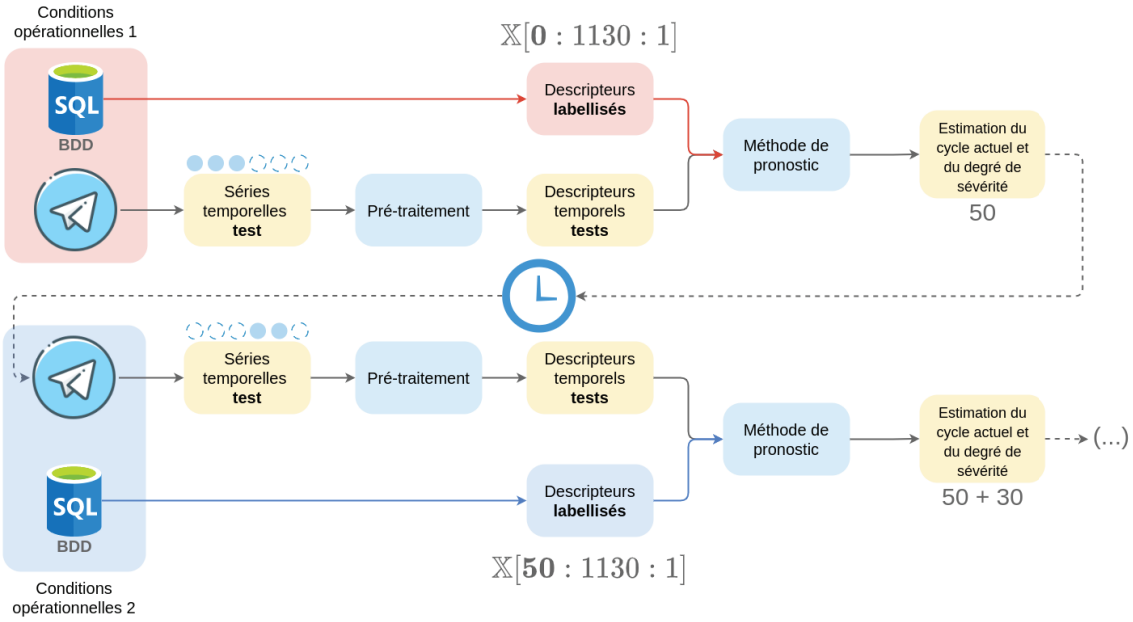


FIGURE 4.32 – Schéma de principe pour le pronostic de deux jeux de données récupérés sur un même aéronef mais sous deux conditions opérationnelles différentes

Notons alors que la base de données considérées peuvent être constituées aussi bien de comportement usuels d'actionneurs que de comportement issus d'essais de vieillissement accélérés. Ce dernier cas représente celui de ces travaux. La capacité de la méthode de pronostic à fournir des résultats d'estimation de durée de vie généralisable à différentes conditions opérationnelles tient donc à la richesse des comportements contenus en base de données et à la robustesse de notre approche. Aujourd'hui, notre base n'est pas suffisamment complète, demain, la qualité du résultat de pronostic augmentera au fur et à mesure que des données d'exploitation, ou d'essais de vieillissements, seront récoltées.

Enfin, cette méthode peut être couplée avec le classifieur présenté à la section 3.3. En effet, ce dernier a permis de montrer comment déterminer les différentes degrés de sévérités présents en base de données. Il suffit alors, après alignement, de déterminer la classe correspondante du dernier point de la source aligné à partir de celui de la cible correspondante comme montré par la figure 4.1. Ainsi, l'état de santé courant des séries est déterminé et le temps de vie restant ou RUL est calculé.

Conclusion

Dans ce chapitre 4 nous avons pu développer une méthode de pronostic à base d'alignement de séries temporelles robuste, applicable à des séries temporelles compressées ou dilatées, incomplètes ou encore à des segments de séries. Dans un premier temps, un état de l'art sur les méthodes de pronostic à partir de mesures de similarité entre séries temporelles a été réalisé à la section 4.1. Cette revue de la littérature a permis de souligner les nouveautés apportées à l'état de l'art par la méthode développée ici, notamment le fait qu'elle n'a pas besoin de définir les paramètres d'une fenêtre glissante pour fournir un résultat de comparaison et de manière générale, fonctionne de manière automatique sans avoir à choisir d'hyperparamètres *a priori*. De plus, elle permet d'aligner des séries qui n'auraient pas la même fréquence d'échantillonnage et il n'est pas non plus nécessaire

de définir des intervalles sur lesquels effectuer l'alignement. La méthode de pronostic complète décrite à la section 4.2, que nous baptiserons Partial Time scaling Invariant Temporal Alignement for Remaining Useful Life Estimation (PARTITA-RULE), est constituée de deux parties principales : une méthode d'alignement de séries temporelles, Partial Time scaling Invariant Temporal Alignement (PARTITA) et un ensemble d'algorithmes permettant de mesurer un incertitude et générer un modèle de fusion d'information. De plus, utilisée en conjonction avec les résultats de partitionnement du chapitre 3.3, donnés par Soft clUsteriNg foR tIme SEries (SUNRISE), elle permet de réaliser le diagnostic ainsi que le pronostic d'actionneurs électromécaniques et, plus généralement, de séries temporelles. PARTITA-RULE permet alors d'obtenir une RUL avec une mesure d'incertitude associée, ce qui pourrait servir de support à la prise de décision pour la maintenance prédictive dans des domaines tels que l'aéronautique. Ses performances ainsi que sa robustesse aux points manquants et (théoriquement) aux changements de conditions opérationnelles ont enfin été présentés à la section 4.3.

Chapitre 5

Conclusion et perspectives

Ces travaux de thèse ont été développés en quatre sections principales. Dans un premier temps, le contexte industriel a été présenté en introduction. Le Prognostics and Health Management (PHM) est un ensemble de méthodes permettant de répondre aux défis climatiques de demain en partie parce qu'il autorise la démocratisation d'actionneurs électromécaniques en aéronautique. Ce *framework* permettrait alors de prévenir de potentiels défauts qui étaient jusque-là non présents dans les actionneurs hydrauliques ou pneumatiques. De plus, l'élaboration de méthodes de détection et de prédiction de défauts contribue au principe de conception durable. En effet, la remontée des défauts récurrents et des compte rendus d'exploitations, permise par la surveillance continue du système considéré, crée une information qui pourrait être exploitée par un bureau d'étude afin de guider la création d'autres systèmes. Au-delà du volet environnemental, cet ensemble de méthodes constituant le PHM, adapté au domaine industriel, produit des gains substantiels sur le volet économique. En effet, en lieu et place des maintenances préventives ou imprévues subies par les compagnies aériennes, le PHM rend possible rend possible la prédiction de l'occurrence de défauts puis la planification de la maintenance dans un calendrier optimal vis à vis du service de l'aéronef. Par effet de conséquence, cela limiterait l'immobilisation des aéronefs au strict nécessaire et donnerait une visibilité plus importante aux compagnies exploitantes pour organiser leur carnet de maintenance. Dans ces travaux, comme vu à la section 1.3, cette méthodologie a été appliquée à un système d'inverseur de poussée électrique développé par Safran Electronics & Defense, l'Electric Thrust Reverser Actuation Systems (E-TRAS). Comme présenté au chapitre 3, les données utilisées seront des signaux vibratoires captés sur un système subissant des essais de vieillissement accélérés pendant l'année 2017. Le fort ancrage industriel de ces travaux tient du fait, en plus de l'utilisation d'un jeu de données industrielles, que toutes les méthodes ont été développées en tenant compte des dynamiques des signaux industriels étudiés, loin d'être parfaitement corrélées au vieillissement de l'actionneur. De plus cette méthode a été pensée pour pouvoir être mise à l'échelle rapidement en production.

Avant de se concentrer sur l'objectif de la thèse qui était de déterminer la durée de vie restante d'un système par des méthodes d'intelligence artificielle, la nature et les méthodes plus classiques utilisées dans le PHM sont explicitées à la section 2.1. Puis, toujours dans le premier chapitre, l'approche connexionniste est présentée à l'aune de la résolution d'un problème de classification, ce qui revient à résoudre un problème de diagnostic dans notre domaine à la section 2.2. Par la suite, cette structure de réseau de neurones a évolué vers des architectures profondes dont les principaux éléments sont présentés à la section 2.3. L'usage de ces techniques dans la littérature est alors étudié au travers de travaux de diagnostic et de pronostic notamment à partir de roulements ou de machines tournantes. Cette recherche a permis de mettre en lumière quelques points, notamment la supériorité des architectures profondes pour la création de descripteurs de défauts, leur usage de plus en plus fréquent dans le domaine ainsi que leurs limites lorsqu'il s'agit de traiter des systèmes pour lesquels peu de données, ou des données fortement biaisées sont possédées. Au-delà du développement d'infrastructures spécifiques, le PHM reste cependant un domaine applicatif dans lequel les algorithmes d'intelligence artificielle sont plus utilisés que créés, c'est pourquoi ces travaux n'ont pas eu pour objectif de créer un nouveau réseau ou une nouvelle méthode d'apprentissage. L'objectif, plus large, était de créer une méthode de PHM complète du pré-traitement de la donnée à la détermination de durée de vie restante d'un actionneur. La première phase de ce projet consiste à créer une méthode d'assignation de labels automatique, décrite et expliquée au chapitre 3.

Cette étape est nécessaire dans un premier temps car les données récupérées sur l'inverseur de poussée E-TRAS ne sont pas labellisées dans le sens où les différents degrés de sévérité de défauts du système, autrement dit son état de santé, nous sont inconnus. Par conséquent, avant de pouvoir rendre ces données utiles pour un algorithme d'apprentissage, il convient d'affecter une étiquette à chacun des points étudiés. L'idée initiale étant plus tard de pouvoir réaliser un apprentissage supervisé à partir de ces mêmes données pour les structures de diagnostic et de pronostic. La méthode résultante, Soft clUsteriNg foR tIme SEries (SUNRISE), a été développée au chapitre 3. Ne possédant pas de labels *a priori*, SUNRISE est une méthode de partitionnement de données ou *clustering*, ce qui justifie la réalisation d'un état de l'art concernant le partitionnement de séries temporelles à la section 3.1. Dans ces travaux, cette méthode a été appliquée aux séries temporelles extraites à partir des données brutes captées sur système. L'extraction des descripteurs de défauts, qui sont des grandeurs statistiques, est détaillée à la section 3.2. De manière générale, tous les algorithmes développés ont été testés sur ces dernières signatures. Ces signaux constituent ainsi notre base de données d'intérêt. Le fonctionnement de la nouvelle méthode de *clustering* de séries temporelles, SUNRISE, est ensuite détaillé à la section 3.3. Résumons ainsi son fonctionnement :

à partir de l'ensemble des données des descripteurs, un schéma de partitionnement est défini pour tous les points étudiés. Pour ce faire, les séries temporelles sont transformées en images, chaque image est alors segmentée par un réseau de neurones profond. C'est à partir de ces images représentant une matrice de distance qu'est extrait un signal de frontière représentant les limites avec mesures d'incertitudes de chaque groupement de données. Enfin, aucun indicateur présent dans la littérature ne permettait de quantifier le résultat de partitionnement obtenu au sens où aucun ne permettait de quantifier les hypothèses faites lors de nos travaux. Partant de cet état de fait nous avons développé notre propre indicateur de qualité à la section 3.4. Ce dernier mesure la compacité des groupes obtenus conjointement avec une mesure de cohérence temporelle du schéma de partitionnement. SUNRISE ainsi que l'indicateur ont pu être comparés aux méthodes usuelles de l'état de l'art dont le résultat nous a permis de montrer l'intérêt de cette approche. Les données utilisées ont alors été étiquetées en fonction de leur degré de sévérité probable. Notons que la mesure d'incertitude du schéma de partitionnement est inhérente à la méthode développée.

Maintenant que chaque point des descripteurs possède un label représentant le degré de sévérité, il convient de développer une méthode pour déterminer la durée de vie restante du système à partir de ces données. Possédant une base de données constituée de comportements probables d'actionneurs jusqu'à fin de vie, l'idée est de comparer de nouvelles données captées en opérationnel pour les comparer à cette base. L'état actuel et futur du système sera ensuite déterminé par l'état d'un mélange de candidats connus. Pour cela, la méthode Partial Time scaling Invariant Temporal Alignement for Remaining Useful Life Estimation (PARTITA-RULE) a été développée au chapitre 4. Le cœur de la méthode repose sur la méthode Partial Time scaling Invariant Temporal Alignement (PARTITA) d'alignement de séries temporelles, c'est pourquoi un état de l'art des méthodes de pronostic à partir d'alignement de séries a été effectué à la section 4.1. Cette nouvelle méthode permet d'aligner deux séries temporelles de tailles différentes, indépendamment de l'existence d'un facteur d'échelle temporel entre les deux séries et ceci avec un alignement partiel. Cet alignement est qualifié de *partiel* au regard des autres méthodes de l'état de l'art. En effet, à la différence de Dynamic Time Wrapping (DTW) qui a besoin de connaître l'intervalle sur lequel aligner deux séries, cette méthode ne le nécessite pas. Elle détermine automatiquement les intervalles pertinents pour un alignement de bonne qualité. Ces trois dernières propriétés procurent à PARTITA un avantage sur les méthodes de la littérature. Cette nouvelle méthode est par la suite complétée de plusieurs étapes algorithmiques afin de pouvoir permettre l'estimation de durée de vie restante ou Remaining Useful Life (RUL) avec mesure d'incertitude, d'un actionneur électromécanique. La section 4.3 clôture alors le chapitre 4 ainsi que les travaux de recherche de cette thèse, en appliquant la nouvelle méthode développée, PARTITA-RULE, aux données de l'inverseur de poussée E-TRAS. Elle permet de montrer la robustesse de la méthode développée et sa pertinence pour effectuer le diagnostic et le pronostic d'actionneurs à partir de profils d'exploitation.

L'ensemble de la méthode de PHM développée dans ces travaux est résumé visuellement par la figure 5.1.

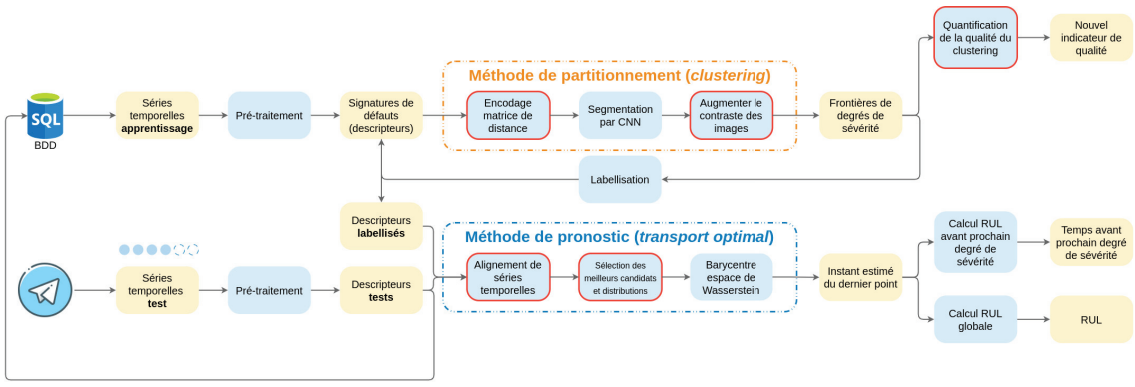


FIGURE 5.1 – Méthode de PHM développée dans ces travaux

La figure 5.1 représente l'ensemble de la méthode développée. Elle a été construite autour de deux axes principaux : la phase de partitionnement de séries afin de labelliser des données inconnues et la méthode de pronostic qui permet d'obtenir à la fois l'état de santé courant et l'état de santé futur du système. Détaillons maintenant la figure 5.1 dans l'ordre de la présentation des travaux. Le processus débute avec la base de données Structured Query Language (SQL) en

haut à gauche. Cette dernière contient les données brutes récoltées sur système, dans notre cas ce sont des signaux vibratoires. Après pré-traitement, ces séries sont labellisées par SUNRISE. La qualité du partitionnement est évaluée *a posteriori* par le nouvel indicateur de qualité χ. Une fois cette étape faite, des descripteurs labellisés sont obtenus. Parallèlement à cette étape, des signaux bruts sont récupérés sur le système en vol (représenté par l'avion en papier sur la figure 5.1) et les mêmes descripteurs de défauts que ceux de la branche précédente sont calculés. Il devient désormais possible de comparer les descripteurs représentant des données incomplètes (les descripteurs **tests**) aux descripteurs **labellisés**. Les descripteurs tests sont alors alignés grâce à PARTITA sur l'ensemble des descripteurs connus et caractérisés, puis deux informations sont obtenues à ce stade : une distribution représentant l'instant du dernier point aligné ainsi que la RUL globale du système. Une fois que les nouvelles données ont été traitées, les descripteurs tests sont enfin rajoutés à la base de données pour venir l'enrichir pour l'étude des prochains relevés. Notons que sur la figure 5.1 certains processus sont entourés en rouge. Cette distinction représente les méthodes qui sont nouvelles par rapport à la littérature.

Soulignons maintenant l'apport de ces travaux par rapport à ceux qui sont présentés dans la littérature existante. Le cœur de la méthode développée repose sur une méthode de partitionnement de séries temporelles, SUNRISE, afin de réaliser la détection de défauts et une méthode d'alignement de séries temporelles, PARTITA, qui, avec le rajout d'une technique de *fusion* de distributions, constitue la méthode de pronostic. La première se distingue des autres méthodes connues dans l'état de l'art car cette méthode de *clustering*, fondée sur une mesure de distance, ne requiert pas le choix du nombre de *clusters a priori* à trouver dans les données. De plus, outre les hyperparamètres d'apprentissage du réseau de neurones, les méthodes de traitement du signal qui extraient les frontières de groupes ne requièrent pas non plus d'hyperparamètres, comme par exemple la taille d'une fenêtre glissante. Ensuite, cette méthode est adaptée à l'étude du regroupement de signaux continus que sont des séries temporelles. Enfin, un nouvel indicateur de qualité χ a été développé pour quantifier la qualité d'un tel partitionnement. Dans le domaine du PHM où l'hypothèse qu'une régénération d'un système sous surveillance est impossible, cet indicateur permet de prendre en compte cette hypothèse en pénalisant tout retour en arrière dans le schéma de partitionnement. Hormis l'indicateur de cohérence temporelle de [Breuneval, 2017], cette propriété n'est pas étudiée dans l'état de l'art. Cependant cet indicateur n'était pas pertinent pour quantifier la pertinence d'un schéma de partitionnement avec nos données. La deuxième méthode d'alignement de séries temporelles, PARTITA, a été développée après le constat de plusieurs limites des méthodes d'alignement de la littérature, au-delà même de la problématique de PHM. Tout d'abord, elle est invariante aux dilatations et aux compressions des signaux à aligner. Ainsi, à la manière de DTW, elle peut sauter des points dans l'assignation de voisins, ce qui la rend adaptable à une grande variété de dynamiques de signaux. De plus, contrairement à DTW, et aux autres méthodes d'alignement de séries, il n'y a pas besoin de déterminer des intervalles de données à partir desquels l'alignement sera effectué. Cette dernière information est déterminée automatiquement par l'algorithme. Donc il n'y a pas besoin non plus de calculer une mesure de similarité sur une fenêtre glissante. Elle est enfin robuste au bruit contenu dans les séries avec lesquelles elle travaille. De manière générale, ces travaux de recherche ont été guidés par une forte volonté d'automatisation des algorithmes développés. Le fait de limiter, voire de supprimer, le choix d'hyperparamètres dans les méthodes développées permet à ces dernières de s'affranchir de tout savoir expert à rajouter au cours du *pipeline* de traitement de la donnée. De fait, cela apporte aux méthodes de partitionnement et de pronostic un pouvoir de généralisation plus important.

Discutons maintenant des hypothèses effectuées pour mener à bien ces travaux. Tout d'abord, la première hypothèse réalisée est l'hypothèse faible 1 qui suppose que l'actionneur surveillé ne se dégrade qu'en fonctionnement. En effet, outre les systèmes de stockage de l'énergie tels que les batteries pour lesquelles le vieillissement calendaire est important [Krupp *et al.*, 2021], il est raisonnable de penser qu'un système mécanique et, dans le cas de notre inverseur de poussée E-TRAS, une machine synchrone à aimants permanents (MSAP) accouplée avec 4 vis à billes via des transmissions flexibles, ne se dégrade pas lorsqu'il est n'est pas utilisé. Toujours dans les hypothèses faibles, l'hypothèse 6 imposant que la séquence source à aligner soit de taille inférieure à la séquence cible dans la méthode PARTITA n'est pas limitante. En effet, sur-échantillonner la source pour permettrait de respecter ce prérequis de l'algorithme. Une hypothèse plus forte concerne l'hypothèse 10 sur la distribution de la variable aléatoire Z_j^h représentant la valeur de l'indice du cycle du dernier point aligné à chaque itération k. Ainsi, sa distribution est supposée suivre une loi normale, ce qui permet de simplifier les calculs pour déterminer le schéma de pondération dans le modèle de fusion. Seulement rien ne garantit que ce soit le cas dans la pratique et la limite de la

méthode induite par la symétrie de ce type de loi est discutée à la section 4.2.5. Enfin, plusieurs hypothèses fortes sont réalisées dans ces travaux, à commencer par celles qui sont effectuées sur l'environnement opérationnel de l'actionneur. L'hypothèse 4 qui interdit toute maintenance dans le cycle de vie, conjuguée à l'hypothèse 3 qui interdit toute régénération du système en opération, mènent naturellement à l'hypothèse 5 qui suppose qu'une défaillance ne peut que s'aggraver au cours du temps. Ces hypothèses seront très probablement non respectées lors de l'exploitation du produit durant laquelle l'inverseur de poussée passera obligatoirement par (au moins) l'ensemble des maintenances préventives. Elles ont été faites car la méthode de partitionnement permet la détection de changement d'état dans les données, mais, étant principalement une méthode de détection, elle ne permet pas de remonter à la cause d'un défaut et donc d'inférer sur la sévérité de ce dernier. Les degrés de sévérité trouvés sont affectés de manière séquentielle. Cette limite pourrait être levée en évaluant non pas des données représentant un cycle de vie produit complet, mais des données qui seraient captées entre deux opérations de maintenance. La méthode fonctionnant pour des séries de tailles quelconques, elle pourrait s'y adapter et nous supposons que les résultats resteraient pertinents. L'hypothèse 2, quant à elle, est fondamentale dans ces travaux et a été réalisée pour trouver un palliatif au verrou 1. Elle concerne l'indépendance des actionneurs étudiés. Rappelons que l'inverseur de poussée E-TRAS est constitué d'une MSAP mettant en mouvement 4 actionneurs liés à la machine tournante et entre eux par des liaisons mécaniques souples. Ainsi, le mouvement d'un actionneur n'est absolument pas indépendant de celui de ses voisins. Imaginons un actionneur grippé ne permettant plus la translation de l'écrou le long de son axe longitudinal, lors de la commande d'ouverture de la nacelle, la translation des trois autres actionneurs sera bloquée par cet élément en défaut. Un actionneur qui serait en parfaite santé serait alors impacté par le blocage et, *de facto*, s'userait prématurément, voire se casserait. Par cet exemple extrême, nous voulons mettre en exergue le fait que le vieillissement d'un actionneur influe sur celui de ses voisins. Ne possédant pas assez de données, il était primordial de multiplier artificiellement les sources de séries temporelles et donc de ne plus considérer l'actionneur à surveiller comme un inverseur de poussée, mais comme une multitude de vis à billes.

Développons maintenant les perspectives de ces travaux, en commençant par les axes d'amélioration des méthodes proposées pour résoudre le problème de PHM. Au-delà des hypothèses fondamentales faites dans ces travaux et passées en revue précédemment, certains points mériteraient une attention particulière. Notamment en reprenant la première phase de pré-traitement des données, il serait intéressant de tester l'apport d'une méthode de sélection de signatures au *framework* développé ici. En effet, même si la sélection est réalisée de manière implicite par le schéma de pondération dans la méthode de pronostic, elle est moins directe dans la méthode de partitionnement où chaque descripteur contribue à l'obtention d'une frontière de groupes globale. Afin d'obtenir des résultats mieux corrélés au vieillissement nous pourrions dès lors sélectionner les descripteurs de défauts sur la base de deux critères : leur corrélation par rapport au vieillissement ainsi que leur monotonie. Ces méthodes sont réalisées dans l'état de l'art avec des critères de corrélation statistiques linéaires ou non-linéaires, de comparaison de la mesure d'entropie entre deux signatures pour mesure leur qualité d'information intrinsèque, ou encore d'une étude plus complète sur l'influence de certains descripteurs directement sur les résultats de pronostic. Toujours dans la méthode SUNRISE, un axe d'amélioration se trouverait dans l'apprentissage de U-NET pour la segmentation d'images. En effet, ce dernier n'ayant été entraîné que pour des motifs de taille $n \times n = 1130 \times 1130$ dans ces travaux, les résultats de segmentation pour des motifs de tailles différentes ne sont pas pertinents. Afin d'augmenter la généralisation de cette méthode il convient donc d'étendre la base d'apprentissage à des motifs de tailles différentes, voire même de modifier la structure du réseau pour le rendre invariant à la mise à l'échelle des motifs produits par les matrices de distances. Enfin, lors de l'estimation de densité par noyau ou Kernel Density Estimation (KDE), la sélection de la bande passante, paramètre de la méthode, reste un sujet de recherche ouvert. À défaut, la règle empirique de [Silverman, 1986] avait été choisie mais lorsque la fonction noyau n'est plus une gaussienne, cette règle n'est plus valable et la sélection d'une bande passante à la pertinence démontrée pour un problème donné reste aujourd'hui un problème d'optimisation non trivial. Concernant la méthode d'alignement de séries temporelles, plusieurs points pourraient être améliorés en commençant par le choix des directions de projection. Aujourd'hui, l'ensemble des directions de projection est évalué par la résolution d'un problème d'optimisation, ce qui représente un coût de calcul important avant de pouvoir effectuer un alignement pertinent. Il s'agirait donc de déterminer théoriquement la forme des projections idéales pour obtenir un bon résultat de partitionnement, sans rajouter de paramètres à choisir de manière empirique afin de rester indépendant de tout réglage manuel. Sur le même sujet, une étude pourrait être faite sur le

nombre de directions optimales à choisir pour améliorer les résultats d'alignement. Enfin, comme il a déjà été évoqué, la création de distributions intermédiaires avant de calculer un barycentre dans l'espace de Wasserstein pourrait être réalisée avec des distributions différentes des gaussiennes, qui modélisent un comportement symétrique.

Au-delà des perspectives d'évolution des méthodes développées, les verrous scientifiques sous-jacents à ces travaux de thèse pourraient être levés de plusieurs manières. Tout d'abord concernant le verrou 1, et plus précisément la captation de signaux relatifs aux actionneurs électromécaniques, des architectures de génération de données artificielles telles que des Generative Adversarial Network (GAN) pourraient être entraînées et utilisées pour pallier le défaut de données. Ces réseaux sont d'ailleurs déjà utilisés dans des travaux de diagnostic [Zirui Wang, J. Wang et Youren Wang, 2018] ou [X. Wang *et al.*, 2018]. Cette approche est d'ailleurs suivie par [S. Shao, P. Wang et Yan, 2019] où, pour pallier l'insuffisance de données pour le diagnostic de machines électriques, des données sont simulées pour rendre la base de données d'apprentissage équilibrée. Cette approche à base de données artificielles permettrait de s'affranchir en partie d'essais de vieillissements coûteux sur des systèmes aéronautiques complexes. Dans la même optique, la création d'un modèle physique, qui serait un jumeau artificiel du système à surveiller, permettrait de générer des données de simulation au même titre qu'un GAN entre autres. De plus, au contraire des architectures d'apprentissage profond dont le fonctionnement reste opaque (difficilement explicable au regard du nombre de neurones et du nombre de couches mises en jeu dans ces modèles), un jumeau numérique permettrait de remonter aux causes des pannes, de prendre en compte l'état d'un système, d'extrapoler sa durée de vie dans certaines conditions opérationnelles et, pour les compagnies d'exploitation, de conserver une traçabilité des défauts survenus. Si un tel modèle est élaboré en parallèle de la phase de conception d'un actionneur, il pourrait orienter la conception de ce dernier et reviendrait à un coût moindre par rapport à la réalisation d'essais de vieillissement accélérés. Concernant le verrou 2 et le changement de conditions opérationnelles, nous avons apporté des éléments de réponse qui attendent d'être testés à la sous-section 4.3.2. De même que précédemment, la prise en compte des différentes conditions pourrait être effectuée dans le modèle de génération de données ou dans le jeu de conditions environnementales renseigné lors d'une simulation d'un jumeau numérique. Enfin, pour le verrou 3 et le transfert des algorithmes de PHM sur cible embarquée, au regard des fréquences d'acquisition des signaux lors de ces travaux, les méthodes SUNRISE et PARTITA n'auraient pas besoin d'être embarquées sur aéronef par exemple. Seul le pré-traitement et donc l'obtention des descripteurs pourrait se faire en ligne, ce qui éviterait de surcharger le dimensionnement des calculateurs embarqués avec des algorithmes d'intelligence artificielle et, de fait, d'éviter le problème encore ouvert de la certification de ce type de modèles en environnement industriel critique. À l'issue de chaque vol, les données des descripteurs pourront être rapatriées sur une station au sol qui, elle, ferait les calculs de diagnostic et de pronostic.

En conclusion, l'implémentation des méthodes de PHM permettrait de réaliser un cycle d'exploitation vertueux au niveau des actionneurs électromécaniques. Ces derniers, de par leur capacité à déterminer leur propre état de santé en temps réel, permettraient de dimensionner au mieux leurs cycles de maintenances respectifs, ce qui occasionnerait un gain économique non négligeable pour les compagnies aériennes. Enfin, la prévention de défauts caractéristiques aux actionnements électriques autoriserait l'adoption accélérée de ces systèmes en vue du développement d'un avion plus respectueux de l'environnement. Le PHM est un domaine d'ingénierie pluridisciplinaire à la frontière de beaucoup de milieux et les défis à résoudre restent nombreux pour permettre son essor dans tous les systèmes aéronautiques. Seulement, nous espérons avec ces travaux avoir démontré que le jeu en vaut la chandelle.

Bibliographie

Abadi, Martín *et al.* (2015). *TensorFlow: Large-scale Machine Learning on Heterogeneous Systems*. URL : https://www.tensorflow.org/.

Abdeljaber, Osama *et al.* (3 fév. 2017). "Real-Time Vibration-Based Structural Damage Detection Using One-Dimensional Convolutional Neural Networks". In : *Journal of Sound and Vibration* 388, p. 154-170. ISSN : 0022-460X. DOI : 10.1016/j.jsv.2016.10.043. URL : https://www.sciencedirect.com/science/article/pii/S0022460X16306204 (visité le 26/10/2021).

Ackley, David H., Geoffrey E. Hinton et Terrence J. Sejnowski (1985). "A Learning Algorithm for Boltzmann Machines*". In : *Cognitive Science* 9.1, p. 147-169. ISSN : 1551-6709. DOI : 10.1207/s15516709cog0901_7. URL : https://onlinelibrary.wiley.com/doi/abs/10.1207/s15516709cog0901_7 (visité le 26/10/2021).

Aghabozorgi, Saeed, Ali Seyed Shirkhorshidi et Teh Ying Wah (1er oct. 2015). "Time-Series Clustering – A Decade Review". In : *Information Systems* 53, p. 16-38. ISSN : 0306-4379. DOI : 10.1016/j.is.2015.04.007. URL : https://www.sciencedirect.com/science/article/pii/S0306437915000733 (visité le 15/07/2021).

Agueh, Martial et Guillaume Carlier (1er jan. 2011). "Barycenters in the Wasserstein Space". In : *SIAM Journal on Mathematical Analysis* 43.2, p. 904-924. ISSN : 0036-1410. DOI : 10.1137/100805741. URL : https://epubs.siam.org/doi/10.1137/100805741 (visité le 13/09/2021).

AIRMES (2019). *The AIRMES Project Focuses on Optimising End-to-End Maintenance Activities within an Operator's Environment - Airmes Project*. airmes-project. URL : http://www.airmes-project.eu/airmes-project-1 (visité le 16/11/2021).

Akiba, Takuya *et al.* (25 juil. 2019). *Optuna: A Next-generation Hyperparameter Optimization Framework*. arXiv : 1907.10902 [cs, stat]. URL : http://arxiv.org/abs/1907.10902 (visité le 07/09/2021).

Ali, Mohammed *et al.* (1er juin 2019). "TimeCluster: Dimension Reduction Applied to Temporal Data for Visual Analytics". In : *The Visual Computer* 35.6, p. 1013-1026. ISSN : 1432-2315. DOI : 10.1007/s00371-019-01673-y. URL : https://doi.org/10.1007/s00371-019-01673-y (visité le 05/11/2021).

Arias Chao, Manuel *et al.* (jan. 2021). "Aircraft Engine Run-to-Failure Dataset under Real Flight Conditions for Prognostics and Diagnostics". In : *Data* 6.1 (1), p. 5. DOI : 10.3390/data6010005. URL : https://www.mdpi.com/2306-5729/6/1/5 (visité le 26/08/2021).

— (1er jan. 2022). "Fusing Physics-Based and Deep Learning Models for Prognostics". In : *Reliability Engineering & System Safety* 217, p. 107961. ISSN : 0951-8320. DOI : 10.1016/j.ress.2021.107961. URL : https://www.sciencedirect.com/science/article/pii/S0951832021004725 (visité le 25/11/2021).

Asadi, Reza et Amelia C. Regan (1er fév. 2020). "A Spatio-Temporal Decomposition Based Deep Neural Network for Time Series Forecasting". In : *Applied Soft Computing* 87, p. 105963. ISSN : 1568-4946. DOI : 10.1016/j.asoc.2019.105963. URL : https://www.sciencedirect.com/science/article/pii/S1568494619307446 (visité le 05/11/2021).

Atamuradov, Vepa *et al.* (11 déc. 2017). "Prognostics and Health Management for Maintenance Practitioners-Review, Implementation and Tools Evaluation". In : *International Journal of Prognostics and Health Management* 8.3, p. 31.

Baldi, Pierre et Kurt Hornik (1er jan. 1989). "Neural Networks and Principal Component Analysis: Learning from Examples without Local Minima". In : *Neural Networks* 2.1, p. 53-58. ISSN : 0893-6080. DOI : 10.1016/0893-6080(89)90014-2. URL : https://www.sciencedirect.com/science/article/pii/0893608089900142 (visité le 20/11/2021).

Bao, Yuequan *et al.* (1er mar. 2019). "Computer Vision and Deep Learning–Based Data Anomaly Detection Method for Structural Health Monitoring". In : *Structural Health Monitoring* 18.2,

p. 401-421. ISSN : 1475-9217. DOI : 10.1177/1475921718757405. URL : https://doi.org/10.1177/1475921718757405 (visité le 05/11/2021).

BASHTANNYK, David M. et Rob J. HYNDMAN (28 mai 2001). "Bandwidth Selection for Kernel Conditional Density Estimation". In : *Computational Statistics & Data Analysis* 36.3, p. 279-298. ISSN : 0167-9473. DOI : 10.1016/S0167-9473(00)00046-3. URL : https://www.sciencedirect.com/science/article/pii/S0167947300000463 (visité le 08/11/2021).

BATISTA, Gustavo E. A. P. A. *et al.* (1er mai 2014). "CID: An Efficient Complexity-Invariant Distance for Time Series". In : *Data Mining and Knowledge Discovery* 28.3, p. 634-669. ISSN : 1573-756X. DOI : 10.1007/s10618-013-0312-3. URL : https://doi.org/10.1007/s10618-013-0312-3 (visité le 14/10/2021).

BEKTAS, Oguz *et al.* (1er mar. 2019). "A Neural Network Filtering Approach for Similarity-Based Remaining Useful Life Estimation". In : *The International Journal of Advanced Manufacturing Technology* 101.1, p. 87-103. ISSN : 1433-3015. DOI : 10.1007/s00170-018-2874-0. URL : https://doi.org/10.1007/s00170-018-2874-0 (visité le 27/09/2021).

BENAMOU, Jean-David *et al.* (1er jan. 2015). "Iterative Bregman Projections for Regularized Transportation Problems". In : *SIAM Journal on Scientific Computing* 37.2, A1111-A1138. ISSN : 1064-8275. DOI : 10.1137/141000439. URL : https://epubs.siam.org/doi/10.1137/141000439 (visité le 24/08/2021).

BENGIO, Yoshua, Patrice SIMARD et Paolo FRASCONI (mar. 1994). "Learning Long-Term Dependencies with Gradient Descent Is Difficult". In : *IEEE Transactions on Neural Networks* 5.2 (2), p. 157-166. ISSN : 1045-9227. DOI : 10.1109/72.279181.

BERGSTRA, James *et al.* (2011). "Algorithms for Hyper-Parameter Optimization". In : *Advances in Neural Information Processing Systems*. T. 24. Curran Associates, Inc. URL : https://proceedings.neurips.cc/paper/2011/hash/86e8f7ab32cfd12577bc2619bc635690-Abstract.html (visité le 22/09/2021).

BOILEAU, Thierry (nov. 2010). "Permanent Magnet Synchronous Drives Service Continuity, a Contribution. Mechanical Sensor Loss, a Solution. Electrical Fault Detection". Theses. Institut National Polytechnique de Lorraine. URL : https://hal.univ-lorraine.fr/tel-01749443 (visité le 22/01/2019).

BONNEEL, Nicolas et David COEURJOLLY (12 juil. 2019). "SPOT: Sliced Partial Optimal Transport". In : *ACM Transactions on Graphics* 38.4 (4), 89:1-89:13. ISSN : 0730-0301. DOI : 10.1145/3306346.3323021. URL : https://doi.org/10.1145/3306346.3323021 (visité le 10/11/2020).

BOREL, Emile (1898). *Leçons sur la théorie des fonctions*. T. 1. De l'observatoire de Paris et du bureau des longitudes. Paris : Gauthier-Villars et fils, Imprimeurs-Libraires. 156 p.

BOTEV, Z. I., J. F. GROTOWSKI et D. P. KROESE (oct. 2010). "Kernel Density Estimation via Diffusion". In : *The Annals of Statistics* 38.5, p. 2916-2957. ISSN : 0090-5364, 2168-8966. DOI : 10.1214/10-AOS799. URL : https://projecteuclid.org/journals/annals-of-statistics/volume-38/issue-5/Kernel-density-estimation-via-diffusion/10.1214/10-AOS799.full (visité le 19/10/2021).

"Structures topologiques" (2007a). In : *Topologie générale: Chapitres 1 à 4*. Sous la dir. de N. BOURBAKI. Berlin, Heidelberg : Springer, p. 1-127. ISBN : 978-3-540-33982-3. DOI : 10.1007/978-3-540-33982-3_1. URL : https://doi.org/10.1007/978-3-540-33982-3_1 (visité le 04/12/2021).

BOURBAKI, N. (2007b). "Utilisation des nombres réels en topologie générale". In : *Topologie Générale: Chapitres 5 à 10*. Sous la dir. de N. BOURBAKI. Berlin, Heidelberg : Springer, p. 115-243. ISBN : 978-3-540-34486-5. DOI : 10.1007/978-3-540-34486-5_5. URL : https://doi.org/10.1007/978-3-540-34486-5_5 (visité le 04/12/2021).

BOURLARD, H. et Y. KAMP (1er sept. 1988). "Auto-Association by Multilayer Perceptrons and Singular Value Decomposition". In : *Biological Cybernetics* 59.4, p. 291-294. ISSN : 1432-0770. DOI : 10.1007/BF00332918. URL : https://doi.org/10.1007/BF00332918 (visité le 20/11/2021).

BRAEI, Mohammad et Sebastian WAGNER (1er avr. 2020). *Anomaly Detection in Univariate Time-series: A Survey on the State-of-the-Art*. arXiv : 2004.00433 [cs, stat]. URL : http://arxiv.org/abs/2004.00433 (visité le 05/11/2021).

BREGMAN, L. M. (1er jan. 1967). "The Relaxation Method of Finding the Common Point of Convex Sets and Its Application to the Solution of Problems in Convex Programming". In : *USSR Computational Mathematics and Mathematical Physics* 7.3, p. 200-217. ISSN : 0041-5553. DOI : 10.1016/0041-5553(67)90040-7. URL : https://www.sciencedirect.com/science/article/pii/0041555367900407 (visité le 20/10/2021).

Breuneval, Romain (21 déc. 2017). "Surveillance de l'état de santé des actionneurs électromécaniques : application à l'aéronautique". Thèse de doctorat. Lyon : Université de Lyon. 262 p.

Brickell, Justin *et al.* (1er jan. 2008). "The Metric Nearness Problem". In : *SIAM Journal on Matrix Analysis and Applications* 30.1, p. 375-396. ISSN : 0895-4798. DOI : 10.1137/060653391. URL : https://epubs.siam.org/doi/abs/10.1137/060653391 (visité le 15/09/2021).

Bridle, John (1989). "Training Stochastic Model Recognition Algorithms as Networks Can Lead to Maximum Mutual Information Estimation of Parameters". In : *Advances in Neural Information Processing Systems*. T. 2. Morgan-Kaufmann. URL : https://proceedings.neurips.cc/paper/1989/hash/0336dcbab05b9d5ad24f4333c7658a0e-Abstract.html (visité le 18/11/2021).

Brualdi, Richard A. (2006). *Combinatorial Matrix Classes*. Encyclopedia of Mathematics and Its Applications. Cambridge : Cambridge University Press. ISBN : 978-0-521-86565-4. DOI : 10.1017/CBO9780511721182. URL : https://www.cambridge.org/core/books/combinatorial-matrix-classes/15AE14E4AF42BDDCC66F5801CB234209 (visité le 16/09/2021).

Cai, Haoshu, Jianshe Feng *et al.* (1er sept. 2020). "Similarity-Based Particle Filter for Remaining Useful Life Prediction with Enhanced Performance". In : *Applied Soft Computing* 94, p. 106474. ISSN : 1568-4946. DOI : 10.1016/j.asoc.2020.106474. URL : https://www.sciencedirect.com/science/article/pii/S1568494620304130 (visité le 27/09/2021).

Cai, Haoshu, Xiaodong Jia *et al.* (1er août 2020). "A Similarity Based Methodology for Machine Prognostics by Using Kernel Two Sample Test". In : *ISA Transactions* 103, p. 112-121. ISSN : 0019-0578. DOI : 10.1016/j.isatra.2020.03.007. URL : https://www.sciencedirect.com/science/article/pii/S0019057820301154 (visité le 27/09/2021).

Caliński, T. et J. Harabasz (1er jan. 1974). "A Dendrite Method for Cluster Analysis". In : *Communications in Statistics* 3.1, p. 1-27. ISSN : 0090-3272. DOI : 10.1080/03610927408827101. URL : https://www.tandfonline.com/doi/abs/10.1080/03610927408827101 (visité le 16/07/2021).

Cannarile, Francesco, Piero Baraldi et Enrico Zio (15 juil. 2019). "An Evidential Similarity-Based Regression Method for the Prediction of Equipment Remaining Useful Life in Presence of Incomplete Degradation Trajectories". In : *Fuzzy Sets and Systems*. Theme: Uncertainty Management 367, p. 36-50. ISSN : 0165-0114. DOI : 10.1016/j.fss.2018.10.008. URL : https://www.sciencedirect.com/science/article/pii/S0165011418304068 (visité le 27/09/2021).

Cauchy, Augustin (juil. 1847). *Méthode générale pour la résolution d'équations simultanées*. Gallica. URL : https://gallica.bnf.fr/ark:/12148/bpt6k2982c (visité le 26/10/2021).

Cazelles, Elsa (21 sept. 2018). "Propriétés statistiques du barycentre dans l'espace de Wasserstein". Thèse de doct. Université de Bordeaux. URL : https://tel.archives-ouvertes.fr/tel-01928219 (visité le 07/09/2021).

Chalapathy, Raghavendra et Sanjay Chawla (23 jan. 2019). *Deep Learning for Anomaly Detection: A Survey*. arXiv : 1901.03407 [cs, stat]. URL : http://arxiv.org/abs/1901.03407 (visité le 05/11/2021).

Chen, J. *et al.* (2020). "Health Monitoring of Landing Gear Retraction/Extension System Based on Optimized Fuzzy C-Means Algorithm". In : *IEEE Access* 8, p. 219611-219621. ISSN : 2169-3536. DOI : 10.1109/ACCESS.2020.3042888.

Chen, Zhongzhe, Shuchen Cao et Zijian Mao (jan. 2018). "Remaining Useful Life Estimation of Aircraft Engines Using a Modified Similarity and Supporting Vector Machine (SVM) Approach". In : *Energies* 11.1 (1), p. 28. DOI : 10.3390/en11010028. URL : https://www.mdpi.com/1996-1073/11/1/28 (visité le 27/09/2021).

Cho, Kyunghyun *et al.* (2 sept. 2014). *Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation*. arXiv : 1406.1078 [cs, stat]. URL : http://arxiv.org/abs/1406.1078 (visité le 22/11/2021).

Chollet, François (25 nov. 2019). *On the Measure of Intelligence*. Version 2. arXiv : 1911.01547 [cs]. URL : http://arxiv.org/abs/1911.01547 (visité le 20/11/2021).

— (oct. 2021). *Deep Learning with Python - Second Edition*. Manning. Manning Shelter Island. 478 p. ISBN : 978-1-61729-686-4.

CleanSky (mai 2021). *Technology Evaluator First Global Assessment 2020 Technical Report*. Union Européenne, p. 124. URL : https://www.cleansky.eu/publication/technology-evaluator-report-synopsis (visité le 15/11/2021).

Colah, Christopher (3 sept. 2015). *Neural Networks, Types, and Functional Programming – Colah's Blog*. Blog. URL : http://colah.github.io/posts/2015-09-NN-Types-FP/ (visité le 28/02/2019).

Cominetti, R. et J. San Martín (1er oct. 1994). "Asymptotic Analysis of the Exponential Penalty Trajectory in Linear Programming". In : *Mathematical Programming* 67.1, p. 169-187. ISSN : 1436-4646. DOI : 10.1007/BF01582220. URL : https://doi.org/10.1007/BF01582220 (visité le 16/09/2021).

Cooper, Tom *et al.* (28 jan. 2021). *Global Fleet and MRO Market Forecast 2021-2031*. Oliver Wyman, p. 58. URL : https://www.oliverwyman.com/our-expertise/insights/2021/jan/global-fleet-and-mro-market-forecast-2021-2031.html (visité le 16/11/2021).

Cornuéjols, Antoine et Laurent Miclet (juin 2010). *Apprentissage artificiel : Concepts et algorithmes*. 2ème édition. Eyrolles. ISBN : 978-2-212-12471-2. URL : https://hal.inria.fr/inria-00538947 (visité le 20/12/2018).

Coxeter, H. S. M. (1947). *Regular Polytopes*. Toronto : Methuen & Co. LTd. London. 346 p. ISBN : 978-0-486-61480-9.

Csiszár, I. (1er mar. 1972). "A Class of Measures of Informativity of Observation Channels". In : *Periodica Mathematica Hungarica* 2.1, p. 191-213. ISSN : 1588-2829. DOI : 10.1007/BF02018661. URL : https://doi.org/10.1007/BF02018661 (visité le 09/10/2021).

Cuturi, Marco (2013). "Sinkhorn Distances: Lightspeed Computation of Optimal Transport". In : *Advances in Neural Information Processing Systems*. T. 26. Curran Associates, Inc. URL : https://papers.nips.cc/paper/2013/hash/af21d0c97db2e27e13572cbf59eb343d-Abstract.html (visité le 15/09/2021).

Cybenko, G. (1er déc. 1989). "Approximation by Superpositions of a Sigmoidal Function". In : *Mathematics of Control, Signals and Systems* 2.4, p. 303-314. ISSN : 1435-568X. DOI : 10.1007/BF02551274. URL : https://doi.org/10.1007/BF02551274 (visité le 17/11/2021).

Davies, David L. et Donald W. Bouldin (avr. 1979). "A Cluster Separation Measure". In : *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-1.2, p. 224-227. ISSN : 1939-3539. DOI : 10.1109/TPAMI.1979.4766909.

Dinis, Duarte, Ana Barbosa-Póvoa et Ângelo Palos Teixeira (1er déc. 2019a). "A Supporting Framework for Maintenance Capacity Planning and Scheduling: Development and Application in the Aircraft MRO Industry". In : *International Journal of Production Economics* 218, p. 1-15. ISSN : 0925-5273. DOI : 10.1016/j.ijpe.2019.04.029. URL : https://www.sciencedirect.com/science/article/pii/S0925527319301586 (visité le 11/11/2021).

— (1er fév. 2019b). "Valuing Data in Aircraft Maintenance through Big Data Analytics: A Probabilistic Approach for Capacity Planning Using Bayesian Networks". In : *Computers & Industrial Engineering* 128, p. 920-936. ISSN : 0360-8352. DOI : 10.1016/j.cie.2018.10.015. URL : https://www.sciencedirect.com/science/article/pii/S0360835218304856 (visité le 11/11/2021).

Dozat, Timothy (2016). "Incorporating Nesterov Momentum into Adam". In : *ICLR Workshop - 2016*. ICLR. Puerto Rico.

Dua, Dheeru et Casey Graff (2017). *UCI Machine Learning Repository*. URL : http://archive.ics.uci.edu/ml.

Duchi, John, Elad Hazan et Yoram Singer (2011). "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization". In : *Journal of Machine Learning Research* 12.61, p. 2121-2159. ISSN : 1533-7928. URL : http://jmlr.org/papers/v12/duchi11a.html (visité le 26/10/2021).

Dumoulin, Vincent et Francesco Visin (mar. 2016). "A Guide to Convolution Arithmetic for Deep Learning". In : *ArXiv e-prints*. eprint : 1603.07285.

Eid, Alexandre *et al.* (jan. 2021). "A Novel Deep Clustering Method and Indicator for Time Series Soft Partitioning". In : *Energies* 14.17 (17), p. 5530. DOI : 10.3390/en14175530. URL : https://www.mdpi.com/1996-1073/14/17/5530 (visité le 03/11/2021).

El Jamal, Dima *et al.* (juin 2021). "Combining Approaches of Brownian Motion and Similarity Principle to Improve the Remaining Useful Life Prediction". In : *2021 IEEE International Conference on Prognostics and Health Management (ICPHM)*. 2021 IEEE International Conference on Prognostics and Health Management (ICPHM), p. 1-7. DOI : 10.1109/ICPHM51084.2021.9486507.

Erlander, Sven et Neil F. Stewart (1990). "The Gravity Model in Transportation Analysis - Theory and Extensions". In :

European Aviation Safety Agency. et EAA. (2019). *European Aviation Environmental: Report 2019*. LU : Publications Office. URL : https://data.europa.eu/doi/10.2822/309946 (visité le 14/11/2021).

Fahlman, Scott E., Geoffrey E. Hinton et Terrence J. Sejnowski (22 août 1983). "Massively Parallel Architectures for AI: Netl, Thistle, and Boltzmann Machines". In : *Proceedings of the Third AAAI Conference on Artificial Intelligence*. AAAI'83. Washington, D.C. : AAAI Press, p. 109-113.

Favin-Lévêque, Martin *et al.* (2019). *Apport de la cyclostationnarité pour l'analyse des défauts de roulement*. Projet d'entreprise Polytechnique. Saclay, p. 42.

Fei, Xiao *et al.* (août 2020). "Literature Review: Framework of Prognostic Health Management for Airline Predictive Maintenance". In : *2020 Chinese Control And Decision Conference (CCDC)*. 2020 Chinese Control And Decision Conference (CCDC), p. 5112-5117. DOI : 10.1109/CCDC49329.2020.9164546.

Fink, Olga *et al.* (1er juin 2020). "Potential, Challenges and Future Directions for Deep Learning in Prognostics and Health Management Applications". In : *Engineering Applications of Artificial Intelligence* 92, p. 103678. ISSN : 0952-1976. DOI : 10.1016/j.engappai.2020.103678. URL : http://www.sciencedirect.com/science/article/pii/S0952197620301184 (visité le 25/01/2021).

Flamary, Rémi *et al.* (2021). "POT: Python Optimal Transport". In : *Journal of Machine Learning Research* 22.78, p. 1-8. ISSN : 1533-7928. URL : http://jmlr.org/papers/v22/20-451.html (visité le 14/09/2021).

Forest, Florent *et al.* (jan. 2021). "An Invariance-guided Stability Criterion for Time Series Clustering Validation". In : *2020 25th International Conference on Pattern Recognition (ICPR)*. 2020 25th International Conference on Pattern Recognition (ICPR), p. 9296-9303. DOI : 10.1109/ICPR48806.2021.9412020.

Fréchet, Maurice (1948). "Les éléments aléatoires de nature quelconque dans un espace distancié". In : *Annales de l'institut Henri Poincaré* 10.4, p. 215-310. URL : http://www.numdam.org/item/AIHP_1948__10_4_215_0/ (visité le 11/10/2021).

Frey, Brendan J. et Delbert Dueck (16 fév. 2007). "Clustering by Passing Messages Between Data Points". In : *Science* 315.5814, p. 972-976. DOI : 10.1126/science.1136800. URL : https://www.science.org/doi/abs/10.1126/science.1136800 (visité le 10/11/2021).

Fu, K. S. et J. K. Mui (1er jan. 1981). "A Survey on Image Segmentation". In : *Pattern Recognition* 13.1, p. 3-16. ISSN : 0031-3203. DOI : 10.1016/0031-3203(81)90028-5. URL : https://www.sciencedirect.com/science/article/pii/0031320381900285 (visité le 05/11/2021).

Fukushima, Kunihiko (1er sept. 1975). "Cognitron: A Self-Organizing Multilayered Neural Network". In : *Biological Cybernetics* 20.3, p. 121-136. ISSN : 1432-0770. DOI : 10.1007/BF00342633. URL : https://doi.org/10.1007/BF00342633 (visité le 19/11/2021).

— (1er avr. 1980). "Neocognitron: A Self-Organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position". In : *Biological Cybernetics* 36.4, p. 193-202. ISSN : 1432-0770. DOI : 10.1007/BF00344251. URL : https://doi.org/10.1007/BF00344251 (visité le 19/11/2021).

Funahashi, Ken-Ichi (1er jan. 1989). "On the Approximate Realization of Continuous Mappings by Neural Networks". In : *Neural Networks* 2.3, p. 183-192. ISSN : 0893-6080. DOI : 10.1016/0893-6080(89)90003-8. URL : https://www.sciencedirect.com/science/article/pii/0893608089900038 (visité le 18/11/2021).

Garnier, Josselin et Sylvie Méléard (sept. 2017). *Aléatoire*. Ecole Polytechnique. Département de Mathématiques Appliquées. Saclay.

Gath, I. et A.B. Geva (juil. 1989). "Unsupervised Optimal Fuzzy Clustering". In : *IEEE Transactions on Pattern Analysis and Machine Intelligence* 11.7, p. 773-780. ISSN : 01628828. DOI : 10.1109/34.192473. URL : http://ieeexplore.ieee.org/document/192473/ (visité le 10/11/2021).

Geffrier, François (30 sept. 2021). *Le focus Eco*. Avec la coll. de Marc Durance. URL : https://www.radioclassique.fr/radio/emissions/la-matinale-economique/le-focus-eco/.

Geman, Stuart et Chii-Ruey Hwang (1982). "Nonparametric Maximum Likelihood Estimation by the Method of Sieves". In : *The Annals of Statistics* 10.2, p. 401-414. ISSN : 0090-5364. JSTOR : 2240675.

Glorot, Xavier, Antoine Bordes et Yoshua Bengio (14 juin 2011). "Deep Sparse Rectifier Neural Networks". In : *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. Proceedings of the Fourteenth International Conference on Artificial

Intelligence and Statistics. JMLR Workshop and Conference Proceedings, p. 315-323. URL : https://proceedings.mlr.press/v15/glorot11a.html (visité le 22/11/2021).

GOMEZ, Antonio *et al.* (2017). *Nacelles Du COMAC C919 | Safran*. www.safrangroup.com. URL : https://www.safran-group.com/fr/produits-services/nacelles-du-comac-c919 (visité le 13/11/2021).

GRAVER, Brandon, Kevin ZHANG et Dan RUTHERFORD (19 sept. 2019). *CO2 Emissions from Commercial Aviation, 2018*. Working Paper 2019-16. The International Council of Clean Transportation, p. 13. URL : https://theicct.org/publications/co2-emissions-commercial-aviation-2018 (visité le 27/10/2021).

GRETTON, Arthur *et al.* (mar. 2012). "A Kernel Two-Sample Test". In : *Journal of Machine Learning Research* 13, 723-773. ISSN : 1533-7928. URL : http://jmlr.csail.mit.edu/papers/v13/gretton12a.html (visité le 17/03/2020).

GU, Mengyao et Youling CHEN (15 juil. 2016). "Two Improvements of Similarity-Based Residual Life Prediction Methods". In : *Journal of Intelligent Manufacturing* 30.1, p. 303-315. ISSN : 1572-8145. DOI : 10.1007/s10845-016-1249-3. URL : https://doi.org/10.1007/s10845-016-1249-3 (visité le 27/09/2021).

GUO, Yifan *et al.* (4 nov. 2018). "Multidimensional Time Series Anomaly Detection: A GRU-based Gaussian Mixture Variational Autoencoder Approach". In : *Proceedings of The 10th Asian Conference on Machine Learning*. Asian Conference on Machine Learning. PMLR, p. 97-112. URL : https://proceedings.mlr.press/v95/guo18a.html (visité le 05/11/2021).

GÜTSCHOW, Johannes, Annika GÜNTHER et Mika PFLÜGER (22 sept. 2021). *The PRIMAP-hist National Historical Emissions Time Series (1750-2019) v2.3.1*. Version 2.3.1. Zenodo. DOI : 10.5281/ZENODO.5494497. URL : https://zenodo.org/record/5494497 (visité le 27/10/2021).

GÜTSCHOW, Johannes, M. Louise JEFFERY *et al.* (9 nov. 2016). "The PRIMAP-hist National Historical Emissions Time Series". In : *Earth System Science Data* 8.2, p. 571-603. ISSN : 1866-3516. DOI : 10.5194/essd-8-571-2016. URL : https://essd.copernicus.org/articles/8/571/2016/ (visité le 27/10/2021).

HAHNLOSER, Richard H. R. *et al.* (juin 2000). "Digital Selection and Analogue Amplification Coexist in a Cortex-Inspired Silicon Circuit". In : *Nature* 405.6789 (6789), p. 947-951. ISSN : 1476-4687. DOI : 10.1038/35016072. URL : https://www.nature.com/articles/35016072 (visité le 22/11/2021).

HALKIDI, M. et M. VAZIRGIANNIS (nov. 2001). "Clustering Validity Assessment: Finding the Optimal Partitioning of a Data Set". In : *Proceedings 2001 IEEE International Conference on Data Mining*. Proceedings 2001 IEEE International Conference on Data Mining, p. 187-194. DOI : 10.1109/ICDM.2001.989517.

HALKIDI, M., M. VAZIRGIANNIS et Yannis BATISTAKIS (2000). "Quality Scheme Assessment in the Clustering Process". In : *PKDD*. DOI : 10.1007/3-540-45372-5_26.

HAMADACHE, Moussa *et al.* (1er juin 2019). "A Comprehensive Review of Artificial Intelligence-Based Approaches for Rolling Element Bearing PHM: Shallow and Deep Learning". In : *JMST Advances* 1.1, p. 125-151. ISSN : 2524-7913. DOI : 10.1007/s42791-019-0016-y. URL : https://doi.org/10.1007/s42791-019-0016-y (visité le 24/11/2021).

HAN, Te *et al.* (15 fév. 2019). "An Adaptive Spatiotemporal Feature Learning Approach for Fault Diagnosis in Complex Systems". In : *Mechanical Systems and Signal Processing* 117, p. 170-187. ISSN : 0888-3270. DOI : 10.1016/j.ymssp.2018.07.048. URL : https://www.sciencedirect.com/science/article/pii/S0888327018304503 (visité le 26/10/2021).

HAO, Yifan, Huiping CAO et Erick DRAAYER (oct. 2020). "CNN Approaches to Classify Multivariate Time Series Using Class-specific Features". In : *2020 IEEE International Conference on Smart Data Services (SMDS)*. 2020 IEEE International Conference on Smart Data Services (SMDS), p. 1-8. DOI : 10.1109/SMDS49396.2020.00008.

HATAMI, Nima, Yann GAVET et Johan DEBAYLE (avr. 2018). "Classification of Time-Series Images Using Deep Convolutional Neural Networks". In : *Proceedings of SPIE, the International Society for Optical Engineering* 10696, UNSP 106960Y. DOI : 10.1117/12.2309486. URL : https://hal.archives-ouvertes.fr/hal-01869027 (visité le 31/01/2021).

HE, Kaiming *et al.* (10 déc. 2015). *Deep Residual Learning for Image Recognition*. arXiv : 1512.03385 [cs]. URL : http://arxiv.org/abs/1512.03385 (visité le 05/11/2021).

HENDRICKX, Kilian *et al.* (1er mai 2020). "A General Anomaly Detection Framework for Fleet-Based Condition Monitoring of Machines". In : *Mechanical Systems and Signal Processing* 139, p. 106585. ISSN : 0888-3270. DOI : 10.1016/j.ymssp.2019.106585. URL : http://www.sciencedirect.com/science/article/pii/S0888327019308064 (visité le 31/01/2021).

HINTON, Geoffrey E et Richard ZEMEL (1994). "Autoencoders, Minimum Description Length and Helmholtz Free Energy". In : *Advances in Neural Information Processing Systems*. Sous la dir. de J. COWAN, G. TESAURO et J. ALSPECTOR. T. 6. Morgan-Kaufmann. URL : https://proceedings.neurips.cc/paper/1993/file/9e3cfc48eccf81a0d57663e129aef3cb-Paper.pdf.

HINTON, Geoffrey E. (1er août 2002). "Training Products of Experts by Minimizing Contrastive Divergence". In : *Neural Computation* 14.8, p. 1771-1800. ISSN : 0899-7667. DOI : 10.1162/089976602760128018. URL : https://doi.org/10.1162/089976602760128018 (visité le 22/11/2021).

— (1er oct. 2007). "Learning Multiple Layers of Representation". In : *Trends in Cognitive Sciences* 11.10, p. 428-434. ISSN : 1364-6613, 1879-307X. DOI : 10.1016/j.tics.2007.09.004. pmid : 17921042. URL : https://www.cell.com/trends/cognitive-sciences/abstract/S1364-6613(07)00217-3 (visité le 26/10/2021).

HINTON, Geoffrey E., Alexander KRIZHEVSKY *et al.* (2 août 2016). "System and Method for Addressing Overfitting in a Neural Network". Brev. amér. 9406017B2. GOOGLE LLC. URL : https://patents.google.com/patent/US9406017B2/en/endropout (visité le 22/11/2021).

HINTON, Geoffrey E. et J. SEJNOWSKI (juin 1983). "Optimal Perceptuel Inference". In : *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE CVPR. Washington, D.C. : IEEE, p. 448-453. URL : https://www.semanticscholar.org/paper/OPTIMAL-PERCEPTUAL-INFERENCE-Hinton-Sejnowski/1718965f492d4e9fe1d98a3fb83efe671a4aed2c (visité le 22/11/2021).

HINTON, Geoffrey E., Nitish SRIVASTAVA *et al.* (3 juil. 2012). *Improving Neural Networks by Preventing Co-Adaptation of Feature Detectors*. arXiv : 1207.0580 [cs]. URL : http://arxiv.org/abs/1207.0580 (visité le 22/11/2021).

HOANG, Duy-Tang et Hee-Jun KANG (28 mar. 2019). "A Survey on Deep Learning Based Bearing Fault Diagnosis". In : *Neurocomputing* 335, p. 327-335. ISSN : 0925-2312. DOI : 10.1016/j.neucom.2018.06.078. URL : https://www.sciencedirect.com/science/article/pii/S0925231218312657 (visité le 26/10/2021).

HOCHREITER, Sepp et Jürgen SCHMIDHUBER (15 nov. 1997). "Long Short-Term Memory". In : *Neural Computation* 9.8, p. 1735-1780. ISSN : 0899-7667. DOI : 10.1162/neco.1997.9.8.1735. URL : https://doi.org/10.1162/neco.1997.9.8.1735 (visité le 26/10/2021).

HOPFIELD, J. J. (1er avr. 1982). "Neural Networks and Physical Systems with Emergent Collective Computational Abilities". In : *Proceedings of the National Academy of Sciences* 79.8, p. 2554-2558. ISSN : 0027-8424, 1091-6490. DOI : 10.1073/pnas.79.8.2554. pmid : 6953413. URL : https://www.pnas.org/content/79/8/2554 (visité le 22/11/2021).

HORNIK, Kurt, Maxwell STINCHCOMBE et Halbert WHITE (1er jan. 1989). "Multilayer Feedforward Networks Are Universal Approximators". In : *Neural Networks* 2.5, p. 359-366. ISSN : 0893-6080. DOI : 10.1016/0893-6080(89)90020-8. URL : https://www.sciencedirect.com/science/article/pii/0893608089900208 (visité le 17/11/2021).

HOU, Mengru, Dechang PI et Bingrong LI (15 juil. 2020). "Similarity-Based Deep Learning Approach for Remaining Useful Life Prediction". In : *Measurement* 159, p. 107788. ISSN : 0263-2241. DOI : 10.1016/j.measurement.2020.107788. URL : https://www.sciencedirect.com/science/article/pii/S0263224120303262 (visité le 27/09/2021).

HU, Yang *et al.* (1er jan. 2022). "Prognostics and Health Management: A Review from the Perspectives of Design, Development and Decision". In : *Reliability Engineering & System Safety* 217, p. 108063. ISSN : 0951-8320. DOI : 10.1016/j.ress.2021.108063. URL : https://www.sciencedirect.com/science/article/pii/S0951832021005652 (visité le 25/11/2021).

HUANG, Cheng-Geng *et al.* (1er oct. 2019). "Improved Trajectory Similarity-Based Approach for Turbofan Engine Prognostics". In : *Journal of Mechanical Science and Technology* 33.10, p. 4877-4890. ISSN : 1976-3824. DOI : 10.1007/s12206-019-0928-3. URL : https://doi.org/10.1007/s12206-019-0928-3 (visité le 27/09/2021).

HUBEL, D. H. et T. N. WIESEL (jan. 1962). "Receptive Fields, Binocular Interaction and Functional Architecture in the Cat's Visual Cortex". In : *The Journal of Physiology* 160.1, p. 106-154.2. ISSN : 0022-3751. pmid : 14449617. URL : https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1359523/ (visité le 19/11/2021).

IEA, Paris (2021). *Transport Sector CO2 Emissions by Mode in the Sustainable Development Scenario, 2000-2030*. www.iea.org. URL : https://www.iea.org/data-and-statistics/charts/transport-sector-co2-emissions-by-mode-in-the-sustainable-development-scenario-2000-2030.

IQBAL, Haris (22 sept. 2020). *HarisIqbal88/PlotNeuralNet*. URL : https://github.com/HarisIqbal88/PlotNeuralNet (visité le 22/09/2020).

ISMAIL FAWAZ, Hassan, Germain FORESTIER *et al.* (1er juil. 2019). "Deep Learning for Time Series Classification: A Review". In : *Data Mining and Knowledge Discovery* 33.4, p. 917-963. ISSN : 1573-756X. DOI : 10.1007/s10618-019-00619-1. URL : https://doi.org/10.1007/s10618-019-00619-1 (visité le 05/11/2021).

ISMAIL FAWAZ, Hassan, Benjamin LUCAS *et al.* (1er nov. 2020). "InceptionTime: Finding AlexNet for Time Series Classification". In : *Data Mining and Knowledge Discovery* 34.6 (6), p. 1936-1962. ISSN : 1573-756X. DOI : 10.1007/s10618-020-00710-y. URL : https://doi.org/10.1007/s10618-020-00710-y (visité le 15/10/2020).

JAIN, A. K., M. N. MURTY et P. J. FLYNN (1er sept. 1999). "Data Clustering: A Review". In : *ACM Computing Surveys* 31.3, p. 264-323. ISSN : 0360-0300. DOI : 10.1145/331499.331504. URL : https://doi.org/10.1145/331499.331504 (visité le 15/07/2021).

JANSSENS, Olivier *et al.* (1er sept. 2016). "Convolutional Neural Network Based Fault Detection for Rotating Machinery". In : *Journal of Sound and Vibration* 377, p. 331-345. ISSN : 0022-460X. DOI : 10.1016/j.jsv.2016.05.027. URL : https://www.sciencedirect.com/science/article/pii/S0022460X16301638 (visité le 26/10/2021).

JAPKOWICZ, Nathalie, Stephen JOSE HANSON et Mark A. GLUCK (1er mar. 2000). "Nonlinear Autoassociation Is Not Equivalent to PCA". In : *Neural Computation* 12.3, p. 531-545. ISSN : 0899-7667. DOI : 10.1162/089976600300015691. URL : https://doi.org/10.1162/089976600300015691 (visité le 20/11/2021).

JARRETT, Kevin *et al.* (sept. 2009). "What Is the Best Multi-Stage Architecture for Object Recognition?" In : *2009 IEEE 12th International Conference on Computer Vision*. 2009 IEEE 12th International Conference on Computer Vision, p. 2146-2153. DOI : 10.1109/ICCV.2009.5459469.

JAVED, Ali, Byung Suk LEE et Donna M. RIZZO (15 sept. 2020). "A Benchmark Study on Time Series Clustering". In : *Machine Learning with Applications* 1, p. 100001. ISSN : 2666-8270. DOI : 10.1016/j.mlwa.2020.100001. URL : https://www.sciencedirect.com/science/article/pii/S2666827020300013 (visité le 19/07/2021).

JIA, Xiaodong, Haoshu CAI *et al.* (22 sept. 2019). "A Novel Similarity-based Method for Remaining Useful Life Prediction Using Kernel Two Sample Test". In : *Annual Conference of the PHM Society* 11.1 (1). ISSN : 2325-0178. DOI : 10.36001/phmconf.2019.v11i1.788. URL : https://papers.phmsociety.org/index.php/phmconf/article/view/788 (visité le 27/09/2021).

JIA, Xiaodong, Bin HUANG *et al.* (24 sept. 2018). "A Review of PHM Data Competitions from 2008 to 2017: Methodologies and Analytics". In : *Annual Conference of the PHM Society* 10.1 (1). ISSN : 2325-0178. DOI : 10.36001/phmconf.2018.v10i1.462. URL : https://papers.phmsociety.org/index.php/phmconf/article/view/462 (visité le 27/09/2021).

JIAO, Ruihua *et al.* (juil. 2019). "A Novel Scheme for Remaining Useful Life Prediction and Safety Assessment Based on Hybrid Method". In : *2019 CAA Symposium on Fault Detection, Supervision and Safety for Technical Processes (SAFEPROCESS)*. 2019 CAA Symposium on Fault Detection, Supervision and Safety for Technical Processes (SAFEPROCESS), p. 395-400. DOI : 10.1109/SAFEPROCESS45799.2019.9213446.

KANTOROVICH, L. V. (1er juil. 1960). "Mathematical Methods of Organizing and Planning Production". In : *Management Science* 6.4, p. 366-422. ISSN : 0025-1909. DOI : 10.1287/mnsc.6.4.366. URL : https://pubsonline.informs.org/doi/abs/10.1287/mnsc.6.4.366 (visité le 12/10/2021).

KEFALAS, Marios *et al.* (9 juin 2021). "Automated Machine Learning for Remaining Useful Life Estimation of Aircraft Engines". In : 2021 IEEE International Conference on Prognostics and Health Management (ICPHM). Détroit (digital).

KHAN, Samir et Takehisa YAIRI (1er juil. 2018). "A Review on the Application of Deep Learning in System Health Management". In : *Mechanical Systems and Signal Processing* 107, p. 241-265. ISSN : 0888-3270. DOI : 10.1016/j.ymssp.2017.11.024. URL : https://www.sciencedirect.com/science/article/pii/S0888327017306064 (visité le 26/10/2021).

KIEFER, J. et J. WOLFOWITZ (sept. 1952). "Stochastic Estimation of the Maximum of a Regression Function". In : *The Annals of Mathematical Statistics* 23.3, p. 462-466. ISSN : 0003-4851. DOI : 10.1214/aoms/1177729392. URL : http://projecteuclid.org/euclid.aoms/1177729392 (visité le 26/10/2021).

KIM, Yoon (2 sept. 2014). *Convolutional Neural Networks for Sentence Classification*. arXiv : 1408.5882 [cs]. URL : http://arxiv.org/abs/1408.5882 (visité le 26/10/2021).

KINGMA, Diederik P. et Jimmy BA (29 jan. 2017). "Adam: A Method for Stochastic Optimization". In : *3rd International Conference for Learning Representations, San Diego, 2015*. arXiv : 1412.6980. URL : http://arxiv.org/abs/1412.6980 (visité le 26/10/2021).

KOLMOGOROV, Andreï (1957). "On the reprensentation of continuous functions of many variables by superposition of continuous functions of one variable and addition". In : Dokl. Akad. Nauk SSSR 114, p. 953-956. URL : http://mi.mathnet.ru/dan22050 (visité le 17/11/2021).

KOORNNEEF, H., W. J. C. VERHAGEN et R. CURRAN (1er août 2020). "A Decision Support Framework and Prototype for Aircraft Dispatch Assessment". In : *Decision Support Systems* 135, p. 113338. ISSN : 0167-9236. DOI : 10.1016/j.dss.2020.113338. URL : https://www.sciencedirect.com/science/article/pii/S0167923620300932 (visité le 11/11/2021).

KRUPP, Amelie *et al.* (19 nov. 2021). "Calendar Aging Model for Lithium-Ion Batteries Considering the Influence of Cell Characterization". In : *Journal of Energy Storage*, p. 103506. ISSN : 2352-152X. DOI : 10.1016/j.est.2021.103506. URL : https://www.sciencedirect.com/science/article/pii/S2352152X21011889 (visité le 06/12/2021).

KUHN, Max et Kjell JOHNSON (2013). "Over-Fitting and Model Tuning". In : *Applied Predictive Modeling*. Sous la dir. de Max KUHN et Kjell JOHNSON. New York, NY : Springer, p. 61-92. ISBN : 978-1-4614-6849-3. DOI : 10.1007/978-1-4614-6849-3_4. URL : https://doi.org/10.1007/978-1-4614-6849-3_4 (visité le 06/11/2021).

KULLBACK, S. et R. A. LEIBLER (mar. 1951). "On Information and Sufficiency". In : *The Annals of Mathematical Statistics* 22.1, p. 79-86. ISSN : 0003-4851, 2168-8990. DOI : 10.1214/aoms/1177729694. URL : https://projecteuclid.org/journals/annals-of-mathematical-statistics/volume-22/issue-1/On-Information-and-Sufficiency/10.1214/aoms/1177729694.full (visité le 09/10/2021).

LEBOEUF, Nicolas (déc. 2012). "Contribution to the Study of Permanent Magnet Synchronous Machines Under Inter-turn Faults Conditions: Modelling, Inter-turn Fault Detection". Theses. Université de Lorraine. URL : https://hal.univ-lorraine.fr/tel-01750066 (visité le 28/01/2019).

LECUN, Y. (1989). *Generalization and Network Design Strategies*. CRG-TR-89-4. Department of Computer Science, University of Toronto.

LECUN, Y. *et al.* (déc. 1989). "Backpropagation Applied to Handwritten Zip Code Recognition". In : *Neural Computation* 1.4, p. 541-551. ISSN : 0899-7667. DOI : 10.1162/neco.1989.1.4.541.

LECUN, Yann, Yoshua BENGIO et Geoffrey HINTON (mai 2015). "Deep Learning". In : *Nature* 521.7553 (7553), p. 436-444. ISSN : 1476-4687. DOI : 10.1038/nature14539. URL : https://www.nature.com/articles/nature14539 (visité le 18/12/2018).

LEI, Yaguo (1er jan. 2017). "2 - Signal Processing and Feature Extraction". In : *Intelligent Fault Diagnosis and Remaining Useful Life Prediction of Rotating Machinery*. Sous la dir. d'Yaguo LEI. Butterworth-Heinemann, p. 17-66. ISBN : 978-0-12-811534-3. DOI : 10.1016/B978-0-12-811534-3.00002-0. URL : http://www.sciencedirect.com/science/article/pii/B9780128115343000020.

LEVENBERG, Kenneth (1944). "A Method for the Solution of Certain Non-Linear Problems in Least-Squares". In : *Quarterly of Applied Mathematics* 2.2, p. 164-168. ISSN : 0033-569X. JSTOR : 43633451.

LEVITT, Jonathan B., Thinsley Kalsang BHUTIA et Kara ROGERS (1er juin 2018). *Receptive Field | Physiology | Britannica*. britannica. URL : https://www.britannica.com/science/receptive-field (visité le 21/11/2021).

LI, Xiaochuan, David MBA et Tianran LIN (oct. 2019). "A Similarity-based and Model-based Fusion Prognostics Framework for Remaining Useful Life Prediction". In : *2019 Prognostics and System Health Management Conference (PHM-Qingdao)*. 2019 Prognostics and System Health Management Conference (PHM-Qingdao), p. 1-5. DOI : 10.1109/PHM-Qingdao46334.2019.8943006.

LIAO, Yixiao, Lei ZHANG et Weihua LI (2017). "Regrouping Particle Swarm Optimization-Based Neural Network for Bearing Fault Diagnosis". In : *2017 International Conference on Sensing, Diagnostics, Prognostics, and Control (SDPC)*, p. 628-631. DOI : 10.1109/SDPC.2017.123.

LIAW, Richard *et al.* (13 juil. 2018). *Tune: A Research Platform for Distributed Model Selection and Training*. arXiv : 1807.05118 [cs, stat]. URL : http://arxiv.org/abs/1807.05118 (visité le 07/09/2021).

LIN, Tsung-Yi *et al.* (20 fév. 2015). *Microsoft COCO: Common Objects in Context*. arXiv : 1405.0312 [cs]. URL : http://arxiv.org/abs/1405.0312 (visité le 11/07/2021).

Lines, Jason, Sarah Taylor et Anthony Bagnall (déc. 2016). “HIVE-COTE: The Hierarchical Vote Collective of Transformation-Based Ensembles for Time Series Classification”. In : *2016 IEEE 16th International Conference on Data Mining (ICDM)*. 2016 IEEE 16th International Conference on Data Mining (ICDM). Barcelona, Spain : IEEE, p. 1041-1046. isbn : 978-1-5090-5473-2. doi : 10.1109/ICDM.2016.0133. url : http://ieeexplore.ieee.org/document/7837946/ (visité le 10/11/2021).

Linnainmaa, Seppo (1er juin 1976). “Taylor Expansion of the Accumulated Rounding Error”. In : *BIT Numerical Mathematics* 16.2, p. 146-160. issn : 1572-9125. doi : 10.1007/BF01931367. url : https://doi.org/10.1007/BF01931367 (visité le 26/10/2021).

Liu, Ruonan *et al.* (1er août 2018). “Artificial Intelligence for Fault Diagnosis of Rotating Machinery: A Review”. In : *Mechanical Systems and Signal Processing* 108, p. 33-47. issn : 0888-3270. doi : 10.1016/j.ymssp.2018.02.016. url : https://www.sciencedirect.com/science/article/pii/S0888327018300748 (visité le 24/11/2021).

Liu, Yanchi *et al.* (déc. 2010). “Understanding of Internal Clustering Validation Measures”. In : *2010 IEEE International Conference on Data Mining*. 2010 IEEE International Conference on Data Mining, p. 911-916. doi : 10.1109/ICDM.2010.35.

Liu, Yingchao, Xiaofeng Hu et Wenjuan Zhang (1er mai 2019). “Remaining Useful Life Prediction Based on Health Index Similarity”. In : *Reliability Engineering & System Safety* 185, p. 502-510. issn : 0951-8320. doi : 10.1016/j.ress.2019.02.002. url : https://www.sciencedirect.com/science/article/pii/S0951832018309220 (visité le 27/09/2021).

Liu, Zhen *et al.* (2017). “Similarity-Based Difference Analysis Approach for Remaining Useful Life Prediction of GaAs-Based Semiconductor Lasers”. In : *IEEE Access* 5, p. 21508-21523. issn : 2169-3536. doi : 10.1109/ACCESS.2017.2759325.

Lopez Pinaya, Walter Hugo *et al.* (1er jan. 2020). “Chapter 10 - Convolutional Neural Networks”. In : *Machine Learning*. Sous la dir. d’Andrea Mechelli et Sandra Vieira. Academic Press, p. 173-191. isbn : 978-0-12-815739-8. doi : 10.1016/B978-0-12-815739-8.00010-9. url : https://www.sciencedirect.com/science/article/pii/B9780128157398000109 (visité le 19/11/2021).

Lyu, Jianhua *et al.* (1er oct. 2020). “Remaining Useful Life Estimation with Multiple Local Similarities”. In : *Engineering Applications of Artificial Intelligence* 95, p. 103849. issn : 0952-1976. doi : 10.1016/j.engappai.2020.103849. url : https://www.sciencedirect.com/science/article/pii/S0952197620302104 (visité le 27/09/2021).

Ma, Sai et Fulei Chu (1er fév. 2019). “Ensemble Deep Learning-Based Fault Diagnosis of Rotor Bearing Systems”. In : *Computers in Industry* 105, p. 143-152. issn : 0166-3615. doi : 10.1016/j.compind.2018.12.012. url : https://www.sciencedirect.com/science/article/pii/S0166361518304731 (visité le 26/10/2021).

Mai, Carole et Robin Chevalier (2016). “Equipment Diagnostics Based on Comparison of Past Abnormal Behaviors Using a Big Data Platform”. In : *PHM Society European Conference* 3.1 (1). issn : 2325-016X. doi : 10.36001/phme.2016.v3i1.1602. url : https://papers.phmsociety.org/index.php/phme/article/view/1602 (visité le 27/09/2021).

Marquardt, Donald W. (1963). “An Algorithm for Least-Squares Estimation of Nonlinear Parameters”. In : *Journal of the Society for Industrial and Applied Mathematics* 11.2, p. 431-441. issn : 0368-4245. JSTOR : 2098941.

Martínez-Arellano, Giovanna, German Terrazas et Svetan Ratchev (1er oct. 2019). “Tool Wear Classification Using Time Series Imaging and Deep Learning”. In : *The International Journal of Advanced Manufacturing Technology* 104.9, p. 3647-3662. issn : 1433-3015. doi : 10.1007/s00170-019-04090-6. url : https://doi.org/10.1007/s00170-019-04090-6 (visité le 05/11/2021).

McCann, Robert J. (nov. 1995). “Existence and Uniqueness of Monotone Measure-Preserving Maps”. In : *Duke Mathematical Journal* 80.2, p. 309-323. issn : 0012-7094, 1547-7398. doi : 10.1215/S0012-7094-95-08013-2. url : https://projecteuclid.org/journals/duke-mathematical-journal/volume-80/issue-2/Existence-and-uniqueness-of-monotone-measure-preserving-maps/10.1215/S0012-7094-95-08013-2.full (visité le 13/09/2021).

MCTG (2019). *Airline Maintenance Cost Executive Commentary*. MCTG December 2019. International Air Transport Association, p. 21.

Metz, Luke *et al.* (10 nov. 2021). *Gradients Are Not All You Need*. arXiv : 2111.05803 [cs, stat]. url : http://arxiv.org/abs/2111.05803 (visité le 18/11/2021).

Min, Erxue *et al.* (2018). “A Survey of Clustering With Deep Learning: From the Perspective of Network Architecture”. In : *IEEE Access* 6, p. 39501-39514. ISSN : 2169-3536. DOI : 10.1109/ACCESS.2018.2855437.

Mofokeng, Tseko, Paul T. Mativenga et Annlizé Marnewick (1er jan. 2020). “Analysis of Aircraft Maintenance Processes and Cost”. In : *Procedia CIRP*. 27th CIRP Life Cycle Engineering Conference (LCE2020)Advancing Life Cycle Engineering : From Technological Eco-Efficiency to Technology That Supports a World That Meets the Development Goals and the Absolute Sustainability 90, p. 467-472. ISSN : 2212-8271. DOI : 10.1016/j.procir.2020.01.115. URL : http://www.sciencedirect.com/science/article/pii/S2212827120302869 (visité le 31/01/2021).

Monge, Gaspard (1781). “Mémoire sur la théorie des déblais et des remblais”. In : p. 40.

Mori, Usue, Alexander Mendiburu et Jose A. Lozano (2016). “Similarity Measure Selection for Clustering Time Series Databases”. In : *IEEE Transactions on Knowledge and Data Engineering* 28.1, p. 181-195. DOI : 10.1109/TKDE.2015.2462369.

Mousavi, S. Mostafa *et al.* (nov. 2019). “Unsupervised Clustering of Seismic Signals Using Deep Convolutional Autoencoders”. In : *IEEE Geoscience and Remote Sensing Letters* 16.11, p. 1693-1697. ISSN : 1558-0571. DOI : 10.1109/LGRS.2019.2909218.

Munir, Mohsin *et al.* (2019). “DeepAnT: A Deep Learning Approach for Unsupervised Anomaly Detection in Time Series”. In : *IEEE Access* 7, p. 1991-2005. ISSN : 2169-3536. DOI : 10.1109/ACCESS.2018.2886457.

Nair, Vinod et Geoffrey E. Hinton (21 juin 2010). “Rectified Linear Units Improve Restricted Boltzmann Machines”. In : *Proceedings of the 27th International Conference on International Conference on Machine Learning*. ICML’10. Madison, WI, USA : Omnipress, p. 807-814. ISBN : 978-1-60558-907-7.

Nanduri, Anvardh et Lance Sherry (19-21 avr. 2016). “Anomaly Detection in Aircraft Data Using Recurrent Neural Networks (RNN)”. In : *2016 Integrated Communications Navigation and Surveillance (ICNS)*. 2016 Integrated Communications Navigation and Surveillance (ICNS). Herndon, Virginia, USA, p. 5C2-1. DOI : 10.1109/ICNSURV.2016.7486356.

Nannan, Hu, Su Xiaohao et Liu Baolong (avr. 2018). “A Review on Prognostics and Health Management”. In : 2018 Second International Conference of Sensor Network and Computer Engineering (ICSNCE 2018). Atlantis Press, p. 176-182. ISBN : 978-94-6252-498-9. DOI : 10.2991/icsnce-18.2018.35. URL : https://www.atlantis-press.com/proceedings/icsnce-18/25894542 (visité le 24/11/2021).

Neal, Radford M. (1996). “Priors for Infinite Networks”. In : *Bayesian Learning for Neural Networks*. Sous la dir. de Radford M. Neal. Lecture Notes in Statistics. New York, NY : Springer, p. 29-53. ISBN : 978-1-4612-0745-0. DOI : 10.1007/978-1-4612-0745-0_2. URL : https://doi.org/10.1007/978-1-4612-0745-0_2 (visité le 19/11/2021).

Nesterov, Yurii (1983). “A Method for Unconstrained Convex Minimization Problem with the Rate of Convergence o(1/K²)”. In :

Nguyen, Van Duc *et al.* (2019). “A Review: Prognostics and Health Management in Automotive and Aerospace”. In : *International Journal of Prognostics and Health Management* 10.2 (2). ISSN : 2153-2648. DOI : 10.36001/ijphm.2019.v10i2.2730. URL : https://papers.phmsociety.org/index.php/ijphm/article/view/2730 (visité le 24/11/2021).

Nicchiotti, Gianluca (2 juil. 2018). “Data-Driven Prediction of Unscheduled Maintenance Replacements in a Fleet of Commercial Aircrafts”. In : *PHM Society European Conference* 4.1 (1). ISSN : 2325-016X. DOI : 10.36001/phme.2018.v4i1.237. URL : http://papers.phmsociety.org/index.php/phme/article/view/237 (visité le 11/11/2021).

Normalisation, Organisatsion Internationale de (sept. 2015). *ISO 13381-1:2015(fr), Surveillance et diagnostic des machines*. Norme. ISO, p. 22. URL : https://www.iso.org/obp/ui/#iso:std:iso:13381:-1:ed-2:v2:fr (visité le 16/11/2021).

Oliphant, Travis (5 déc. 2006). “A Bayesian Perspective on Estimating Mean, Variance, and Standard-Deviation from Data”. In : *Faculty Publications*. URL : https://scholarsarchive.byu.edu/facpub/278.

Organisatsion Internationale de Normalisation (sept. 2012). *ISO 13372:2012, Surveillance et diagnotic de l’état des machines*. Norme. ISO, p. 15. URL : https://www.iso.org/cms/render/live/fr/sites/isoorg/contents/data/standard/05/22/52256.html (visité le 16/11/2021).

Ozaki, Yoshihiko *et al.* (25 juin 2020). “Multiobjective Tree-Structured Parzen Estimator for Computationally Expensive Optimization Problems”. In : *Proceedings of the 2020 Genetic and*

Evolutionary Computation Conference. GECCO '20. New York, NY, USA : Association for Computing Machinery, p. 533-541. ISBN : 978-1-4503-7128-5. DOI : 10.1145/3377930.3389817. URL : https://doi.org/10.1145/3377930.3389817 (visité le 22/09/2021).

PAN, Sinno Jialin et Qiang YANG (oct. 2010). "A Survey on Transfer Learning". In : *IEEE Transactions on Knowledge and Data Engineering* 22.10, p. 1345-1359. ISSN : 1558-2191. DOI : 10.1109/TKDE.2009.191.

PARZEN, Emanuel (sept. 1962). "On Estimation of a Probability Density Function and Mode". In : *Annals of Mathematical Statistics* 33.3, p. 1065-1076. ISSN : 0003-4851, 2168-8990. DOI : 10.1214/aoms/1177704472. URL : https://projecteuclid.org/euclid.aoms/1177704472 (visité le 27/01/2021).

PASZKE, Adam *et al.* (2019). "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In : *Advances in Neural Information Processing Systems 32*. Sous la dir. de H. WALLACH *et al.* Curran Associates, Inc., p. 8024-8035. URL : http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

PEDREGOSA, F. *et al.* (2011). "Scikit-Learn: Machine Learning in Python". In : *Journal of Machine Learning Research* 12, p. 2825-2830.

PERAFÁN-LÓPEZ, Juan Carlos et Julián SIERRA-PÉREZ (16 oct. 2020). "An Unsupervised Pattern Recognition Methodology Based on Factor Analysis and a Genetic-DBSCAN Algorithm to Infer Operational Conditions from Strain Measurements in Structural Applications". In : *Chinese Journal of Aeronautics*. ISSN : 1000-9361. DOI : 10.1016/j.cja.2020.09.035. URL : http://www.sciencedirect.com/science/article/pii/S1000936120304775 (visité le 31/01/2021).

PERSLEV, Mathias *et al.* (24 oct. 2019). *U-Time: A Fully Convolutional Network for Time Series Segmentation Applied to Sleep Staging*. arXiv : 1910.11162 [cs, eess, stat]. URL : http://arxiv.org/abs/1910.11162 (visité le 05/11/2021).

PEYRÉ, Gabriel et Marco CUTURI (18 mar. 2020). *Computational Optimal Transport*. arXiv : 1803.00567 [stat]. URL : http://arxiv.org/abs/1803.00567 (visité le 04/02/2021).

PURI, Siddhartha (29 juin 2010). "Training Convolutional Neural Networks on Graphics Processing Units". Brev. amér. 7747070B2. MICROSOFT CORP. URL : https://patents.google.com/patent/US7747070B2/en (visité le 22/11/2021).

QIAN, Ning (1er jan. 1999). "On the Momentum Term in Gradient Descent Learning Algorithms". In : *Neural Networks* 12.1, p. 145-151. ISSN : 0893-6080. DOI : 10.1016/S0893-6080(98)00116-6. URL : https://www.sciencedirect.com/science/article/pii/S0893608098001166 (visité le 26/10/2021).

RAINA, Rajat, Anand MADHAVAN et Andrew Y. NG (14 juin 2009). "Large-Scale Deep Unsupervised Learning Using Graphics Processors". In : *Proceedings of the 26th Annual International Conference on Machine Learning*. ICML '09. New York, NY, USA : Association for Computing Machinery, p. 873-880. ISBN : 978-1-60558-516-1. DOI : 10.1145/1553374.1553486. URL : https://doi.org/10.1145/1553374.1553486 (visité le 26/10/2021).

RAISSI, Maziar et George Em KARNIADAKIS (15 mar. 2018). "Hidden Physics Models: Machine Learning of Nonlinear Partial Differential Equations". In : *Journal of Computational Physics* 357, p. 125-141. ISSN : 0021-9991. DOI : 10.1016/j.jcp.2017.11.039. URL : http://www.sciencedirect.com/science/article/pii/S0021999117309014.

RAMEZANI, Saeed, Alireza MOINI et Mohamad RIAHI (1er avr. 2019). "Prognostics and Health Management in Machinery: A Review of Methodologies for RUL Prediction and Roadmap". In : *International Journal of Industrial Engineering and Management Science* 6.1, p. 38-61. ISSN : 2409-1871. URL : http://www.ijiems.com/article_86909.html (visité le 24/11/2021).

RANASINGHE, Kavindu *et al.* (1er jan. 2022). "Advances in Integrated System Health Management for Mission-Essential and Safety-Critical Aerospace Applications". In : *Progress in Aerospace Sciences* 128, p. 100758. ISSN : 0376-0421. DOI : 10.1016/j.paerosci.2021.100758. URL : https://www.sciencedirect.com/science/article/pii/S0376042121000622 (visité le 25/11/2021).

RANZATO, Marc'Aurelio *et al.* (4 déc. 2006). "Efficient Learning of Sparse Representations with an Energy-Based Model". In : *Proceedings of the 19th International Conference on Neural Information Processing Systems*. NIPS'06. Cambridge, MA, USA : MIT Press, p. 1137-1144.

REDDI, Sashank J., Satyen KALE et Sanjiv KUMAR (2018). *On the Convergence of Adam and Beyond*. arXiv : 1904.09237. URL : http://arxiv.org/abs/1904.09237 (visité le 26/10/2021).

REZAEIANJOUYBARI, Behnoush et Yi SHANG (15 oct. 2020). "Deep Learning for Prognostics and Health Management: State of the Art, Challenges, and Opportunities". In : *Measurement* 163,

p. 107929. ISSN : 0263-2241. DOI : 10.1016/j.measurement.2020.107929. URL : https://www.sciencedirect.com/science/article/pii/S026322412030467X (visité le 24/11/2021).

RIPLEY, Brian D. (1996). *Pattern Recognition and Neural Networks*. Cambridge : Cambridge University Press. ISBN : 978-0-521-71770-0. DOI : 10.1017/CBO9780511812651. URL : https://www.cambridge.org/core/books/pattern-recognition-and-neural-networks/4E038249C9BAA06C8F4EE6F044D09C5C (visité le 06/11/2021).

RITCHIE, Hannah (18 sept. 2020). *Sector by Sector: Where Do Global Greenhouse Gas Emissions Come From?* Our World in Data. URL : https://ourworldindata.org/ghg-emissions-by-sector (visité le 11/11/2021).

RITCHIE, Hannah et Max ROSER (sept. 2021). "Transport". In : *Our World in Data*.

ROBBINS, Herbert et Sutton MONRO (sept. 1951). "A Stochastic Approximation Method". In : *The Annals of Mathematical Statistics* 22.3, p. 400-407. ISSN : 0003-4851, 2168-8990. DOI : 10.1214/aoms/1177729586. URL : https://projecteuclid.org/journals/annals-of-mathematical-statistics/volume-22/issue-3/A-Stochastic-Approximation-Method/10.1214/aoms/1177729586.full (visité le 26/10/2021).

RONNEBERGER, Olaf, Philipp FISCHER et Thomas BROX (18 mai 2015). *U-Net: Convolutional Networks for Biomedical Image Segmentation*. arXiv : 1505.04597 [cs]. URL : http://arxiv.org/abs/1505.04597 (visité le 17/09/2020).

ROSENBLATT, Frank (1958). *The Perceptron: A Theory of Statistical Separability in Cognitive Systems*. 1 t. Buffalo, N.Y. : Cornell Aeronautical Laboratory. 268 p.

ROUSSEEUW, Peter J. (1er nov. 1987). "Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis". In : *Journal of Computational and Applied Mathematics* 20, p. 53-65. ISSN : 0377-0427. DOI : 10.1016/0377-0427(87)90125-7. URL : https://www.sciencedirect.com/science/article/pii/0377042787901257 (visité le 16/07/2021).

RUDER, Sebastian (15 juin 2017). *An Overview of Gradient Descent Optimization Algorithms*. arXiv : 1609.04747 [cs]. URL : http://arxiv.org/abs/1609.04747 (visité le 26/10/2021).

RUMELHART, David E., Geoffrey E. HINTON et Ronald J. WILLIAMS (oct. 1986). "Learning Representations by Back-Propagating Errors". In : *Nature* 323.6088 (6088), p. 533-536. ISSN : 1476-4687. DOI : 10.1038/323533a0. URL : https://www.nature.com/articles/323533a0 (visité le 20/11/2021).

RUSSAKOVSKY, Olga *et al.* (2015). "ImageNet Large Scale Visual Recognition Challenge". In : *International Journal of Computer Vision (IJCV)* 115.3, p. 211-252. DOI : 10.1007/s11263-015-0816-y.

SAFRANGROUP (26 oct. 2021). *Aviation Industry Chief Technology Officers Issue Joint Call to Action to Deliver Sustainable Aviation Plans | Safran*. www.safrangroup.com. URL : https://www.safran-group.com/pressroom/aviation-industry-chief-technology-officers-issue-joint-call-action-deliver-sustainable-aviation-2021-10-26 (visité le 27/10/2021).

SAHA, B. et K. GOEBEL (2007). *Battery Data Set*. URL : http://ti.arc.nasa.gov/project/prognostic-data-repository.

SALAKHUTDINOV, Ruslan et Geoffrey HINTON (15 avr. 2009). "Deep Boltzmann Machines". In : *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics*. T. 5. PMLR, p. 448-455. URL : https://proceedings.mlr.press/v5/salakhutdinov09a.html.

SAVITZKY, Abraham. et M. J. E. GOLAY (1er juil. 1964). "Smoothing and Differentiation of Data by Simplified Least Squares Procedures." In : *Analytical Chemistry* 36.8, p. 1627-1639. ISSN : 0003-2700. DOI : 10.1021/ac60214a047. URL : https://doi.org/10.1021/ac60214a047 (visité le 20/10/2021).

SAXENA, A. et K. GOEBEL (2008). "Turbofan Engine Degradation Simulation Data Set". In : NASA Ames, Moffett Field, CA. URL : https://ti.arc.nasa.gov/c/13/.

SCHWENK, Holger et Maurice MILGRAM (1er jan. 1994). "Transformation Invariant Autoassociation with Application to Handwritten Character Recognition". In : *Proceedings of the 7th International Conference on Neural Information Processing Systems*. NIPS'94. Cambridge, MA, USA : MIT Press, p. 991-998.

SCOTT, David W. (13 mar. 2015). *Multivariate Density Estimation: Theory, Practice, and Visualization, Second Edition*. Wiley Series in Probability and Statistics. John Wiley & Sons, Ltd. ISBN : 978-1-118-57557-4. DOI : 10.1002/9781118575574.ch6. URL : https://onlinelibrary.wiley.com/doi/abs/10.1002/9781118575574.ch6 (visité le 08/05/2020).

SHANNON, C. E. (1948). "A Mathematical Theory of Communication". In : *Bell System Technical Journal* 27.3, p. 379-423. ISSN : 1538-7305. DOI : 10.1002/j.1538-7305.1948.tb01338.x. URL :

https://onlinelibrary.wiley.com/doi/abs/10.1002/j.1538-7305.1948.tb01338.x (visité le 12/10/2021).

Shao, Haidong, Hongkai Jiang, Ying Lin *et al.* (1er mar. 2018). "A Novel Method for Intelligent Fault Diagnosis of Rolling Bearings Using Ensemble Deep Auto-Encoders". In : *Mechanical Systems and Signal Processing* 102, p. 278-297. ISSN : 0888-3270. DOI : 10.1016/j.ymssp.2017.09.026. URL : http://www.sciencedirect.com/science/article/pii/S0888327017305034 (visité le 18/02/2020).

Shao, Haidong, Hongkai Jiang, Haizhou Zhang *et al.* (1er fév. 2018). "Rolling Bearing Fault Feature Learning Using Improved Convolutional Deep Belief Network with Compressed Sensing". In : *Mechanical Systems and Signal Processing* 100, p. 743-765. ISSN : 0888-3270. DOI : 10.1016/j.ymssp.2017.08.002. URL : https://www.sciencedirect.com/science/article/pii/S0888327017304260 (visité le 26/10/2021).

Shao, Siyu, Pu Wang et Ruqiang Yan (1er avr. 2019). "Generative Adversarial Networks for Data Augmentation in Machine Fault Diagnosis". In : *Computers in Industry* 106, p. 85-93. ISSN : 0166-3615. DOI : 10.1016/j.compind.2019.01.001. URL : https://www.sciencedirect.com/science/article/pii/S0166361518305657 (visité le 26/10/2021).

SHERLOC – SHERLOC Project (2020). http://sherloc-project.com/. URL : http://sherloc-project.com/ (visité le 16/11/2021).

Silverman, Bernard W. (1er avr. 1986). *Density Estimation for Statistics and Data Analysis*. CRC Press. 190 p. ISBN : 978-0-412-24620-3.

Simonyan, Karen et Andrew Zisserman (10 avr. 2015). *Very Deep Convolutional Networks for Large-Scale Image Recognition*. arXiv : 1409.1556 [cs]. URL : http://arxiv.org/abs/1409.1556 (visité le 21/11/2021).

Sinkhorn, Richard (juin 1964). "A Relationship Between Arbitrary Positive Matrices and Doubly Stochastic Matrices". In : *The Annals of Mathematical Statistics* 35.2, p. 876-879. ISSN : 0003-4851, 2168-8990. DOI : 10.1214/aoms/1177703591. URL : https://projecteuclid.org/journals/annals-of-mathematical-statistics/volume-35/issue-2/A-Relationship-Between-Arbitrary-Positive-Matrices-and-Doubly-Stochastic-Matrices/10.1214/aoms/1177703591.full (visité le 13/10/2021).

— (1967). "Diagonal Equivalence to Matrices with Prescribed Row and Column Sums". In : *The American Mathematical Monthly* 74.4, p. 402-405. ISSN : 0002-9890. DOI : 10.2307/2314570. JSTOR : 2314570.

Sinkhorn, Richard et Paul Knopp (1er mai 1967). "Concerning Nonnegative Matrices and Doubly Stochastic Matrices". In : *Pacific Journal of Mathematics* 21.2, p. 343-348. ISSN : 0030-8730. URL : https://msp.org/pjm/1967/21-2/p14.xhtml (visité le 13/10/2021).

Smolensky, P. (3 jan. 1986). "Information Processing in Dynamical Systems: Foundations of Harmony Theory". In : *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations*. Cambridge, MA, USA : MIT Press, p. 194-281. ISBN : 978-0-262-68053-0.

Soons, Youri *et al.* (2020). "Predicting Remaining Useful Life with Similarity-Based Priors". In : *Advances in Intelligent Data Analysis XVIII*. Sous la dir. de Michael R. Berthold, Ad Feelders et Georg Krempl. Lecture Notes in Computer Science. Cham : Springer International Publishing, p. 483-495. ISBN : 978-3-030-44584-3. DOI : 10.1007/978-3-030-44584-3_38.

Sprong, Jorben Pieter, Xiaoli Jiang et Henk Polinder (22 sept. 2019). "A Deployment of Prognostics to Optimize Aircraft Maintenance - A Literature Review: A Literature Review". In : *Annual Conference of the PHM Society* 11.1 (1). ISSN : 2325-0178. DOI : 10.36001/phmconf.2019.v11i1.776. URL : https://papers.phmsociety.org/index.php/phmconf/article/view/776 (visité le 24/11/2021).

Student (mar. 1908). "The Probable Error of a Mean". In : *Biometrika* 6.1, p. 1. ISSN : 00063444. DOI : 10.2307/2331554. JSTOR : 2331554.

Sun, Wenjun *et al.* (1er juil. 2016). "A Sparse Auto-Encoder-Based Deep Neural Network Approach for Induction Motor Faults Classification". In : *Measurement* 89, p. 171-178. ISSN : 0263-2241. DOI : 10.1016/j.measurement.2016.04.007. URL : https://www.sciencedirect.com/science/article/pii/S0263224116300641 (visité le 26/10/2021).

Susto, Gian Antonio, Angelo Cenedese et Matteo Terzi (1er jan. 2018). "Chapter 9 - Time-Series Classification Methods: Review and Applications to Power Systems Data". In : *Big Data Application in Power Systems*. Sous la dir. de Reza Arghandeh et Yuxun Zhou. Elsevier, p. 179-220. ISBN : 978-0-12-811968-6. DOI : 10.1016/B978-0-12-811968-6.00009-7. URL :

https://www.sciencedirect.com/science/article/pii/B9780128119686000097 (visité le 05/11/2021).

Tavakoli, Neda *et al.* (20 avr. 2020). "An Autoencoder-Based Deep Learning Approach for Clustering Time Series Data". In : *SN Applied Sciences* 2.5, p. 937. ISSN : 2523-3971. DOI : 10.1007/s42452-020-2584-8. URL : https://doi.org/10.1007/s42452-020-2584-8 (visité le 25/10/2021).

Tavenard, Romain *et al.* (2020). "Tslearn, A Machine Learning Toolkit for Time Series Data". In : p. 6.

Tchakoua, Pierre *et al.* (22 avr. 2014). "Wind Turbine Condition Monitoring: State-of-the-Art Review, New Trends, and Future Challenges". In : *Energies* 7.4 (4), p. 2595-2630. DOI : 10.3390/en7042595. URL : https://www.mdpi.com/1996-1073/7/4/2595 (visité le 29/11/2021).

Turlach, Berwin A. (1993). "Bandwidth Selection in Kernel Density Estimation: A Review". In : *CORE and Institut de Statistique*.

Tzinis, Irene (6 mai 2015). *Technology Readiness Level*. NASA. URL : http://www.nasa.gov/directorates/heo/scan/engineering/technology/technology_readiness_level (visité le 29/11/2021).

UNFCCC (1997). *Kyoto Protocol Reference Manual - On Accounting of Emissions and Assigned Amount*. United Nations Framework Convention on Climate Change. Kyoto Protocol - Reference Manual. Kyoto : United Nations, p. 130. URL : https://unfccc.int/resource/docs/publications/08_unfccc_kp_ref_manual.pdf.

Villani, Cédric (2003). *Topics in Optimal Transportation*. Graduate Studies in Mathematics 58. American Mathematical Society. ISBN : 978-1-4704-6726-5.

— (2009). *Optimal Transport: Old and New*. Grundlehren Der Mathematischen Wissenschaften. Berlin Heidelberg : Springer-Verlag. ISBN : 978-3-540-71049-3. DOI : 10.1007/978-3-540-71050-9. URL : https://www.springer.com/gp/book/9783540710493 (visité le 13/09/2021).

Virtanen, Pauli *et al.* (2020). "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In : *Nature Methods* 17, p. 261-272. DOI : 10.1038/s41592-019-0686-2.

Vrignat, Pascal, Frédéric Kratz et Manuel Avila (1er fév. 2022). "Sustainable Manufacturing, Maintenance Policies, Prognostics and Health Management: A Literature Review". In : *Reliability Engineering & System Safety* 218, p. 108140. ISSN : 0951-8320. DOI : 10.1016/j.ress.2021.108140. URL : https://www.sciencedirect.com/science/article/pii/S095183202100630X (visité le 25/11/2021).

Wang, Mengni *et al.* (2020). "Combining Autoencoder with Similarity Measurement for Aircraft Engine Remaining Useful Life Estimation". In : *Proceedings of the International Conference on Aerospace System Science and Engineering 2019*. Sous la dir. de Zhongliang Jing. Lecture Notes in Electrical Engineering. Singapore : Springer, p. 197-208. DOI : 10.1007/978-981-15-1773-0_14.

Wang, Tianyi *et al.* (oct. 2008). "A Similarity-Based Prognostics Approach for Remaining Useful Life Estimation of Engineered Systems". In : *2008 International Conference on Prognostics and Health Management*. 2008 International Conference on Prognostics and Health Management, p. 1-6. DOI : 10.1109/PHM.2008.4711421.

Wang, Xingjian *et al.* (19-22 juin 2018). "Fault Diagnosis of Electromechanical Actuator Based on Principal Component Analysis and Support Vector Machine". In : *CSAA/IET International Conference on Aircraft Utility Systems (AUS 2018)*. CSAA/IET International Conference on Aircraft Utility Systems (AUS 2018), p. 1467-1471. DOI : 10.1049/cp.2018.0322.

Wang, Zhiguang et Tim Oates (1er juin 2015a). "Imaging Time-Series to Improve Classification and Imputation". In : URL : https://arxiv.org/abs/1506.00327v1 (visité le 10/11/2021).

— (24 sept. 2015b). *Spatially Encoding Temporal Correlations to Classify Temporal Data Using Convolutional Neural Networks*. arXiv : 1509.07481 [cs]. URL : http://arxiv.org/abs/1509.07481 (visité le 14/10/2020).

Wang, Zirui, Jun Wang et Youren Wang (8 oct. 2018). "An Intelligent Diagnosis Scheme Based on Generative Adversarial Learning Deep Neural Networks and Its Application to Planetary Gearbox Fault Pattern Recognition". In : *Neurocomputing* 310, p. 213-222. ISSN : 0925-2312. DOI : 10.1016/j.neucom.2018.05.024. URL : https://www.sciencedirect.com/science/article/pii/S0925231218305617 (visité le 26/10/2021).

Watch, Climate (2020). *CAIT Data for GHG Emissions*. URL : https://www.climatewatchdata.org/ghg-emissions..

Wen, LIU, TAN Jian Ping et SANG Yan Wei (jan. 2021). "An Evidential Similarity-Based Regression Method for the Prediction of Equipment Remaining Useful Life under Complex

Conditions". In : *2021 IEEE International Conference on Consumer Electronics and Computer Engineering (ICCECE)*. 2021 IEEE International Conference on Consumer Electronics and Computer Engineering (ICCECE), p. 362-366. DOI : 10.1109/ICCECE51280.2021.9342372.

WESENDRUP, Kevin et Bernd HELLINGRATH (18 juil. 2020). "A Process-based Review of Post-Prognostics Decision-Making". In : *PHM Society European Conference* 5.1 (1), p. 12-12. ISSN : 2325-016X. DOI : 10.36001/phme.2020.v5i1.1203. URL : https://papers.phmsociety.org/index.php/phme/article/view/1203 (visité le 24/11/2021).

WILSON, Hugh R. et Jack D. COWAN (jan. 1972). "Excitatory and Inhibitory Interactions in Localized Populations of Model Neurons". In : *Biophysical Journal* 12.1, p. 1-24. ISSN : 0006-3495. pmid : 4332108. URL : https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1484078/ (visité le 18/11/2021).

XINXIN, Xiong, Li QING et Cheng NONG (août 2016). "Remaining Useful Life Prognostics of Aircraft Engine Based on Fusion Algorithm". In : *2016 IEEE Chinese Guidance, Navigation and Control Conference (CGNCC)*. 2016 IEEE Chinese Guidance, Navigation and Control Conference (CGNCC), p. 628-633. DOI : 10.1109/CGNCC.2016.7828859.

XU, Fan et Peter W. TSE (1er jan. 2019). "Combined Deep Belief Network in Deep Learning with Affinity Propagation Clustering Algorithm for Roller Bearings Fault Diagnosis without Data Label". In : *Journal of Vibration and Control* 25.2, p. 473-482. ISSN : 1077-5463. DOI : 10.1177/1077546318783886. URL : https://doi.org/10.1177/1077546318783886 (visité le 05/11/2021).

XU, Fan, Wai tai Peter TSE et Yiu Lun TSE (1er déc. 2018). "Roller Bearing Fault Diagnosis Using Stacked Denoising Autoencoder in Deep Learning and Gath–Geva Clustering Algorithm without Principal Component Analysis and Data Label". In : *Applied Soft Computing* 73, p. 898-913. ISSN : 1568-4946. DOI : 10.1016/j.asoc.2018.09.037. URL : https://www.sciencedirect.com/science/article/pii/S1568494618305544 (visité le 05/11/2021).

XU, Jiuping, Yusheng WANG et Lei XU (avr. 2014). "PHM-Oriented Integrated Fusion Prognostics for Aircraft Engines Based on Sensor Data". In : *IEEE Sensors Journal* 14.4, p. 1124-1132. ISSN : 1558-1748. DOI : 10.1109/JSEN.2013.2293517.

YANG, Xiaoyu *et al.* (1er jan. 2020). "Similarity-Based Information Fusion Grey Model for Remaining Useful Life Prediction of Aircraft Engines". In : *Grey Systems: Theory and Application* 11.3, p. 463-483. ISSN : 2043-9377. DOI : 10.1108/GS-05-2020-0066. URL : https://doi.org/10.1108/GS-05-2020-0066 (visité le 27/09/2021).

YOU, M-Y et G MENG (1er août 2011). "A Generalized Similarity Measure for Similarity-Based Residual Life Prediction". In : *Proceedings of the Institution of Mechanical Engineers, Part E: Journal of Process Mechanical Engineering* 225.3, p. 151-160. ISSN : 0954-4089. DOI : 10.1177/0954408911399832. URL : https://doi.org/10.1177/0954408911399832 (visité le 27/09/2021).

YOU, Ming-Yi et Guang MENG (mar. 2013a). "A Framework of Similarity-Based Residual Life Prediction Approaches Using Degradation Histories With Failure, Preventive Maintenance, and Suspension Events". In : *IEEE Transactions on Reliability* 62.1, p. 127-135. ISSN : 1558-1721. DOI : 10.1109/TR.2013.2241203.

— (1er fév. 2013b). "Toward Effective Utilization of Similarity Based Residual Life Prediction Methods: Weight Allocation, Prediction Robustness, and Prediction Uncertainty". In : *Proceedings of the Institution of Mechanical Engineers, Part E: Journal of Process Mechanical Engineering* 227.1, p. 74-84. ISSN : 0954-4089. DOI : 10.1177/0954408912449947. URL : https://doi.org/10.1177/0954408912449947 (visité le 27/09/2021).

YU, Wennian, II Yong KIM et Chris MECHEFSKE (1er juil. 2020). "An Improved Similarity-Based Prognostic Algorithm for RUL Estimation Using an RNN Autoencoder Scheme". In : *Reliability Engineering & System Safety* 199, p. 106926. ISSN : 0951-8320. DOI : 10.1016/j.ress.2020.106926. URL : https://www.sciencedirect.com/science/article/pii/S0951832019307902 (visité le 27/09/2021).

ZAPOROWSKA, Anna *et al.* (1er jan. 2020). "A Clustering Approach to Detect Faults with Multi-Component Degradations in Aircraft Fuel Systems". In : *IFAC-PapersOnLine* 53.3, p. 113-118. ISSN : 2405-8963. DOI : 10.1016/j.ifacol.2020.11.018. URL : http://www.sciencedirect.com/science/article/pii/S2405896320301622 (visité le 31/01/2021).

ZEILER, Matthew D. (22 déc. 2012). *ADADELTA: An Adaptive Learning Rate Method*. arXiv : 1212.5701 [cs]. URL : http://arxiv.org/abs/1212.5701 (visité le 26/10/2021).

ZHANG, Chong *et al.* (30 avr. 2018). *A Multi-State Diagnosis and Prognosis Framework with Feature Learning for Tool Condition Monitoring*. arXiv : 1805.00367 [cs, eess]. URL : http://arxiv.org/abs/1805.00367 (visité le 21/02/2019).

ZHANG, Chuxu *et al.* (17 juil. 2019). "A Deep Neural Network for Unsupervised Anomaly Detection and Diagnosis in Multivariate Time Series Data". In : *Proceedings of the AAAI Conference on Artificial Intelligence* 33.01 (01), p. 1409-1416. ISSN : 2374-3468. DOI : 10.1609/aaai.v33i01.33011409. URL : https://ojs.aaai.org/index.php/AAAI/article/view/3942 (visité le 05/11/2021).

ZHANG, Li *et al.* (1er août 2017). "A Deep Learning-Based Recognition Method for Degradation Monitoring of Ball Screw with Multi-Sensor Data Fusion". In : *Microelectronics Reliability* 75, p. 215-222. ISSN : 0026-2714. DOI : 10.1016/j.microrel.2017.03.038. URL : https://www.sciencedirect.com/science/article/pii/S0026271417300896 (visité le 26/10/2021).

ZHANG, Liangwei *et al.* (2019). "A Review on Deep Learning Applications in Prognostics and Health Management". In : *IEEE Access* 7, p. 162415-162438. ISSN : 2169-3536. DOI : 10.1109/ACCESS.2019.2950985.

ZHANG, Sen-Ju, Rui KANG et Yan-Hui LIN (1er avr. 2021). "Remaining Useful Life Prediction for Degradation with Recovery Phenomenon Based on Uncertain Process". In : *Reliability Engineering & System Safety* 208, p. 107440. ISSN : 0951-8320. DOI : 10.1016/j.ress.2021.107440. URL : https://www.sciencedirect.com/science/article/pii/S0951832021000119 (visité le 27/09/2021).

ZHANG, Zhengxin *et al.* (16 déc. 2018). "Degradation Data Analysis and Remaining Useful Life Estimation: A Review on Wiener-process-based Methods". In : *European Journal of Operational Research* 271.3, p. 775-796. ISSN : 0377-2217. DOI : 10.1016/j.ejor.2018.02.033. URL : https://www.sciencedirect.com/science/article/pii/S0377221718301486 (visité le 24/11/2021).

ZHAO, H. *et al.* (1er jan. 2021). "Aero Engine Rul Prediction Based on the Combination of Similarity and PSO-SVR". In : p. 912-917. DOI : 10.1049/icp.2021.0462. URL : https://digital-library.theiet.org/content/conferences/10.1049/icp.2021.0462 (visité le 27/09/2021).

ZHAO, Rui *et al.* (15 jan. 2019). "Deep Learning and Its Applications to Machine Health Monitoring". In : *Mechanical Systems and Signal Processing* 115, p. 213-237. ISSN : 0888-3270. DOI : 10.1016/j.ymssp.2018.05.050. URL : http://www.sciencedirect.com/science/article/pii/S0888327018303108 (visité le 07/05/2019).

ZHOU, Lei *et al.* (1er mar. 2020). "Airline Planning and Scheduling: Models and Solution Methodologies". In : *Frontiers of Engineering Management* 7.1, p. 1-26. ISSN : 2096-0255. DOI : 10.1007/s42524-020-0093-5. URL : https://doi.org/10.1007/s42524-020-0093-5 (visité le 11/11/2021).

ZUCKER, Steven W. (1977). "Algorithms for Image Segmentation". In :

Notions d'algèbre

Définitions

Définition .0.1 : Espace euclidien

Un espace euclidien E est un espace préhilbertien réel de dimension finie. Autrement dit, une espace euclidien est un espace de dimension finie muni d'un produit scalaire.

Définition .0.2 : Jacobien

Soit F une fonction de plusieurs variables définie sur un ouvert de $\mathbb{R}^n$ à valeurs dans $\mathbb{R}^m$. Cette dernière est définie par la valeur de ses composantes réelles telle que :

$$F : \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \to \begin{pmatrix} f_1(x_1, \ldots, x_n) \\ \vdots \\ f_m(x_1, \ldots, x_n) \end{pmatrix}$$

Si les dérivées partielles de ces fonctions composantes existent, elle peuvent être rassemblées dans une matrice $\mathcal{M}_{m,n}(\mathbb{R})$ appelée la matrice Jacobienne *de F et définie telle que :*

$$\mathcal{J}_F = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{pmatrix}$$

Avec $\frac{\partial f_i}{\partial x_j}$ représentant la dérivée partielle de la composante i de F par rapport à la variable x_j.

Définition .0.3 : Produit scalaire

Un produit scalaire défini sur un $\mathbb{K}$-espace vectoriel est une forme bilinéaire symétrique, définie positive à valeurs dans $\mathbb{K}$ qui sera notée $\cdot$[a] possédant les propriétés suivantes :

1. Symétrie : $\forall x, y \in E, \langle x, y \rangle = \langle y, x \rangle$
2. Définie : $\forall x \in E, \langle x, x \rangle = 0 \Rightarrow x = 0$
3. Positive : $\forall x \in E, \langle x, x \rangle \geq 0$

[a] En notations matricielle le point sera omis dès lors que la transposée apparaît.

Définition .0.4 : Produit matriciel

Soit deux matrices $\mathbb{A} \in \mathcal{M}_{np}(\mathbb{R})$ et $\mathbb{B} \in \mathcal{M}_{pm}(\mathbb{R})$. Le produit matriciel entre $\mathbb{A}$ et $\mathbb{B}$, noté $\mathbb{AB}$, est une matrice $\mathbb{C} \in \mathcal{M}_{nm}(\mathbb{R})$ dont les éléments (c_{ij}) sont définis tels que :

$$\forall i, j, \; c_{ij} = \sum_{k=1}^{p} a_{ik} b_{kj}$$

Avec (a_{ij}) et (b_{ij}) qui sont les éléments respectifs de $\mathbb{A}$ et $\mathbb{B}$.

Définition .0.5 : Norme

Une norme définie sur un $\mathbb{K}$-espace vectoriel E est une application $||.|| : E \to \mathbb{R}_+$ telle que :

1. $||x|| = 0 \Leftrightarrow x = 0$
2. *Homogénéité : Soit* $(\lambda, x) \in \mathbb{K} \times E, ||\lambda x|| = |\lambda| x$
3. *Inégalité triangulaire :* $\forall (x, y) \in E^2, ||x + y|| \leq ||x|| + ||y||$

Notions d'analyse

Définitions

La fonction sigmoïde est une fonction f définie comme :

$$\forall x \in \mathbb{R}, f_\lambda(x) = \frac{1}{1 + \exp -\lambda x} \tag{1}$$

La fonction tangente hyperbolique est une fonction définie par :

$$\begin{aligned} \tanh : \mathbb{C} \backslash i\pi \left(\mathbb{Z} + \frac{1}{2} \right) &\to \mathbb{C} \\ z &\mapsto \frac{\sinh(z)}{\cosh(z)} \end{aligned} \tag{2}$$