System identification enabled health state prognostics using deep learning. Application to predictive maintenance of business jet aircraft

Martin Herve De Beaulieu

UNIVERSITÉ DE LORRAINE

École doctorale IAEM Lorraine

Identification et pronostics de l'état de santé des systèmes non linéaires par apprentissage profond.

Application à la maintenance prévisionnelle des avions d'affaires.

THÈSE

présentée et soutenue publiquement le 4 décembre 2023

pour l'obtention du

Doctorat de l'Université de Lorraine

(mention Automatique, Traitement du signal et des images, Génie informatique)

par

Martin Hervé de Beaulieu

Composition du jury

Président :	Benoît IUNG	Professeur, Université de Lorraine
Rapporteurs :	Kamal MEDJAHER	Professeur, Université de Toulouse
	Jonas SJÖBERG	Professeur, Université de Chalmers
Examinateurs :	Marie ALBISSER	Chercheuse, Institut franco-allemand de recherches de Saint-Louis
	Chetan KULKARNI	Chercheur, KBR Inc, National Aeronautics and Space Administration
Directeurs de thèse :	Hugues GARNIER	Professeur, Université de Lorraine
	Mayank Shekhar JHA	Maître de conférences, Université de Lorraine
Co-encadrant :	Farid CERBAH	Chercheur, Dassault Aviation

Université de Lorraine, CRAN, CNRS — UMR 7039

Mis en page avec la classe thesul.

System identification enabled health state prognostics using deep learning.

Application to predictive maintenance of business jet aircraft

Acknowledgments

First and foremost, I would like to thank my three PhD supervisors, Hugues Garnier, Mayank Shekhar Jha and Farid Cerbah. Thank you for encouraging and challenging me. Without your dedication to study and guide my research, to help me gain perspective and to review my works, this PhD would not have been the same. In the same way I would like to thank the many colleagues at CRAN, in the iModel team and at Polytech, with whom I have had the opportunity to talk, to teach and to collaborate on many subjects. A special thanks to Antoine Courteau, intern at CRAN from April 11 to July 15, 2022, who assisted me in developing the system identification strategy.

I would like to address my acknowledgement to my PhD committee members, Jonas Sjöberg, Kamal Medjaher, Benoit Iung, Chetan Kulkarni and Marie Albisser, for their evaluation of my PhD work, for their appreciation and interesting insights to complete and enrich the PhD contributions.

Likewise, I would like to thank the members of my comité de suivi, Romain Postoyan and Cédric Pradalier for their advice all along the PhD.

I also owe a debt of gratitude to Aurore Tranchina and Souad Boutaguermouchet, the administrative managers who helped me to submit the manuscript and organize the defense.

I would like to extend my warmest thanks to my fellow PhD students at CRAN, in particular Louis Massucci and Axelle Hego, who preceded me and gave me the benefit of their experience. A particular thank to Soha Kanso, with whom I had the pleasure of sharing my office during the last year of the PhD, for her cheerfulness and our invaluable mutual support. The writing process can sometimes be a source of doubts and discouragement, so for that I warmly thank Julien Thuillier, who offered me his support and precious experience. Thanks also to Yanis Ouakrim for proofreading my manuscript, for our mutual assistance discussions at various stages of our respective theses and good times of friendship.

Still on the subject of writing, I would like to thank the Fathers of Chéméré who generously welcomed me, offering me a propitious place to work and think. Thanks to the many intercessors I know I can count on, on Earth and in Heaven.

Throughout this PhD, I also received extremely generous support from sincere friends, among whom I would particularly like to thank Emile Kuffel, an ever-available confidant, a relevant advisor and a highly effective coach.

Last but not least, I would naturally like to express my gratitude to my parents, who instilled in me the values of hard work and abnegation needed to successfully complete these three years of work.

"Sumus quasi nanos, gigantium humeris insidentes,
ut possimus plura eis et remotiora videre,
non utique proprii visus acumine, aut eminentia corporis,
sed quia in altum subvenimur et extollimur magnitudine gigantea."

Bernard de Chartres

"Nous sommes sur un milieu vaste,
toujours incertains et flottants entre l'ignorance et la connaissance [...].
Nous brûlons du désir d'approfondir tout,
et d'édifier une tour, qui s'élève jusqu'à l'infini.
Mais tout notre édifice craque,
et la terre s'ouvre jusqu'aux abîmes."

Blaise Pascal, Les Pensées.
En l'année du quatre-centième anniversaire de sa naissance.

Contents

List of Figures

List of Tables

List of Algorithms

Acronyms

AI Artificial Intelligence.

ANN Artificial Neural Network.

AR AutoRegressive.

C-MAPSS Commercial Modular Aero-Propulsion System Simulation.

CDF Cumulative Distribution Function.

CNN Convolutional Neural Network.

DL Deep Learning.

DTW Dynamic Time Warping.

EKF Extended Kalman Filter.

EMD Empirical Mode Decomposition.

EOL End of Life.

FCL Fully Connected Layer.

FNN Feedforward Neural Network.

GAN Generative Adversarial Network.

GRU Gated Recurrent Unit.

GUI Graphical User Interface.

HI Health Index.

HPC High Pressure Compressor.

HPT High Pressure Turbine.

ICA Independent Component Analysis.

IV Instrumental Variable.

LPC Low Pressure Compressor.

LPT Low Pressure Turbine.

LSTM Long Short-Term Memory.

MGU Minimal Gated Unit.

ML Machine Learning.

MLE Maximum Likelihood Estimation.

MLP Multilayer Perceptron.

MME Method-of-Moments Estimator.

MSE Mean Squared Error.

MTTF Mean Time To Failure.

NN Neural Network.

NRMSE Normalized Root Mean Square Error.

PCA Principal Component Analysis.

PDF Probability Density Function.

PEM Prediction Error Method.

PHI Physics Health Index.

PHM Prognostics and Health Management.

PI Proportional-Integral.

PINN Physics-Informed Neural Network.

PoF Physics-of-Failure.

RIVC Refined Instrumental Variable for Continuous-time systems.

RMS Root Mean Square.

RMSE Root Mean Squared Error.

RNN Recurrent Neural Network.

RUL Remaining Useful Life.

SISO Single-Input Single-Output.

SOH State of Health.

SRIVC Simplified Refined Instrumental Variable method for Continuous-time systems.

STL Seasonal and Trend decomposition using Loess.

SVM Support Vector Machine.

TCN Temporal Convolutional Network.

TTF Time to Failure.

VHI Virtual Health Index.

Résumé en français (French summary)

1 Contexte de la thèse

L'objectif de la maintenance, dans un contexte industriel, est d'assurer la disponibilité maximale d'un système, au meilleur niveau de fonctionnement possible, et avec des coûts réduits (Lee et al., 2014). Ces dernières années, la maintenance prévisionnelle des systèmes (NF EN 13306, 2018) a fait l'objet d'une attention croissante de la part de la communauté des chercheurs en raison des perspectives d'améliorations de la fiabilité, de la sécurité et des gains de coûts associés dans de nombreux domaines, notamment l'aérospatiale, l'automobile, l'énergie et les soins de santé.

Le Prognostics and Health Management (PHM) est le domaine d'étude qui se concentre sur la définition de méthodes et d'outils permettant de concevoir des politiques de maintenance optimales pour un système spécifique, en tenant compte de ses conditions de fonctionnement et de dégradation (Kalgren et al., 2006).

Le pronostic est l'étape du cycle PHM qui se concentre sur l'évaluation de l'état de santé (State of Health (SOH)) d'un système ou d'un composant et sur la prévision de sa fin de vie (End of Life (EOL)), qui correspond au moment où le système (ou le composant) ne remplit plus sa fonction prévue (Ellefsen et al., 2019).

Deux indicateurs principaux sont généralement utilisés dans ce but:

- L'indicateur d'état de santé (Health Index (HI)) est un indicateur qui estime le véritable SOH du système étudié.
- La durée de vie résiduelle (Remaining Useful Life (RUL)) est le temps restant entre l'instant présent t et la fin de vie (EOL) du système étudié (Si et al., 2011).

Les approches de prédiction de la durée de vie résiduelle sont classiquement classées en trois grandes familles (J. Guo et al., 2019):

- Les approches basées sur des modèles physiques
- Les approches basées sur l'apprentissage à partir des données
- Les approches hybrides

Ces dernières années, la recherche s'est concentrée en grande partie sur la prédiction de la durée de vie résiduelle (RUL) à l'aide d'approches basées sur l'intelligence artificielle, et notamment sur les modèles d'apprentissage profond dont le très grand nombre de paramètres est ajusté sur la base d'une grande quantité de données d'entraînement (Si et al., 2011).

Bien qu'elles offrent des perspectives attrayantes, les méthodes basées sur l'intelligence artificielle souffrent de deux inconvénients majeurs. Tout d'abord, ces méthodes s'avèrent très consommatrices de données. En effet, les modèles neuronaux complexes nécessitent de grandes

quantités de données pour être entraînés. En outre, presque toutes les approches basées sur l'intelligence artificielle proposées dans le cadre du pronostic exploitent des données étiquetées avec la durée de vie résiduelle, réalisant une corrélation directe entre les valeurs des capteurs et les prédictions de durée de vie résiduelle, de manière supervisée. Néanmoins, dans le cadre de situations industrielles réelles, de telles données ne sont pratiquement jamais disponibles. Elles nécessitent en effet de mener des expériences longues et coûteuses, et souvent inapplicables pour des systèmes multi-composants et/ou critiques pour la sécurité des utilisateurs (comme c'est le cas en aéronautique).

Le deuxième inconvénient majeur des méthodes basées sur l'intelligence artificielle est leur manque d'exploitation des connaissances et de la physique disponibles *a priori*. Il s'agit de méthodes basées sur les données, qui excluent bien souvent certaines connaissances susceptibles d'améliorer les modèles.

2 Objectifs scientifiques et industriels

Sur la base de ce contexte et des limites associées aux approches de pronostic à base d'intelligence artificielle, deux objectifs de recherche ont été soulignés pour guider les travaux de cette thèse :

- **Objectif de recherche n°1 :** Limiter au maximum l'utilisation de données étiquetées avec la durée de vie résiduelle pour l'entraînement des modèles de pronostic à base d'intelligence artificielle.
- **Objectif de recherche 2 :** Exploiter les connaissances existantes sur le système et sur la physique de la dégradation afin de renforcer les méthodes de pronostic basées sur l'intelligence artificielle.

Les travaux présentés ici sont financés par Dassault Aviation dans le cadre d'un partenariat de recherche avec le CRAN. En tant que constructeur d'avions (militaires et civils), Dassault Aviation cherche constamment à améliorer et optimiser la maintenance de ses appareils. L'objectif de cette collaboration est donc d'évaluer le potentiel d'utilisation des méthodes basées sur l'intelligence artificielle pour le pronostic de manière générale, avec une application spécifique à un système embarqué du Falcon 6X, un avion récemment développé.

Enfin, les approches proposées dans le cadre de ce travail de doctorat doivent être testées sur les données réelles collectées par Dassault Aviation. Cela signifie que des efforts significatifs seront nécessaires pour adapter les méthodes de recherche aux problèmes pratiques de l'étude de cas, offrant ainsi un travail de recherche réellement applicable.

Deux sources de données ont été utilisées dans cette thèse :

1. Les méthodes proposées ont d'abord été appliquées à un jeu de données de référence bien connu de la communauté PHM, à savoir le jeu de données C-MAPSS. Cela permet d'assurer la reproductibilité des approches proposées en utilisant une référence pour comparer les méthodes non-supervisées explorées au cours de ce travail de thèse avec des méthodes supervisées de corrélation directe entre la durée de vie résiduelle et les données collectées sur les capteurs.
2. Dans un second temps, l'approche de pronostic proposée a été appliquée dans son intégralité au cas expérimental de Dassault Aviation, en capitalisant sur les approches précédemment testées sur C-MAPSS et en les améliorant. Ce cas expérimental démontre que les méthodes proposées sont réalistes d'un point de vue industriel, et permettent d'envisager un développement dans des situations réelles, et non simulées.

3 Approche proposée

Pour résoudre les deux problèmes de recherche identifiés ci-dessus, une approche de pronostic en plusieurs étapes est proposée. Inspirée des réseaux de neurones informés par la physique (Physics-Informed Neural Network (PINN)), elle repose sur l'intégration de connaissances de la physique dans les données d'apprentissage elles-mêmes, par le biais d'une augmentation de données orientée vers le pronostic, générant ainsi des données nominales et dégradées. Cette approche permet ensuite une extraction non supervisée de l'indicateur d'état de santé de manière réaliste d'un point de vue industriel, c'est-à-dire sans utiliser de données de dégradation. Enfin, sur la base de l'indicateur d'état de santé (HI) extrait, des prédictions de durée de vie résiduelle (RUL) sont réalisées. En outre, une procédure d'évaluation basée sur la fiabilité est proposée pour garantir la cohérence des prédictions vis-à-vis des lois de fiabilité en usage.

Les différentes étapes de l'approche générale proposée sont détaillées ci-dessous :

1. **Augmentation de données orientée vers le pronostic.**
   Tout d'abord, sur la base d'un volume réduit de données nominales collectées sur le système étudié (c'est-à-dire des données provenant d'un système en bonne santé), une double augmentation de données, orientée vers le pronostic, est effectuée. D'une part, un processus d'augmentation de données nominales basé sur l'identification de système permet de générer de nouvelles données nominales en limitant l'utilisation des données d'entraînement, tout en garantissant la cohérence des données générées par rapport au système réel. D'autre part, une stratégie d'enrichissement des données de dégradation basée sur la physique permet d'injecter une dégradation qui respecte le mécanisme physique sous-jacent de la défaillance, générant ainsi de nouvelles trajectoires sans utiliser de données de dégradation préalables. Ainsi donc, les données générées par ce double processus d'augmentation de données intègrent à la fois les connaissances physiques liées au système lui-même, et au mécanisme de dégradation. Ces nouvelles données permettent, par la suite, d'entraîner des modèles à base de réseaux de neurones profonds, d'où l'intérêt de l'intégration des connaissances physiques *a priori*.

2. **Extraction non supervisée de l'indice de santé.**
   Sur la base des données nominales (qui peuvent être à la fois des données nominales réelles collectées sur le système étudié et des données nominales supplémentaires générées par le processus d'augmentation de données nominales en utilisant l'identification du système), une structure à base de réseaux de neurones de type autoencodeur est entraînée à effectuer la reconstruction des données collectées par les capteurs. Par conséquent, l'autoencodeur apprend le comportement nominal du système étudié. Ensuite, une fois la phase d'entraînement terminée, les paramètres de l'autoencodeur sont figés et la structure est utilisée pour traiter des trajectoires complètes de vie du système contenant une dégradation progressive jusqu'à la défaillance totale. La structure ayant été entraînée uniquement sur des données nominales (c'est-à-dire lorsque le système est en bonne santé), par conséquent l'erreur de reconstruction augmente à mesure que les données issues des capteurs s'éloignent de leur distribution nominale initiale, en raison de la dégradation. Cette erreur de reconstruction est donc utilisée comme indicateur du véritable état de santé du système, c'est-à-dire comme HI. Ce processus d'extraction de l'état de santé, basé sur l'évolution de l'erreur de reconstruction d'un autoencodeur, est totalement non supervisé, c'est-à-dire qu'il n'utilise aucune donnée étiquetée. En outre, il est entraîné exclusivement sur la base de données nominales, ce qui est beaucoup plus réaliste d'un point de vue industriel.

3. **Prédiction à long terme de l'indicateur d'état de santé et de la durée de vie résiduelle.**
   Sur la base d'un échantillon initial de trajectoires d'état de santé, un modèle neuronal de prédiction est appliqué pour produire des prévisions d'état de santé à long terme, jusqu'à ce qu'une valeur seuil prédéfinie correspondant à la fin de vie soit atteinte. Cela conduit donc à une estimation récursive de la durée de vie résiduelle (RUL). Plusieurs modèles de réseaux de neurones récurrents, utilisant les cellules mémoire de type Long Short-Term Memory (LSTM) ont été testés au cours de cette thèse, notamment des modèles d'encodeur-décodeur basés sur les LSTM et des modèles profonds de type LSTM superposés. En outre, une procédure d'évaluation basée sur la fiabilité est proposée pour s'assurer que les prédictions faites sont cohérentes avec les lois de fiabilité classiques et peuvent donc être utilisées en toute confiance pour guider les décisions de maintenance.

L'extraction d'indicateur d'état de santé grâce à des autoencodeurs et la prédiction à long terme de l'indicateur d'état de santé pour déduire la durée de vie résiduelle (c'est-à-dire les étapes 2 et 3) sont testées sur le jeu de données public C-MAPSS. Ce jeu de données, publié par la NASA, contient un grand nombre de trajectoires de vie complètes de turboréacteurs obtenues grâces à un simulateur complexe. Ces données sont largement répandues dans la communauté PHM et constituent une excellent référence de comparaison des méthodes explorées avec l'état de l'art. L'approche générale proposée est également appliquée dans son intégralité au cas expérimental fourni par Dassault Aviation, à savoir le système de distribution d'air du cockpit d'un Falcon 6X, qui vise à réguler la température ambiante dans le cockpit.

4 Contributions

Suivant les deux objectifs de recherche définis dans la Section 2, plusieurs propositions ont été faites, dans le cadre d'une approche de pronostic globale (Section 3). Les différentes contributions de cette thèse sont résumées ci-après :

- L'utilisation de l'identification de systèmes reposant sur des données réduit considérablement la quantité de données d'apprentissage requise pour assurer la tâche d'augmentation de données. Par rapport aux méthodes génératives d'augmentation de données (par exemple, les réseaux de neurones de type GAN), cela permet de réaliser une augmentation nominale des données dans un contexte industriel réaliste où très peu de données mesurées sont disponibles, répondant ainsi à l'objectif de recherche n°1. Par ailleurs, en comparaison des méthodes génératives d'augmentation de données, l'utilisation de l'identification de système permet d'éviter de générer des données aberrantes, et fournit un cadre de génération beaucoup plus sûr, sans pour autant nécessiter une connaissance extensive des mécanismes physiques sous-jacents.

- L'augmentation hybride des données sous dégradation, obtenue en injectant un modèle de dégradation basé sur la physique à l'intérieur du modèle nominal identifié à l'étape précédente permet d'exploiter à la fois les connaissances disponibles sur le système et sur la physique de la défaillance (Physics-of-Failure (PoF)), afin de générer des trajectoires temporelles de dégradation complète jusqu'à la défaillance (Time to Failure (TTF)) sans utiliser aucune données de dégradation mesurées. Cette approche, qui intègre des connaissances explicites directement dans les données d'apprentissage qui sont ensuite utilisées pour entraîner des structures neuronales, fait partie de la famille des réseaux de neurones

informés par la physique (PINN) (Pan et al., 2022; Sobie et al., 2018). Elle répond donc simultanément aux objectifs de recherche n°1 et n°2.

- L'approche entièrement non supervisée pour l'extraction de l'indicateur d'état de santé (HI) à partir des données des capteurs en utilisant l'erreur de reconstruction d'un autoencodeur permet d'obtenir cet indicateur d'état de santé uniquement sur la base de données nominales (c'est-à-dire lorsque le système est en bonne santé). Cela est particulièrement réaliste d'un point de vue industriel, étant donné que dans la majeure partie des situations réelles, les données de dégradation mesurées sont peu ou pas disponibles. En outre, la prédiction à long terme de l'indicateur d'état de santé à l'aide de réseaux de neurones récurrents de type LSTM permet de déduire la durée de vie résiduelle sans utiliser de données étiquetées. Par conséquent, ces deux propositions évitent toute utilisation de données étiquetées (avec la durée de vie résiduelle) dans l'ensemble de l'approche de pronostic, ce qui permet de répondre à l'objectif de recherche n° 1.

- L'évaluation par rapport aux lois de la fiabilité qui est proposée dans cette thèse permet d'évaluer les résultats de la prédiction de la durée de vie résiduelle par rapport aux lois établies de la fiabilité. La vérification de l'hypothèse de la distribution de Weibull préalablement définie (en utilisant le test de Kolmogorov-Smirnov) garantit la validité des approches proposées utilisant l'intelligence artificielle, bien qu'elles soient de type "boîte noire".

- Enfin, une double source de données a été utilisée pour valider l'approche proposée. Les applications ont d'abord été réalisées sur l'ensemble de données de référence C-MAPSS, qui est largement utilisé dans la communauté PHM. Cela garantit la reproductibilité des résultats obtenus d'un point de vue académique. Ensuite, le cadre global de pronostic a été appliqué à un cas industriel réel en collaboration avec Dassault Aviation. Cette expérimentation confère un intérêt particulier au travail présenté en le confrontant à des problèmes industriels réalistes.

Introduction

The maintenance of systems, that is, ensuring that they remain in operational condition to perform their intended function, is a crucial issue in the industry. In particular, this represents an important opportunity for cost savings (Mobley, 2002).

The will to improve customer experience and safety, the need to cope with increasing competition among industries, the prospect of significantly reducing costs, and, in recent years, the awareness of sustainability and energetic issues (Iung and Levrat, 2014), have conducted this domain to receive increased attention in the last decades. Therefore, considerable efforts have been made to transition from corrective maintenance, where equipment/subsystem is repaired after a failure has occurred, to increasingly proactive maintenance. Such predictive maintenance relies on the forecast of a process behavior, based on current health state assessment and known characteristics of the significant parameters of the degradation (NF EN 13306, 2018).

In this context, Artificial Intelligence (AI)-based approaches, which have grown very rapidly over recent years, represent a promising prospect for the future of predictive maintenance strategies. This is coupled with the development of new data collection and storage capabilities, as well as the massive deployment of sensors on industrial systems leading to improvements in the monitoring capabilities. Nevertheless, although these AI-based approaches seem promising in different areas of research, they suffer from several limitations in an industrial context (Chaoub et al., 2022).

In particular, the limited availability of degradation data and the value of incorporating existing physics-based knowledge within the learning framework are the principal issues studied in this thesis work. The aim of the work presented in this PhD is thus to propose an approach based on the modern AI-based methods which takes into account realistic constraints associated with industrial systems.

This work was carried out in collaboration with Dassault Aviation, an aircraft manufacturer, enabling the proposed approach to be tested on a real industrial case study. This is therefore an applied research work, in which a significant amount of effort has been devoted to the practical implementation of the proposed approach.

The first major contribution of this PhD concerns the use of data-driven system identification methods, coupled with the injection of physics-based degradation model, for data augmentation purposes to compensate for the lack of available data under degradation. In addition, this allows *a priori* knowledge about the system to be integrated within the data augmentation process, thus influencing all subsequent prediction structures using this augmented data.

Another important contribution of the PhD concerns the unsupervised extraction of the Health Index (HI), in an indirect approach, through the use of autoencoder reconstruction error. Such an unsupervised approach avoids the dependency on data under degradation, and is therefore much more appropriate for real industry applications.

The application of these AI-based methods to a practical industrial system enables their performance to be assessed in a real-life situation, which is not common in the existing literature

within the community. In addition, a reliability-based assessment is proposed at the end of the overall approach, with the aim of ensuring the consistency of the results obtained with the established laws of reliability.

The overall context of prognostics, in particular the challenges and the most promising research opportunities, is presented in Chapter 1. The research challenges of this PhD are also outlined, and the overall proposed approach which will be applied to the Dassault Aviation use case is introduced.

Chapter 2 focuses on the problem of data augmentation. In particular, the solution adopted here relies on the data-driven system identification and the injection of a physics-based degradation model. Chapter 3 presents the second stage of the proposed approach, which is the unsupervised extraction of HI, using AI-based methods (in this case, autoencoder structures). Two initial approaches are presented and applied to the C-MAPSS academic dataset. Then a new indirect approach, based on the exploitation of the reconstruction error, is applied to the Dassault Aviation data. Finally, the third stage of the proposed approach, which consists of long-term HI forecasting in order to predict the RUL, is described in Chapter 4. As in the previous chapter, different methods are explored through two first studies applied to the C-MAPSS dataset, then the experimental use case of Dassault Aviation is addressed and discussed. In addition, a new reliability-based assessment is presented and applied, demonstrating the credibility of the AI-based methods used. Finally, in the last chapter, general conclusions are drawn and the remaining perspectives that can serve as a basis for future work on the subject are mentioned.

List of publications

Journal papers

- Hervé de Beaulieu, M., Jha, M. S., Garnier, H., & Cerbah, F. (2023). End-to-end remaining useful life prediction using physics-informed data augmentation. *Currently in submission process to Reliability Engineering & System Safety.*
  This publication covers the entire prognostics approach applied to the Dassault Aviation use case, including data augmentation (Chapter 2), unsupervised HI extraction using reconstruction error (Chapter 3), and RUL prediction (Chapter 4).

International conferences

- Hervé de Beaulieu, M., Jha, M. S., Garnier, H., & Cerbah, F. (2022b). Long range health index estimation based unsupervised RUL prediction using encoder-decoders. *11th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes.*
  This publication corresponds to C-MAPSS Application 1, mainly presented in Sections 3.2.1 and 4.4.1. It describes the two-step approach of direct HI extraction using vanilla autoencoder and RUL prediction with threshold overshooting, together with its application to the C-MAPSS dataset.

- Hervé de Beaulieu, M., Jha, M. S., Garnier, H., & Cerbah, F. (2022c). Unsupervised prognostics based on deep virtual health index prediction. *7th PHM Society European Conference*, 193–199.
  This publication corresponds to C-MAPSS Application 2, mainly presented in Sections 3.2.1 and 4.4.1. It describes the two-step approach of direct HI extraction with CNN-based autoencoder and RUL prediction with EOL pattern recognition using DTW algorithm, together with its application to the C-MAPSS dataset.

Posters

- Hervé de Beaulieu, M., Jha, M. S., Garnier, H., & Cerbah, F. (2021). Data-driven Health Index prognostics using Deep Learning (Poster). *Annual PhD students conference CRAN.*

- Hervé de Beaulieu, M., Jha, M. S., Garnier, H., & Cerbah, F. (2022a). Data-driven system identification and unsupervised health index prognostics using deep learning. application to predictive maintenance of business aircraft (poster). *HCERES Day CRAN (Haut Conseil de l'Evaluation de la Recherche et de l'Enseignement Supérieur).*

Chapter 1

The context of prognostics

The aim of this first chapter is to present a general overview of the state of the art regarding the current prognostics approaches. More specifically, the focus is on prognostics methods using artificial intelligence, which is the most widely used approach today, outlining both their advantages and their limitations. On the basis of this general presentation, some key research challenges have been identified to orient the present PhD work, while integrating the specific industrial issues associated with a collaboration with Dassault Aviation. The remainder of this chapter is arranged as follows. PHM fundamentals are presented in Section 1.1. Data-driven prognostics approaches, with their benefits and limitations, are studied in Section 1.2, and their main directions of improvement currently being explored by researchers are introduced. The case study resulting from the partnership with Dassault Aviation, together with its specific features, is presented in Section 1.3. Finally, an overview of the general solution to this specific experimental use case is given in Section 1.4.

Contents

1.1 Prognostics and Health Management background

1.1.1 Prognostics and Health Management overview

The purpose of maintenance, in an industrial context, is to ensure maximum availability of a system, at the highest possible level of operation, and with reduced costs (Lee et al., 2014). In recent years, Prognostics and Health Management (PHM) of systems has received increasing attention from the research community due to its potential for improving the reliability, safety and maintenance of complex systems in various domains including aerospace, automotive, energy and healthcare.

PHM is the field of study that focuses on providing methods and tools to design optimal maintenance policies for a specific asset in consideration of its specific operating and degradation conditions (Kalgren et al., 2006). PHM can be regarded as a holistic framework integrating fault detection, diagnostics, prognostics, maintenance and logistics decisions (Gouriveau et al., 2017).

The different steps that are typically followed by PHM are represented in Figure 1.1 (Atamuradov et al., 2017).

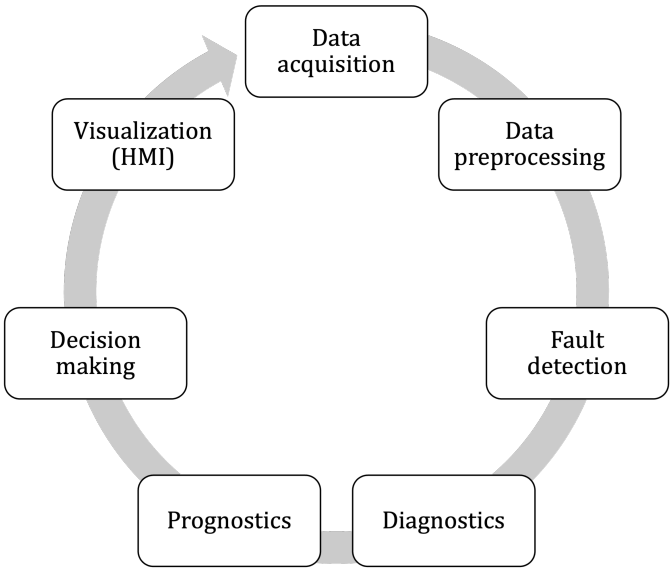


Figure 1.1: The PHM process (adapted from Jardine et al., 2006).

1.1.2 Prognostics methodology and metrics

Prognostics focuses on assessing the State of Health (SOH) of a system or a component and predicting its End of Life (EOL), which is the moment at which a system or a component will no longer perform its intended function (Ellefsen et al., 2019). Two main indicators are then used for this purpose.

The first one is the Health Index (HI).

Definition 1.1.1 (HI) *The Health Index (HI) is an indicator that estimates the true SOH of the system studied.*

HI is crucial as it helps describing the degradation process as accurately as possible. It is typically constructed using inherent information within condition monitoring signals (Lei et al., 2018). HI can be distinguished in two categories, namely Physics Health Index (PHI) and Virtual Health Index (VHI) (C. Hu et al., 2012). A PHI is related to the Physics-of-Failure (PoF) and is typically extracted from sensor signals using signal processing methods or statistical methods. For instance, the Root Mean Square (RMS) of vibration signals has been widely used to built PHI in the field of bearings (N. Li et al., 2015, H. Li and Wang, 2013). On the other hand, a VHI is usually built by fusing multiple sensor signals or multiple PHI together. Therefore, a VHI loses its physics meaning and simply presents a virtual description of the degradation trend of the system (Lei et al., 2018).

Secondly, the Remaining Useful Life (RUL) is defined as follows:

Definition 1.1.2 (RUL) *The Remaining Useful Life (RUL) is the time remaining between the current instant t and the End of Life (EOL) of the system under study (Si et al., 2011).*

RUL can be written as follows:

$$\mathrm{RUL}(t) = t_{EOL} - t \tag{1.1}$$

where t_{EOL} is the EOL instant and t is the current time.

The EOL is defined as:

Definition 1.1.3 (EOL) *The End of Life (EOL) is the time in the life of a system when it is no longer able to perform its intended function.*

In most of the studies, including the present one, the objective of prognostics is resumed to predict the RUL, given the current machine condition and its past operation profile (Jouin et al., 2013, Tobon-Mejia et al., 2012).

Prognostics can be performed at component or sub-component level. It concerns the prediction of the time progression of a specific failure mode from its incipience to the time of component failure (EOL) (Sikorska et al., 2011). HI can be used as an intermediate variable in order to predict RUL (Lei, 2016), as illustrated in Figure 1.2.

Traditionally, any estimate of RUL is accompanied by a confidence interval. Such confidence limits are critical in prediction as degradation is a stochastic process (Saxena et al., 2010). Risk-dependent maintenance decisions heavily rely on such uncertainty quantification (Vachtsevanos, 2006, Fallahi et al., 2022).

RUL prediction approaches are usually categorized into three main families: physics-based approaches, data-driven approaches and hybrid approaches (J. Guo et al., 2019).

Physics model-based approaches attempt to describe the degradation process of the system being studied by constructing mathematical models based on existing physics laws and on the available knowledge or observation of the degradation mechanism (Cubillo et al., 2016). Such approaches are calling for availability of expert knowledge in the different scientific fields related to the degradation of the observed system (mechanics, thermodynamics, fluid physics, etc.) as well as an in-depth expertise about the system itself. However, for more complex systems, it may be difficult to understand the physics of damage in a comprehensive way, therefore restricting applicability of these strictly physics-based methods (Lei et al., 2018).

Data-driven approaches consist of deriving models directly from historical (observed) data, in which the characteristics of the degradation process are identified (D. An et al., 2015).

Numerous data-driven approaches have been developed over the years, leading to a large panel of statistical and Machine Learning (ML) approaches, including Bayesian network, Hidden Markov Model, Gaussian process regression, Artificial Neural Network (ANN), Fuzzy Logic, etc.

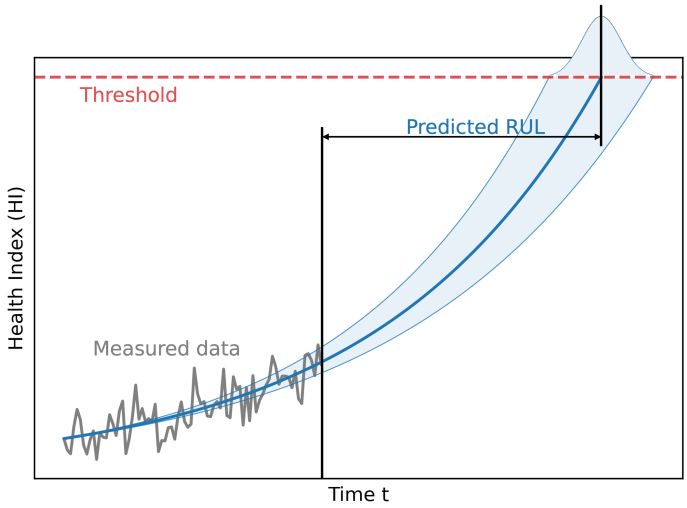


Figure 1.2: The prognostics step of PHM and its inherent uncertainty. The RUL prediction is accomplished at the moment where the failure is detected. The uncertainty on the prediction ends up with the resulting distribution of possible RUL.

In particular, in recent years, research has focused in large part on the prediction of RUL using AI-based approaches, which rely on highly complex computation models with coefficients that are adjusted on the basis of a large number of observations (Si et al., 2011). The advancement of AI opens extensive perspectives in terms of model complexity and flexibility, especially for highly complex systems for which physics or traditional statistical models turn out to be limited (Fink et al., 2020).

Finally, hybrid approaches are the integration of both data-driven and physics-based prognostics. The objective is therefore to leverage the strength of both aforementioned approaches to improve the performance and range of application of RUL prediction (Liao and Köttig, 2014). It is in this latter category, which offers more realistic prospects for application from an industrial point of view, that this study will focus.

1.2 Data-driven prognostics

1.2.1 Overview of Artificial Intelligence-based approaches

In recent years, AI-based prognostics approaches have become increasingly popular, in particular due to the increase in data collection and storage capacity, as well as advances in computing power (Nguyen et al., 2023).

An ANN consists of an input layer of neurons (also called units), several intermediate layers of neurons named hidden layers, and a final output layer. The objective of the ANN structure is to learn the unknown function that links the output layer to the input layer by adjusting the weights between the neurons, on the basis of multiple observations of input-output pairs. When inputs are processed only in the forward direction (that is, from the input neurons, through the hidden layer(s) to the output neurons, without any internal cycles or loops), the ANN is called a Feedforward Neural Network (FNN). An ANN with limited number of layers and neurons is not suitable for complex models. Thereby, Deep Learning (DL) is a field of study where complex

neural networks, integrating multiple hierarchical layers, are used to transform the input data into more abstract representations, thus offering better modeling abilities.

The most common DL structures used in prognostics are briefly introduced hereafter.

Despite their low computational capacity, one-hidden-layer FNN, have first been applied by S.-j. Wu et al., 2007 and Saon, Hiyama, et al., 2010 to perform bearing RUL prediction. A similar but deeper architecture, containing four hidden layers, was implemented by L. Wang et al., 2015, still trying to forecast RUL for bearings. In addition, many techniques of signal processing were used to prepare the input features passed through the FNN, such as RMS, kurtosis (Qui and Lee, 2004), wavelet and Fourier transforms (Z. Zhang et al., 2013), in order to improve the RUL prediction performance.

Convolutional Neural Network (CNN) is a class of neural structure very popular in visual imagery and machine vision applications (LeCun, Bengio, et al., 1995). CNNs are specifically suitable for processing data in grid form (Forsyth et al., 1999), by applying multiple convolution operations (Goodfellow et al., 2016) in layers called "convolutional layers". Such a structure is particularly valuable to extract hidden features that will then allow for recognition of warning signs of failure. Pooling layers can be inserted between convolutional layers in order to perform sub-sampling, thus offering better generalization properties by reducing the resolution of the dimensionality of intermediate representations, as well as the sensitivity to shifts and distortions (A. Zhang et al., 2021). The hidden features extracted through this process are finally fed to a FNN in order to end with a RUL prediction. A typical architecture of CNN structure combined with FNN applied to prognostics proposed by X. Li et al., 2018 can be found in Figure 1.3.

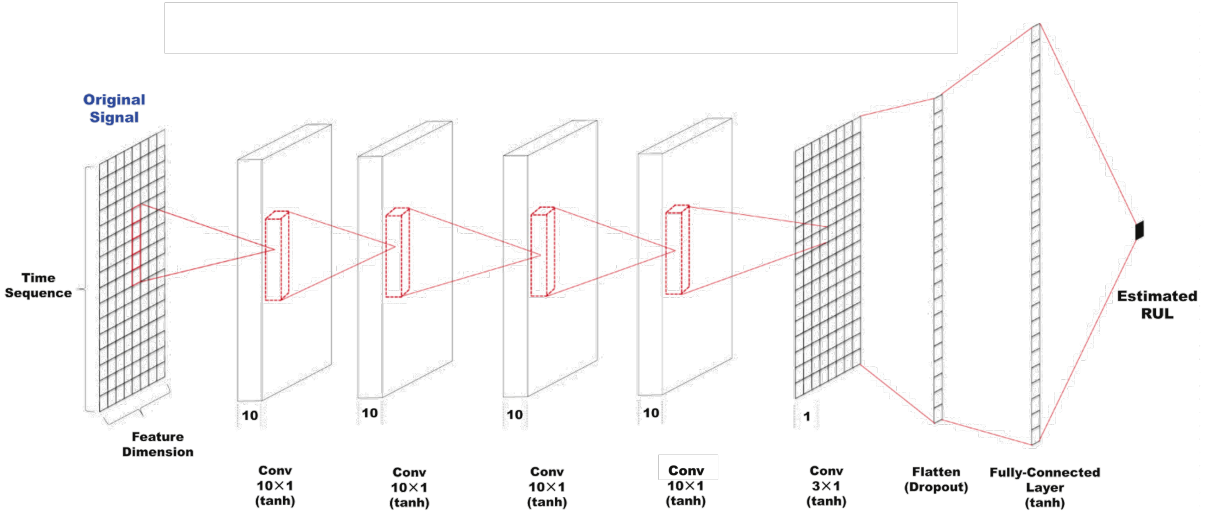


Figure 1.3: DL architecture combining CNN with FNN for RUL prediction (from X. Li et al., 2018).

A first architecture using CNNs has been proposed by Babu et al., 2016, with two convolutional layers along with two average pooling layers followed by one fully connected layer. Applied to a multi-sensor turbofan simulation dataset (Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) dataset), the architecture gave experimental results which showed a significant improvement compared to other regression algorithms in the state of the art, including ANN and Support Vector Machine (SVM). The CNN architecture was able to capture the patterns of the sensor signals at different time scales to deliver a more accurate RUL prediction.

X. Li et al., 2018 confirmed the interest of CNN architecture for RUL prediction, by proposing a deeper structure, thus demonstrating that larger convolution kernel size and increased number of convolution layers lead to a finer vision of the patterns of the sensor signals.

As a variant of CNN for sequence modeling tasks, Bai et al., 2018 introduced Temporal Convolutional Network (TCN). Whereas CNNs can be seen as being temporally naive, in TCN the convolutions are said to be "causal", i.e. the i^{th} element of an output sequence is only obtained by convolutions performed on previous elements $(i-1)^{th}, (i-2)^{th}, ..., (i-n)^{th}$, of the input sequence. TCN architecture was firstly applied to C-MAPSS dataset by Wenqiang et al., 2019, producing results that outperform other DL structures. A very similar application case of TCN was published by C. Liu et al., 2019 on bearing RUL prediction, demonstrating faster learning combined with a better accuracy in prediction when compared to other DL methods. Finally, another application was successfully performed on Lithium-Ion battery prognostics (D. Zhou et al., 2020).

Recurrent Neural Network (RNN) are another way of processing the sequential data that constitutes the bulk of sensor data. It can be thought of as multiple recurrent standard cells whose states are affected by both past states and current input. Figure 1.4 gives an unrolled overview of an RNN.

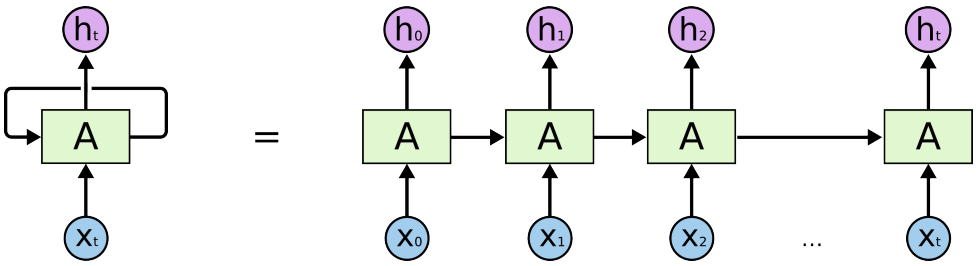


Figure 1.4: An unrolled Recurrent Neural Network (RNN) (from Olah, 2015).

An improved type of recurrent cell called Long Short-Term Memory (LSTM) has been proposed by Hochreiter and Schmidhuber, 1997, significantly improving the learning process (notably solving the so-called "long-term dependencies problem" highlighted by Bengio et al., 1994). The exact operation of this type of cell will be described in more detail in Chapter 4. Many variants have also been developed, such as peephole LSTM by Gers et al., 1999, Minimal Gated Unit (MGU) by G.-B. Zhou et al., 2016, or Gated Recurrent Unit (GRU) by Cho et al., 2014. In addition, complexifications of the architecture of LSTM-based neural networks have been proposed. For instance, stacked-LSTM (also called deep-LSTM) structures proposed by Graves et al., 2007 increases the depth of the structure. Bidirectional LSTM proposed by Graves et al., 2005 consists of training the network in both time directions simultaneously, through the use of two separate hidden layers, namely a forward layer and a backward layer, thus offering a wider use of the time context.

The first results of LSTMs for RUL prediction were proposed by Yuan et al., 2016 with one single LSTM layer, applied to C-MAPSS dataset. Y. Wu et al., 2018 proposed a detailed comparison of several recurrent architectures, demonstrating the superiority of LSTM cell. Different architectures of stacked-LSTM were applied to C-MAPSS dataset by Zheng et al., 2017 and J. Wang et al., 2018. A similar architecture has been tested on Lithium-Ion battery for RUL prediction by Y. Liu et al., 2017. Finally, an innovating semi-supervised architecture embedding LSTMs has been proposed by Ellefsen et al., 2019.

Finally, it is worth noting that all sorts of combinations of the types of neural network mentioned above can be envisaged, thus increasing model complexity. For instance, an architecture that combines a CNN with a stacked-bidirectional and unidirectional LSTM was proposed by Q. An et al., 2020.

The attention mechanism, invented by Bahdanau et al., 2014 improves the processing of long time series for sequence-to-sequence tasks (Sutskever et al., 2014). A so-called "context vector" creates links between the output and the whole input sequence, thus avoiding long-term forgetting issue. Applied to C-MAPSS dataset in conjunction with LSTMs by da Costa et al., 2019, attention mechanism demonstrated significant improvement for RUL prediction under multiple operating conditions. Such attention mechanism have also been implemented by H. Zhang et al., 2020 in rotary machine prognostics, by Y. Chen et al., 2020 in bearing prognostics, by Xiang et al., 2020 in gear RUL prediction, and by Z. Chen et al., 2020 on C-MAPSS dataset.

Therefore, through multiple structural improvements, AI-based approaches have demonstrated their ability to offer increasingly complex computational models for predicting RUL, across a broad range of application domains.

1.2.2 Limitations of Artificial Intelligence-based approaches

Although they offer attractive prospects, AI-based methods suffer from two major drawbacks.

The first drawback of AI-based methods is that they turn out to be very data-consuming. Indeed, complex neural models require huge amounts of data to be trained on. Furthermore, almost all the AI-based approaches mentioned in Section 1.2.1 leverage RUL-labeled data to perform direct mapping between sensor values and RUL in a supervised manner. However, in most of the real-life applications, such labeled data are hardly available (Chaoub et al., 2022). Indeed, obtaining RUL-labeled data requires conducting experiments (accelerated degradation tests) until the EOL of the system is reached, which is time consuming, costly and often unrealistic in face of safety critical industrial systems. As a result, as most of the AI-based existing approaches for prognostics rely heavily on the availability of labeled ground truth RUL data, the usefulness of such approaches remains limited in the absence of such a ground truth RUL in real use cases. This is identified as the research challenge 1.

Research challenge 1: RUL-labeled data limitation
Limiting reliance on measured RUL-labeled data for model training.

The second major drawback of AI-based methods is their lack of exploitation of *a priori* knowledge and physics. These are data-driven, black-box methods that exclude certain knowledge that could improve the models. Moreover, they do not allow any physical understanding of the predictions performed. Finally, since these approaches fall outside of the traditional deterministic framework as well as the usual statistical approaches, It is more difficult to assess the credibility of the predictions made. For these reasons, such methods are hardly applicable to real industrial cases where safety and security issues require available knowledge to be incorporated into the models. This limit is formalized in the current study as research challenge 2.

Research challenge 2: *A priori* knowledge and physics integration
Leveraging the existing system knowledge and physics of degradation in order to strengthen the AI-based prognostics.

1.2.3 Possible avenues to address the limitations of Artificial Intelligence-based approaches

To deal with the first two challenges mentioned above, several machine learning based approaches have emerged recently. The two most commonly encountered approaches are briefly introduced hereafter.

Data augmentation

Data augmentation consists of using different techniques to generate more training samples from existing ones in order to improve the performance of a prediction made by a neural network. It is a very promising domain that has already demonstrated its efficiency in many machine learning fields, especially in image processing (Shorten and Khoshgoftaar, 2019). Data augmentation has recently been comprehensively surveyed for time series by Iwana and Uchida, 2021a, and more specifically for prognostics by Gay et al., 2022. In addition to compensating for the lack of historical data during the training phase, data augmentation can also improve model robustness to variability (Fields et al., 2019). Data augmentation transformations are traditionally classified in four families (Gay, 2023):

- Random transformations
- Pattern mixing
- Decompositions
- Generative models

Random transformations is a collection of techniques borrowed from image data augmentation. For time series, it consists of modifying the shape of an original signal using mathematical function involving random factors. It includes for instance jittering (which consists in adding noise to the signal, as presented in Um et al., 2017), scaling (which is a multiplication of the signal by some random factor), time warping (which is the process of compressing or extending the time axis of the signal as implemented in Rashid and Louis, 2019), and various other transformations. However, for prognostics applications, the input data is time-ordered, which means that the order of the samples must absolutely be preserved (e.g. time permutation is not relevant).

Pattern mixing combines one or more patterns to generate new ones. This can be achieved for instance by making a weighted average of two existing trajectories, often referred to as interpolation (Sawicki and Zieliński, 2020). New trajectories can be generated in the same way using deviation from the mean (Yeomans et al., 2019). Guided warping (Iwana and Uchida, 2021b) is another variant of pattern mixing where several time series are mixed by warping the features of a sample pattern, matching the time steps of a reference pattern.

Decomposition methods typically decompose time series signals by extracting characteristics or underlying patterns. Then, these features can either be used independently, recombined, or perturbed in order to generate new data samples. These methods include Empirical Mode Decomposition (EMD) (N. E. Huang et al., 1998), Independent Component Analysis (ICA) (Eltoft, 2002), Seasonal and Trend decomposition using Loess (STL) (Bergmeir et al., 2016) and many others.

Finally, generative models are based on the training of deep neural networks to produce new samples. An increasingly popular approach is Generative Adversarial Network (GAN). A

GAN is a generative model that consists of two neural network models: a "generator" and a "discriminator" (Creswell et al., 2018). Such a model is efficient to create new realistic data samples on demand (K. Wang et al., 2017). It can therefore be used to augment the amount of data available in order to help training a RUL prediction algorithm for instance. Special architectures of GAN, using notably recurrent neural networks such as LSTM, have proven to be effective for generating realistic time series data (Yoon et al., 2019). Yet, such approaches are black-boxes, i.e. the models that govern the generated dynamics are unknown, and usually do not consider the physical plausibility of the generated data. Moreover, GANs require a significant amount of training data to be powerful, making them poorly suitable for data augmentation based on small initial datasets.

The first three families (random transformations, pattern mixing and decompositions) are convenient to use with small datasets but often prove to be limited in their ability of generating rich new data samples. On the other hand, GANs are a very powerful and promising tool, but they generally require a large amount of data for training and, as black-box models, they offer poor transparency and few guarantees of consistency with regard to the laws of physics. However, this conclusion should be tempered by a very recent study (Xiong et al., 2023) that proposes a controlled physics-informed generative adversarial network (referred to as "CPI-GAN"). This novel framework could generate synthetic degradation trajectories that are physically interpretable, thus offering an answer to the usual problems of transparency and physical consistency of GANs, but still keeping the problem of the required volume of degradation data in training.

Physics-Informed Neural Network (PINN)

Physics-Informed Neural Network (PINN), which belong to a hybrid class of neural networks combining machine learning and physical laws can facilitate learning algorithms to converge to the optimal solution, offer better generalization, and improve consideration of the underlying physics. In recent years, PINNs have gained significant attention due to their ability to learn from limited data while respecting the fundamental principles of physics (Raissi et al., 2019). Prior knowledge is incorporated in the data-driven model, making it possible to improve the performance of the model in the absence of some data and/or improving interpretability. When the knowledge relies more on some expertise about the system than on physics law, such an approach is often referred to as "Knowledge-Informed Neural Networks". The integration of knowledge can be divided into three primary ways (Xu et al., 2022, Von Rueden et al., 2021): inside training data, using hypothesis set, and intervening in the learning process.

- The first way is to integrate explicit knowledge into the training data through simulations (Sobie et al., 2018). These simulations are created using prior knowledge, and can then be used to train a model that will assimilate this knowledge contained inside of the simulations. Such an approach can be referred to as physics-based data augmentation (Pan et al., 2022).

- The second approach is to constrain the search space of the machine-learning model by delimiting it with a predefined hypothesis set based on conservative laws. This ensures that the results of the AI-based model will adhere to specific physical laws selected beforehand. These conservative laws can be embedded as linear equations in the neural network through weighted connections between the input layer and the first hidden layer, similar to a "pre-neural-network" concept (Lu et al., 2017, Ma et al., 2023). It is also possible to integrate physical laws directly into a standard neural network cell. Such a hybrid model has been proposed by Yucesan and Viana, 2022, creating a custom-LSTM cell embedding reduced-order physics models for grease damage prediction evolution in wind turbine bearings. This

approach proved to be capable of providing accurate prognostics with a low amount of data used for training.

- The last and most extensively used approach to integrate knowledge into machine learning approaches is to act in the learning process. The most common approach is to add one or more external physics-based (or knowledge-based) losses to the traditional loss function of the learning algorithm (R. Guo et al., 2023). The additional losses act as regularizing constraints during the training, guiding the network to a model that respects the desired knowledge/physics. Such a loss has notably been implemented in PHM applications using Weibull-law (von Hahn and Mechefske, 2022). Wen et al., 2023 proposed a combination of the second and third approaches presented here, in the field of Li-Ion battery degradation. Various empirical or physical dynamic models and surrogate neural networks are fused in order to establish a complex dynamic model, thus providing multiple loss functions that can be adaptively balanced to train the PINN architecture.

To conclude, such physics-induced approaches are an excellent avenue of investigation to address both research challenges 2 and 1, improving the performance of the learning algorithms, helping in reducing the amount of required training data and offering greater recognition and understanding of the underlying physics of the process (Fink et al., 2020).

Hybrid Prognostics

Hybrid approaches are mostly referred to as the integration of both data-driven and physics-based approaches in prognostics methods Pecht, 2010, thus encompassing a wide variety of studies. The intuitive purpose behind this is to leverage the strengths of both data-driven and physics-based prognostics approaches to improve prediction performance and robustness J. Liu et al., 2012. The aim of hybrid models is to combine the computational and representational power of data-driven approaches with the physical consistency and high fidelity of physics-based approaches J. Guo et al., 2019.

The conceptual framework of hybrid approaches was suggested by Medjaher and Zerhouni, 2013a and then applied to a mechatronic system (Medjaher and Zerhouni, 2013b). It essentially consists of building a physics-based model of the nominal system and its degradation process (offline phase), then using these two models to assess the current SOH of the system and predict its RUL by estimating the physics-based model parameters using data-driven approaches, based on new information collected by sensors over the life of the system (online phase). Within this framework, several Bayesian estimation methods have been employed for the online phase, such as Extended Kalman Filter (EKF) (Chelouati et al., 2021, Kanso et al., 2022) or Particle filter (M. J. Daigle and Goebel, 2012, M. J. Daigle and Goebel, 2011, Jha, Bressel, et al., 2016, and Jha, Dauphin-Tanguy, et al., 2016). In this way, various types of filters have been compared by M. Daigle et al., 2012. Deep learning models have also been implemented, for instance by Nascimento et al., 2021, for Lithium-Ion battery prognostics.

With the emergence of hybrid approaches, comprehensive reviews of hybrid prognostics methods have been conducted (Rezamand et al., 2020, Liao and Köttig, 2014, Meng and Li, 2019), proposing different classifications of hybrid approaches. Nevertheless, this is a vast field of research that is still evolving. It should be noted that, although hybrid prognostics approaches have been distinguished from PINNs, the latter can very well be understood as hybrid approaches since they use models and/or physical knowledge to improve the performance of neural networks. The following two main classes can be distinguished, without necessarily encompassing all the work published in recent years (Javed et al., 2017).

- **Series approaches** consist of a sequential arrangement of the models. Data-driven or physics-based models serve as *a priori* for each other, both orders being possible.
- **Parallel approaches** refer to cases where the data-driven and physics-based models are used simultaneously, so that their results are fused.

In series approaches, a physics-based model can be first used to extract features that will then be passed on to the data-driven model such as in P. Wang et al., 2020. In this hybrid RUL prediction approach for wind turbine bearings, features are first extracted from raw data using traditional time-domain and frequency-domain analysis techniques before being selected (Principal Component Analysis (PCA)) and fused to be used in the RUL prediction model. Chao et al., 2019 even proposed a hybrid sequential approach in which unobserved process variables are inferred by a physics-based performance model in order to enhance the input space of a data-driven diagnostics model.

A physics-based model can also be first used to model the Physics-of-Failure (PoF), followed by a data-driven model in charge of predicting the RUL. In this way, Khorasgani et al., 2013 first modeled the PoF for electrolytic capacitors, then used a particle filter approach to derive the dynamic form of the degradation model and estimate the current SOH, leading to more accurate estimation of the RUL of the electrolytic capacitors.

In the reverse order, a data-driven model can be first used to infer a prognostics model, followed by a classical prediction model. Such an approach has been proposed by B. Wang et al., 2018 for bearing prognostics. In this work, degradation data of bearings are sparsely represented using relevance vector machine regressions, and then a degradation curve identification is performed using exponential degradation models to ensure RUL prediction.

Finally, hybrid approaches involving complex sequences of multiple stages (anomaly detection, fault isolation, failure mode identification, prognostics, etc.) for which either physics-based or data-driven models are applied, depending on which is most appropriate in each case have been proposed by Kumar et al., 2008 and Cheng and Pecht, 2009 for electronic products prognostics.

On the other hand, the parallel combination approaches consists of training several models of RUL prediction together, and then refining the obtained results using an optimization algorithm. Such a parallel approach has been applied to bearing prognostics by Du et al., 2012. Hanachi et al., 2019 also proposed a hybrid parallel approach that fuses together the prediction results of an empirical wear-time model with a measurement-based inference model for tool wear prediction.

The approach proposed in this PhD certainly belongs to the hybrid method family, since it is divided into multiple stages using both data-driven methods (in particular neural networks) and physics-based methods. This approach is also related to PINNs as it involves physics-based data augmentation, therefore integrating explicit knowledge into the training data (corresponding to the first way described in Section 1.2.3).

1.3 Industrial context and objectives

1.3.1 Dassault Aviation experimental case study

The work presented here is supported by Dassault Aviation as part of a research partnership with CRAN. As a manufacturer of aircraft (both military and civil), Dassault Aviation is constantly striving to improve and optimize aircraft maintenance. The aim of this collaboration is therefore to research and evaluate the potential use of AI-based methods for prognostics on a general level, with a specific application to a system used on commercial aircraft, namely the Falcon 6X depicted in Figure 1.5. This aircraft is equipped with a new-generation on-board computer, that

provides extensive observability of the dynamic behavior of all aircraft systems, resulting in the availability of full flight multivariate time series.

Figure 1.5: The Falcon 6X, application system for the study.

The real-life industrial case in which Dassault Aviation is engaged meets the two research challenges presented above.

First of all, business jets generally fly less frequently than airliners. Moreover, the aircraft studied here is a recent model, that has flown little up to the present time, and only under test conditions, for development purposes, not in normal commercial use. Therefore, the amount of in-flight data collected on the aircraft is limited. In addition, the data collected are test data on which no degradation representative of the wear of the aircraft is yet apparent. This means firstly that the data is unlabeled (i.e. no RUL associated), and secondly that it is not possible to train supervised models as presented in Section 1.2.1 since there are no Time to Failure (TTF) trajectories. This is therefore a typical case of the research challenge 1.

Moreover, aeronautical systems are safety-critical, since the lives of passengers are at stake. For this reason, the AI-based prognostics strategies must exploit and incorporate all available *a priori* knowledge, as well as existing physics, in order to offer the most reliable and consistent results. This corresponds exactly to the research challenge 2.

Last but not least, the proposed approaches developed in the course of this PhD work must be tested on the real data collected by Dassault Aviation. This means that significant efforts will be required to adapt the research methods to the practical problems of the case study, thereby providing a truly applicable piece of research.

1.3.2 Problem constraints and data sources

As indicated in Section 1.3.1, the overall research work is intended to be applied to the Dassault Aviation use case. However, in the first part of the PhD, measured in-flight data from the Falcon 6X were not yet available, nor was a simulator. Then, in the second part of the PhD, some nominal data became available (but only nominal, i.e. when the system is healthy).

To compensate for the limited availability of nominal data and the absence of degradation data, a comprehensive three-stage approach has been designed. Firstly, data augmentation approach based on system identification and injection of a physics-based degradation model is proposed. Secondly, the HI is extracted from the raw data using an unsupervised approach. Thirdly, the RUL is predicted from the HI. This overall method will be introduced in Section

1.4.2.

However, since no data was yet available in the first part of the PhD, certain stages of the overall strategy were first tested on a public dataset, while awaiting experimental data from Dassault.

As a result, two sources of data were used in this PhD:

1. For stages 2 and 3 of the overall approach, the proposed methods were first applied to a public academic dataset that is well known to the PHM community, namely C-MAPSS (presented in Appendix A).

2. Once the data from the Falcon 6X aircraft became available, the global approach has been applied in its entirety to the Dassault Aviation experimental case, capitalizing on the approaches previously explored in first part of the PhD and developing them further.

In the end, this situation leads to the use of two different sources of data: one commonly used in the PHM community, and the other enabling approaches to be tested under real conditions. This therefore helps to consolidate the proposed work.

For the sake of clarity, the organization of this thesis manuscript follows the logical structure of the proposed overall approach (i.e. data augmentation (Chapter 2), then HI extraction (Chapter 3), and finally RUL prediction (Chapter 4)). The application to the Dassault Aviation experimental case is systematically presented in each of the chapters. In addition, for Chapters 3 and 4, it is preceded by an initial application to the C-MAPSS academic dataset.

1.4 Conclusion

1.4.1 Industrial and research challenges

Various RUL prediction approaches were discussed in this chapter. It turns out that methods based on physical models are limited for complex systems for which the physics is not extensively known. On the other hand, AI-based models, while powerful for dealing with multi-dimensional problems, are black-box approaches that are generally not transparent and and do not exploit *a priori* physical and expert knowledge of the system, raising safety concerns for critical systems. This is research challenge 2.

In addition, the vast majority of existing approaches in the literature require large quantities of labeled data, which will be processed in a supervised way in order to perform a direct mapping between sensor data and RUL. This situation certainly works with simulation datasets like C-MAPSS, but is not realistic for industrial applications, where obtaining such data requires long, costly experiments that are often unfeasible for safety-critical systems. While advanced data augmentation methods such as GANs can partly compensate for the lack of data, it turns out that they are also black-box methods and therefore fall back on the first scientific problem. It is essential to be able to control the physical consistency of artificially generated data so as to be sure of the quality of the data subsequently supplied to prognostics methods. This is research challenge 1.

The application case proposed as part of the collaboration with Dassault Aviation brings together these two challenges:

- Since an aircraft is a safety-critical system, a purely black-box data-driven approach is not appropriate, and it is worth taking advantage of the knowledge available about the system to enhance the robustness and reliability of data-driven models by incorporating *a priori* knowledge in some way (Research challenge 2).

- Though providing an extended sensor-based observability on all the monitored aircraft systems, there is no degradation data available, but only a few "healthy" data samples (Research challenge 1).

1.4.2 Overview of the general solution for Dassault Aviation experimental use case

As described in Section 1.3.2, the second part of the thesis consisted in developing a global and coherent prognostics approach to meet the two research challenges mentioned above, with a specific application to the Falcon 6X use case proposed by Dassault Aviation. In the following lines, this methodological proposal is briefly described in a holistic manner.

To address the two research challenges identified above, a multi-step prognostics approach is proposed. Inspired by PINNs, it relies on the integration of physics into the training data itself through a prognostics-oriented data augmentation, thus generating both nominal and degraded data. It then offers an unsupervised extraction of HI in a realistic manner, i.e. without using any degradation data. Finally, based on this extracted HI, RUL predictions are made. In addition, a reliability-based assessment procedure is proposed, to ensure the consistency of predictions with the established reliability laws.

The various stages of the multi-step approach are outlined above, along with the research challenges which they address respectively.

1. **Prognostics-oriented data augmentation.** Firstly, on the basis of a limited volume of nominal data collected on the studied system (i.e. data from a system in healthy condition), a dual prognostics-oriented data augmentation is conducted. On the one hand, a *system identification enabled nominal data augmentation process* generates new nominal data with limited use of training data while guaranteeing consistency with regard to the real system. On the other hand, a physics-based degradation data enrichment strategy allows the injection of degradation that respects the underlying physics of the failure mechanism, thereby generating new trajectories without using any prior degradation data. This dual data augmentation stage therefore contributes to addressing both research challenge 1, as well as research challenge 2. It is described in Chapter 2.

2. **Unsupervised Health Index Extraction.** On the basis of nominal data (which can be both real nominal data collected on the system and additional nominal data generated by the *system identification enabled nominal data augmentation process*), a neural autoencoder is trained to perform sensor data reconstruction. Consequently, the autoencoder learns the nominal behavior of the system under study. Then, once the training phase is complete, the autoencoder parameters are frozen and it is used to process TTF trajectories containing progressive degradation until total failure. As a result, the reconstruction error will increase as the sensor measurements move away from the initial nominal distribution, due to degradation. This reconstruction error is therefore used as an indicator of SOH, i.e. as HI. This HI extraction process, based on the evolution of the reconstruction error, is completely unsupervised, and therefore responds appropriately to the research challenge 1 (RUL-labeled data limitation). The detailed approach is described in Chapter 3

3. **Health Index and Remaining Useful Life long-term prediction.** Based on an initial sample of HI trajectories, a predictive neural model is applied to produce long-term HI predictions, until a predefined threshold value corresponding to the EOL is reached. This therefore leads to a recursive estimate of the RUL. The complete approach, along with the

associated experimental results, is presented in Chapter 4. In addition, a reliability-based assessment procedure is proposed to ensure that the predictions made are consistent with typical reliability laws and can therefore be used with trust to guide maintenance decisions.

The overall prognostics method including the three steps of prognostics-oriented data augmentation, HI extraction and RUL prediction is depicted in Figure 1.6.

Figure 1.6: General overview of the proposed prognostics approach.

Chapter 2

Prognostics-oriented hybrid data augmentation

As briefly mentioned in the conclusion of Chapter 1, to overcome the problem of the lack of training data, while ensuring physical consistency, it is proposed in this PhD work to perform data augmentation while integrating a priori knowledge and physics elements.

The first step is to perform system identification enabled data augmentation whose aim is to increase the amount of nominal data (i.e. when the system is healthy). System identification is chosen for this data augmentation task because it offers two major advantages. Firstly, it is data-driven, which means that extensive knowledge about the physics of the system is not required to model it. Secondly, the models obtained by this method offer certain behavioral guarantees, ensuring with certainty that the data generated will respect the limits of the identified model.

After that, a second step consists of augmenting the data, not only quantitatively, but rather by enriching it by injecting a physics-based degradation. In this way, it is ensured that the behavior of the generated data respects both the Physics-of-Failure (PoF) as well as the dominant system dynamics identified in the previous system identification step.

It should be clearly noted that these two separate data augmentation phases will then provide the data needed to fulfill two distinct goals. The nominal data generated by the system identification enabled data augmentation process will be used to train the unsupervised HI extractor in Chapter 3, while the degraded data generated in the second step will serve to train the HI long-term prediction model presented in Chapter 4.

The rest of the chapter is therefore divided into two parts. Section 2.1 deals with the system identification problem under the particular constraints of the case study. Then, Section 2.2 will describe the process of injecting a physical degradation model to generate degraded data.

Contents

2.1 System identification enabled nominal data augmentation

In this section, fundamental elements of system identification will be progressively presented, with a focus on the specific constraints of the case study.

Section 2.1.1 describes the three possible modeling approaches. Then, the basics of linear system identification are introduced in Section 2.1.2, followed by the special case of closed-loop identification in Section 2.1.3. Elements concerning identification based on historical data only are then provided in Section 2.1.4, as well as some thoughts on model approximation in Section 2.1.5. Finally, the particular constraints encountered in the present case study along with the formulation of the problem are summarized in Section 2.1.6, leading to the proposed original approach in Section 2.1.7, with application to the case study in Section 2.1.8.

2.1.1 Modeling approaches

Three broad approaches can be distinguished for obtaining a model of a dynamic system. The layout presented here is the one suggested by Sjöberg et al., 1995.

The first one, known as white-box modeling, refers to constructing the entire model using scientific relationships that fully describe the process, from a physics perspective. This presupposes that the model is perfectly known.

The second one is called grey-box modeling. In this approach, prior knowledge about the process is used to derive a model from first principles using laws of physics, and the unknown parts of the model are estimated by using the available data. This applies when some physical insight is available, but several parameters still need to be determined from observed data. By doing so, it avoids estimating what is already known, and the remaining parameters are tested against experimental data (Sohlberg and Jacobsen, 2008, Bohlin, 2006).

Finally, black-box model identification approach is intended for situations where little or no physical insight is available. A parametric model is selected and its parameters are estimated by using data-driven techniques, usually from data collected during an experiment. Here, this aim is to capture the most dominant dynamics of the process. This approach is typically called system identification and is often used for control design because it is relatively easy to implement. It is also worth noting that the black-box approach is mainly data-driven.

2.1.2 Linear system identification

A vast literature covering system identification methods is available (Ljung, 1999, Pintelon and Schoukens, 2001, Garnier et al., 2008, Young, 2011).

Most industrial systems are complex and non-linear, and this is why a part of the research is dedicated to non-linear identification (Sjöberg et al., 1995, Nelles, 2001). Nevertheless, in most cases it is possible to linearize the process at a local level, thereby reducing the problem to linear model identification. With this simpler approach, multiple local linear models can be identified around different operating points. Indeed, despite the fact that most industrial processes are complex and non-linear, they usually operate along certain fixed trajectories composed of predetermined operating points around which linear local models can be used.

In the case of linear system identification, the following assumptions are thus formulated:

Assumption 2.1.1 (Linearity) *It is assumed that the system linearly operates around a working point and can, as such, be well described by a linear model.*

Assumption 2.1.2 (Time-invariance) *The characteristics of the system, in absence of degradation, are assumed not to vary over time.*

Assumption 2.1.3 (SISO) *The system under study is a Single-Input Single-Output (SISO) system.*

The traditional system identification methodology starts by exciting the process with a carefully designed input signal under open-loop conditions. Then, the observed input-output data are used to generate a suitable model of the process. Almost always, reduced-complexity models are generated and evaluated to figure out if they are able to capture the most dominant dynamics of the process. The general data-driven system identification workflow in open loop can be summarized as follows (Ljung, 1999):

1. The first step is to collect a sufficiently rich set of data from the system.
2. It is then necessary to choose a class of models on which the identification will be carried out, in order to determine the model structure and the number of parameters to be estimated.
3. The third step then consists of estimating the parameters of the selected model on the basis of a chosen identification criterion.
4. Finally, the model is evaluated and, if necessary, one or more of the previous stages is repeated to refine the result.

Therefore, system identification is typically an iterative process, where user insight and judgment can be mixed with practical considerations, extensive data handling and complex algorithms.

The theory of system identification has primarily been developed considering the parametric estimation of discrete-time models, which is largely attributable to the development of numerical calculation and to the discrete nature of the data acquired. Nevertheless, the identification methods for continuous-time linear models based on sampled data offer a number of valuable benefits. Indeed, most of the systems governed by the laws of physics (such as electronic, mechanical, hydraulic systems, etc.) evolve continuously over time. Their behavior can therefore be described, after being linearized as discussed above, by differential equations with constant coefficients. Such a representation in the continuous domain is therefore natural for the study of

this kind of systems. The CRAN research team has contributed extensively to the development of continuous-time model identification approaches (e.g. Garnier et al., 2008, Garnier et al., 2018), so these are the approaches used in this PhD work.

As mentioned earlier, the model structure used to approximate industrial processes is often among the family of simple process models. These models are characterized by steady-state gain, possible dead time and dominating time-constants. Examples of this type of model structure include first-order models plus time-delay of the form:

$$G(s) = \frac{K_p}{1 + T_{p1}s} e^{-T_d s} \tag{2.1}$$

where s denotes the Laplace variable, but also second-order models with two real poles of the form

$$G(s) = \frac{K_p(1 + T_z s)}{(1 + T_{p1}s)(1 + T_{p2}s)} e^{-T_d s} \tag{2.2}$$

or second order models with complex poles of the form

$$G(s) = \frac{K_p(1 + T_z s)}{1 + 2\zeta T_w s + (T_w s)^2} e^{-T_d s} \tag{2.3}$$

Once the low-order model form is selected, its parameters θ must be estimated. The standard approaches for parameter estimation in the literature are the Prediction Error Method (PEM) and the Instrumental Variable (IV) approach (Ljung, 1999, Young, 2011, Garnier, 2015).

The parametric estimation method that is used in this PhD work is the Refined Instrumental Variable for Continuous-time systems (RIVC) (Young, 2008, Garnier, 2015). It is a statistically efficient algorithm suitable for continuous-time model identification that has been developed and used successfully over many years for the data-driven modeling of many real systems, including aeronautical systems. Moreover, this method has already been used in a previous collaboration with Dassault Aviation, for the purpose of modeling bleed air system feedback control loops (Garnier et al., 2018). Among the standard criteria usually used to evaluate the model quality, the goodness of fit value using Normalized Root Mean Square Error (NRMSE) is defined as follows:

$$\text{fit}(\%) = (1 - \text{NRMSE}) \times 100 = (1 - \frac{\|y - y_{sim}\|}{\|y - \bar{y}\|}) \times 100 \tag{2.4}$$

where y and $\bar{y}$ are the measured output and its mean value respectively, and y_{sim} is the model simulated response. The closer to 100% the criterion, the better the model fits the data.

For successful application of the identification procedure, it is almost always necessary to have some user-friendly software tools to facilitate the user's modeling. The CONtinuous-Time System IDentification (CONTSID) toolbox, developed at CRAN over many years, offers the latest and most sophisticated time-domain identification algorithm (Garnier et al., 2021). A brief overview of the CONTSID toolbox, with its main estimation commands, can be found in Appendix B.

2.1.3 Closed-loop system identification

In many cases, the controlled process is run under closed-loop conditions, i.e. it uses feedback, where the output signal is fed back to the input, to reduce steady-state errors and improve stability and/or safety. The objective of closed-loop identification is to use closed-loop operating data to build a dynamic model of the process.

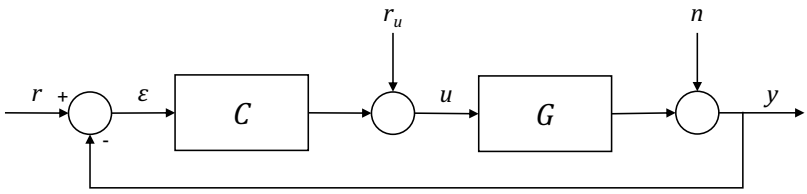


Figure 2.1: Standard block diagram of a closed-loop system.

Over the last 30 years, closed-loop identification has been an area of considerable academic and industrial interest. Important issues such as identifiability under closed-loop conditions have received attention from many researchers (Box and MacGregor, 1976, Söderström et al., 1978, MacGregor and Fogal, 1995) and a number of identification strategies have been developed (Van den Hof, 1998, Forssell and Ljung, 1999, Ljung, 1999, Gilson and Van den Hof, 2005). There are three main approaches, of which the first is the most popular (Söderström and Stoica, 1989):

- Direct identification
- Indirect identification
- Joint input/output identification

Figure 2.1 displays the standard block diagram of a process under feedback control, usually referred to as closed-loop system.

It is well known that special problems may occur in the context of closed-loop identification. In the block diagram presented in Figure 2.1:
G is the transfer function of the system to be identified
C is the transfer function of the controller
r is the reference (or setpoint)
u is the command
r_u is a possible extra-signal on the command u
n is the output noise or sensor noise
y is the output
ε is the error ($\varepsilon = r - y$)

From the block diagram, the output can be expressed as:

$$y = Gu + n = G(C(r - y) + r_u) + n \tag{2.5}$$

hence

$$(1 + CG)y = G(Cr + r_u) + n \tag{2.6}$$

In a similar way, the command can be expressed as:

$$u = C(r - Gu - n) + r_u \tag{2.7}$$

hence

$$(1 + CG)u = C(r - n) + r_u \tag{2.8}$$

The ratio of the output signal over the command signal leads to:

$$\frac{y}{u} = \frac{CGr + Gr_u + n}{Cr + r_u - Cn} \tag{2.9}$$

r	r_u	n	$\frac{y}{u}$
0	0	0	No identification possible
0	0	n	$-\frac{1}{C}$, inverse of the controller
0	r_u	0	G, true transfer function
0	r_u	n	$\frac{Gr_u+n}{r_u-Cn}$
r	0	0	G, true transfer function
r	0	n	$\frac{CGr+n}{C(r-n)}$
r	r_u	0	G, true transfer function
r	r_u	n	$G + \frac{1+CG}{((Cr+r_u)/n)-C}$

Table 2.1: Closed-loop system identification results depending on the extra-signals available.

As a result, the cases listed in Table 2.1 may be encountered (Eykhoff, 1984).

It is clear, then, that only a limited amount of information about the process and controller can be derived from the observed data in a particular identification situation and under particular circumstances. External signals must be rich enough for the system to be excited and to provide sufficient information. Let us investigate the case where $r_u = 0$, i.e. when no extra signal is added to the command. Then:

$$u = C(r - y) \tag{2.10}$$

hence

$$y = \frac{Cr - u}{C} \tag{2.11}$$

Then, when the setpoint r remains constant (or null), the equation becomes:

$$y = -\frac{1}{C} \times u \tag{2.12}$$

In closed-loop model identification approaches, it is generally assumed that the setpoint and/or the extra signal on the command sufficiently excite the system (Van den Hof, 1998). However, this is not the case in this study. As shown in second line of Table 2.1 and in Equation 2.12, without external excitations, any identification algorithm will naturally identify the inverse of the controller transfer function. This result is well known (see e.g. MacGregor and Fogal, 1995, B. Huang and Kadali, 2008) and will be exploited in the proposed approach.

2.1.4 System identification procedure for historical closed-loop data

It has been stated that for system identification, both in open loop and more so in closed loop, a design of experiment adapted to the situation must be established in order to acquire, from the considered system, input-output signals that are sufficiently rich to be informative.

However, in many industrial situations, as it is the case of the application study, such experiment design cannot be implemented, and historical data from the process operating in closed loop must be used only. This is why historical informative segments of data must be carefully selected to perform identification of dynamic models (Xing et al., 2020, Peretzki et al., 2011). It is worth mentioning that, where large quantities of data are available, various data mining methods can be used to locate suitable segments for system identification (Isaksson, 2013). However, given the limited amount of data available for this study, it was not necessary to use such methods (a hand analysis was sufficient).

This specific characteristic of exclusive use of historical closed-loop data leads to the following formulation of the data selection problem. From a collection of data:

$$Z_N = [Z(t_1), Z(t_2), \cdots, Z(t_N)] \tag{2.13}$$

where $Z(t_k)$ is a pair of input-output $[u(t_k), y(t_k)]$. The objective is to find time segments $[t_{init}, t_{end}]$ such that the data in Z_N is sufficiently informative to be used for system identification.

The iterative data-driven system identification methodology resulting from this specific constraint is described in Figure 2.2. Historical informative data segments must first be selected, based on the expert assessment and careful data observation. After that, a low-order model structure is selected, within which the parameters will be estimated. The parameters of the selected model structure are estimated from the retained input-output data.

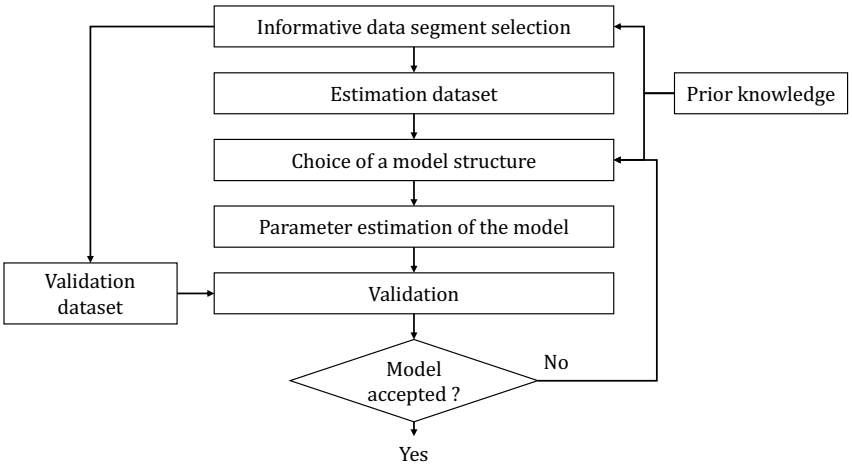


Figure 2.2: The data-driven system identification procedure based on historical informative data segments selection.

Goodwin et al., 2001 also add that, in practice, it is common to use a combination of both black-box and phenomenological insights, which can be seen as prior knowledge, to build a model. Phenomenological insights are very often crucial to understand the key dynamics, the non-linearities and the significant time variations of a given system. Bearing this in mind, a substantial part of the work will involve analyzing and exploring the available data (see Section 2.1.8).

2.1.5 Some considerations about model approximation

In traditional system identification literature, the identifiability issues and quality of identification are addressed mainly under the assumption that the model set $\mathcal{M}$ contains the true process $\mathcal{S}$,

i.e. the model can describe the true process dynamics ($\mathcal{S} \in \mathcal{M}$).

The more typical case is that of under modeling (or identifying reduced-complexity models) when the plant is not in the model set. This is the focus of the present study, and is a more practical situation since a plant is generally of relatively high-order, and the model structure used to approximate such a process is almost always of lower order. Stated simply, identification can therefore be considered as a model approximation problem.

Moreover, when building a model, it is important to remember that all real processes are complex and so any attempt to build an exact description of the plant is generally an unreachable goal (Goodwin et al., 2001). Furthermore, as stated by Box, 1976, "All models are wrong, but some are useful".

In fact, both art and science are involved in deriving a model that captures the desired plant characteristics. Hence, it is usually best to start with a simple solution and then add functionality as the solution evolves. This is why, in the present study, rather simple linear low-order models are employed, provided that they capture the essential features of the problem.

2.1.6 Problem formulation of the specific closed-loop system identification

The different constraints specific to the case study are summarized below:

- The system operates in closed loop.
- Data from limited historical test flights[1] is available only.
- The controller is not known and must be identified.
- Experiment design is not possible. Only a few and not sufficiently informative changes of the setpoint occur in the historical data.
- Each loop element (i.e. controller and process) must be identified.

The data driven system identification problem can therefore be stated as follows:

From the limited historical data available, determine linear models of the controller C and the process G in order to approximate the dynamic behavior of the feedback loop (as shown in Figure 2.3), with the aim of generating additional data and later integrating additional degradation in the closed-loop components.

2.1.7 Proposed data-driven system identification approach

The specific constraints described above have led us to develop a taylor-made system identification procedure to identify both controller and process models as shown in the feedback control loop displayed in Figure 2.3.

Two main scenarios are to be hoped for in order to obtain appropriate data for the identification of the C and G models (Horch, 2000):

- The control is operating in "manual mode". The system can be considered as in an open loop, and the command signal u is varied enough to identify the model of the process G.

[1] Test flights correspond to flights on which various systems of the aircraft were adjusted and inspected. The tests carried out are however not known.

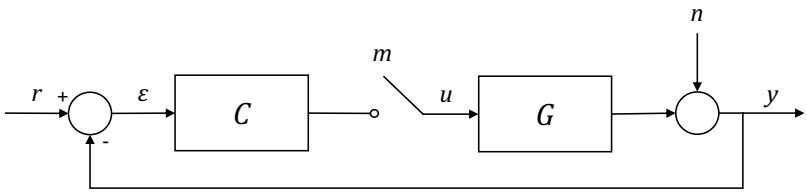


Figure 2.3: Block diagram of the feedback control loop with an option to switch to "manual mode" (corresponding to an open-loop control).

- The control is in "automatic mode" (or in closed-loop control) and the setpoint r remains constant. In this case, following Equation 2.12, the controller model C can be deduced through the identification of its inverse.

As a consequence, the proposed methodology is to treat these two cases separately as follows:

1. Find historical data segments where the control is in "manual mode" and identify the process model G from "open-loop data".
2. Find historical data segments where the control is in "automatic mode" and identify the inverse of the controller model of C from "closed-loop data".
3. Adjust the variance of the white noise input of a simple low-pass AutoRegressive (AR) model to reproduce the measurement noise effects.
4. Validate the overall model by testing its ability to reproduce the general behavior of several test flights.

2.1.8 Application results

The system under study is an "air distribution system", whose purpose is to regulate the temperature inside the cockpit of the aircraft. Hot air (around $200°C$) is collected from the aircraft engines, then cooled down to an unknown temperature, and finally sent to the cockpit through a series of pipes. A valve allows more or less warm air to enter the cockpit.

Based on *a priori* knowledge about the temperature control and preliminary data observation, a traditional block diagram of the feedback system has been built in Figure 2.4. A temperature setpoint r is selected by the pilots. The controller C is driven by the error ε, formed from the temperature setpoint r subtracted by the measured temperature y_m, which is the true system output y corrupted by noise n. The command u is provided to the actuator A which is a solenoid valve, thus making it possible to act on the cockpit temperature P by controlling the valve position.

In the observed flights, most of the time, the setpoint is set around $22°C$. This is thereby the working point around which the system can be well described by a linear model (see Assumption 2.1.1)

Many historical variables are collected on the air distribution system during test flights. The following will be used for identifying the closed-loop elements:

- The temperature setpoint r
- The command u which is here the opening position of the valve
- The measured temperature y_m

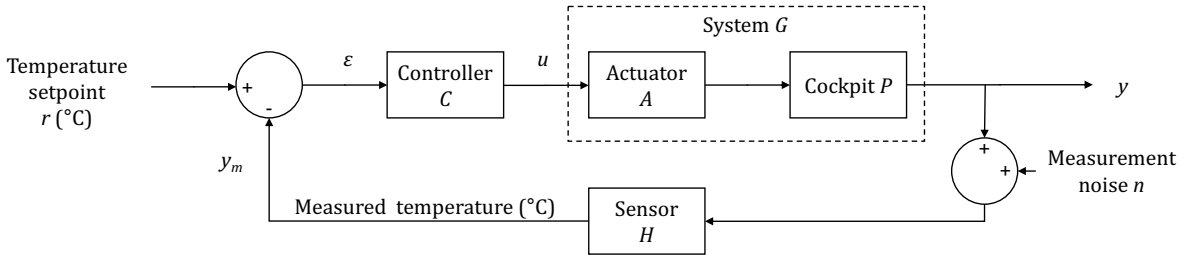


Figure 2.4: The block diagram of the air distribution system of the cockpit.

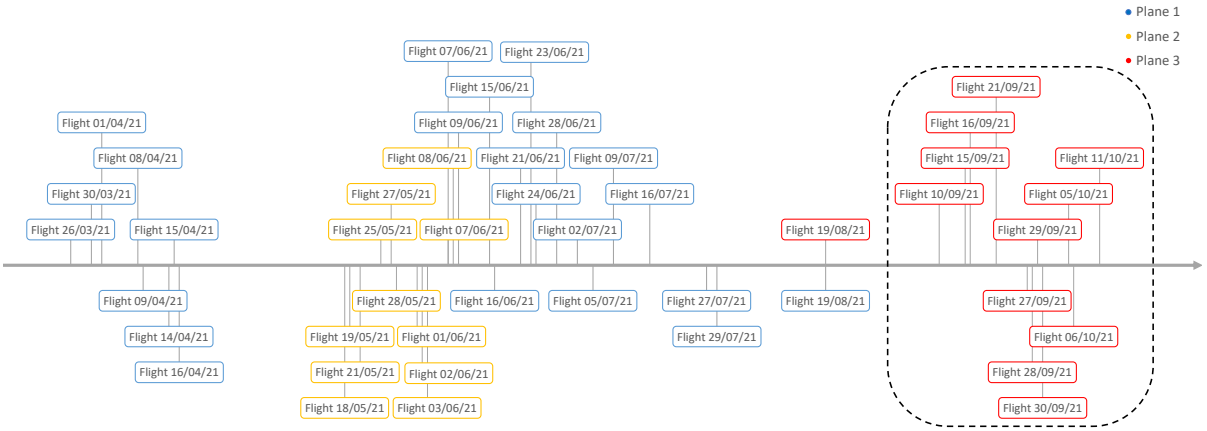


Figure 2.5: Time distribution of the available test flights.

Data selection

As described in the proposed procedure (Section 2.1.7), it is necessary to find data segments corresponding to a "manual mode" (i.e. open-loop control) and an "auto mode" (i.e. closed-loop control) in the available historical data. A careful analysis of the available flight recordings has therefore been carried out.

The data available is made up of a series of test flights carried out during the year 2021. As test flights, they were used to carry out various adjustment tests and control operations on different systems of the aircraft. It was not possible to know what these tests consisted of. The time distribution of all the available test flights is displayed in Figure 2.5.

The flights were made on three different aircraft, over a period of eight months. Data acquisition difficulties were encountered for some flights collected from aircraft 1 and 2. Moreover, it turns out that the controllers were only correctly set starting from August 2021. For these reasons, the set of usable data has been reduced to the flights of aircraft 3, from September onwards, i.e. the last eleven flights collected, as highlighted in the dashed frame in Figure 2.5.

Four main variables are collected and displayed in the following figures:

- The temperature setpoint ($°C$)
- The measured temperature in the cockpit ($°C$)
- The command (position of the valve) ((°)

- The altitude over the flight (feet)

A careful study of each of the eleven available flights reveals that some of them are not informative. The main characteristics of the eleven flights are summarized in Table 2.2. For each flight, it is indicated whether it contains data segments where the control system operates in "manual mode" (i.e. in open-loop control, to identify G), data segments where it is in "auto mode", (i.e. in closed-loop control, to identify C) and whether it exhibits setpoint variations. As described in Section 2.1.3, the inverse of the controller can only be identified if the temperature setpoint r remains constant.

Flight date	"Manual mode" segments?	"Auto mode" segments?	Setpoint variations during flight?	Observations
10/09/2021	Yes	Yes	No	One manual segment but during a takeoff phase
15/09/2021	No	Yes	No	Few unexplained temperature behaviors during the flight
16/09/2021	No	Yes	Yes (one change)	-
21/09/2021	No	Yes	No	-
27/09/2021	No	Yes	Yes (one change)	-
28/09/2021	No	Yes	No	Unexplained temperature behaviors during the flight
29/09/2021	No	Yes	Yes Minor	Suitable for identifying controller C Setpoint changes too rapid to observe response
30/09/2021	-	-	-	Data acquisition problems
05/10/2021	Yes	Yes	Yes	Suitable for identifying system G
06/10/2021	-	-	-	Data acquisition problems
11/10/2021	No	No	No	No activity (the aircraft did not take off)

Table 2.2: Summary of the main characteristics of the test flights studied.

Looking at Table 2.2, the following observations can be formulated:

- Some flight recordings are non-informative (October 11) or corrupted (September 30, October 6) and therefore cannot be used for identification purposes. For instance, Figure 2.6 displays the flight record corresponding to the last line in Table 2.2, in which the air distribution system is inactive for the entire duration of the record (it can even be seen that the aircraft has not taken off).
- Most flights are entirely in "auto mode" (i.e. in closed-loop control), with little or no setpoint variation. They could therefore be used to identify the inverse of the controller, as explained in Section 2.1.3.
- For some flights, the measured temperature behavior for a few segments of data cannot be explained from the feedback control. These data segments have therefore been excluded.

Data segment selection for the process model identification: The careful analysis of the 11 flight recordings thus revealed that the only two flights offering possible data segments to identify the G model are the September 10 and October 5 flights. However, in the first case, the short segment available in "manual mode" (i.e. in open-loop control) happens right in the takeoff phase, and is therefore not very suitable for identification. Indeed, as described above, hot air is drawn from the aircraft engines, then cooled before being sent to the cockpit through valve A. Take-off and descent phases are therefore periods of disturbance for the system under study, since variations in engine load tend to modify the temperature of the intake air.

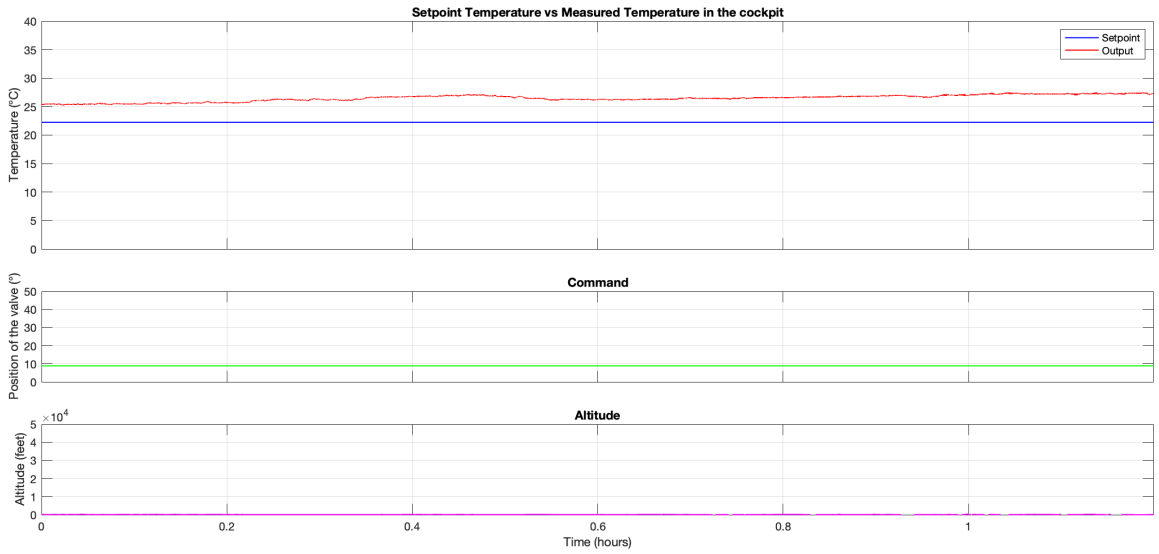


Figure 2.6: Flight of October 11, 2021, Aircraft 3 (the aircraft did not take off).

The only flight record that offers such a situation in cruise phase, is the flight of October 5, 2021, which is displayed in Figure 2.7. In this flight, it turns out that at some moments, the actuator A is forced to a closed position by an upstream system, thus leading the system to behave as in "manual mode" (i.e. in open-loop control). These situations do not correspond to normal system operation, but we take advantage of the opportunity presented by such situations to identify the system G, as proposed in Section 2.1.7. Historical data segments selected to identify the process model G can be visualized in Figure 2.7.

Data segment selection for the identification of the inverse of the controller: Figure 2.8 displays the flight recording of September 29, where the setpoint r remained constant over most of the flight duration, while the system was in "auto mode" (i.e. in closed-loop control). This record is therefore an excellent data segment for identifying the inverse of controller C, as it is particularly long and therefore provides a large number of data points. Furthermore, it is one of the most recent flights, so the aircraft configuration (including any controller settings) is the most up-to-date. For these two reasons (number of data points and most recent date), this data segment has been selected to identify the inverse of the controller C. Another possibility might have been to select l similar segments on different flights, identify l controller C models, then average them out, as proposed by Ljung, 1999, p. 464). However, since the settings made over the course of the recorded flights are not known, it is preferable to carry out the identification with the most recent flight.

Identification of the system model G

During a full closure command of the valve, the system operates as in open loop such that the block diagram becomes the one shown in Figure 2.9.

The actuator A and the cockpit P are grouped together into the lumped system model G, as described in Figure 2.9.

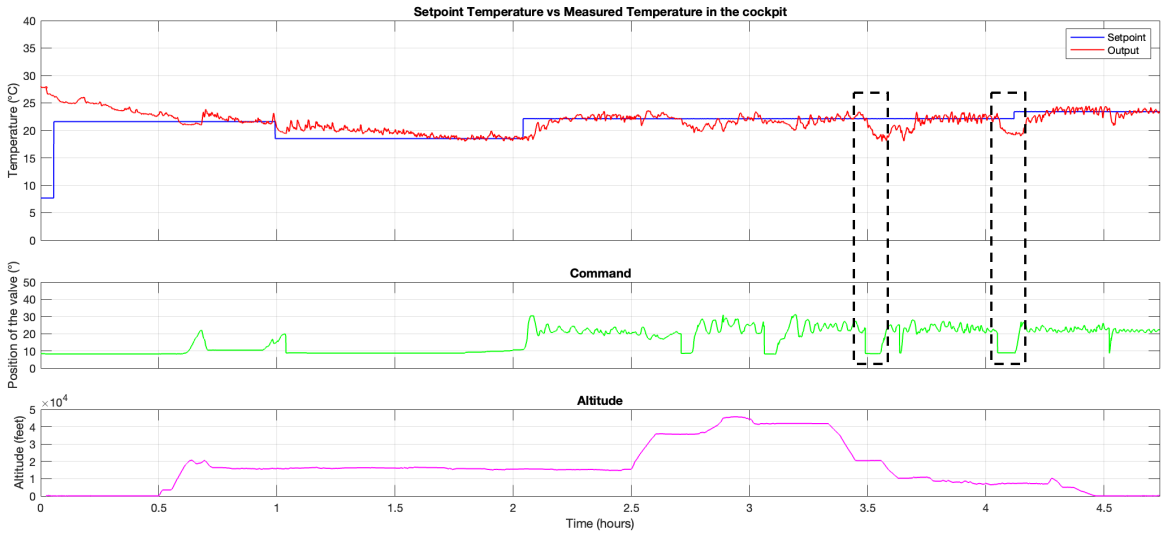


Figure 2.7: Flight of October 5, 2021, Aircraft 3. The data segments selected for the estimation and the validation of the system model G are framed in dashed lines.

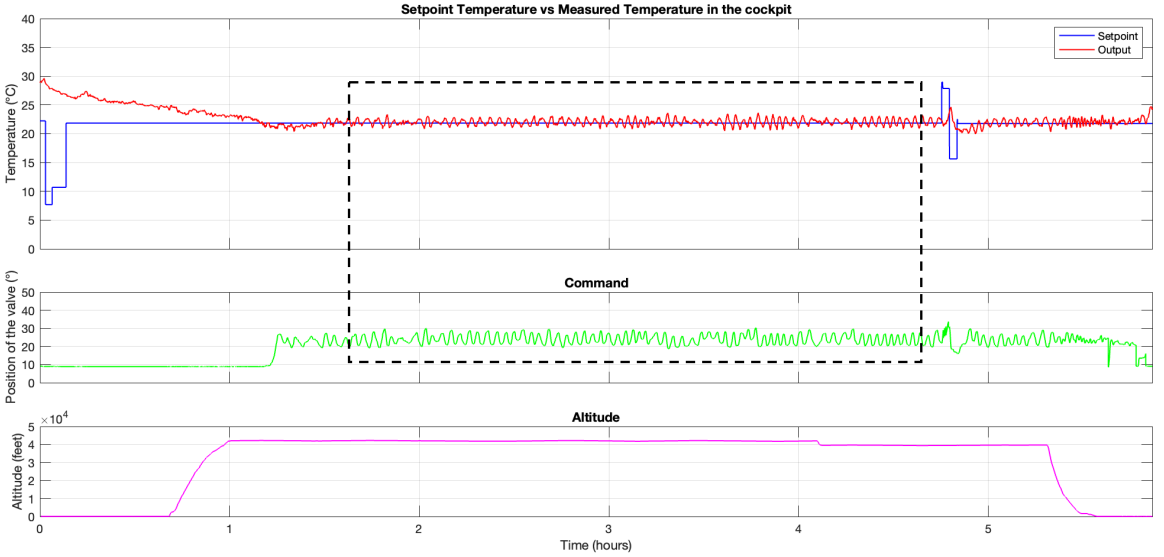


Figure 2.8: Flight of September 29, 2021, Aircraft 3. The data segment selected for identifying the inverse of the controller C is framed in dashed lines.

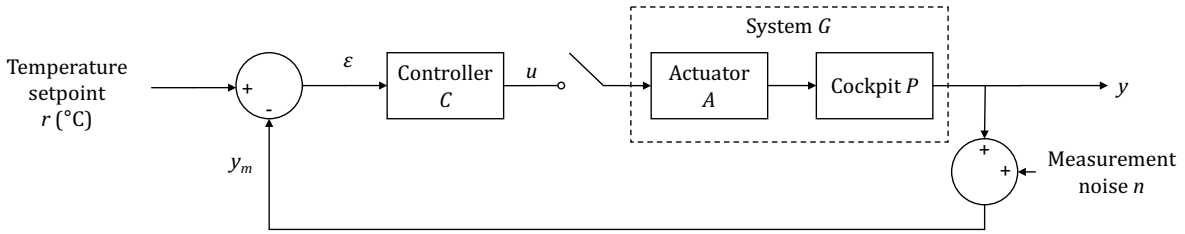


Figure 2.9: The air distribution system in "open-loop mode" for a full closure command of the valve. It is when the system operates in this "manual mode" that G can be identified.

It is worth recalling that actuators are, in many cases, highly nonlinear and often have their own dynamic behavior. This is the case, for example, with valves and hydraulic actuators (Goodwin et al., 2001). This is why, although the actuator and the plant may be lumped together at this stage (since the actuator is assumed to be linear), in the fault injection stage (Section 2.2) they will be treated as two separate blocks.

The dynamics of the sensor is much faster than the dynamics of the system G and is therefore considered negligible. The model is therefore described by:

$$y_m = Gu + e \tag{2.14}$$

where G is chosen from the temperature response as first-order transfer function plus delay:

$$G(s) = \frac{Ke^{-\tau s}}{1 + Ts} \tag{2.15}$$

where τ is the delay, T the time-constant, K the steady-state gain and e is a white noise. Nevertheless, note that the measurement noise is not likely to be white. It will be modeled afterward by a low-pass AR model. The white noise assumption corresponds to the use of the Simplified Refined Instrumental Variable method for Continuous-time systems (SRIVC) algorithm.

Figure 2.10 displays the measured step-like response that is used to identify the system model G. This data segment corresponds to the right frame of the flight shown in Figure 2.7.

The parameters of the model have been then determined using the RIVC method (Young, 2011, Garnier, 2015), available in the CONTSID toolbox[2] (`PROCSRIVC`, See Appendix B). The following parameters are obtained:

$$\widehat{G}(s) = \frac{(0.3 \pm 0.002)e^{(-8 \pm 0.94)s}}{1 + (50.5 \pm 1.6)s} \tag{2.16}$$

where the value after the $\pm$ sign represents the estimated standard deviation.

A validation of the identified model is then performed on the second historical data segment selected from the flight (left frame in Figure 2.7). This is depicted in Figure 2.11. On both estimation (Figure 2.11a) and cross-validation data (Figure 2.11b), the simulation (noise-free) obtained by the approximate low-order model $\widehat{G}$ is able to capture the main temperature response of the real system, meaning that the model parameters are correctly estimated.

It is worth noting that without these two data segments where the control can be seen as operating in "manual mode" (i.e. in open-loop control), it would not have been possible to

[2]Similar results have been obtained by using the PEM-based `PROCEST` routine from the System Identification toolbox.

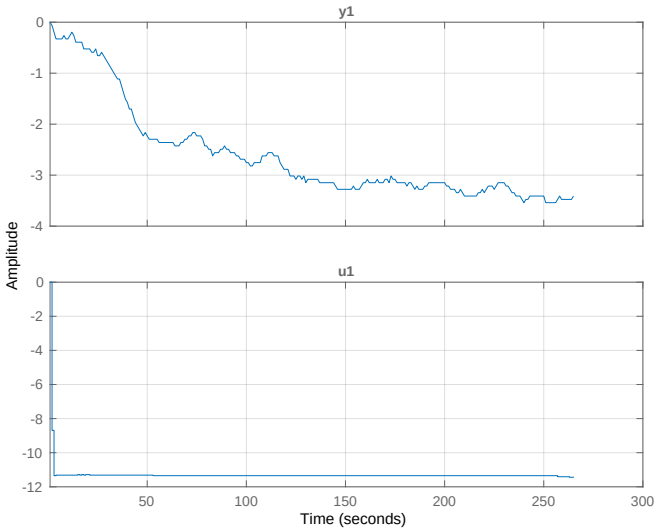
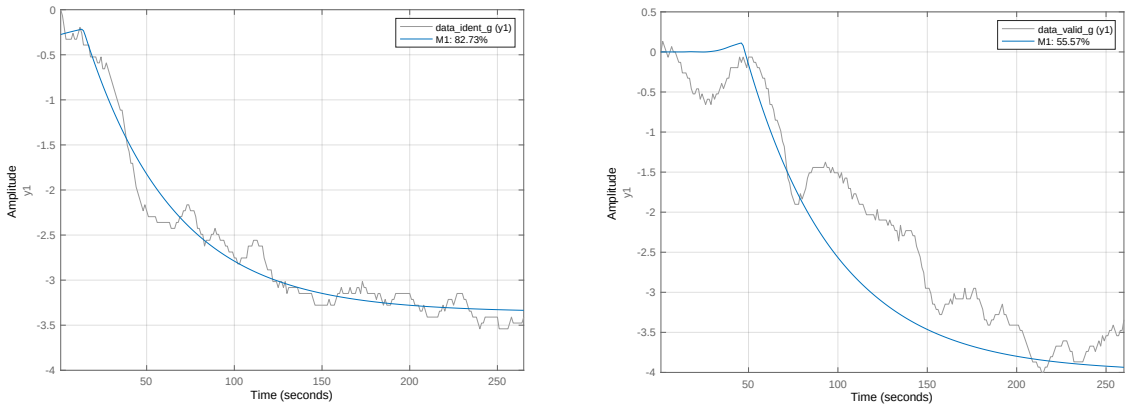


Figure 2.10: Estimation data used to identify the model of the system G. The valve closing step is shown on the bottom plot, while the temperature measured y_m in the cockpit is shown on the top plot. The operating points have been removed from the original data.



(a) Estimation data.

(b) Cross-validation data.

Figure 2.11: Comparison of the measured and simulated $\widehat{G}$ model responses.

identify the system model G without proper external excitations, as it is usually recommended in traditional closed-loop system identification (Van den Hof, 1998). This demonstrates how crucial the data study phase was for this identification procedure.

Identification of the controller model C

Given the limited amount of knowledge available about the controller, the following assumption has been made:

Assumption 2.1.4 (PI controller) *The controller C is assumed to be a Proportional-Integral (PI) controller*

As described in Section 2.1.7, a data segment in which the system is in "auto mode" (i.e. closed-loop configuration) and where the setpoint remains constant is used to identify the inverse of the controller. Figure 2.12 shows the data segment selected for this purpose in the October 5 flight (which is depicted in Figure 2.7).

The transfer function $L(s)$ is then identified, using the CONTSID toolbox:

$$\frac{y_m(s)}{u(s)} = L(s) = -\frac{1}{C(s)} \tag{2.17}$$

with

$$L(s) = \frac{b_1 s + b_0}{s + a_0} \tag{2.18}$$

Then deducing $\widehat{C}(s)$ from $\widehat{L}(s)$ in the form of a proportional-integral controller, the following estimated model is obtained:

$$\widehat{C}(s) = K_p\left(\frac{1 + T_i s}{T_i s}\right) = 1.6\left(\frac{1 + 8.6s}{8.6s}\right) \tag{2.19}$$

Model $\widehat{C}$ will be validated in the complete closed-loop validation stage, since it is difficult to validate the controller individually in this closed-loop context.

Noise Model

An estimate $\widehat{n}$ of the measurement noise has been generated from a low-pass AR(1) model (Ljung, 1999, Young, 2011).

$$\widehat{n}(t_k) = \frac{1 - d_1}{1 + d_1 q^{-1}} e(t_k) \tag{2.20}$$

where q^{-1} is the delay operator and $e(t_k)$ is a white noise process of variance σ_e^2. d_1 has been empirically set to 0.95 to low-pass filter the white noise. The variance of the latter has been empirically adjusted to reproduce the measurement noise observed in the historical data.

Validation

In general, validation consists of confronting the identified model with as much information about the real system as is practical (Bohlin, 1991). A model is said to be valid if it is useful in relation to the purpose in mind (Ljung and Glad, 2016). The current objective is to generate additional data samples in order to train AI-based models for prognostics. Consequently, the

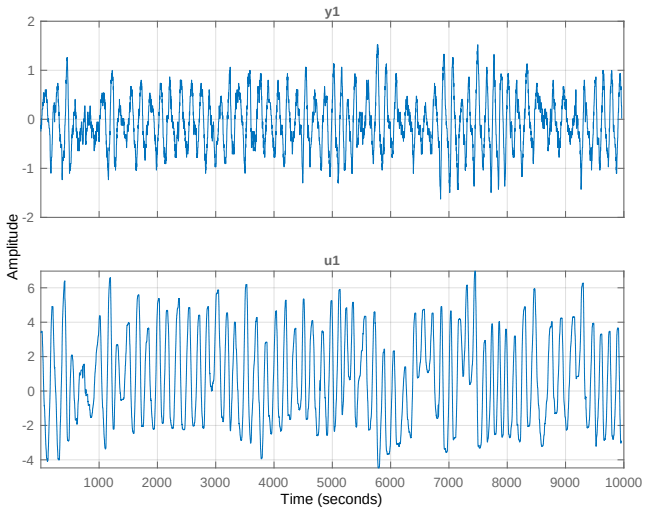


Figure 2.12: Estimation data used to identify the model of the inverse of the controller C, when the setpoint remains constant. The command u is shown on the bottom plot, while the temperature measured in the cockpit y_m is shown on the top plot. The operating points have been removed from the original data.

aim is not to reproduce the true system perfectly, but to capture its general dynamics in order to increase the amount of initial data. Moreover, it is worth recalling that the data available were collected during test flights carried out during the aircraft development phase for which the tests and settings carried out are not known. Consequently, it is not necessary to try to perfectly reproduce each observed behavior, the overall dynamic is what matters.

The model is validated by replicating existing flights in simulation, in order to assess whether the main dynamics have been captured. Such plots are very intuitive for evaluating a model, and it is therefore easy to see exactly which features the model is capable of reproducing, and which features it has not captured (Ljung, 1999). Moreover, such simulations reveal the combined effects of the different approximated models.

In this way, three observed flights (of September 21, 27 and 29) are replicated and illustrated in Figures 2.13, 2.14 and 2.15. In each case, the simulated flight is displayed on the top plot, while the actual measured flight is displayed on the bottom plot. For each flight, at the start of the flight recording, the measured temperature evolves freely until it falls below the temperature setpoint, which activates the control loop. This phenomenon of temperature drop is approximately reproduced in the simulation, but it should be noted that the validation period corresponds to the time when the aircraft is in cruise flight and the system is regulated.

On the basis of flight replications, it can be concluded that the temperature control behavior is globally reproduced, but that there are non-linearities that are not taken into account in the model approximation and therefore cannot be reproduced. More specifically, when the experimental conditions broadly correspond to the operating point at which the linearity assumption is made, the reproduction is reasonably satisfactory. However, it turns out, as expected, that some non-linear behaviors are encountered at certain times. This is notably the case on the flight of September 27 (Figure 2.14), on the rising front step just after 3h40, and on the flight of September 29 (Figure 2.15), on the rising front step around 4h50. It should be noted in addition that the identification of G was made on a falling edge, but it is not certain that the response would

be the same on a rising edge. This shows the limits of the linearity assumption. The overall good replication of these flights was also submitted to system experts from Dassault Aviation, who validated the system identification results.

2.1.9 Data augmentation

From the identified process and controller models, it is now possible to generate additional nominal (i.e. healthy) data samples. Figure 2.16 depicts the overall model of the air distribution system as used for nominal data augmentation. To do this, the flight duration must be selected and the setpoint indicated. To respect the linearity assumption around an operating point, it is preferable to choose a setpoint close to $22.5°C$. Similarly, to ensure that the data generated remain close to reality, setpoint changes should be limited (1 step per flight generated) and remain close to the operating point (for instance a step from $22.5°C$ to $23.5°C$).

One example of such a flight generated by the nominal data augmentation process can be seen in Figure 2.17. It should be noted that the data augmentation only concerns the part of the flights when the system is being controlled, and not the phases when the control system is switched off (in this case, the temperature would evolve freely, without any control).

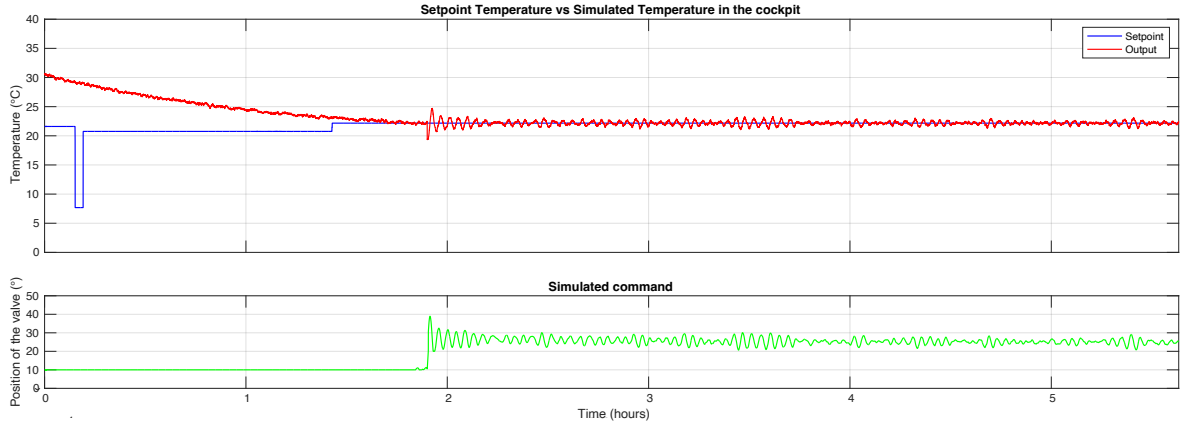
2.1.10 Conclusion

In this first part, a method of data augmentation enabled by system identification has been developed. This involves first identifying the general dynamics of the system under study using data-driven system identification methods. articular constraints were encountered in the study case, namely:

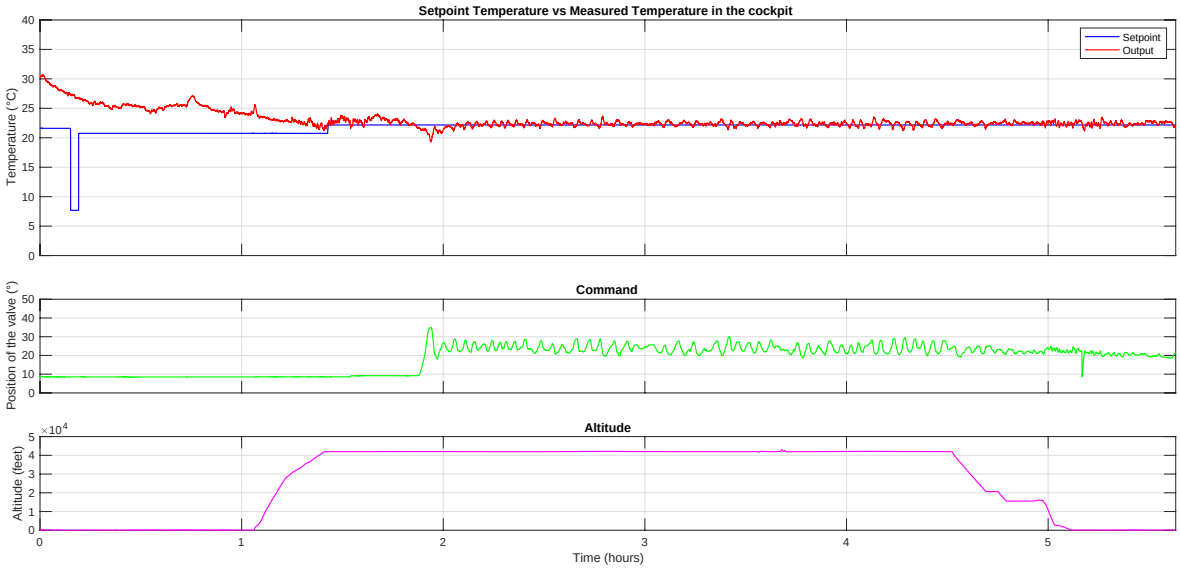
- The system operates in closed loop
- Experiment design is not possible and only limited historical data is available
- The controller C is not known and must be identified, as well as the process G

As a result, a linear tailor-made model identification approach was proposed, relying on the selection of historical data segments under specific conditions (“manual mode”, “auto mode” without setpoint variation). The analysis and choice of historical data segments used for identification were crucial. Continuous-time approaches from CONTSID toolbox have been used here, but it would of course be possible to apply discrete-time approaches, or even non-linear methods. Models are then validated by the reproduction of real flights. The estimated models are able to reproduce the system dynamics in a general manner, around the operating point, but the results obtained are limited by the experimental conditions. If a better fidelity of the model to real behaviors was required, either an approach based on physics (i.e. build a complete physics-based model) would be appropriate, or it would be necessary to be able to excite the system on the setpoint r or on the command u. Nevertheless, it should be remembered that the data available are experimental data taken from test flights. Consequently, it was not worth trying to reproduce perfectly the behavior of a system that is still in the adjustment phase.

Finally, the identified models allow to generate new nominal (i.e. healthy) data samples that will be used to train the autoencoder to extract HI (Chapter 3). The major advantage of system identification in the context of data augmentation is that it is possible to obtain results with very limited amount of estimation data and system knowledge. Moreover, the use of low-order models offers guarantees on the behavior of the data generated, whereas the use of highly complex deep

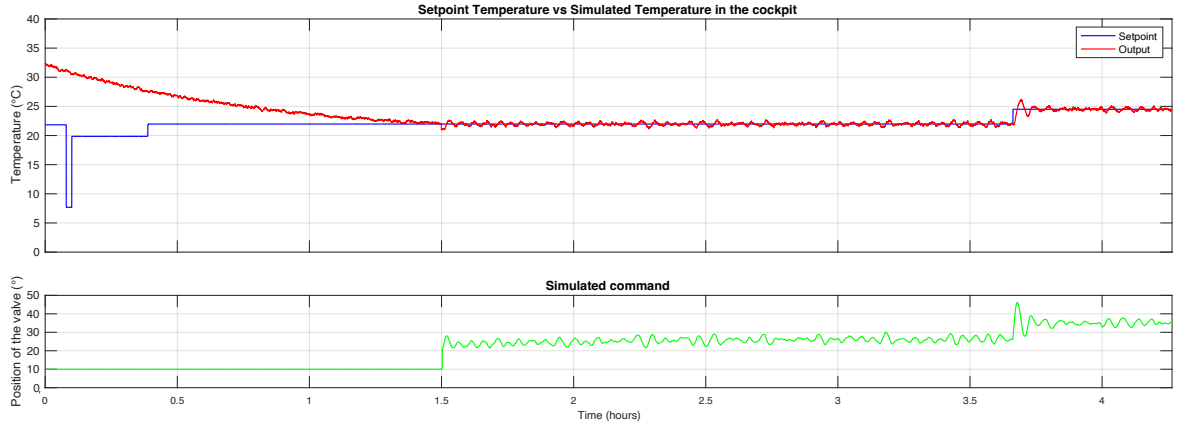


(a) Replicated flight

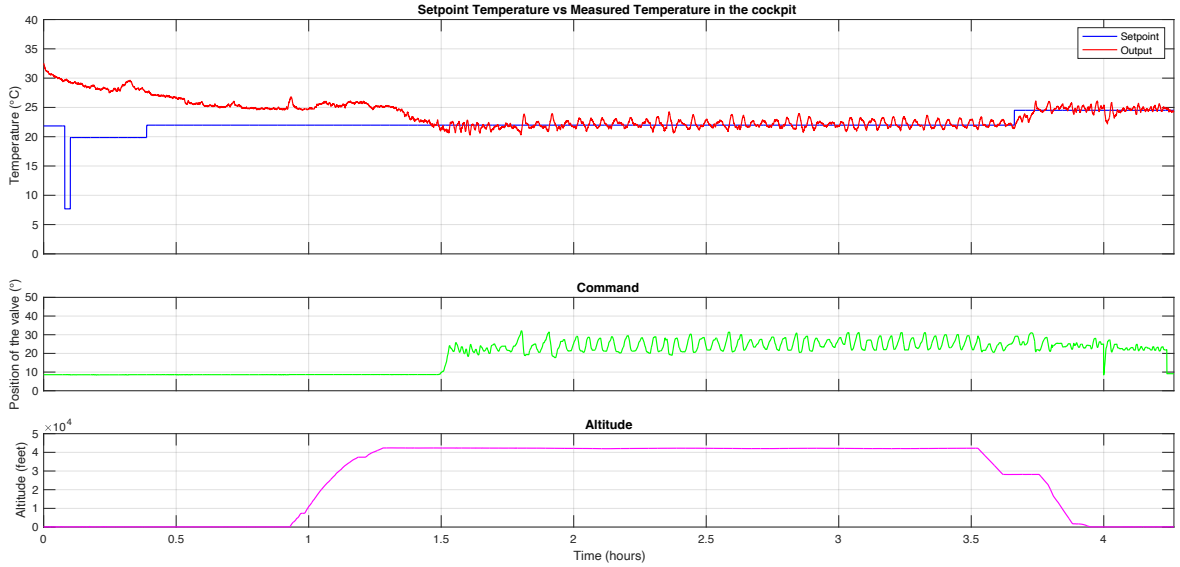


(b) Measured flight data

Figure 2.13: Replication of flight of September 21, 2021 with the *system identification-enabled nominal data augmentation process*. During the flight phase, when control is activated, the general dynamics are correctly reproduced. At the end of the recording (i.e. at landing and when the aircraft is on the tarmac), the behavior is no longer linear, which may be due to unknown external disturbances (outside temperature, door opening, etc.).

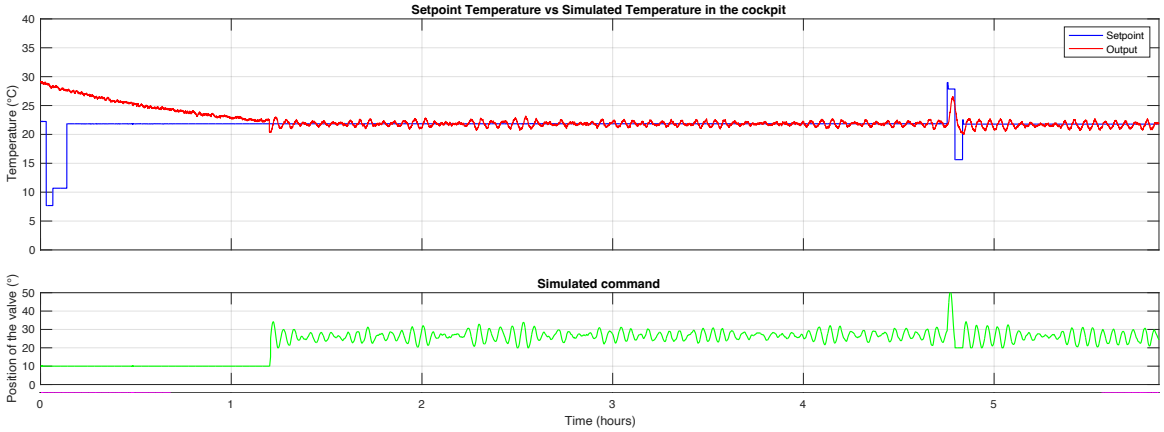


(a) Replicated flight

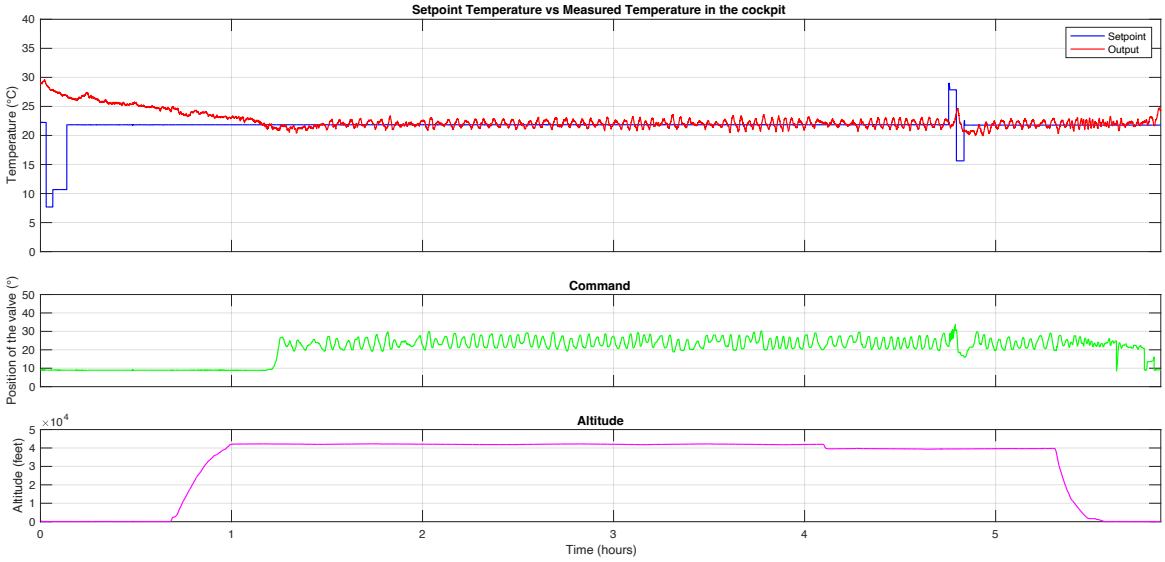


(b) Measured flight data

Figure 2.14: Replication of flight of September 27, 2021 with the *system identification-enabled nominal data augmentation process*. During the flight phase, when control is activated, the general dynamics are correctly reproduced, except for the temperature setpoint change, where the command on the measured flight does not correspond to the ideal linear response (bottom plot).



(a) Replicated flight



(b) Measured flight data

Figure 2.15: Replication of flight of September 29, 2021 with the *system identification-enabled nominal data augmentation process*. During the flight phase, when control is around the operating point, the general dynamics are correctly reproduced. However, similarly to the September 27 flight (Figure 2.14), when the temperature setpoint varies the behavior of the real system is not well reproduced in simulation.

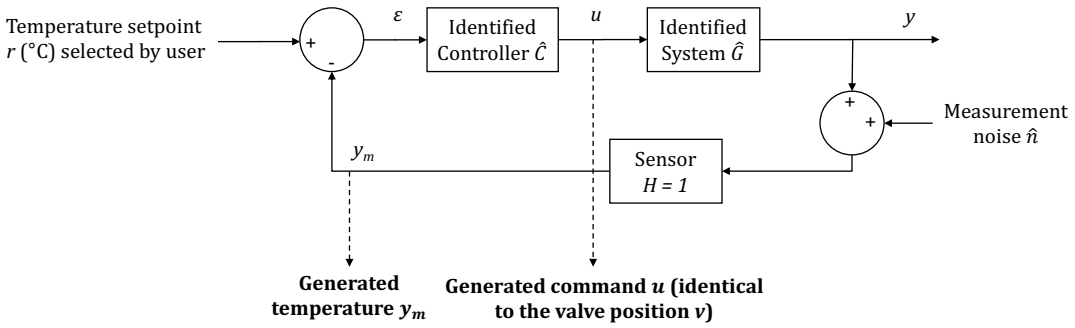


Figure 2.16: Block diagram of the global air distribution system model used to generate additional nominal data. Generated variables are in bold.

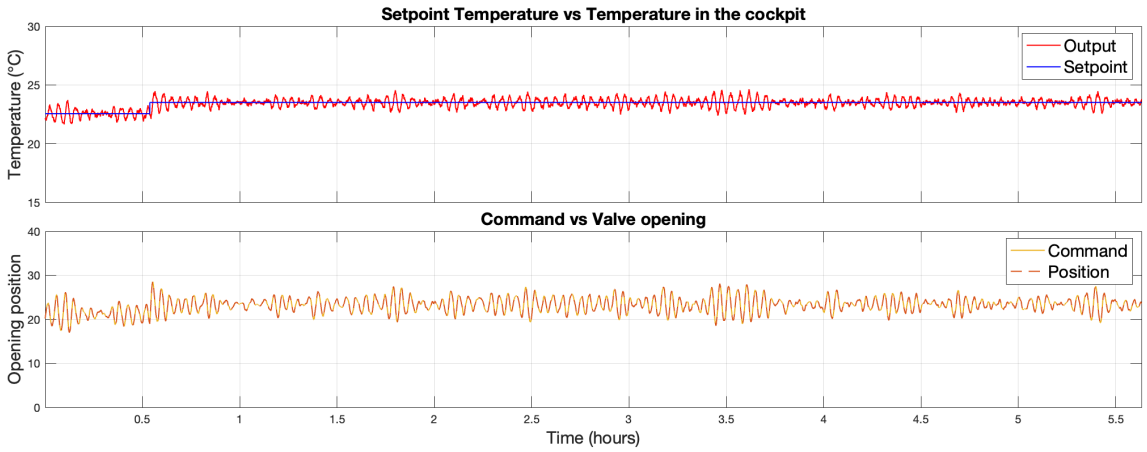


Figure 2.17: New nominal flight generated by the *system identification-enabled nominal data augmentation process*. In nominal data generation, the valve position coincides perfectly with the command, since the actuator is assumed to be perfectly linear, but in the following a degradation will be injected into the actuator, creating a non-linearity which will distinguish the two signals.

learning models (such as GANs) is generally limited by the absence of consistency guarantees on the dynamics of the data generated.

In the next Section, degradation will be injected in this nominal model (thus creating non-linearities) in order to enrich the data and obtain TTF trajectories that will later serve to train a prediction model.

2.2 Physics-based degraded data enrichment

2.2.1 The hybrid procedure

The data enrichment process can be seen as a hybrid process since it is based on data-driven models supplemented by a physics-based degradation model. Indeed, nominal (or healthy) data are available, from which a data-driven model can be obtained (Section 2.1). On the other hand, since these data are nominal, i.e. the system studied is in a healthy condition, such data cannot be used to identify a degradation process. For this reason, a physics-based degradation model must be built from scratch, later injected into the previously identified nominal model, thus forming a hybrid model. This hybrid data generation process is depicted in Figure 2.18. It should be remembered that this hybrid process for obtaining complete degradation trajectories (usually referred to as Time to Failure (TTF) trajectories) will only be used in the RUL prediction stage, and not for the HI extraction stage, which is totally unsupervised and does not require the use of any degradation data (only employing healthy data).

To this end, a realistic degradation model adapted to the case study must first be designed (Section 2.2.2). This model can then be injected into the corresponding sub-model of the air distribution system previously identified using data-driven methods. In doing so, TTF trajectories are generated (Section 2.2.3).

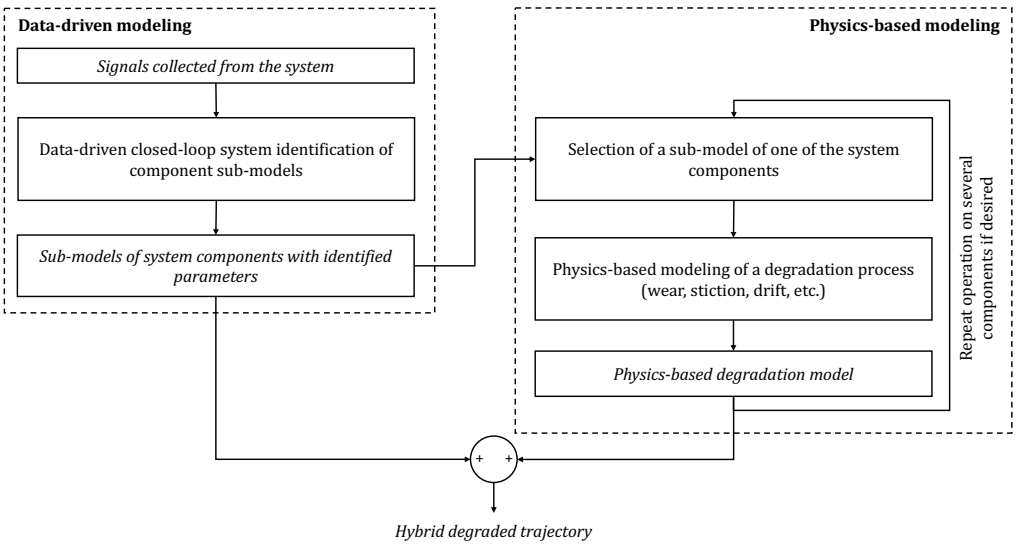


Figure 2.18: The hybrid process of degraded data generation.

2.2.2 The physics-based degradation model

Degraded data augmentation is handled by injecting additional artificial degradation trajectories within the nominal model previously identified. Various models have been developed to describe degradation mechanisms, tailored to specific fields of application. For instance, Arrhenius model is traditionally used for various non-mechanical failure mechanisms, mostly depending on chemical reactions, diffusion processes or migration processes (Arrhenius, 1967). Paris' law and Coffin-Mason model are typically applied to mechanical failure, material fatigue or material deformation, especially crack growth (Pugno et al., 2006). Eyring Model is used to describe changes in the rate of a chemical reaction as a function of temperature or stress (Heckert et al., 2002). These different functional degradation mechanisms across various domains exhibit certain characteristics such as irreversibility and monotinicity. It turns out that the exponential behavior of the failure evolution is common to all these degradation models. Moreover, in practice, similar exponential degradation trends can be observed in most areas. Therefore, these conclusions motivate the choice of a generic exponential degradation model, as had already been done in the design of the C-MAPSS simulator (Saxena et al., 2008).

In the test flight data, one can notice slight oscillations on the valve position. Such a phenomenon can have various origins, including valve clogging (Horch, 2000), leading to a stiction defect. It is therefore decided to build a degradation model based on the effect of stiction on the actuator (i.e. the valve). Figure 2.19 displays the new block diagram of the system. In contrast to block diagram shown in Figure 2.4, the actuator A is now separated from the model G, and integrates the non-linear dynamics of its degradation.

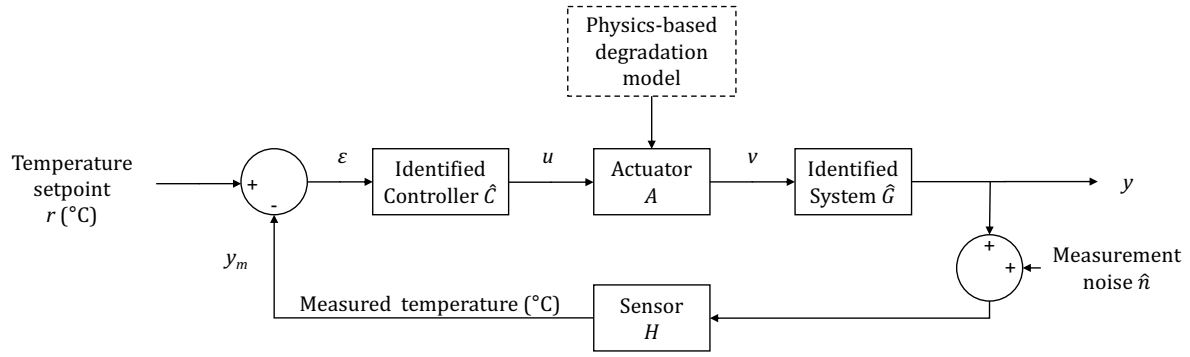


Figure 2.19: The block diagram of the air distribution system of the cockpit integrating the non-linear effects of degradation in the actuator A.

Firstly, a stiction model is implemented at the actuator level. Secondly, an exponential increase of the stiction mechanism is applied by adjusting the parameters of the stiction model as a function of time, following an exponential degradation profile.

One well-established stiction model is as follows (Siraskar, 2021, Choudhury et al., 2004):

$$x_k = \begin{cases} x_{k-1} + (e_k - sign(e_k) f_D), & \text{if } |e_k| > f_S \\ x_{k-1}, & \text{if } |e_k| \leq f_S \end{cases} \tag{2.21}$$

with $e_k = u_k - x_{k-1}$. f_S and f_D are respectively the static and dynamic stiction parameter.

The graphical characteristic of a valve under such a stiction phenomenon is depicted in Figure 2.20.

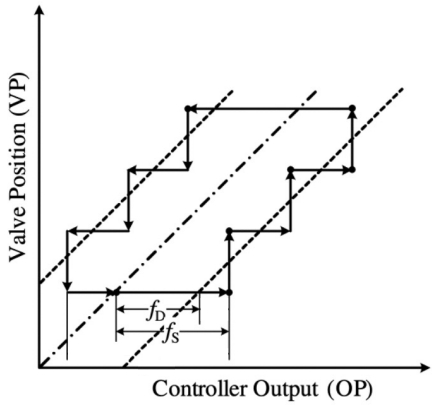


Figure 2.20: Valve stiction modeling (from He and Wang, 2010).

2.2.3 Generation of Time To Failure trajectories

As explained in the previous section, to build a degradation model from the stiction model, it is necessary to adjust the parameters of this model over time. In this case, by increasing the parameters f_S and/or f_D, a degradation of the valve can be simulated. A first failure mode is simulated by increasing the value of the parameter f_S alone, following a time t-dependent exponential degradation model.

$$f_S = \beta e^{\alpha t} \tag{2.22}$$

In practical terms, the static stiction resistance to be overcome increases over the life of the valve, forcing the controller to increment the value of the command sent to the valve for the same movement. This progressively induces a larger oscillation in the setpoint and a jolting behavior of the valve, as it can be seen in Figure 2.21.

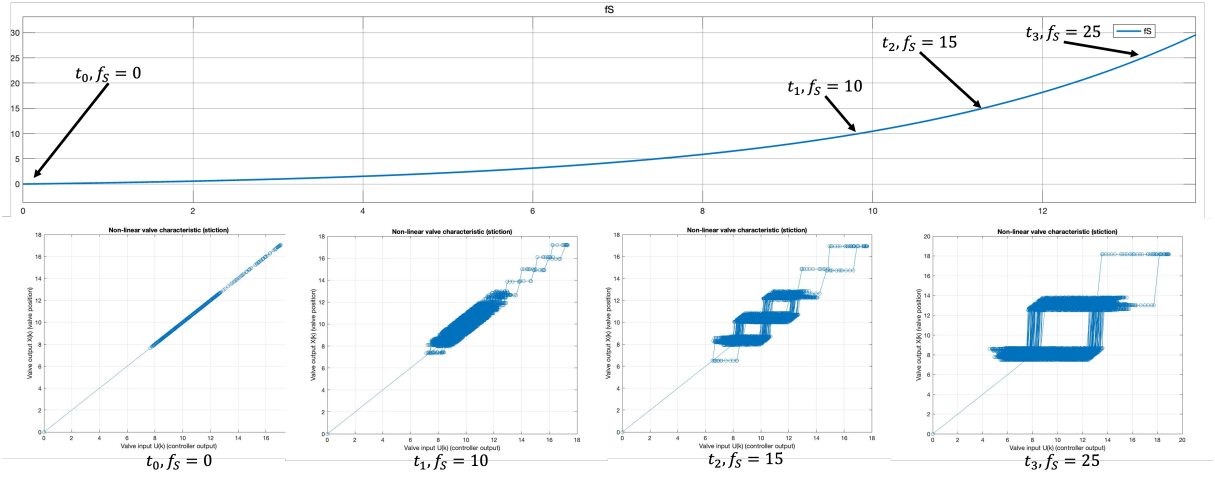
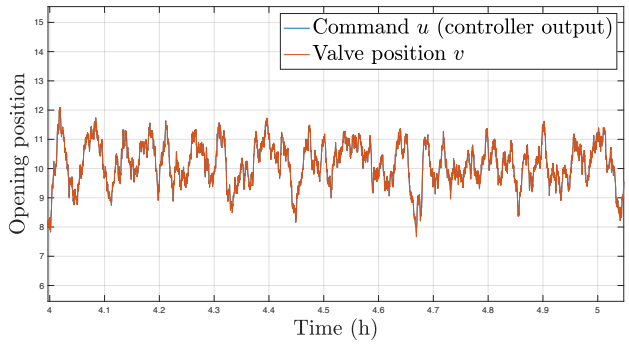
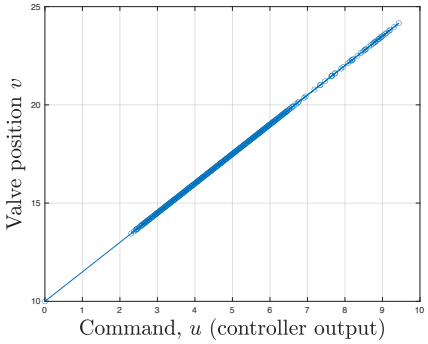


Figure 2.21: The evolution of the valve characteristic as a function of the exponential increase of the f_S coefficient.



(a) Sample plot of the command sent by the controller and the valve position when$f_S = 0$. Both signals are identical in this situation.

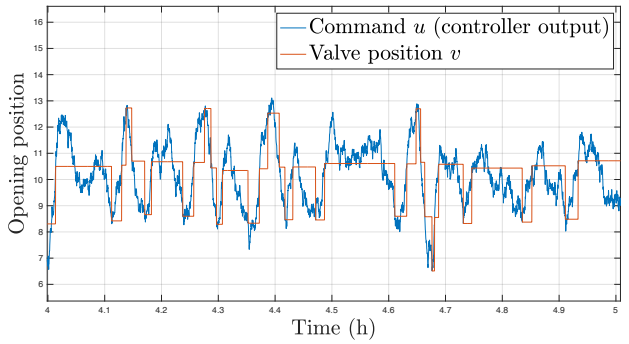


(b) The valve characteristic when $f_S = 0$.

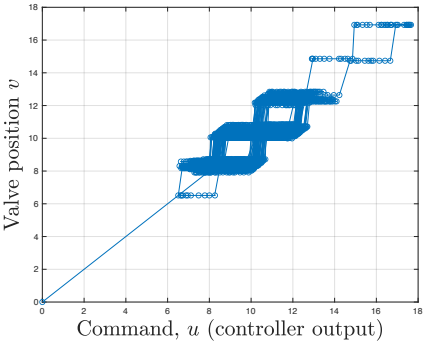
Figure 2.22: Valve behavior when $f_S = 0$ (perfectly linear).

At the very beginning of the life of the system, the valve is considered to be operating without any stiction phenomenon (that is, $f_S = 0$), i.e. respecting a perfectly linear relation between the controller command and the valve position. Such a behavior is described in Figure 2.22, where both variables overlap perfectly.

With increasing age, the valve becomes clogged, resulting in higher dry stiction. It can then be observed that the command sent by the controller is not perfectly followed by the valve, which operates in fits and starts. Figure 2.23a describes such a behavior, for $f_S = 15$.



(a) Sample plot of the command sent by the controller and the valve position when $f_S = 15$.



(b) The valve characteristic when $f_S = 15$.

Figure 2.23: Valve behavior when $f_S = 15$ (i.e. half degradation).

At the very EOL of the system, the valve gets stuck in a position from which it cannot move. This is because the range of the command no longer allows a high enough command to move the valve. The failure is then considered complete and the system has reached its EOL. Multiple TTF trajectories can thus be generated by varying α and β parameters in the exponential profile of the f_S parameter evolution, leading to more or less aggressive degradation and longer or shorter lifetimes. An illustration of such a TTF trajectory is shown in Figure 2.24.

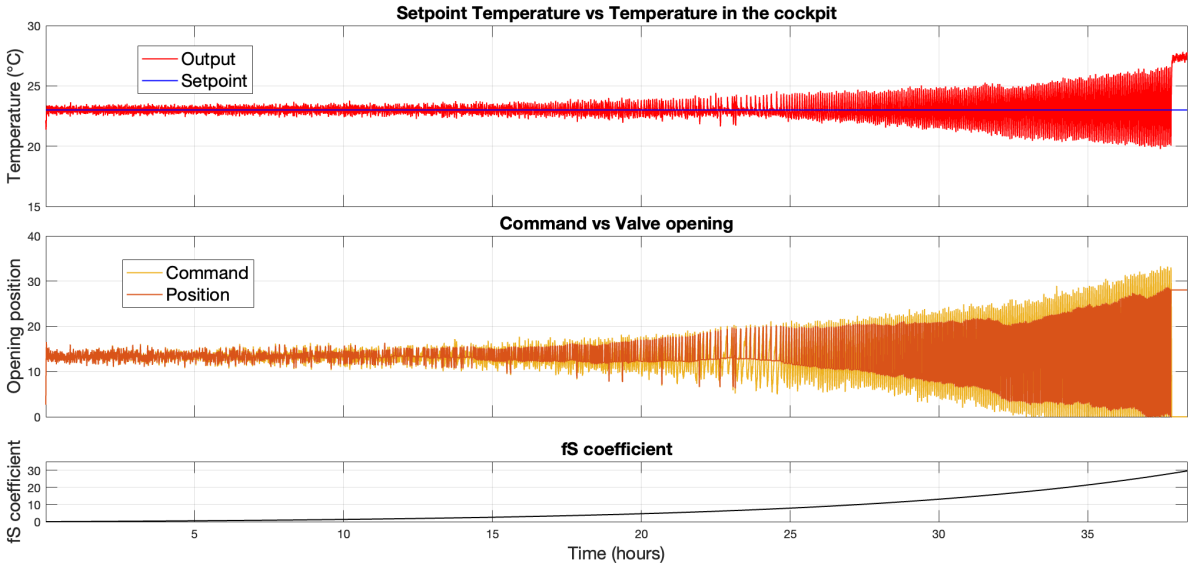


Figure 2.24: A TTF trajectory generated by the prognostics-oriented data augmentation process. It can be seen that the oscillation of the measured temperature increases around the setpoint with the growth of the parameter f_S. At the end of life, the command sent by the controller saturates in 0. Therefore, the valve can no longer be opened and thus remains permanently blocked, ending all control. This corresponds to the EOL instant.

2.3 Conclusion

As demonstrated in Chapter 1, one of the crucial limitations of the PHM is the lack of real data. Most AI-based approaches require large quantities of data, which are not available in real industrial cases. In response to this major problem, a prognostics-oriented data augmentation method is proposed. This approach is divided into two distinct parts.

In the first part, a *system identification-enabled nominal data augmentation process* is proposed (Section 2.1). Such an approach combines three advantages:

1. Firstly, unlike physical models, it is not necessary to have extensive knowledge of the underlying physics laws governing the global system being studied.

2. Secondly, data-driven system identification methods can be used with very few data, allowing new data to be generated with an initial set of reduced size. This is a considerable advantage over AI-based data augmentation methods such as GANs, which require a much larger amount of input data.

3. Thirdly, unlike AI-based data augmentation techniques such as GANs, system identification offers better guarantees of transparency and control over the data generation process. The model structure is chosen on the basis of physical and expert knowledge. Once parameters are tuned, there is mathematical assurance that the data generated by the identified models will not fall outside the scope of these models, whereas data generated by GANs can sometimes be unpredictable and/or outliers. This is a particularly crucial criterion in the present case of application to an aeronautical system. This is a safety-critical system for which the generated data that will be supplied to the neural structures during training

must be fully under control. It is therefore essential to be able to guarantee the physics consistency of the training data.

In the second part, the data is enriched by the injection of degradation derived from a physics-based model inside of a component (Section 2.2). A stiction model is introduced to model the non-linear effects in the actuator (namely, the valve). Note that another component could have been chosen, such as the sensor for example. By varying the model parameters according to an exponential profile (corresponding to most of the degradation mechanisms encountered in mechanical systems), it is then possible to generate TTF trajectories. Once again, this provides complete transparency to the data generated, and ensures consistency with the physics of the degradation mechanism.

This twofold proposal therefore addresses the first two research challenges:

- Firstly, by increasing the amount of data available, it will then be possible to train neural structures for prognostics purposes. Nominal data are increased in quantity, and degraded data are generated from scratch. This provides an answer to research challenge 1.
- Secondly, the proposed approach allows *a priori* knowledge to be integrated in the data augmentation process. This is done both in the modeling of the global system and the choice of models for the nominal data augmentation (Section 2.1), and in the physics-based modeling of the component degradation for the degraded data augmentation (Section 2.2). As a result, since the resulting data will be used as training data for the AI-based models in the next two chapters, this knowledge incorporated into the data will inevitably improve their consistency with physical assumptions about the system and the degradation. This helps to address research challenge 2, offering a reasonable compromise between purely AI-based methods and purely physics-based methods.

Such an approach where explicit prior knowledge is integrated into the training data through physics-based data augmentation is referred to as one of the approaches belonging to the PINN family (see Section 1.2.3).

Chapter 3

Unsupervised Health Index extraction

In this chapter, approaches are developed for extracting the Health Index (HI) from multi-sensor data collected on the system. As mentioned in Section 1.3.2, research work was conducted in two phases. To begin with, studies investigated several approaches based on autoencoder for extracting HI, with an application onto the C-MAPSS academic dataset. Then, in a second phase, an original approach, totally unsupervised and without any use of degradation data, was proposed to address a typical industrial use case (from Dassault Aviation). In this proposed approach, an autoencoder is trained to reconstruct nominal (i.e. healthy) data and latter applied to a Time to Failure (TTF) trajectory, using the reconstruction error as a HI, thus addressing research challenge 1. The rest of this chapter is organized as follows. Section 3.1 describes the basic theory behind autoencoder structures. Section 3.2 introduces two studies applied to C-MAPSS dataset using autoencoders for HI extraction, with their limitations. Section 3.3 presents the conceptual framework for the use of reconstruction error as a HI. Finally, Section 3.4 describes the experimental results obtained on the real industrial case provided by Dassault Aviation.

Contents

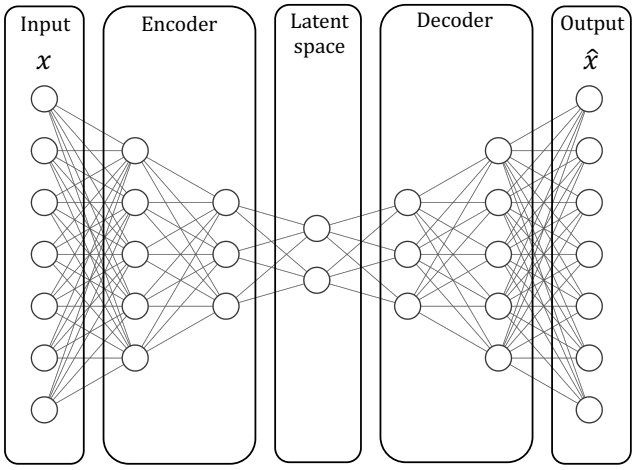


Figure 3.1: An autoencoder structure based on FCLs (called vanilla autoencoder).

3.1 Autoencoder structure

This work proposes to use autoencoders to extract the HI from the data collected on the system by the sensors.

Autoencoder architecture is developed for encoding an input x into a compressed representation called "latent space", denoted as z, and then decoding it to try to reconstruct the original input x (Bengio et al., 2013).

The encoding function is denoted as:

$$z = f_{\theta_e}(x) \tag{3.1}$$

and the decoding function as:

$$\widehat{x} = g_{\theta_d}(z) \tag{3.2}$$

where θ_e and θ_d are the parameters of functions f and g so that the overall learning is accomplished by the following nested function:

$$\widehat{x} = g_{\theta_d}(f_{\theta_e}(x)) \tag{3.3}$$

The training is achieved by minimizing the reconstruction error, which is a function of x and $\widehat{x}$, as defined as follows:

$$J_{AE}(\theta_e, \theta_d) = \sum \mathcal{L}(x, \widehat{x}) = \sum \mathcal{L}(x, g_{\theta_d}(f_{\theta_e}(x))) \tag{3.4}$$

where $\mathcal{L}$ is a loss function such as the Mean Squared Error (MSE).

Encoding and decoding functions can be represented by multiple Fully Connected Layers (FCL) of decreasing size in order to force the computations to flow through a bottleneck representation, namely the latent space. As the latent space has a limited size, the network prioritizes learning the most meaningful features that allow an accurate reconstruction of the input. A typical autoencoder made of FCLs can be seen in Figure 3.1.

Autoencoders offer several advantages that are particularly relevant in the present case:

- Firstly, autoencoders are a class of neural networks characterized as unsupervised learning method, in other words, they do not require any labeled data to train on (Bank et al., 2020). This is a key interest in the field of prognostics where it is very rare to have RUL-labeled data available, as mentionned in Section 1.2.2.

- Secondly, autoencoders have proven to be very efficient at extracting features from raw sensor data (Y. Hu et al., 2016, Gensler et al., 2016). Since extracting a HI from multiple sensors can be seen as a feature extraction task, this is of primary interest for such a task. In particular, it has been shown that the features obtained from autoencoders have monotonicity and clear trendability characteristic that are essential for HI estimation (Y. Hu et al., 2016). Monotonicity is a key characteristic of the HI because it is commonly assumed that industrial components do not undergo self-healing, which would lead to non-monotonic indicator behavior. Trendability is also a crucial factor, as it indicates whether the evolution of the degradation has a regular profile and can thereby be described by a function. Finally, if the HI that is extracted from raw sensor data has both good monotonicity and trendability, the prediction model that will be implemented later will be able to provide good RUL predictions.

- Thirdly, autoencoders are data-related, that is they are able to extract relevant features only from data within similar distribution to data they have been trained on. As such, if the test data presents a drift in its distribution (which is typically the case when a degradation is observed), the reconstruction error will increase following that drift. This specific characteristic of autoencoders will be the basis of the unsupervised approach, presented in Section 3.3.

3.2 Application studies on C-MAPSS dataset

In the first part of the PhD, autoencoders were used in two initial works to extract HI from raw sensor data in different ways. These explorations, both of which have resulted in conference publications, were first steps, applied to C-MAPSS dataset, which then led, in the second phase of the PhD, to the methodological proposition of an approach based on the use of the reconstruction error as HI (presented in Section 3.3). The public C-MAPSS dataset, used in these two studies, is described in detail in Appendix A, where the data selection and preprocessing stages are also described (see Section A.4).

Hereafter, these two works are referred to as "C-MAPSS application 1" (Hervé de Beaulieu et al., 2022b) and "C-MAPSS application 2" (Hervé de Beaulieu et al., 2022c).

3.2.1 C-MAPSS application 1: Direct Health Index extraction with a vanilla autoencoder

Proposed approach

The first approach that has been implemented in this PhD study consists of using the autoencoder structure in a direct manner. Consider a set of raw multi-sensor data of dimension $T_i \times S$ with $T_i \in \mathbb{N}$ as the total lifetime of equipment i and $S \in \mathbb{N}$ the total number of sensors. The problem to be solved can be seen as a dimensionality reduction problem, which is tackled by using a vanilla autoencoder architecture in a temporally-independent manner. At each time step t_k in $t_1 \leq t_k \leq t_{T_i}$ of the multi-dimensional input time series, the vector of sensor readings X_{t_k} is

supplied as input to the autoencoder, which progressively reduces its initial dimension until the latent space, and then reconstructs it through the decoder part.

Algorithm 1: Vanilla autoencoder training for sensor reconstruction.

Input: A set $\mathbf{X}_{train}$ of N multi-dimensional time series $\mathbf{X}_i = \{X_{t_k}\}_{k=1}^{T_i}$ of duration T_i.
Output: A trained vanilla autoencoder model with optimal weights $\theta_e^\star, \theta_d^\star$.
for *iter = 1 to* $\mathcal{K}$ **do**
  for i *= 1 to* N **do**
    for $t_k = t_1$ *to* t_{T_i} **do**
      $\widehat{X}_{t_k} \leftarrow g_{\theta_d}(f_{\theta_e}(X_{t_k}))$;
      $loss \leftarrow loss + \mathcal{L}(X_{t_k}^i, \widehat{X}_{t_k}^i)$;
    end
    $L_{iter} \leftarrow L_{iter} + loss$;
  end
  Update θ_e, θ_d by computing gradient descend on iteration loss L_{iter};
end

The autoencoder is trained in a reconstruction objective, according to the procedure described in Algorithm 1.

After the training phase, the autoencoder is then pruned in the testing phase, in order to conserve only the encoding funnel-shaped part, which reduces the dimensionality of the input data. Each vector of sensor readings data X_{t_k} for t_k in $t_1 \leq t_k \leq t_{\mathcal{T}_i}$ (where $\mathcal{T}_i$ is the total fraction of life available at the current instant) is passed through this pruned autoencoder, thus converting the original multivariate time series of dimension $\mathcal{T}_i \times S$ to a univariate time series of dimension of dimension $\mathcal{T}_i \times 1$. The newly obtained time series is normalized between 0 and 1 using a min-max scaler defined as follows:

$$norm(\mathrm{HI}_{t_k}) = \frac{\mathrm{HI}_{t_k} - min(\mathbf{HI}_{train})}{max(\mathbf{HI}_{train}) - min(\mathbf{HI}_{train})} \tag{3.5}$$

where $\mathbf{HI}_{train} = \{\mathbf{HI}_i\}_{i=1}^N$ is the set of N HI trajectories obtained from the training set. The procedure for obtaining the HI for a single test trajectory is summarized in Algorithm 2.

Algorithm 2: Direct HI extraction procedure using a vanilla autoencoder for one test trajectory.

Input: One test trajectory $\mathbf{X}_i = \{X_{t_k}\}_{k=1}^{\mathcal{T}_i}$ of duration $\mathcal{T}_i$.
Output: A HI trajectory $\mathbf{HI}_i = \{\mathrm{HI}_{t_k}\}_{k=1}^{\mathcal{T}_i}$.
for $t_k = t_1$ *to* $t_{\mathcal{T}_i}$ **do**
  $\mathrm{HI}_{t_k} \leftarrow f_{\theta_e^\star}(X_{t_k})$;
  $\mathrm{HI}_{t_k} \leftarrow norm(\mathrm{HI}_{t_k})$
end

The univariate and normalized time series that is obtained from such a process can be called a Virtual Health Index (VHI), since it is the fusion of an initial set of sensors, which therefore no longer has any physical interpretation.

Assuming that the degradation is an information that is perceptible in the variables recorded by the sensors (even in a hidden way), it is therefore expected that, by merging these different

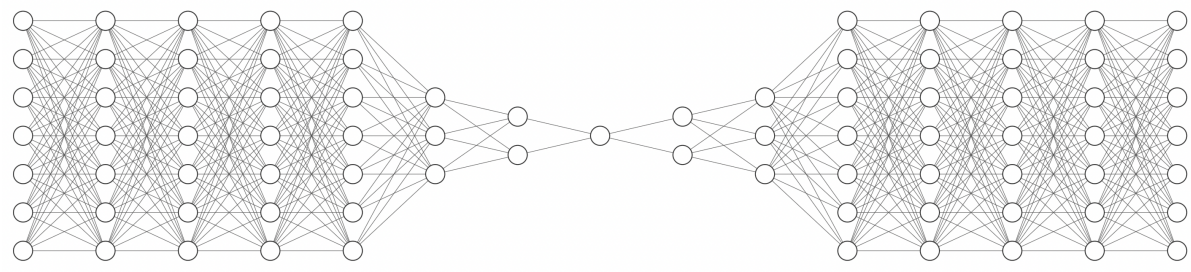


Figure 3.2: Vanilla autoencoder structure for direct HI extraction.

sensors, the main dynamic of the HI thus created will actually be the degradation trend. This is therefore tested on the C-MAPSS dataset.

Results on C-MAPSS dataset

As described in detail in Annex A.4, a subset of seven sensors is pre-selected from the C-MAPSS data. Therefore, the input FCL of the encoder contains seven neurons. The rest of the encoder is made of 5 FCLs of same size, followed by three layers of decreasing size. The initial accumulation of the first five layers of identical size will increase the depth of the network, enabling potential hidden features to be detected from the inputs and building up several layers of abstraction. Next, the layers of decreasing size are used to progressively reduce the size of the input vector, until the desired dimension of 1 is reached in the latent space.The decoder is symmetrical to the encoder. Following best practice proposed by Srivastava et al., 2014, a dropout of 0.2 is applied between each layer to prevent from over-fitting. The architecture of the autoencoder can be seen in Figure 3.2 and the set of hyper-parameters for this implementation is summarized in Table 3.2.

Hyper-parameters	Values
Size of encoder layers	7, 7, 7, 7, 7, 3, 2, 1
Size of decoder layers	2, 3, 7, 7, 7, 7, 7
Dropout	0.2
Iterations (also called Epochs)	100

Table 3.1: Set of hyper-parameters for C-MAPSS Application 1.

The extracted HI of one given turbine from the FD001 set, along with its corresponding raw sensor data can be seen in Figure 3.3. Such a HI presents, through a visual observation, a clear increasing and global monotonic trend. It is worth noting that all the turbines from the test set are normalized together. As such, HI values are all contained in a $[0, 1]$ interval, but might not reach its bounds. For instance, one can see in Figure 3.3 that the maximum value is 0.82, not 1. This creates a certain disparity between the HIs obtained, and hence introduces a variability in the data, that is likely to cause subsequent problems in estimating the RUL. It is worth pointing out that a complete trajectory such as the one shown in Figure 3.3 serves as a validation for the direct HI extraction approach, but in a test case, the raw data is obviously not available until the EOL. The HI is then extracted up to the present moment, denoted as $\mathcal{T}_i$ in Algorithm 2, and it is at this point that the prediction of future HI values becomes necessary.

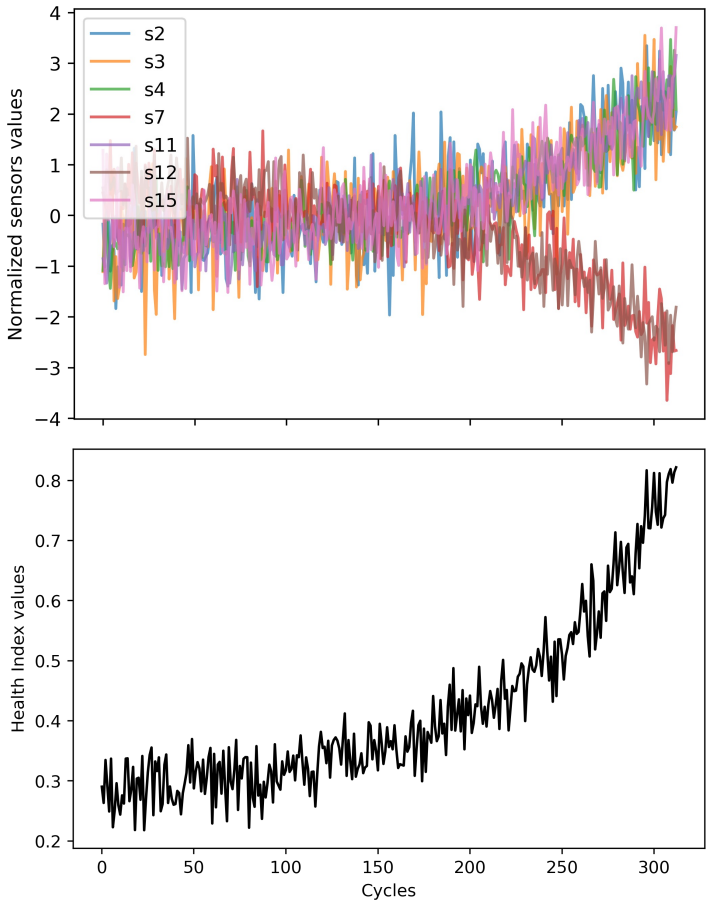


Figure 3.3: C-MAPSS application 1: Example of one complete HI trajectory along with raw sensor data for one turbine of the FD001 set.

3.2.2 C-MAPSS application 2: Direct Health Index extraction with a convolutional autoencoder

The second application based on an autoencoder for HI extraction involves a more elaborate structure incorporating Convolutional Neural Network (CNN). As presented in Section 1.2.1, CNN is a specific kind of deep learning model which has shown excellent abilities for feature extraction, especially in the domain of image classification, speech recognition and time series prediction (LeCun, Bengio, et al., 1995). CNNs have also been effectively applied to prognostics (Babu et al., 2016, X. Li et al., 2018), in the context of direct mapping from raw sensor data to RUL prediction. Furthermore, it has already been pointed out in Section 3.1 that autoencoder structures are very efficient in extracting HI from raw sensor data and are well suited for treating multivariate non-stationary data. In particular, it has been shown that automatically extracted features are preferable to hand-crafted features in the case of bearing vibrations (Y. Hu et al., 2016), exhibiting monotonicity and clear trendability. Therefore, embedding a CNN model inside of an autoencoder structure is a reasonable approach that should provide good HI extraction capabilities from raw sensor data.

In the remainder of this part, fundamental elements concerning CNNs will be provided. Then, the proposed approach combining CNNs with autoencoders for HI extraction will be presented and the experimental results will be described.

Convolutional Neural Networks for time series

In CNN autoencoders, the encoding and decoding functions f_{θ_e} and g_{θ_d} are realized by CNN layers. This consists of performing a convolution product between the input denoted as I and a kernel denoted as F (also called filter).

The convolution operation consists of producing, on the basis of two functions u and v, a third one that expresses how the shape of one is modified by the other. It is defined as follows:

$$s(t) = \int u(a)v(t-a)da. \tag{3.6}$$

The convolution product is usually denoted using an asterisk:

$$s(t) = (u * v)(t). \tag{3.7}$$

Numerical data, such as images or sound clips, are stored as multi-dimensional arrays. They have one or more axes for which ordering matters (for instance, width and height axes for images, time axis for a sound clips) and one axis, called the channel axis, that is used to access the different views of the data (for instance, RGB channels of a color image).

To process such numerical data, a discrete convolution is used, which is a linear transformation that preserves this notion of ordering. It is sparse (i.e. only a small number of input units contribute to a given output unit) and reuses parameters (the same weights are applied to multiple locations in the input) (Dumoulin and Visin, 2016).

In convolutional network terminology, the first argument (in current notation, function u) of the convolution is called the input, and the second argument (in current notation, function v) is called the kernel or filter. The output $s(t)$ is usually called the feature map.

The convolution product for an image or a multivariate time series (2D) is thus expressed as follows (Goodfellow et al., 2016):

$$S(i,j) = (I * K)(i,j) = \sum_{i=1}^{m}\sum_{j=1}^{n} I(m,n)K(i-m, j-n) \tag{3.8}$$

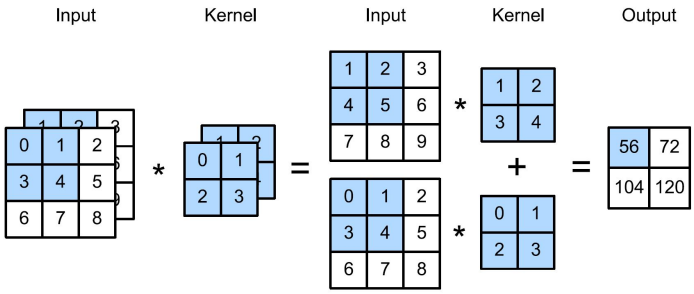


Figure 3.4: Computation of a convolution on a 2-channel 2-dimensional input (A. Zhang et al., 2021) with a $2 \times 3 \times 3$ input and a $2 \times 2 \times 2$ kernel.

with a two-dimensional input I and a two-dimensional kernel K.

Figure 3.4 illustrates a simple example of two-dimensional convolution. Each kernel slides across the input, with respect to the corresponding channel. At each position, the element-wise product between the kernel and the input elements that it is overlapping is calculated and then summed to give the output at the current position. The procedure is then repeated over the whole input.

If the input has multiple channels (such as in Figure 3.4, where the input has two channels), the kernel to be applied must have the same number of channels (i.e. being a 3-dimensional kernel) and the resulting feature maps are then summed together element-wise to form the output feature map.

Therefore, assigning several kernels to the same input has the effect of multiplying the output feature maps, thus increasing the depth of the extracted features. For instance, in Figure 3.5, three kernels are applied, thus performing three convolutions, and therefore leading to an output feature maps of three channels.

The sliding step of the convolution operation is called the stride, of values denoted as $s_h \times s_w$. This can be understood as a sub-sampling operation. In addition, it is worth noting that multiple convolutions will lead to a reduction of the available information at the boundaries of the original input. For instance, 10 layers of 5×5 kernel convolutions over an input of initial size of 240×240 would reduce the output feature map size to 200×200, thereby deleting any interesting information on the boundaries. To avoid such a loss of information, padding is often used. It consists of adding extra cells around the boundaries of the input, thus increasing the effective size of the input (the extra cells typically assume the value 0, but many other padding types exist, such as mirror padding, constant padding, etc.). Figure 3.6 illustrates a zero-padding operation with a dimension of 1×1.

Considering stride, padding and the size reduction phenomenon inherent to convolution, the dimensions of an output feature map can be computed as follows:

$$\frac{n_i - k_i + p_i + s_i}{s_i} \times \frac{n_j - k_j + p_j + s_j}{s_j} \tag{3.9}$$

with (n_i, n_j) the input dimensions, (k_i, k_j) the kernel dimensions, (s_i, s_j) the stride and (p_i, p_j) the padding. Various results can then be deduced by applying certain best practices such as symmetrical padding, odd kernel dimensions, etc.

Finally, pooling operation aims to reduce the size of feature maps by using some function to

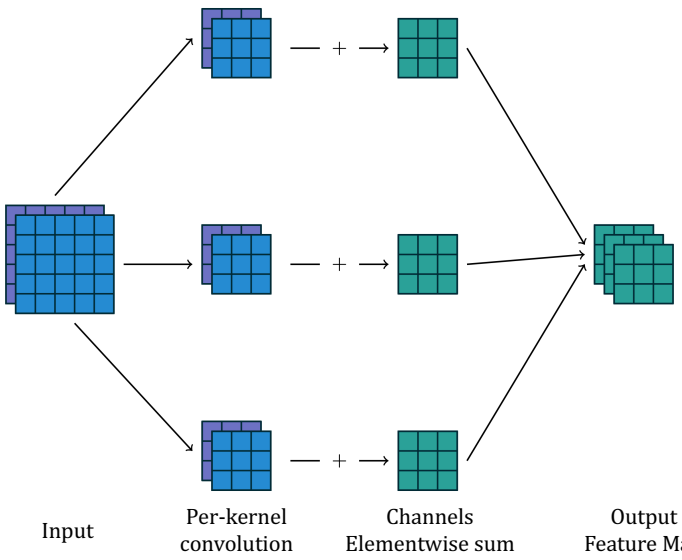


Figure 3.5: Convolution operation with multiple kernels (Dumoulin and Visin, 2016). The input is of dimension 2 channels×5×5, and 3 kernels of size 2 channels×3×3 are applied. A convolution is performed for each kernel, thus producing each time 2 feature maps which are summed together element-wise. At the end, the output feature map is therefore of size 3 channels × 3 × 3.

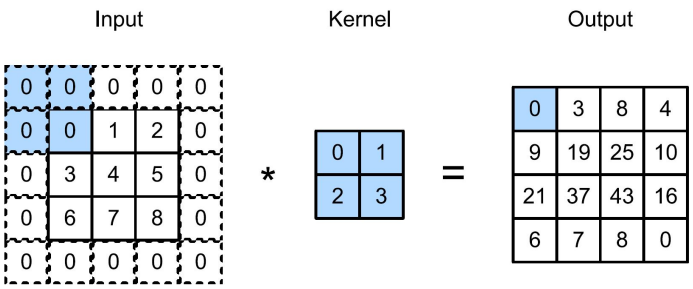


Figure 3.6: Zero-padding operation (A. Zhang et al., 2021).

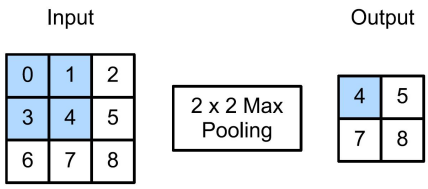


Figure 3.7: Max-pooling operation (from A. Zhang et al., 2021).

summarize sub-regions, such as the average or the maximum value. Pooling works in a similar way to discrete convolution, but instead of using the linear combination described by the kernel, it uses a different function. The most common pooling operations are average-pooling and max-pooling (which can be visualized in Figure 3.7).

In the specific context of time series processing, a one-dimensional convolution is applied along time, since this dimension must be kept intact. This operation is expressed as follows:

$$S(t) = (I * F)(t) = \sum_{t=1}^{T} I(t)F(t-T) \tag{3.10}$$

where S is the output (also called feature map), I is the input, F is the kernel, T is the overall length of the input and $*$ is the convolution product (Goodfellow et al., 2016).

Proposed approach

Consider a set of multi-sensor data of dimension $T_i \times S$ with $T_i \in \mathbb{N}$ the total lifetime of equipment i and $S \in \mathbb{N}$ the total number of sensors. These series can therefore be treated as one-dimensional data with S channels. Then, by adjusting the number of kernels, it is possible to artificially increase the number of channels (corresponding to constructing additional artificial features) or to reduce it, in order to obtain a one-dimensional HI. An autoencoder structure embedding CNN layers is thus proposed, compressing in the encoding part the multi-dimensional data to a one-dimensional time series in the latent space, and then expanding it in the decoding part to its original dimension. The autoencoder structure is thus used in the same spirit as in C-MAPSS Application 1 (see Section 3.2.1). However, in the current method, the time series are supplied to the CNN-based autoencoder in their entirety, and are therefore processed all at once, rather than time step-by-time step. The training procedure of the CNN-based autoencoder is summarized in Algorithm 3.

The procedure for obtaining the HI using the CNN-based autoencoder for a single test trajectory is summarized in Algorithm 4. In the same way as for C-MAPSS Application 1, the test trajectory is denoted as $\mathbf{X}_i = \{X_{t_k}\}_{k=1}^{\mathcal{T}_i}$ where $\mathcal{T}_i$ is the total fraction of life available at the current instant ($\mathcal{T}_i \neq T_i$).

Results on C-MAPSS dataset

The proposed CNN-based autoencoder architecture is depicted in Figure 3.8.

The encoder is made of seven one-dimensional convolution layers. In order to extract deep features, the first three layers are designed to increase the depth of the input data by expanding the number of channels from $S = 7$ to $S' = 56$. The four next layers then compress it to obtain

Algorithm 3: CNN-based autoencoder training for sensor reconstruction.

Input: A set $\mathbf{X}_{train}$ of N multi-dimensional time series $\mathbf{X}_i = \{X_{t_k}\}_{k=1}^{T_i}$ of duration T_i.
Output: A trained CNN-based autoencoder model with optimal weights $\theta_e^{\text{CNN}\star}, \theta_d^{\text{CNN}\star}$.
for *iter = 1 to* $\mathcal{K}$ **do**
  for i *= 1 to* N **do**
    $\widehat{\mathbf{X}_i} \leftarrow g_{\theta_d^{\text{CNN}}}(f_{\theta_e^{\text{CNN}}}(\mathbf{X}_i))$;
    $L_{iter} \leftarrow L_{iter} + \mathcal{L}(\mathbf{X}_i, \widehat{\mathbf{X}_i})$;
  end
  Update $\theta_e^{\text{CNN}}, \theta_d^{\text{CNN}}$ by computing gradient descend on iteration loss L_{iter};
end

Algorithm 4: Direct HI extraction procedure using a CNN-based autoencoder for one test trajectory.

Input: One test trajectory $\mathbf{X}_i = \{X_{t_k}\}_{k=1}^{\mathcal{T}_i}$ of duration $\mathcal{T}_i$.
Output: A HI trajectory $\mathbf{HI}_i = \{\text{HI}_{t_k}\}_{k=1}^{\mathcal{T}_i}$.
$\mathbf{HI}_i \leftarrow f_{\theta_e^{\text{CNN}\star}}(\mathbf{X}_i)$;

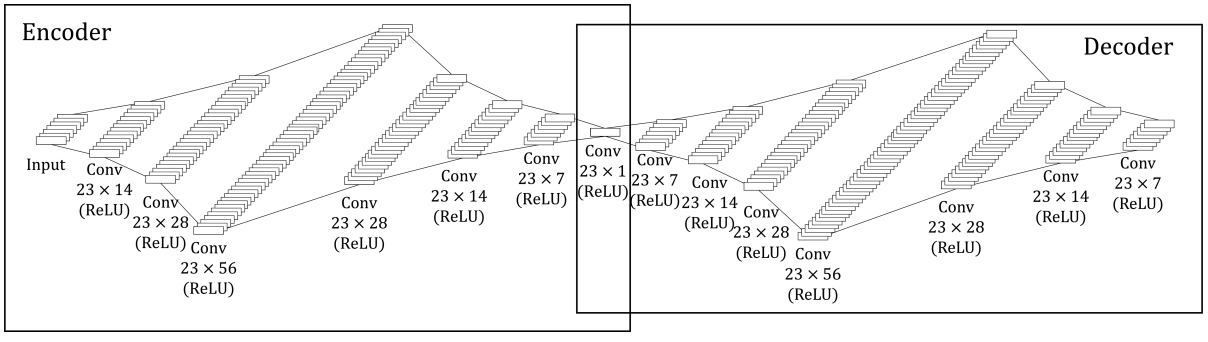


Figure 3.8: Proposed deep CNN autoencoder structure for HI extraction.

a univariate HI in the latent space (i.e. $S = 1$). The CNN-based decoder is built as a mirror of the encoder, similarly to the architecture proposed in Section 3.2.1.

As pointed out in Equation 3.9, the succession of convolutions has the effect of reducing the size of the data at the extremes. However, it is crucial to keep the duration of the time series unchanged, since the aim is to compress the sensor dimension but not to change the time dimension. Therefore, the hyper-parameters of the convolution layers are carefully chosen to ensure that the initial length of the time series remains unchanged over the convolution operations. Following the formula indicated in Equation 3.9, stride $s = 1$; kernel length $k_l = 23$ and symmetrical padding $p = 11$ guarantee that the length of the feature map is equal to the length of the input. Between each convolution, a ReLU activation function is applied to ensure non-linearity, complying with best practices recommended by the community. The complete set of hyper-parameters for this implementation is summarized in Table 3.2.

Hyper-parameters	Values
Size of encoder layers (i.e. number of kernel applied)	14, 28, 56, 28, 14, 7, 1
Size of decoder layers	7, 14, 28, 56, 28, 14, 7
Iterations (also called Epochs)	100
Kernel length k_l	23
Padding (on both sides)	11
Stride s	1

Table 3.2: Set of hyper-parameters for C-MAPSS Application 2.

The data preparation step is the same as the one applied for C-MAPSS Application 1. It is described in detail in Appendix A.4. However, since the time series are supplied to the CNN-based autoencoder in their totality and not time step-by-time step, it is consequently necessary to normalize their length. Therefore, a zero pre-padding operation is applied, in order to force all time series to have the same length (Dwarampudi and Reddy, 2019). After the HI has been obtained, the univariate HI is "unpadded" to return to its original length. Figure 3.9 depicts an example of padded and normalized multi-variate sensor time series given as input to the CNN-based autoencoder.

The HI obtained has no physical sense and can be classified as VHI. As a result, it should not be considered as a physical measure of the equipment SOH. In particular, the amplitude variations observed on the curve do not indicate deterioration or recovery in the SOH of the studied system.

In fact, it turns out that, as in the case of image processing, the CNN layers have treated the EOL as a pattern. This can be clearly seen in Figure 3.10, where the extracted HI along with the true RUL are plotted together, after "unpadding" the data.

The so-called "EOL pattern", which characterizes the moment when the EOL of the studied equipment occurs, is a local minimum whose shape and position are always identical for all the turbines available in the FD001 training set. All of the EOL patterns for these turbines can be seen in Figure 3.11. Since this pattern is found immediately preceding the EOL of each of the turbines in the training set, it is assumed that it will also be present for unseen turbines, and that it can therefore be used as a marker to detect the EOL. This will be described fully in Section 4.3.2.

It is worth noting that in Figures 3.9 to 3.11, complete trajectories (i.e. until EOL) are displayed, in order to analyze the proposed HI extraction approach using CNN-based autoencoder. In a test case, the HI is extracted up to the present moment (i.e. up to the point where no more

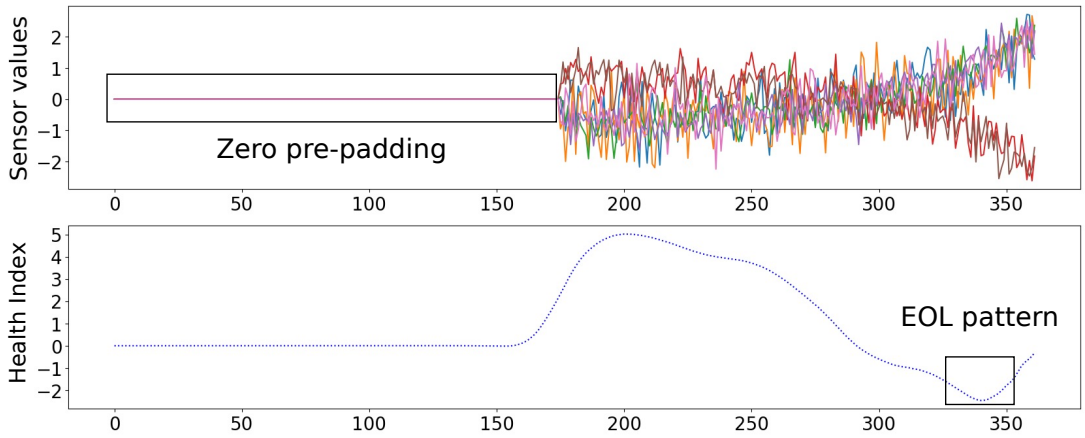


Figure 3.9: Example of HI obtained with CNN-based autoencoder architecture. Top plot: the multi-variate input data of one turbine from FD001 training set. The data is standardized and pre-padding has been applied to normalize the length. Bottom plot: the corresponding HI extracted by the CNN-based autoencoder.

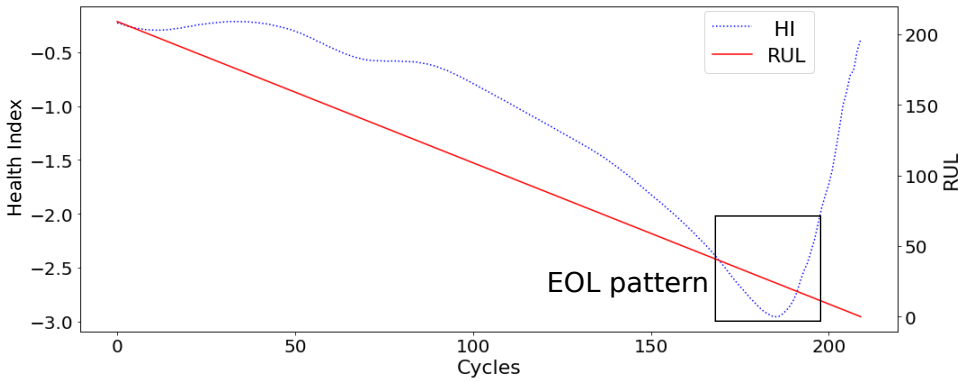


Figure 3.10: Example of the extracted HI along with true RUL for a complete trajectory of one turbine from the FD001 training set. It can be clearly seen that the EOL pattern appears right before the true EOL (i.e. when the RUL reaches zero).

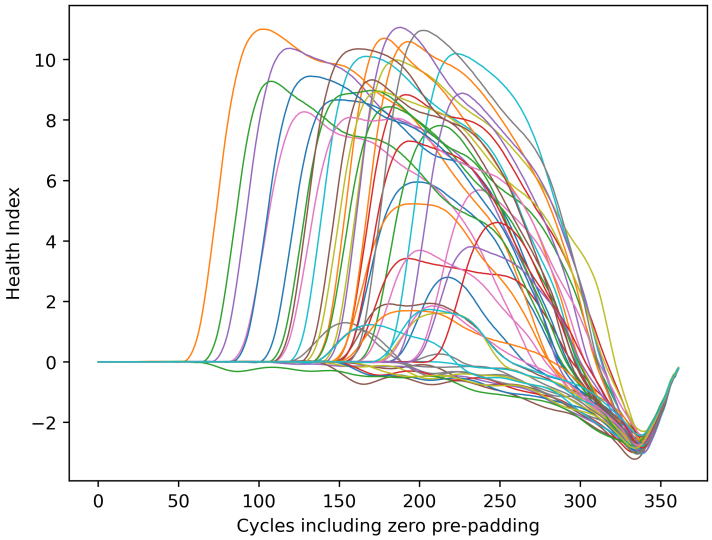


Figure 3.11: The EOL patterns of all turbines in the FD001 training set.

raw data is available), denoted as $\mathcal{T}_i$ in Algorithm 4, and from this point HI forecast must be performed.

3.2.3 Conclusion of C-MAPSS applications 1 and 2

The two C-MAPSS applications explored two approaches to extract HI in a direct manner, using various autoencoder structures. Such structures have proven to be efficient in extracting HI from raw sensor data.

In the first one, a vanilla autoencoder was exploited time step-by-time step as a tool for dimension reduction of the raw input data. It allows to easily obtain a HI but it remains naive from a temporal point of view, and therefore cannot consider the dynamics that drive the system over time. Moreover, there is no guarantee that the residual dynamics which is extracted from the sensor compression is representative of the system SOH. Finally, it is not easy to set the value of the HI level corresponding to the EOL, given the variability of the indicator.

In the second approach, a more complex CNN-based autoencoder processes the entire time series supplied. The HI obtained is deprived of any possibility of physical interpretation and the EOL is marked by a pattern rather than by a threshold value. The existence of this pattern is highly advantageous, as it eliminates the requirement to normalize the HI. It is then simply a matter of spotting the pattern to detect the EOL, thus avoiding the thorny problem of choosing a HI threshold.

While both approaches avoid direct dependence on RUL-labeled data, both require complete TTF trajectories as training data. However, as mentioned above, such data is generally unavailable for most industrial applications, including the Dassault Aviation case study. For this reason, an indirect approach is proposed in the following, which totally avoids the use of TTF trajectories.

3.3 Health Index extraction based on reconstruction error

Since in most real industrial cases, nominal (i.e. healthy) data is available but no data under degradation, the proposition made in this work is to train the autoencoder exclusively on nominal data. Therefore, the autoencoder structure is trained to reconstruct nominal operating time series, i.e. when the system has not yet been degraded. The autoencoder therefore learns a basic representation of the nominal operating behavior of the system. During this training phase, both real data samples and generated ones (obtained by the *system identification enabled nominal data augmentation* process described in Chapter 2) can be used.

Later on, this nominal reconstruction model is used to reconstruct complete TTF time series, including degraded operation of the system. The reconstruction error, i.e. the difference between real signals and reconstructed signals is then computed to obtain a HI estimation. With this approach, no degradation data is used during training. The degradation progressively appears in the HI through the gradual shift that it creates in the distribution of data collected by sensors over the life of the system (Malhotra et al., 2016).

Consider a nominal training domain

$$D_N = \left\{\mathbf{X}_N^i\right\}_{i=1}^{N_N} \tag{3.11}$$

where N_N is the number of nominal training samples. Each sample $\mathbf{X}_N^i$ belongs to a *nominal feature space* $\mathcal{X}_N$. $\mathbf{X}_N^i$ denotes a nominal multivariate sequential data sample of duration $\mathcal{T}_i$ and with S features. The nominal training data samples $\mathbf{X}_N^i$ are sliced into several windows:

$$\mathbf{X}_N^i(t_w) = \left\{X_{t_k}^i\right\}_{k=w}^{w+\Delta} \tag{3.12}$$

with t_w as the start time step of the window and Δ as its total duration.

Parameters θ_e and θ_d of the encoder and decoder are learned on the basis of the nominal training data, i.e. with respect to the *nominal feature space* $\mathcal{X}_N$. To that end, the parameters are updated at each training iteration following the gradient of the reconstruction error:

$$E(t_w) = \left\|\mathbf{X}_N^i(t_w) - g_{\theta_d}(f_{\theta_e}(\mathbf{X}_N^i(t_w)))\right\| \tag{3.13}$$

Once the training has been accomplished, the optimal parameters $\theta_e^\star$ and $\theta_d^\star$ of the encoder and decoder are frozen.

Let us now consider a system that is deteriorating. A TTF trajectory is then denoted as $\mathbf{X}_D^i$ and belongs to a *degraded feature space* $\mathcal{X}_D$.

To be more precise, a TTF trajectory that is being recorded on a running device belongs to the *nominal feature space* $\mathcal{X}_N$ only at the very beginning of the life of the system. Then, as the life of the system evolves, the actual feature space, including degradation, moves away from the initial *nominal feature space*.

As the *nominal feature space* and *degraded feature space* are different ($\mathcal{X}_N \neq \mathcal{X}_D$), the associated probability distribution are different as well. More formally, the distribution of data samples belonging to the *degraded feature space* drifts increasingly from the distribution of nominal samples as the system approaches its failure (i.e its EOL):

$$P(\mathbf{X}_N^i) \neq P(\mathbf{X}_D^i). \tag{3.14}$$

For each window of a TTF trajectory, the HI is obtained by computing the total reconstruction error:

$$E_{total}(t_w) = \sum_{k=w}^{w+\Delta} \left\| \mathbf{X}_D^i(t_k) - g_{\theta_d^\star}(f_{\theta_e^\star}(\mathbf{X}_D^i(t_k))) \right\| \tag{3.15}$$

Due to the distribution shift in $P(\mathbf{X}_D^i)$, the reconstruction error E_{total} continues to grow until the system fails completely. This time varying, certainly increasing reconstruction error is used as an indicator of the SOH, namely as HI.

3.4 Experimental implementation on Dassault Aviation use case

The method of unsupervised extraction of HI based on the reconstruction error is evaluated on the industrial case study provided by Dassault Aviation. The process can be visualized in Figure 3.12, which distinguishes the training part (Figure 3.12a) from the testing part (Figure 3.12b).

3.4.1 Experiment details

For the training phase, it has been stated in the previous section that N_N training samples are considered. In other words, let us consider N_N nominal flights operated by identical category aircraft, with index $1 \leq i \leq N_N$, $i \in \mathbb{N}$ for which data from S sensors $1 \leq s \leq S$, $s \in \mathbb{N}$ are collected. Precisely, four signals are collected here (temperature setpoint r, measured temperature y_m, valve position v, command u), i.e. $S = 4$.

Each training flight record i has a total duration denoted as $\mathcal{T}_i$, with index $t_1 \leq t_k \leq t_{\mathcal{T}_i}$, $k \in \mathbb{N}$.

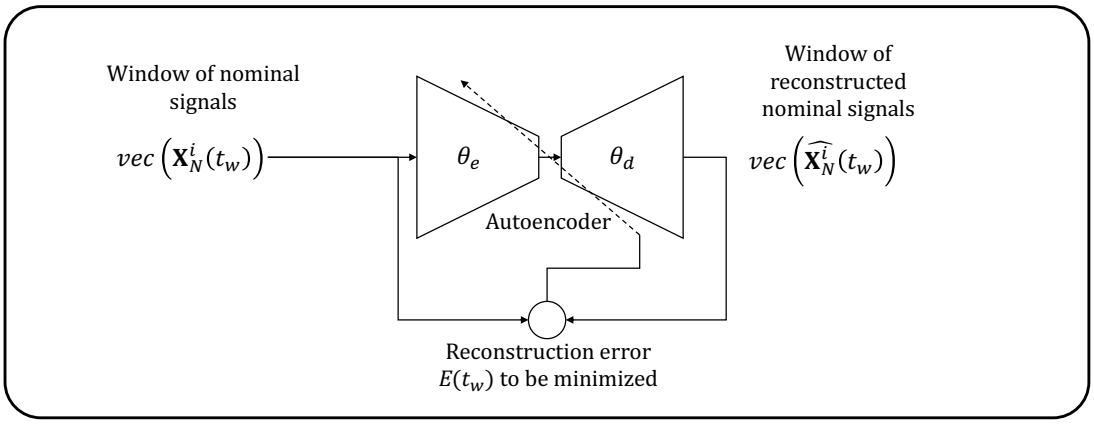
The nominal training set is thus the collection of objects: $\{\mathbf{X}_N^i\}_{i=1}^{N_N}$ with each object $\mathbf{X}_N^i \in \mathbb{R}^{\mathcal{T}_i \times S}$. Therefore, the s-th column $X^{s(i)}$ corresponds to the vector of values of sensor s for the entire duration of the flight i (i.e. for all time steps $t_1 \leq t_k \leq t_{\mathcal{T}_i}$) and the t_k-th row ${X_{t_k}}^{(i)}$ corresponds to the vector of values of all sensor $1 \leq s \leq S$ for the given time step t_k. Finally, the scalar ${X_{t_k}^s}^{(i)}$ is the single value recorded by sensor s at time step t_k on flight i. Note that the sensor number S is identical for all flights, while the flight duration $\mathcal{T}_i$ varies for each flight i. This can be summarized as follows:

$$\mathbf{X}_N^i = \begin{bmatrix} {X_{t_1}^1}^{(i)} & {X_{t_1}^2}^{(i)} & \cdots & {X_{t_1}^S}^{(i)} \\ {X_{t_2}^1}^{(i)} & {X_{t_2}^2}^{(i)} & \cdots & {X_{t_2}^S}^{(i)} \\ \cdots & \cdots & \ddots & \cdots \\ {X_{t_{\mathcal{T}_i}}^1}^{(i)} & {X_{t_{\mathcal{T}_i}}^2}^{(i)} & \cdots & {X_{t_{\mathcal{T}_i}}^S}^{(i)} \end{bmatrix} = \begin{bmatrix} {X_{t_1}}^{(i)} \\ {X_{t_2}}^{(i)} \\ \cdots \\ {X_{t_{\mathcal{T}_i}}}^{(i)} \end{bmatrix} = \begin{bmatrix} X^{1(i)} & X^{2(i)} & \cdots & X^{S(i)} \end{bmatrix} \tag{3.16}$$

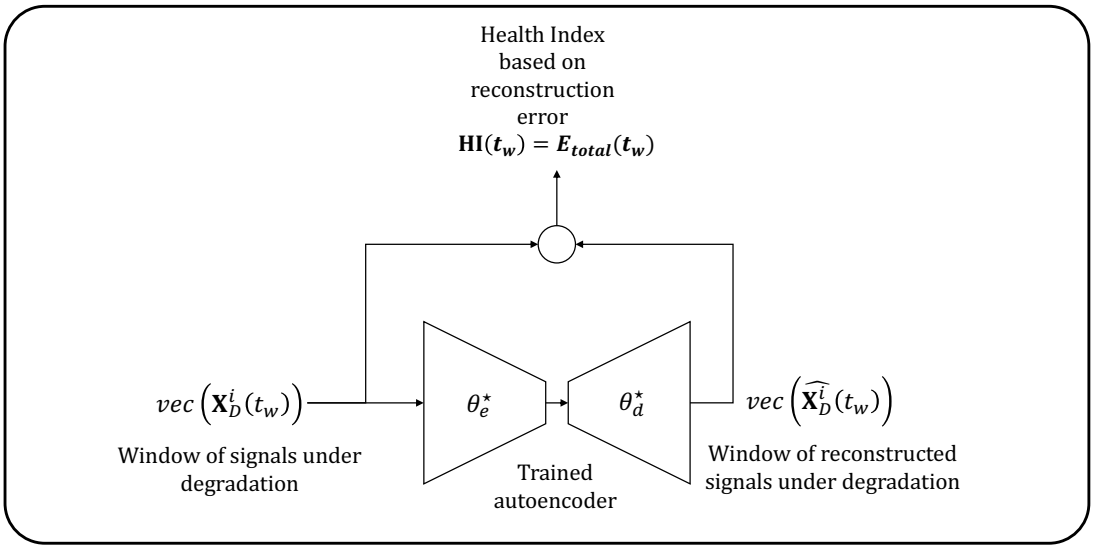
As described in Section 3.3, the nominal training data samples are divided into several windows:

$$\mathbf{X}_N^i(t_w) = \left\{ {X_{t_k}}^{(i)} \right\}_{k=w}^{w+\Delta} \in \mathbb{R}^{\Delta \times S} \tag{3.17}$$

with t_w as the start time step of the time window and $\Delta = 60s$ as its total duration. Δ is a hyper-parameter that depends on the system under study and its dynamics. In particular, $\Delta = 60s$ proves to be an appropriate range for capturing the dynamics of the control in operation. Using the sensor notation of the case study, this gives:



(a) Training phase. Learning of the parameters θ_e and θ_d of the encoder and decoder by minimizing reconstruction error E



(b) Testing phase. The optimal parameters $\theta_e^\star$ and $\theta_{d^\star}$ are frozen and the total reconstruction error E_{total} is used as HI.

Figure 3.12: Overall process of the unsupervised extraction of HI based on the reconstruction error.

$$\begin{aligned}\mathbf{X}_N^i(t_w) &= \begin{bmatrix} X_{t_w}^{1\ (i)} & X_{t_w}^{2\ (i)} & \cdots & X_{t_w}^{S\ (i)} \\ X_{t_{w+1}}^{1\ (i)} & X_{t_{w+1}}^{2\ (i)} & \cdots & X_{t_{w+1}}^{S\ (i)} \\ \cdots & \cdots & \ddots & \cdots \\ X_{t_{w+\Delta}}^{1\ (i)} & X_{t_{w+\Delta}}^{2\ (i)} & \cdots & X_{t_{w+\Delta}}^{S\ (i)} \end{bmatrix} \\ &= \begin{bmatrix} r_{\ t_w}^{\ (i)} & u_{\ t_w}^{\ (i)} & y_{m\ t_w}^{\ (i)} & v_{\ t_w}^{\ (i)} \\ r_{\ t_{w+1}}^{\ (i)} & u_{\ t_{w+1}}^{\ (i)} & y_{m\ t_{w+1}}^{\ (i)} & v_{\ t_{w+1}}^{\ (i)} \\ \cdots & \cdots & \cdots & \cdots \\ r_{\ t_{w+\Delta}}^{\ (i)} & u_{\ t_{w+\Delta}}^{\ (i)} & y_{m\ t_{w+\Delta}}^{\ (i)} & v_{\ t_{w+\Delta}}^{\ (i)} \end{bmatrix}\end{aligned} \tag{3.18}$$

As the autoencoder is made of FCL, each window is vectorized so that:

$$\begin{aligned}vec(\mathbf{X}_N^i(t_w)) = [&r_{\ t_w}^{\ (i)}, r_{\ t_{w+1}}^{\ (i)}, \cdots, r_{\ t_{w+\Delta}}^{\ (i)}, u_{\ t_w}^{\ (i)}, u_{\ t_{w+1}}^{\ (i)}, \cdots, u_{\ t_{w+\Delta}}^{\ (i)}, \\ &y_{m\ t_w}^{\ (i)}, y_{m\ t_{w+1}}^{\ (i)}, \cdots, y_{m\ t_{w+\Delta}}^{\ (i)}, v_{\ t_w}^{\ (i)}, v_{\ t_{w+1}}^{\ (i)}, \cdots, v_{\ t_{w+\Delta}}^{\ (i)}]^T \in \mathbb{R}^{(\Delta S)\times 1}\end{aligned} \tag{3.19}$$

with $\Delta S = 60 \times 4 = 240$.

The training task can then be formulated as the reconstruction of each nominal vector $vec(\mathbf{X}_N^i(t_w))$ in the collection of objects $\left\{\mathbf{X}_N^i\right\}_{i=1}^{N_N}$. The reconstruction of one nominal vector is expressed as:

$$\widehat{vec(\mathbf{X}_N^i(t_w))} = g_{\theta_d}(f_{\theta_e}(vec(\mathbf{X}_N^i(t_w)))) \tag{3.20}$$

minimizing the reconstruction error:

$$E(t_w) = \left\| vec(\mathbf{X}_N^i(t_w)) - g_{\theta_d}(f_{\theta_e}(vec(\mathbf{X}_N^i(t_w)))) \right\| \tag{3.21}$$

The training part is summarized in Algorithm 5.

On the other hand, the testing part, i.e. the HI extraction from one trajectory under degradation of duration $\mathcal{T}_i$ can be formulated as the total reconstruction of vector $vec(\mathbf{X}_D^i(t_w))$ in the trajectory, with t_w in t_1 to $t_{\mathcal{T}_i}$.

The testing part is summarized in Algorithm 6.

3.4.2 Application to the experimental Dassault Aviation use case

Data preprocessing

The healthy data supplied to the autoencoder during the training phase are a set of ten healthy flights data generated via the *system identification-enabled nominal data augmentation* process described in Section 2.1. The parameters of the data generation are chosen so that they reproduce the observed flights as accurately as possible. That is, the generated flights have a random duration comprised between four and seven hours, the initial setpoint is randomly fixed between 22.5 and 24°C, and within each flight, a setpoint step change of random amplitude and time is set. Each sensor vector $X^{s(i)}$ is standardized in order to ensure that each feature has the same weight in the reconstruction training process.

$$X_{t_k}^{s\ (i)}{}_{scaled} = \frac{X_{t_k}^{s\ (i)} - \mu(s)}{\sigma(s)} \tag{3.22}$$

Algorithm 5: Autoencoder training for sensor reconstruction.

Input: A training set of N_N nominal (i.e. healthy) flight records $\left\{\mathbf{X}_N^i\right\}_{i=1}^{N_N}$, each of length $\mathcal{T}_i$.
Output: A trained autoencoder model with optimal weights $\theta_e^\star, \theta_d^\star$.
for *iter = 1 to* $\mathcal{K}$ **do**
  for i *= 1 to* N_N **do**
    Divide the flight record $\mathbf{X}_N^i$ into vectorized windows $vec(\mathbf{X}_N^i(t_w))$ of duration Δ;
    for ***each*** *vectorized window* **do**
      $\widehat{vec(\mathbf{X}_N^i(t_w))} \leftarrow g_{\theta_d}(f_{\theta_e}(vec(\mathbf{X}_N^i(t_w))))$;
      $E(t_w) = \left\| vec(\mathbf{X}_N^i(t_w)) - \widehat{vec(\mathbf{X}_N^i(t_w))} \right\|$;
      $loss \leftarrow loss + E(t_w)$;
    end
    $L_{iter} \leftarrow L_{iter} + loss$;
  end
  Update θ_e, θ_d by computing gradient descend on iteration loss L_{iter};
end

Algorithm 6: HI extraction procedure using the reconstruction error for one test trajectory.

Input: One test trajectory $\mathbf{X}_D^i$ of duration $\mathcal{T}_i$.
Output: A HI trajectory $\mathbf{HI}_i$ of duration $\mathcal{T}_i$.
Divide the trajectory $\mathbf{X}_D^i$ into vectorized windows $vec(\mathbf{X}_D^i(t_w))$ of duration Δ;
for ***each*** *vectorized window* **do**
  $\widehat{vec(\mathbf{X}_D^i(t_w))} \leftarrow g_{\theta_d^\star}(f_{\theta_e^\star}(vec(\mathbf{X}_D^i(t_w))))$;
  $\mathrm{HI}_{t_w} \leftarrow \sum_{k=w}^{w+\Delta} \left\| vec(\mathbf{X}_D^i(t_k)) - \widehat{vec(\mathbf{X}_D^i(t_k))} \right\|$;
end

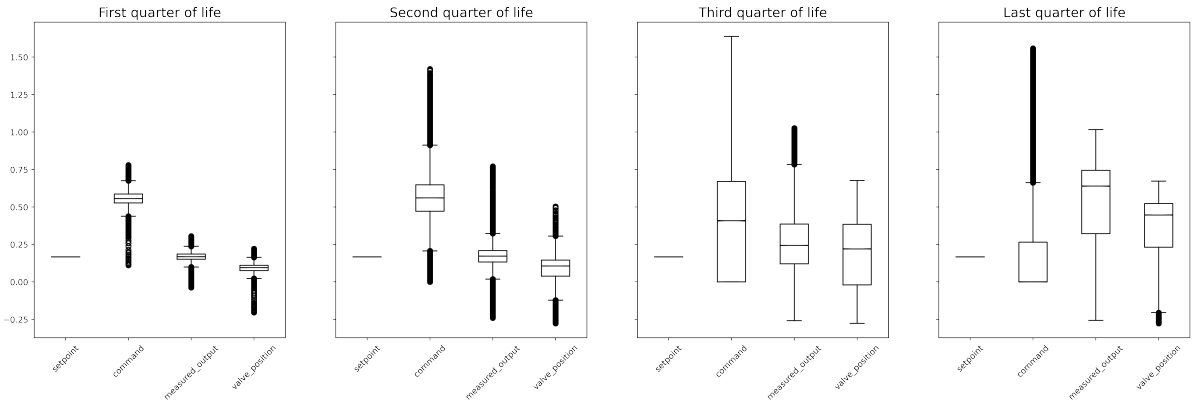


Figure 3.13: The deviation of the *degraded feature space* $\mathcal{X}_D$ over time. TTF trajectories are divided into quarters of life, for each of which the distribution of features (i.e. sensors) is plotted in boxplots.

with mean of sensor s:

$$\mu(s) = \frac{1}{N_N \times T_i} \sum_{i=1}^{N_N} \sum_{k=1}^{T_i} X_{t_k}^{s\ (i)} \tag{3.23}$$

and standard deviation of sensor s:

$$\sigma(s) = \sqrt{\frac{1}{N_N \times T_i} \sum_{i=1}^{N_N} \sum_{k=1}^{T_i} \left[X_{t_k}^{s\ (i)} - \mu(s) \right]^2} \tag{3.24}$$

A brief analysis of the evolution of the distribution of features over time in the TTF trajectories generated can be found in Figure 3.13. Taking the first quarter of life as a reference, the following observations can be formulated:

- In the second quarter of life, although the medians for each of the sensors remain fairly unchanged, the outliers have increased, as well as the boundaries of the whiskers.
- In the third quarter of life, the medians have deviated slightly from their initial value and the interquartile range has greatly increased, as have the whiskers.
- In the last quarter of life (i.e. the fraction of life in which the EOL is reached), it can be seen that the general distribution of each of the sensors is totally different from their initial distribution.

All this demonstrates that the feature space deviation hypothesis formulated in Section 3.3 ($\mathcal{X}_N \neq \mathcal{X}_D$) is evident.

Autoencoder structure and hyper-parameters

As explained in Section 3.4.1, the data supplied to the autoencoder are vectorized over 60-second time windows, which is an appropriate duration for capturing the dynamics of the control in this case. Therefore, one value of HI is computed per minute. This indicates that the first layer of the autoencoder must have a size of 60×4 as there are $S = 4$ sensors.

The autoencoder structure is therefore composed of multiple FCL of decreasing size following a geometric sequence with common ratio 1/2, i.e. 240 neurons, then 120, 60 and finally 30, so that the latent space size is reduced by 87.5% compared with the initial size of the input data. The decoder is built symmetrically to the encoder, except that the setpoint signal is not reconstructed by the decoder, thus leading to a final layer of size 180. The structure of the network can be seen in Figure 3.14 and the set of hyper-parameters for this implementation is summarized in Table 3.3.

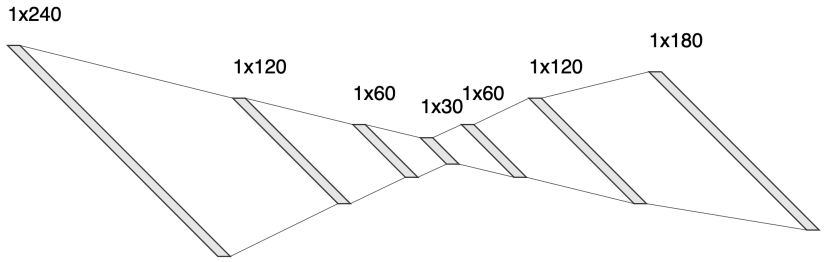


Figure 3.14: The structure of the autoencoder for signal reconstruction, along with the different FCL sizes.

Hyper-parameters	Values
Size of encoder layers	240, 120, 60, 30
Size of decoder layers	60, 120, 180
Window size Δ	60 seconds
Range of durations $\mathcal{T}_i$ of flight records for training	4 to 7 hours
Iterations (also called Epochs)	200

Table 3.3: Set of hyper-parameters for Dassault Aviation experiment.

Experimental results

Once the autoencoder has been trained to learn the nominal behavior of the system under study, it is tested to reconstruct a randomly selected window of nominal data $\mathbf{X}_N^i(t_w)$. Figure 3.15 displays the result of such a validation. As expected, because of the drastic reduction in information in latent space, nominal reconstruction is necessarily lossy. However, the aim is not to reconstruct the nominal data perfectly, but to effectively detect distribution deviation in the data as the degradation progresses.

To ensure that this objective is fulfilled, the autoencoder is tested on a TTF trajectory. For the present study, a trajectory obtained using the degraded data augmentation process developed in Section 2.2 is used as test data. According to the process described in Figure 3.12, the reconstructed signals are compared to the original ones, providing the evolution of the reconstruction error E_{total} over time. To limit excessive noise in the data, the HI is sub-sampled at 10-minute intervals. This sampling rate is more than sufficient for RUL monitoring, since the degradation phenomenon is progressive. The signals collected during the entire life of one example aircraft, together with the associated HI at each instant are displayed in Figure 3.16.

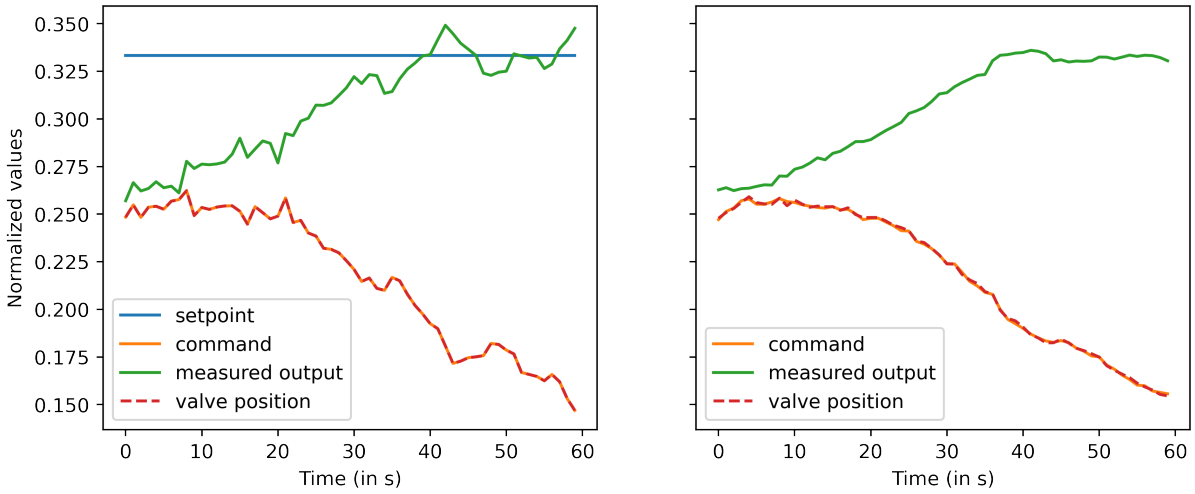


Figure 3.15: An example of the reconstruction of a nominal (i.e. healthy) data window. The input nominal window (denoted as $\mathbf{X}_N^i(t_w)$) is plotted on the left-hand graph, while the reconstructed window (denoted as $\widehat{\mathbf{X}}_N^i(t_w)$) is plotted on the right-hand graph. Note that the setpoint r is not reconstructed, but is used as input information for the autoencoder, and that the command and the valve position are identical, because in nominal operation the valve is supposed to be perfectly linear, as described in Figure 2.22.

It can be clearly seen that the reconstruction error increases along with the degradation of the system until the EOL, immediately before 2000 minutes of life[3].

The HI extraction process then is tested over a set of ten test TTF trajectories. The median HI trajectory, together with the standard deviation with respect to the fraction of total life passed is depicted in Figure 3.17

Finally, HI trajectories are normalized between zero and one on the basis of the training trajectories (since test trajectories are not known), according to Equation 3.5. This last stage is optional and merely allows the HI trajectories to be visualized between 0 and 1, according to the well-established practices of the PHM community.

3.5 Conclusion

In this chapter, several ways of using autoencoders have been explored, with the aim of extracting a HI from raw multi-variate sensor data. At first, a vanilla autoencoder has been used in time-independent approach, as a tool to achieve time step-by-time step dimension reduction. In a second phase, a more complex structure, based on CNN layers embedded inside of an autoencoder, was applied to search for deeply hidden features, thus leading to an HI where the EOL is represented by a recurrent pattern, so called the EOL pattern. These two first proposals have been tested on the public C-MAPSS dataset developed by NASA (detailed in Appendix A). Although they both provide useful insights into the issue of avoiding the use of RUL-labeled data (Research challenge 1), they are nonetheless still limited:

[3]This valve lifetime is not necessarily representative of reality, but is obtained using a TTF obtained by data augmentation for test purposes. The parameters used to generate this TTF trajectory have been arbitrarily selected.

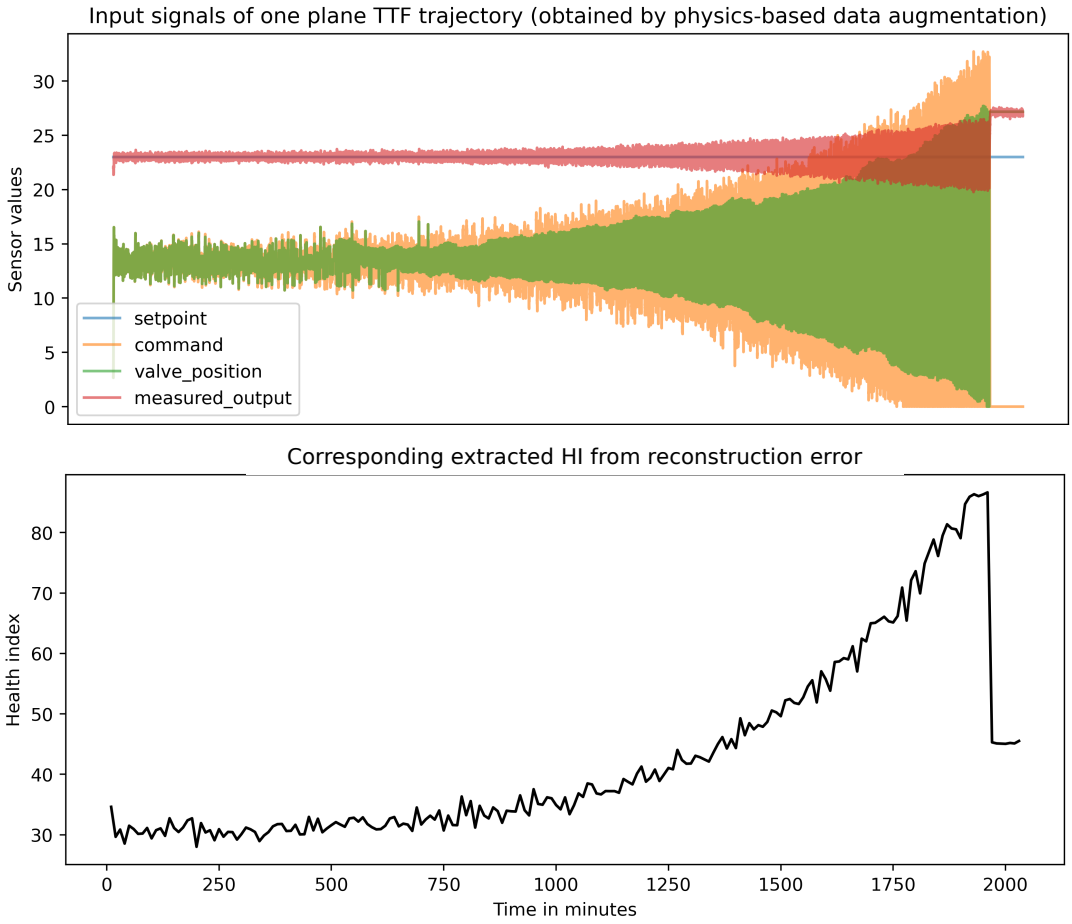


Figure 3.16: The HI obtained from the reconstruction error of the signals collected on one example TTF trajectory. The input sensor data is plotted on the top graph, while the corresponding HI trajectory is plotted on the bottom graph. The reconstruction error increases gradually up to the EOL. Once this moment has passed, the system no longer operates and, as a result, the reconstruction error (i.e. the HI) falls back.

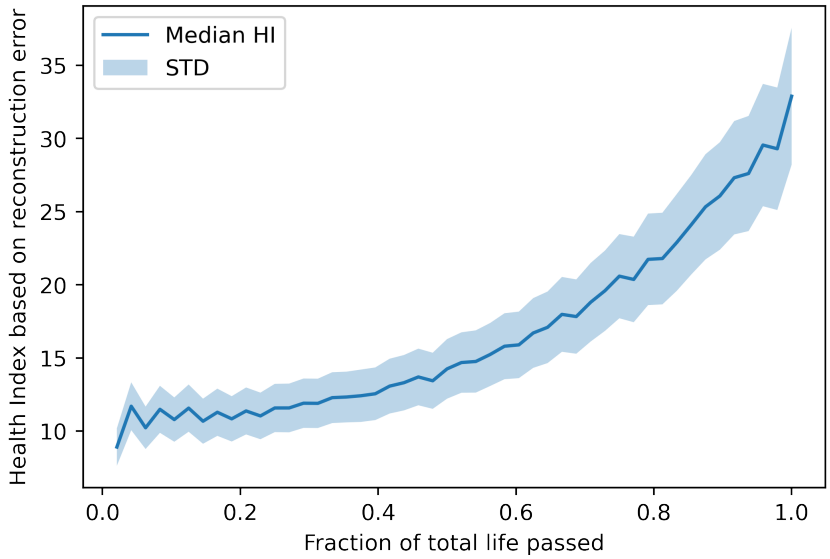


Figure 3.17: Median HI trajectory along with standard deviation based on the reconstruction error for a test set of ten TTF trajectories, w.r.t. fraction of total life passed.

- C-MAPSS Application 1 appears to be limited in terms of computational complexity, and does not guarantee that the HI dynamics obtained is strictly representative of the degradation.
- C-MAPSS Application 2 requires the use of complete trajectories during training, which is hardly realistic from an industrial point of view.

Therefore, a third approach is proposed, based exclusively on nominal data, i.e. when the system is healthy. This alternative approach relies on the evolution of the reconstruction error of an autoencoder trained exclusively on nominal data. As the degradation progresses, the sensor variables collected on the system under study are subject to a drift which is reflected in the quality of the data reconstruction by the autoencoder (whose weights are frozen). Such an approach offers excellent generalization capabilities as it relies only on the nominal behavior of the system. Therefore, it has the potential to detect any degradation, regardless of the component involved and the failure mode. This proposal is a major contribution of the thesis, addressing specifically research challenge 1.

In the next chapter, various prediction methods based on neural networks will be explored, in order to forecast the future HI values of a system and thus deduce the RUL.

Chapter 4

Health Index prediction for Remaining Useful Life estimation

An approach for extracting the HI from the data collected by the sensors has been developed in Chapter 3. In this chapter, an approach for forecasting this HI over long time horizon is proposed. The aim is to perform a long-term prediction of the HI in order to deduce the RUL. It is a two-stage predictive approach that consists firstly of predicting the HI until the EOL is detected, secondly recursively deducing the RUL. In the following, Section 4.1 presents the proposed approach for estimating the RUL via the long-term prediction of the HI. Different HI prediction models are proposed in 4.2. Then, various EOL detection methods are described in 4.3. Section 4.4 will present, as in the previous chapter, firstly the results obtained on an academic reference, i.e. the C-MAPSS dataset, and secondly the results obtained on the Dassault Aviation use case. Finally, a reliability-based assessment procedure is proposed in order to demonstrate the consistency of the proposed AI-based approach with respect to the laws of reliability.

Contents

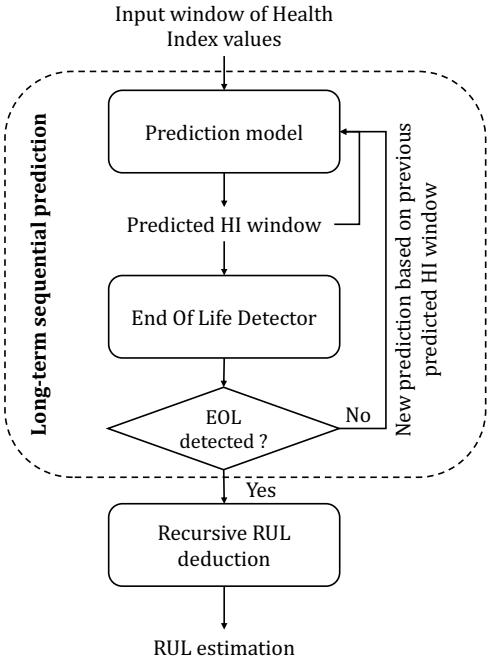


Figure 4.1: Overview of the proposed prediction approach.

4.1 A two-step proposed approach

In order to obtain a prediction of the RUL from the HI extracted using the method proposed in Chapter 3, a two-stage approach is proposed here. Firstly, a long-term prediction of the HI is carried out from an initial sample of the HI extracted from raw sensor data collected at the beginning of the system life. This long-term prediction process is said to be sequential, since it reuses previously predicted windows as input for future predictions, in an incremental manner. It is then necessary to know when to stop this sequential prediction task. To do this, the EOL is detected directly in the predicted HI data. Various EOL detection tools are presented in this sense in Section 4.3.

Finally, once the HI has been predicted until the EOL of the system, the RUL associated can be estimated recursively, by measuring the duration of the HI prediction made up to that point.

An overall overview of the proposed prediction approach can be found in Figure 4.1.

4.2 Sequential prediction using Recurrent Neural Network (RNN)-based models

The first step of the proposed approach consists of a long-term sequential prediction of the HI. Therefore, in this section, some foundations about sequential problem formulation are provided in Section 4.2.1. Then, a special LSTM-based encoder-decoder architecture is described in Section 4.2.2.

4.2.1 Sequential prediction problem formulation

Sequence prediction is a specific kind of supervised learning. In time series context, two main prediction scenarios can be distinguished. Firstly, sequence prediction (also referred to as sequence learning) consists of predicting the next value for a given input sequence (Sun and Giles, 2001).

$$\left[X_{t_{k=1}}, ..., X_{t_{k=\Delta_{input}}}\right] \mapsto \widehat{X}_{t_{k=\Delta_{input}+1}} \tag{4.1}$$

where Δ_{input} is the length of the input sequence and $\mapsto$ represents the mapping model between the input sequence and the output (which can typically be an RNN or its variants).

For instance, such a kind of prediction scenario has been used by Zheng et al., 2017 to predict RUL from raw input data, in a direct mapping manner. In this case, the sequence prediction is expressed by:

$$\left[X_{t_1}, ..., X_{t_{\Delta_{input}}}\right] \mapsto \widehat{\text{RUL}}_{t_{\Delta_{input}}} \tag{4.2}$$

where X_{t_k} is the multi-dimensional vector of sensor measurements at time t_k and $\widehat{\text{RUL}}_{t_{\Delta_{input}}}$ is the RUL prediction at time $t_{\Delta_{input}}$.

Sequence prediction can then be extended in time, by including past predictions within the input sequence for future predictions:

$$\begin{gathered}\left[X_{t_{k=1}}, ..., X_{t_{k=\Delta_{input}}}\right] \mapsto \widehat{X}_{t_{k=\Delta_{input}+1}} \\ \left[X_{t_{k=2}}, ..., X_{t_{k=\Delta_{input}}}, \widehat{X}_{t_{k=\Delta_{input}+1}}\right] \mapsto \widehat{X}_{t_{k=\Delta_{input}+2}} \\ \text{etc.}\end{gathered} \tag{4.3}$$

Secondly, sequence-to-sequence prediction (sometimes referred to as multi-step time series forecasting) involves predicting an output sequence given an input sequence (Brownlee, 2017).

$$\left[X_{t_{k=1}}, ..., X_{t_{k=\Delta_{input}}}\right] \mapsto \left[\widehat{X}_{t_{k=\Delta_{input}+1}}, ..., \widehat{X}_{t_{k=\Delta_{input}+\Delta_{pred}}}\right] \tag{4.4}$$

where Δ_{input} is the length of the input sequence and Δ_{pred} is the length of the predicted output sequence.

Sequence-to-sequence prediction has been used for Lithium-Ion battery capacity prediction by Y. Liu et al., 2017, with input and output window length of 50, thus expressed as follows:

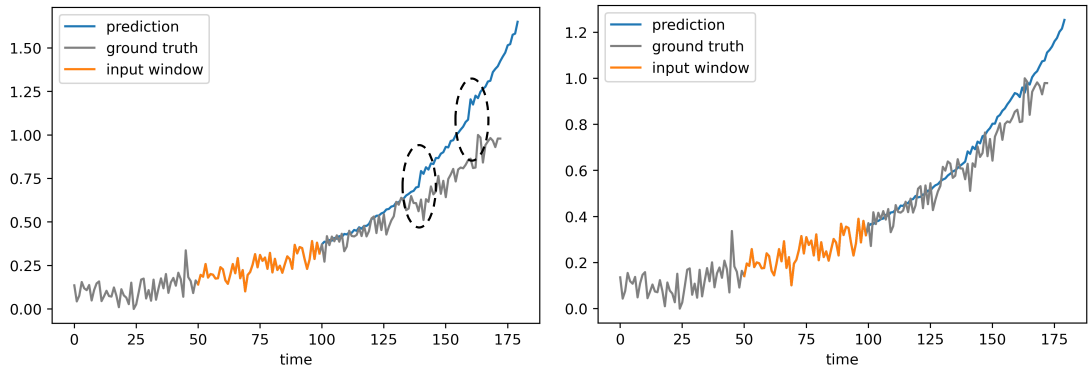
$$[X_{t_1}, ..., X_{t_{50}}] \mapsto \left[\widehat{X}_{t_{51}}, ..., \widehat{X}_{t_{100}}\right] \tag{4.5}$$

where X_{t_k} is the actual battery capacity value of t_k-th cycle and $\widehat{X}_{t_k}$ is the predicted battery capacity of t_k-th cycle.

Sequence-to-sequence prediction can be extended to make longer-term predictions. To do this, previous predicted output sequences are reused as input for future predictions.

$$\begin{gathered}\left[X_{t_{k=1}}, ..., X_{t_{k=\Delta_{input}}}\right] \mapsto \left[\widehat{X}_{t_{k=\Delta_{input}+1}}, ..., \widehat{X}_{t_{k=\Delta_{input}+\Delta_{pred}}}\right] \\ \left[\widehat{X}_{t_{k=\Delta_{input}+1}}, ..., \widehat{X}_{t_{k=\Delta_{input}+\Delta_{pred}}}\right] \mapsto \left[\widehat{X}_{t_{k=\Delta_{input}+\Delta_{pred}+1}}, ..., \widehat{X}_{t_{k=\Delta_{input}+2\Delta_{pred}}}\right] \\ \text{etc.}\end{gathered} \tag{4.6}$$

Such a multi-step sequence-to-sequence prediction is handled in Y. Liu et al., 2017 for long-term battery capacity prediction.



(a) Prediction without overlapping ($\Delta_{input} = 50$, $\Delta_{pred} = 20, \delta = 0$). Discontinuities between predicted windows are circled in dashed lines.

(b) Prediction with overlapping ($\Delta_{input} = 50$, $\Delta_{pred} = 30, \delta = 10$). The overall prediction obtained ensures better continuity between predicted windows

Figure 4.2: Comparison of two chained sequence-to-sequence predictions made with a simple LSTM network, one with overlapping and the other without. In both cases, 4 chained predictions have been performed.

When predictions are executed in chain, i.e. by reusing previous predictions as input for subsequent predictions, the error accumulates from prediction to prediction. In one-step ahead sequence prediction, as expressed in Equation 4.3, more predictions must be made than in multi-step ahead sequence-to-sequence prediction expressed in Equation 4.6 to reach the same time point. Hence, as stated by Y. Liu et al., 2017, the cumulative error is larger in the first case than in the second. Therefore, the sequence-to-sequence prediction is selected for the present study.

In addition, the concept of overlapping for sequence-to-sequence prediction is proposed in this work, in order to increase the degree of correlation between the input window and the output window. The rationale behind using overlapping is that it could improve continuity between different prediction windows (i.e. avoiding breaks between the chained windows in a long-term sequence-to-sequence prediction). As an example, Figure 4.2 illustrates two cases of prediction using a simple LSTM network, based on a set of generalized exponential curves. In Figure 4.2a, the input window length is $\Delta_{input} = 50$, the prediction window length is $\Delta_{pred} = 20$, and overlapping is null ($\delta = 0$). Discontinuities between predicted windows are visible. On the other hand, in Figure 4.2b, the input window length is the same ($\Delta_{input} = 50$) but the prediction window length is $\Delta_{pred} = 30$, with an overlapping $\delta = 10$. Such overlapping windows can be seen in Figure 4.3. In this case, the effective prediction is of the same length as in the previous case, but the distribution is different (i.e. out of the 30 predicted values, 20 are supplementary and 10 are overlapping the previous window). As a result, it can be seen in Figure 4.2b that the discontinuities that appeared in the absence of overlapping are considerably smoothed out, thus producing a better prediction. The choice of hyper-parameters Δ_{input}, Δ_{pred} and δ should be made empirically, depending on the situation. A specific study of the impact of these hyper-parameters is carried out for the Dassault Aviation case study in Section 4.4.2.

In the case studied here, the aim is to generate HI predictions based on previous HI observations. Therefore, consider a HI input window denoted as follows:

$$\text{Input window} : \mathbf{HI}(t_w) = \{\text{HI}_{t_k}\}_{k=w}^{w+\Delta_{input}} \tag{4.7}$$

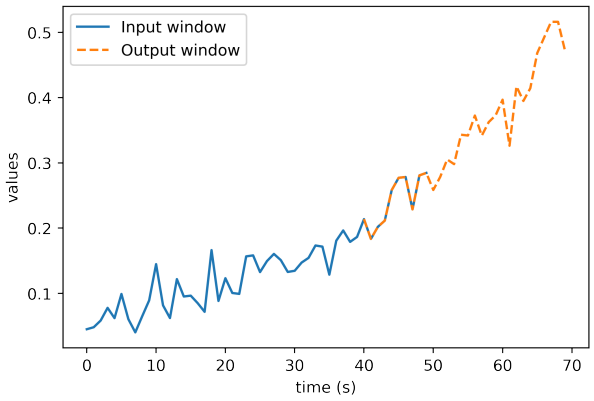


Figure 4.3: Input and output windows with overlapping. In this example, the input window duration is $\Delta_{input} = 50$, the overlapping is $\delta = 10$, and the target window duration is $\Delta_{pred} = 20$.

where t_w is the initial time step and Δ_{input} is the total duration (or length) of the input sequence (also called input window).

The aim of the predictive model is to predict the next future HI window, denoted as:

$$\text{Output window} : \mathbf{HI}(t_{w+\Delta_{input}-\delta}) = \{\text{HI}_{t_k}\}_{k=w+\Delta_{input}-\delta}^{w+\Delta_{input}-\delta+\Delta_{pred}} \tag{4.8}$$

where δ is the overlapping and Δ_{pred} is the duration of the prediction window.

If the overlapping is chosen to be null (i.e. $\delta = 0$), then the sequence of output windows forms a consecutive chain. But if the overlapping is chosen to be greater, i.e. $\delta \geq 1$, then the output windows are overlapping the input windows, as depicted in Figure 4.3.

After q predictions, the duration T_{total} of the total chain made of the predicted HI windows (including the first input window) is thus defined as:

$$T_{total} = \Delta_{input} + q \times (\Delta_{pred} - \delta) \tag{4.9}$$

To be trained, the predictive model thus needs a training set composed of pairs of input-output windows. Sequential prediction process is supervised learning (unlike the HI extraction process applied to Dassault Aviation data, developed in Section 3.3). Historical TTF trajectories are therefore required for training. As explained in Chapter 1, such trajectories are generally unavailable in real industrial cases. For this reason, a prognostics-oriented data augmentation approach was developed in Chapter 2, specifically in Section 2.2. Nevertheless, the different HI prediction models presented in Section 4.2, as well as the diverse EOL detection methods described in Section 4.3, were first tested on the academic C-MAPSS dataset (see Section 4.4.1), before being integrated into the realistic framework of the Dassault Aviation application case in Section 4.4.2.

The TTF HI trajectories are divided into pairs of input-target windows, so that predictive model can be trained by minimizing the following error:

$$J_{model} = \sum \mathcal{L}(\mathbf{HI}(t_{w+\Delta_{input}-\delta}), \widehat{\mathbf{HI}}(t_{w+\Delta_{input}-\delta})) \tag{4.10}$$

where the loss function $\mathcal{L}$ is the MSE, since it is essentially a regression problem.

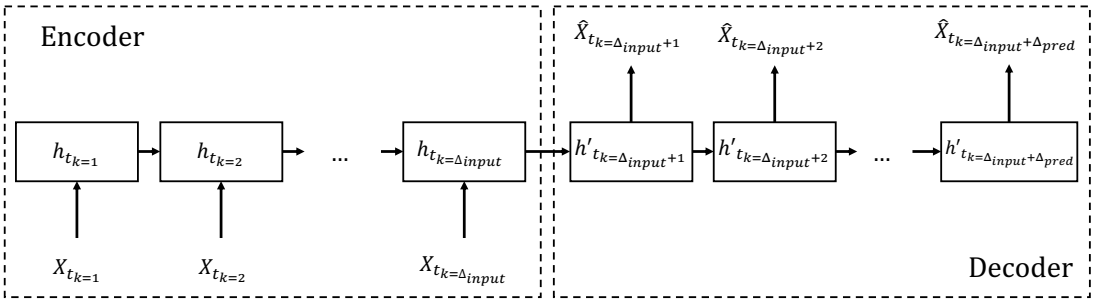


Figure 4.4: Traditional structure of an RNN-based encoder-decoder.

4.2.2 Long Short-Term Memory (LSTM)-based encoder-decoder

RNN is a neural network structure well-adapted to sequential data, as it uses prior information along with current input in order to perform prediction of the desired output. As described in Section 1.2.1, RNNs have been widely used for RUL prediction tasks, and in particular LSTM cells have provided highly effective results. A detailed presentation of the state of the art concerning LSTM architecture is available in Appendix C. A more sophisticated architecture is used for RUL prediction on the C-MAPSS dataset, namely an LSTM-based encoder-decoder. Similarly to CNN-based autoencoders (presented in Section 3.2.2), LSTM-based encoder-decoder is a special case of encoder-decoder where the encoding and decoding functions are performed by LSTM models (Cho et al., 2014). Therefore, the LSTM-based encoder transforms the input time series into a fixed-dimension representation vector (generally referred to as "context vector") while the LSTM-based decoder maps this context vector to the target time series (Du et al., 2020). Figure 4.4 presents the traditional recurrent encoder-decoder structure.

More precisely, consider a sequence-to-sequence learning with input and output windows defined as follows (overlapping is neglected here for simplicity):

$$\left[X_{t_{k=1}}, ..., X_{t_{k=\Delta_{input}}}\right] \mapsto \left[\widehat{X}_{t_{k=\Delta_{input}+1}}, ..., \widehat{X}_{t_{k=\Delta_{input}+\Delta_{pred}}}\right] \tag{4.11}$$

The hidden state of the LSTM-based encoder at time t_k is denoted as h_{t_k}. Relevant information is captured by the LSTM-based encoder as it is supplied with the input time series. When the last time step $t_{k=\Delta_{input}}$ of the input sequence is reached, the hidden state $h_{t_{k=\Delta_{input}}}$ (which is the context vector that embeds all the retained information from the input time series $\mathbf{X}$) is transmitted as the first hidden state $h'_{t_{k=\Delta_{input}+1}}$ of the decoder, which then constructs the desired output time series.

The weights θ_e^{LSTM} and θ_d^{LSTM} of the model are estimated by minimizing the error between the target sequence and the one output by the model:

$$J(\theta_e^{\text{LSTM}}, \theta_d^{\text{LSTM}}) = \sum \mathcal{L}(\mathbf{X}, g_{\theta_d^{\text{LSTM}}}(f_{\theta_e^{\text{LSTM}}}(\mathbf{X}))) \tag{4.12}$$

Finally, the different LSTM-based architectures presented in Appendix C can be used as encoding and decoding functions. For instance, Yu et al., 2019 proposed a bidirectional RNN-based encoder-decoder. In this PhD work, the proposed architecture is a three-layers stacked-LSTM-based encoder-decoder. The overall architecture of the proposed prediction model can be seen in Figure 4.5. More precisely, the encoder (made of three stacked-LSTM layers) takes as input a sequence of HI denoted as $\mathbf{HI}(t_w)$ (defined in Equation 4.7). The encoder outputs

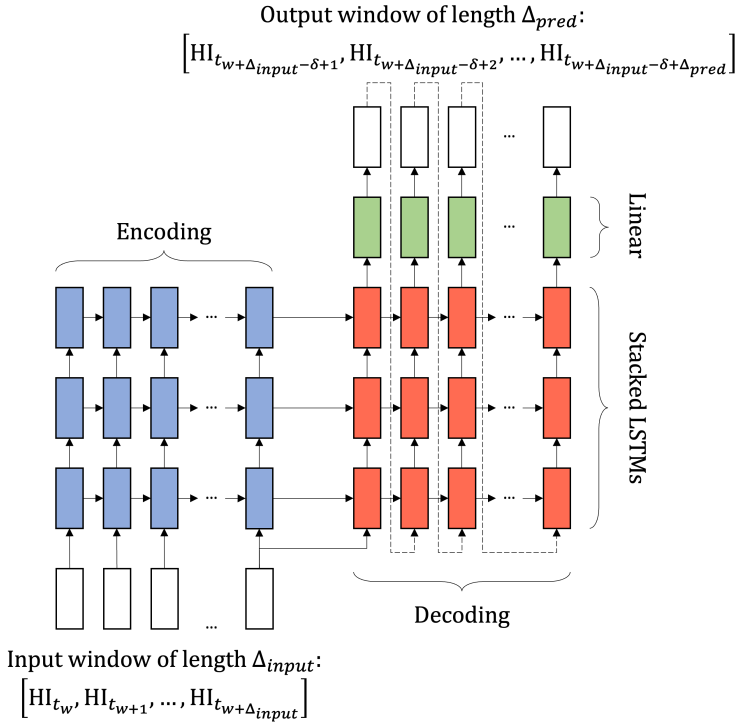


Figure 4.5: Structure of the stacked-LSTM-based encoder-decoder applied to C-MAPSS. The encoder and decoder are both composed of three stacked-LSTM layers. Then, a FCL is used to obtain the predicted window.

one context vector for each LSTM layer, denoted as $h^L_{t_{w+\Delta_{input}}}$ where L is the LSTM layer and Δ_{input} is the length of the input window. Each of these context vectors is then transmitted to the decoder to become the first hidden state of the corresponding decoder LSTM layer, denoted as $h'^L_{t_{w+\Delta_{input}-\delta+1}}$. The output of the decoder is provided by the third LSTM layer of the decoder. It is finally given to a Fully Connected Layer (FCL), thus resulting in a HI output window $\mathbf{HI}(t_{w+\Delta_{input}-\delta})$ defined as in Equation 4.8.

4.3 End of Life (EOL) detection

As mentioned in Section 4.1, it is necessary to define a point at which the chained window prediction loop stops, i.e. to detect the point at which it is assumed, based on the HI prediction, that the system has reached its EOL. In this work, two approaches have been tested. They are each presented in the next two sections.

4.3.1 Threshold overshooting

An intuitive method is to define an upper bound on the HI, considering that when the HI reaches this limit, the system has attained its EOL. In this situation, HI windows are predicted sequentially as described in Section 4.1, and in each window the maximum HI value reached is compared with the pre-set threshold value denoted as $\text{EOL}_{threshold}$.

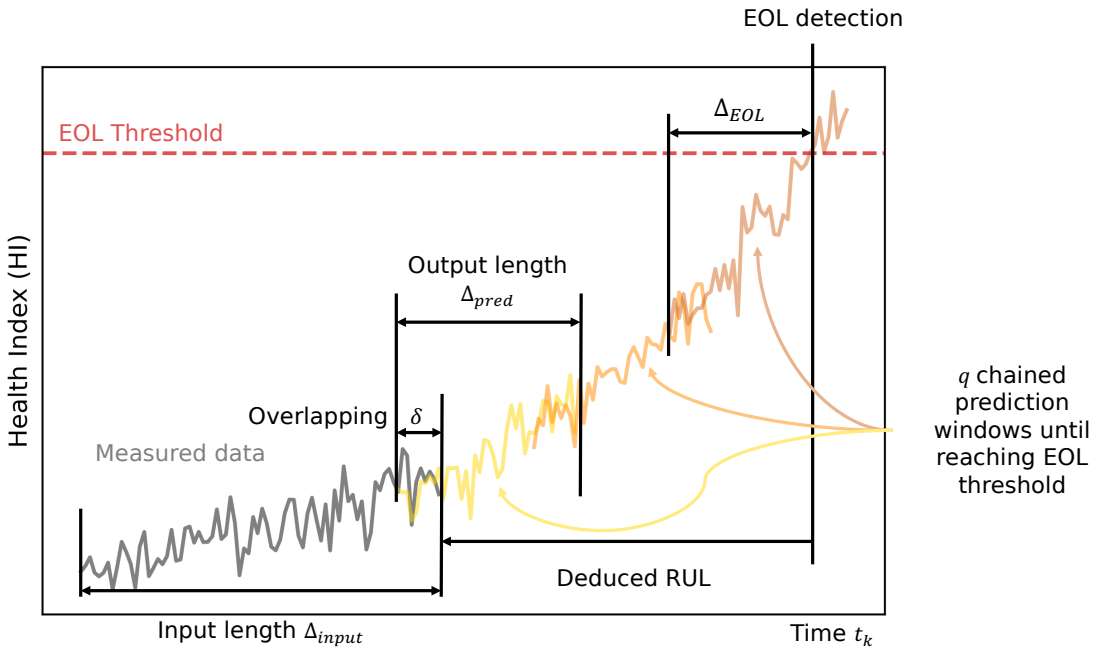


Figure 4.6: Overall view of the threshold overshooting strategy. q windows of length Δ_{pred} are predicted, until the EOL threshold is reached. RUL value is then deduced by counting the time steps covered between the start point and the EOL.

After q iterations, the RUL can then be deduced recursively, counting the iterations. This process is summarized in Figure 4.6.

The deduced RUL value is essentially a count of all the time steps covered before reaching the EOL. It is expressed as:

$$\text{RUL} = \Delta_{input} + (q-1) \times (\Delta_{pred} - \delta) + (\Delta_{EOL} - \delta) \tag{4.13}$$

where Δ_{input} is the length of HI input window, Δ_{pred} is the length of the output window, δ is the overlapping, q is the number of prediction windows and Δ_{EOL} is the length of the partial last predicted window, which is cut off at the time step where the EOL is detected, as indicated in Figure 4.6.

Algorithm 7 describes in detail the iterative process leading to EOL detection and RUL deduction.

This method has the advantage of being relatively intuitive and interpretable from a physical point of view. On the other hand, its main drawback is the need to have a normalized HI, in order to be able to set the EOL threshold. This approach is applied to C-MAPSS dataset in Section 4.4.1 and to Dassault Aviation experimental use case in Section 4.4.2.

4.3.2 Pattern recognition using Dynamic Time Warping (DTW)

The second EOL detection method is strongly connected to the CNN-based autoencoder HI extraction approach introduced in Section 3.2.2. As a reminder, the CNN-based autoencoder structure has extracted a distinctive pattern that always appears just before the EOL.

Algorithm 7: RUL prediction process using HI sequential prediction threshold overshooting strategy.

Input: A HI window of length Δ_{input} : $\mathbf{HI}(t_w) = \{\mathrm{HI}_{t_k}\}_{k=w}^{w+\Delta_{input}}$.
Output: A RUL value (scalar).
$q = 0$;
$\mathbf{HI}(t_{w+\Delta_{input}-\delta}) \leftarrow h_\Theta(\mathbf{HI}(t_w))$ where h_Θ can be any NN-based approximation model parameterized by Θ (such as LSTM and its variants);
while $max(\mathbf{HI}(t_{w+\Delta_{input}-\delta})) < EOL_{threshold}$ **do**
  $q \leftarrow q + 1$;
  $\mathbf{HI}(t_{w+\Delta_{input}-\delta+\Delta_{pred}}) \leftarrow h_\Theta(\mathbf{HI}(t_{w+\Delta_{input}-\delta}))$ (next predicted window from previous predicted window);
end
$\mathrm{RUL} = \Delta_{input} + (q-1) \times (\Delta_{pred} - \delta) + (\Delta_{EOL} - \delta)$;

In this case, HI windows are predicted sequentially until the EOL pattern is detected. To carry out this pattern recognition task, the Dynamic Time Warping (DTW) algorithm is employed (Müller, 2007, Berndt and Clifford, 1994), that defines an "elastic measure of similarity" that is well-suited to map similar patterns which are not strictly aligned in time.

DTW is a widely used time series similarity measure that comes from speech recognition (Sakoe and Chiba, 1978).

Let us consider two time series $\mathbf{b} = (b_1, b_2, ...b_N)$, and $\mathbf{d} = (d_1, d_2, ...d_M)$ of lengths N and M. A local cost matrix C representing all pairwise distances between $\mathbf{b}$ and $\mathbf{d}$ is constructed. To do this, a local cost measure c is computed between each pair of elements of the sequences $\mathbf{b}$ and $\mathbf{d}$.

Therefore, the local cost matrix is expressed as $C \in \mathbb{R}^{N \times M}$ and is defined by

$$C(n, m) = dist(b_n, d_m) \tag{4.14}$$

with $n \in [1 : N], m \in [1 : M]$ and $dist$ being a local distance measure (e.g. $dist(b_n, d_m) = \|b_n - d_m\|$).

The objective of the DTW algorithm is to find the alignment path that passes through the low cost areas of the local cost matrix. The "warping path" the sequence of points $p = (p_1, p_2, ..., p_L)$ with $p_l = (p_n, p_m) \in [1 : N] \times [1 : M]$ for $l \in [1 : L]$, respecting the three following conditions (Müller, 2007):

1. *Boundary condition*: $p_1 = (1, 1)$ and $p_L = (N, M)$. This means that the start and end points of the warping path are effectively the first and the last points of the two sequences $\mathbf{b}$ and $\mathbf{d}$.

2. *Monotonicity condition*: $n_1 \leq n_2 \leq ... \leq n_L$ and $m_1 \leq m_2 \leq ... \leq m_L$. This condition is a guarantee that the points are correctly time-ordered.

3. *Step size condition*: $p_{l+1} - p_l \in \{(1, 0), (0, 1), (1, 1)\}$ for $l \in [1 : L - 1]$. This criterion prevents warping from making large time shifts when aligning the two sequences.

The total cost of a warping path p is given by:

$$C_p(\mathbf{b}, \mathbf{d}) = \sum_{l=1}^{L} dist(b_{n_l}, d_{m_l}) \tag{4.15}$$

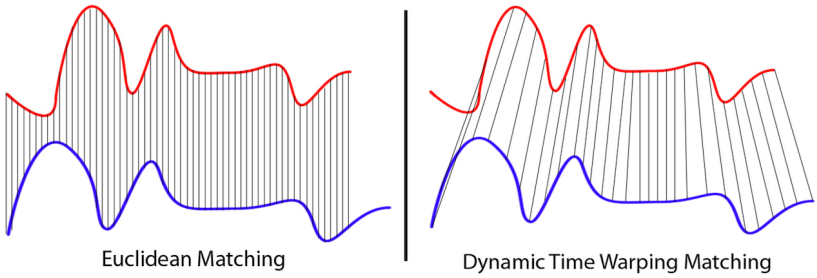


Figure 4.7: Simplified comparison between the Euclidean distance and the DTW distance.

The DTW distance between two time series **b** and **d** is then the total cost of the optimal warping path, denoted as p^*, which is the warping path with the minimum total cost among all possibilities.

The main advantage of DTW is that it minimises the effects of time shift and distortion by allowing flexible transformation of time series in the intention of detecting similar shapes. A simplified comparison between Euclidean and DTW distances is illustrated in Figure 4.7.

For the specific purpose of this study, the DTW algorithm is used to recognize a reference EOL pattern denoted as ν in the HI prediction. This reference pattern ν is simply the average of the patterns observed in the training set. For each predicted HI window $\mathbf{HI}(t_{w+\Delta_{input}-\delta})$, the DTW distance is computed with the reference pattern EOL pattern ν. When this similarity score exceeds a certain value, the EOL pattern is considered to be sufficiently recognized, and the RUL can therefore be deduced. Algorithm 8 summarizes this process.

Algorithm 8: RUL prediction process using HI sequential prediction and DTW EOL pattern recognition.

Input: A HI window of length Δ_{input} : $\mathbf{HI}(t_w) = \{\mathrm{HI}_{t_k}\}_{k=w}^{w+\Delta_{input}}$, a reference EOL pattern denoted as ν.
Output: A RUL value (scalar).
$q = 0$;
$\mathbf{HI}(t_{w+\Delta_{input}-\delta}) \leftarrow h_\Theta(\mathbf{HI}(t_w))$ where h_Θ can be any NN-based approximation model parameterized by Θ (such as LSTM and its variants);
while $DTW(\mathbf{HI}(t_{w+\Delta_{input}-\delta}), \nu) > threshold$ **do**
  $q \leftarrow q + 1$;
  $\mathbf{HI}(t_{w+\Delta_{input}-\delta+\Delta_{pred}}) \leftarrow h_\Theta(\mathbf{HI}(t_{w+\Delta_{input}-\delta}))$ (next predicted window from previous predicted window);
end
$\mathrm{RUL} \leftarrow \Delta_{input} + q \times (\Delta_{pred} - \delta)$;

The main advantage of this approach is that it is no longer required to normalize the HI trajectories. However, it remains dependent on the use of the CNN-based autoencoder structure. Moreover, it is not certain that an EOL pattern would necessarily be recognized in a different use case than the one covered in Section 3.2.2 (i.e. C-MAPSS). Finally, the pattern is most likely to be related to the failure mode, in which scenario all possible patterns would have to be detected, which would require degradation trajectories for all possible system failure modes. This approach is tested on C-MAPSS dataset in Section 4.4.1.

4.4 Application results of the proposed two-step approach

As mentioned in Section 1.3.2, and in the same way as in Chapter 3, two sources of data were used:

1. Firstly, the prediction approaches were tested on the C-MAPSS dataset commonly used in the PHM community. It is a continuation of the two approaches presented in Sections 3.2.1 (C-MAPSS Application 1) and 3.2.2 (C-MAPSS Application 2) respectively.

2. Secondly, the prediction is carried out on the Dassault Aviation experimental case. It is thus the continuation of the HI extraction performed in Section 3.4.

4.4.1 Results on C-MAPSS dataset

C-MAPSS Application 1

For the first application (Hervé de Beaulieu et al., 2022b), the HI was extracted using the direct HI extraction method with the vanilla autoencoder presented in Section 3.2.1.

Then, future values of HI are forecast using the three-layers stacked-LSTM-based encoder-decoder presented in Figure 4.5, until the EOL threshold is attained, following the approach described in Algorithm 7.

Figure 4.8 displays one example of such a prediction of multiple windows of HI up to the selected EOL threshold. Based on the prediction shown in Figure 4.8, the RUL estimated at time step $t_{k=112}$ can be computed following Equation 4.13. It took $q = 12$ iterations to reach the threshold, input window length is $\Delta_{input} = 50$, output window length is $\Delta_{pred} = 50$, overlapping is $\delta = 45$, and the threshold is attained at the 29th time step of the last predicted window (i.e. $\Delta_{EOL} = 29$) hence:

$$\text{RUL}(t_{k=112}) = (50) + (12 - 1) \times (50 - 45) + (29 - 45) = 89 \tag{4.16}$$

By repeating this operation at each time step t_k of the available HI input sequence, a RUL trajectory can be obtained for a turbine. In Figure 4.9, the complete RUL trajectory for the same turbine as in Figure 4.8 is depicted. It can then be seen that, at time step $t_{k=62}$ (which corresponds to time step $t_{k=112}$ when the length of the input window $\Delta_{input} = 50$ is subtracted), the predicted RUL is 89.

C-MAPSS Application 2

In this second study (Hervé de Beaulieu et al., 2022c) applied to C-MAPSS dataset, the HI is extracted from the raw sensor data using the CNN-based autoencoder described in Section 3.2.2. The prediction of future HI values is handled by the same three-layers stacked-LSTM-based encoder-decoder as in C-MAPSS Application 1, but in this case the prediction is stopped when the EOL pattern is detected by the DTW algorithm, following the procedure described in Algorithm 8.

Figure 4.10 depicts the successive predictions of HI windows until the threshold is recognized with a sufficient DTW similarity score. This score value is empirically tuned by using a validation set.

Similarly as in the previous study, the RUL can then be deduced recursively, and by repeating the process over a whole HI input sequence, the RUL trajectory for one turbine can be obtained. It is displayed, for the same turbine as in Figure 4.10, in Figure 4.11.

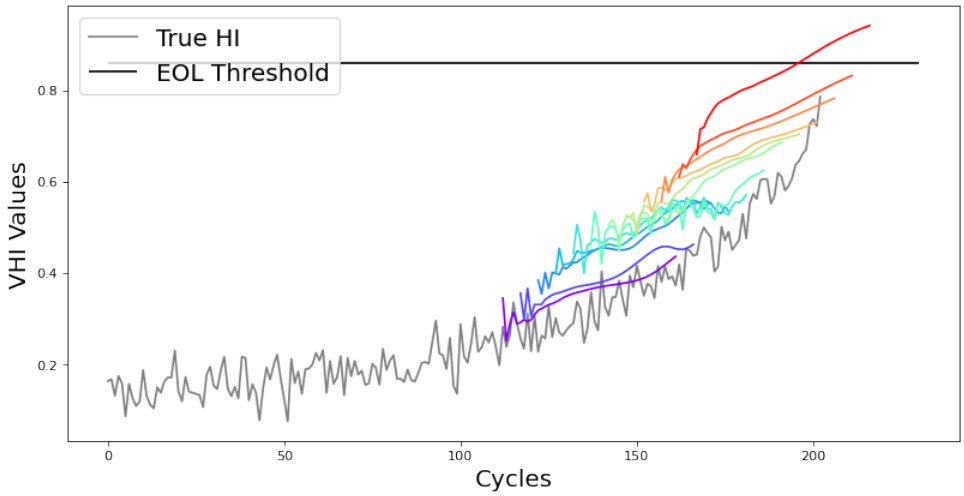


Figure 4.8: C-MAPSS Application 1: Example of the HI prediction for one turbine (Turbine 34) of FD001 test set. Windows of HI are successively predicted until the EOL threshold is attained, thus marking the EOL. In this example, input window length $\Delta_{input} = 50$, output window length $\Delta_{pred} = 50$, overlapping $\delta = 45$ and the prediction starts at time step $t_{k=112}$.

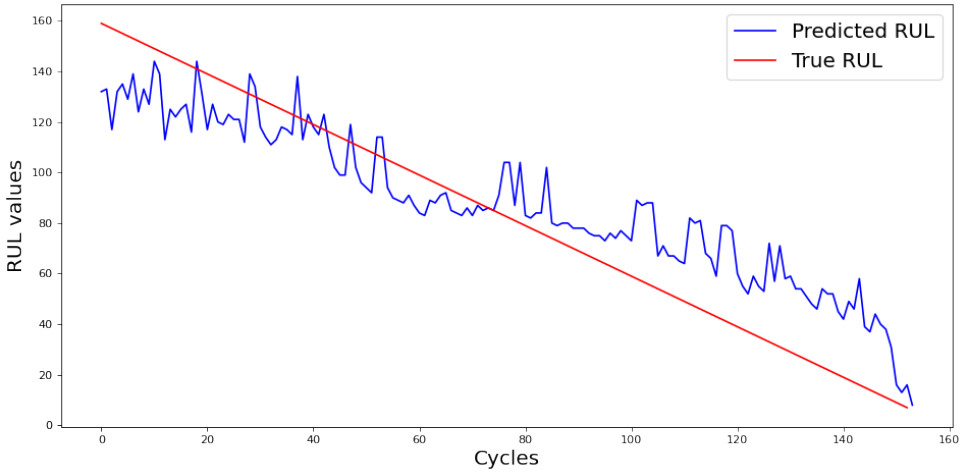


Figure 4.9: C-MAPSS Application 1: Example of a complete predicted RUL trajectory from one turbine (Turbine 34) of the FD001 test set.

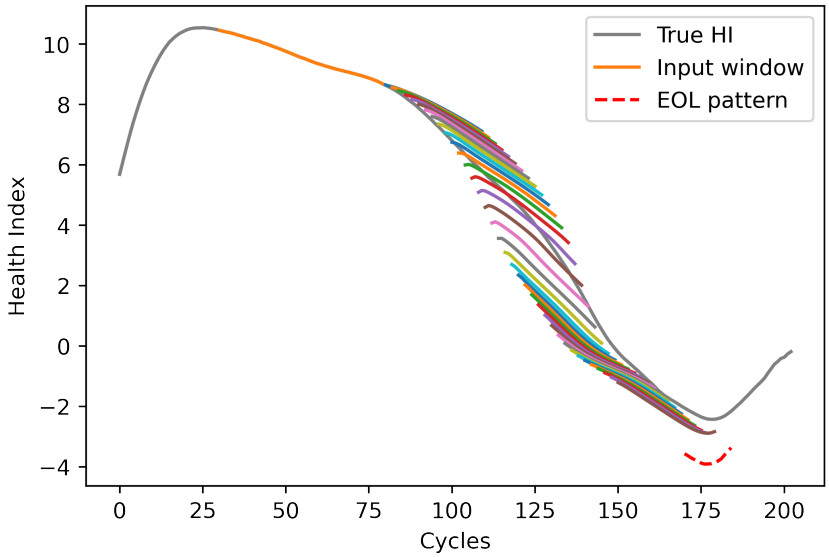


Figure 4.10: C-MAPSS Application 2: Example of the HI prediction for one turbine (Turbine 34) of FD001 test set. The successive predicted HI windows follow each other until the EOL pattern is recognized with a sufficient DTW similarity score, thus marking the EOL.

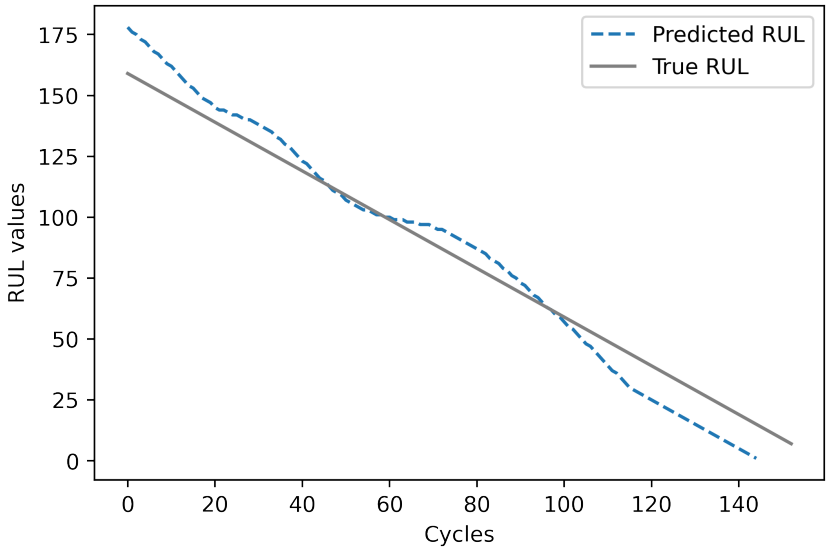


Figure 4.11: C-MAPSS Application 2: Example of a complete predicted RUL trajectory from one turbine (Turbine 34) of the FD001 test set.

Discussion about C-MAPSS Applications 1 and 2

The average RMSE obtained for all the turbines in the FD001 test set in the C-MAPSS dataset is computed for both the applications 1 and 2. The results are given in Table 4.1, together with some results obtained by supervised approaches published in recent years. However, the results of supervised approaches are given here for information and can hardly be compared to the results obtained with applications 1 and 2 on an equivalent basis. Indeed, the C-MAPSS dataset is specifically designed for direct mapping between sensor data and RUL. As a result, fully supervised approaches, using ever deeper and more complex neural networks, will inevitably produce much better results. Nevertheless, they remain highly unrealistic and hardly applicable in real industrial cases, as mentioned in Research challenge 1. On the other hand, the two applications presented here aim to make RUL predictions without using the RUL-labels available in the C-MAPSS dataset, but by predicting HI. It is therefore quite obvious that similar results cannot be expected. In this work, the emphasis is on the realistic aspect of the proposed scientifically sound solutions with respect to industrial issues.

Approach	Mean RMSE on FD001 test set
Supervised approaches relying on the use of RUL-labeled data	
MLP (Babu et al., 2016)	37.6
Simple CNN (Babu et al., 2016)	18.4
Deep CNN (X. Li et al., 2018)	12.6
Simple LSTM (X. Li et al., 2018)	13.5
Deep LSTM (Zheng et al., 2017)	16.1
Bidirectional-LSTM (J. Wang et al., 2018)	13.7
RUL-label-free approaches relying on HI prediction	
C-MAPSS Applications 1 (Vanilla autoencoder + EOL threshold overshooting)	44.7
C-MAPSS Applications 2 (CNN-based autoencoder + EOL pattern detection)	40.1

Table 4.1: RUL prediction performance for various approaches on C-MAPSS dataset.

4.4.2 Application results to the Dassault Aviation experimental use case

Prediction model

After applying the method to the C-MAPSS academic dataset, it is now being applied to the Dassault Aviation experimental case. Similarly to C-MAPSS approaches 1 and 2, the objective in the Dassault Aviation experimental case is to perform sequential predictions of HI until the EOL is detected. For this HI prediction task, a stacked-LSTM architecture is proposed. It takes as input a window of HI values, and forecasts as output a prediction window of future HI values. The specific feature of the structure proposed here is that all the hidden states of the last LSTM layer are vectorized (or "flattened") and transmitted to a sequence of FCL. Therefore, after the last stacked-LSTM layer L of the network, the entire set of hidden states $h^L_{t_{w+\Delta_{input}}}$ to $h^L_{t_{w+\Delta_{input}+\Delta_{pred}}}$ are concatenated and vectorized, to be supplied to a network of FCL that will lead to a predicted time window of the desired size. This ensures an extensive use of all the temporal information, from the start to the end of the input sequence. This detailed architecture can be seen in Figure 4.12. The hyper-parameters of this model and of the sequence-to-sequence prediction process are listed in Table 4.2 and are discussed immediately afterwards.

Hyper-parameter	Value
Number of layers L	3
Hidden state size $\lvert h_{t_k} \rvert$	240
Number of FCL	3
Size of FCL	$\lvert \text{FCL}_1 \rvert = (\lvert h_{t_k} \rvert \times \Delta_{pred})/4$ $\lvert \text{FCL}_2 \rvert = \lvert \text{FCL}_1 \rvert /4$ $\lvert \text{FCL}_3 \rvert = \Delta_{pred}$
Input window length Δ_{input}	8h
Output window length Δ_{pred}	3h
Overlapping δ	1.5h
Dropout	0.2

Table 4.2: Hyper-parameters used in the sequence-to-sequence prediction with the stacked-LSTM model. The values indicated are those used to obtain the results shown in Section 4.4.2.

Dassault Aviation experimental results

To train such a prediction model, it is necessary to have a training set composed of complete HI trajectories similar to the one shown in Figure 3.16. This training set is constructed by generating a set of 10 TTF multi-dimensional trajectories using the *physics-based degraded data augmentation process* presented in Section 2.2. As a reminder, a TTF multi-dimensional trajectory of this kind was shown in Figure 2.24. Each TTF multi-dimensional trajectory is then processed by the unsupervised HI extraction approach presented in Chapter 3, thus delivering a complete training HI trajectory. Once the model has been trained to do sequence-to-sequence prediction of HI windows, it can then be tested.

From an input HI window, future HI windows are sequentially forecast (using the stacked-LSTM network presented in Figure 4.12), until the EOL threshold (equal to 1) is reached, according to the procedure described in Algorithm 7. Figure 4.13 displays two HI predictions carried out until the EOL threshold is reached, for one TTF test trajectory. The first prediction (Figure 4.13a) starts at $t_{k=480}$ minutes while the second (Figure 4.13b) starts at $t_{k=800}$ minutes. It can be observed that the earlier the prediction starts, the less accurate it is, since the total prediction length is longer, accumulating more prediction errors.

As in the two previous C-MAPSS applications, the final objective is to estimate the RUL using the HI prediction. In Figure 4.14, the complete predicted RUL trajectory for the same TTF test trajectory as in Figure 4.13 can be seen. It is clear that the accuracy of the prediction increases as the EOL approaches.

In Table 4.3, the mean RMSE along with its standard deviation has been computed for the ten TTF test trajectories. The average RMSE was calculated for the first fifteen RUL predictions on the one hand and the last fifteen on the other hand, highlighting the considerable increase in accuracy as the EOL is approached.

	Mean RMSE	Standard deviation
15 first RUL predictions	28.8	15.0
15 last RUL predictions	3.7	1.5
Complete RUL trajectories	8.4	1.7

Table 4.3: Mean RMSE for the RUL prediction on the 10 TTF trajectories.

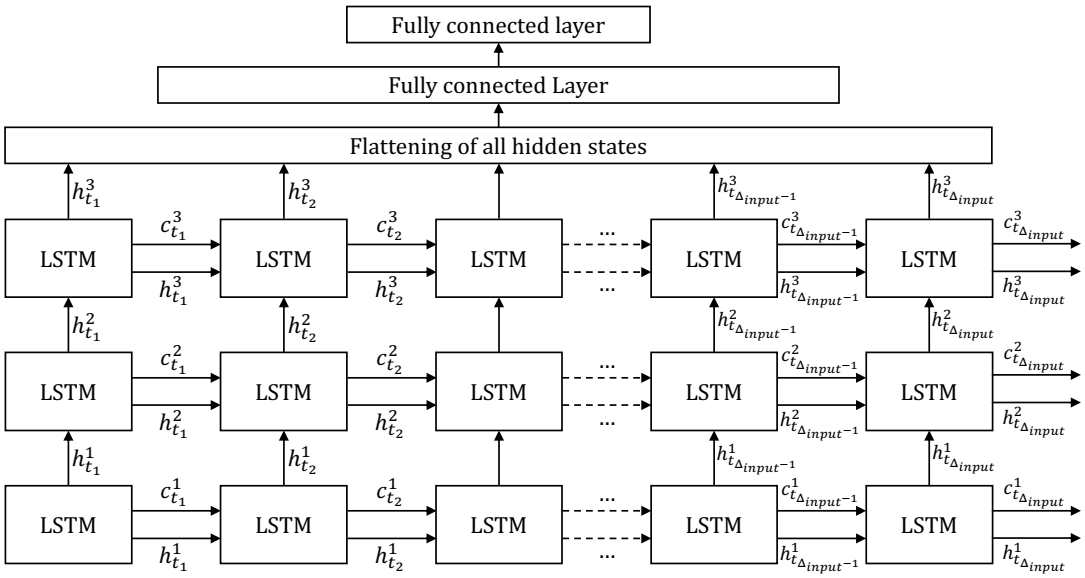
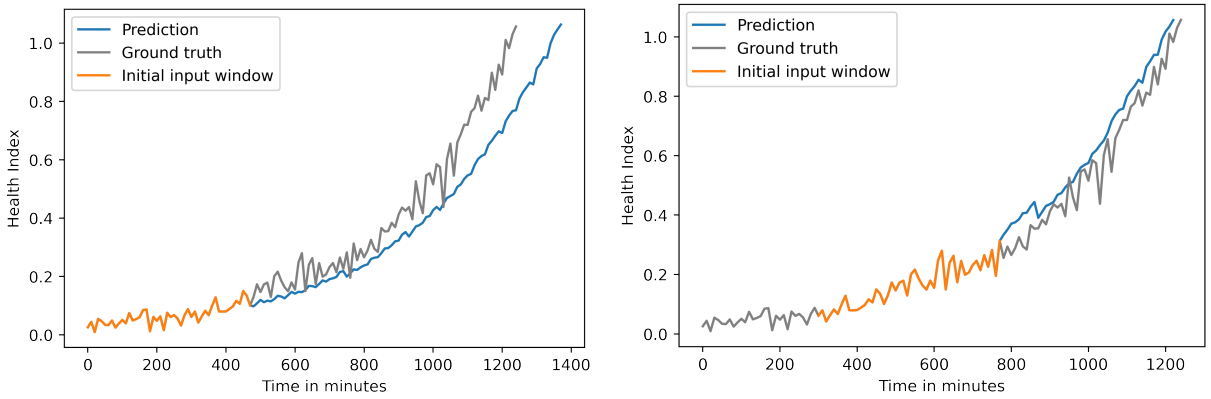


Figure 4.12: Structure of a L-layered stacked-LSTM network. In this figure, $L = 3$ and the hidden state (respectively cell state) of layer l at time step t_k is referred to as $h^l_{t_k}$ (respectively $c^l_{t_k}$). This structure is used to perform sequential predictions of HI in the Dassault Aviation experimental case.



(a) HI prediction for one TTF test trajectory, starting from $t_{k=480}$ minutes.

(b) HI prediction for one TTF test trajectory, starting from $t_{k=800}$ minutes.

Figure 4.13: Two HI predictions starting at different initial time steps.

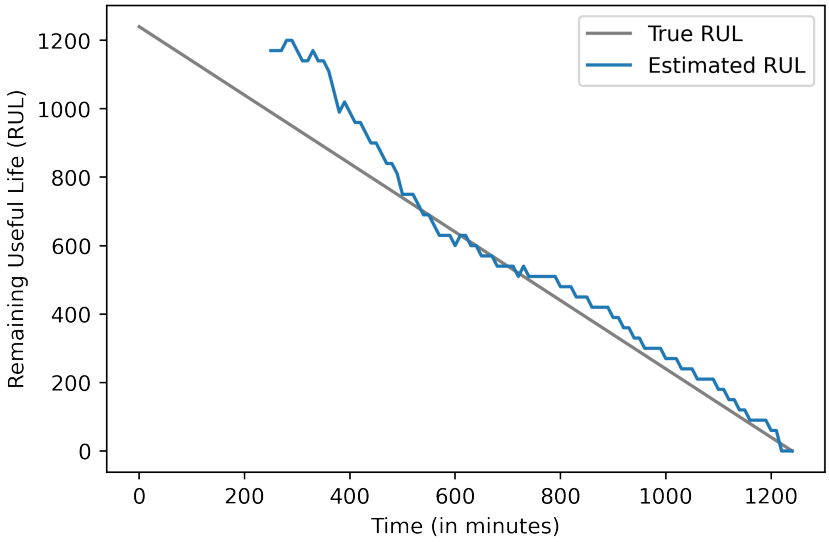


Figure 4.14: Example of a complete predicted RUL trajectory for one TTF test trajectory.

Moreover, the RMSE for each individual TTF test trajectory can be seen in Figure 4.15a. Similarly, the RMSE computed on the first fifteen and last fifteen RUL predictions is depicted in Figure 4.15b. This confirms, at the level of each individual test trajectory, the substantial improvement in prediction as the end of life of the system is approached.

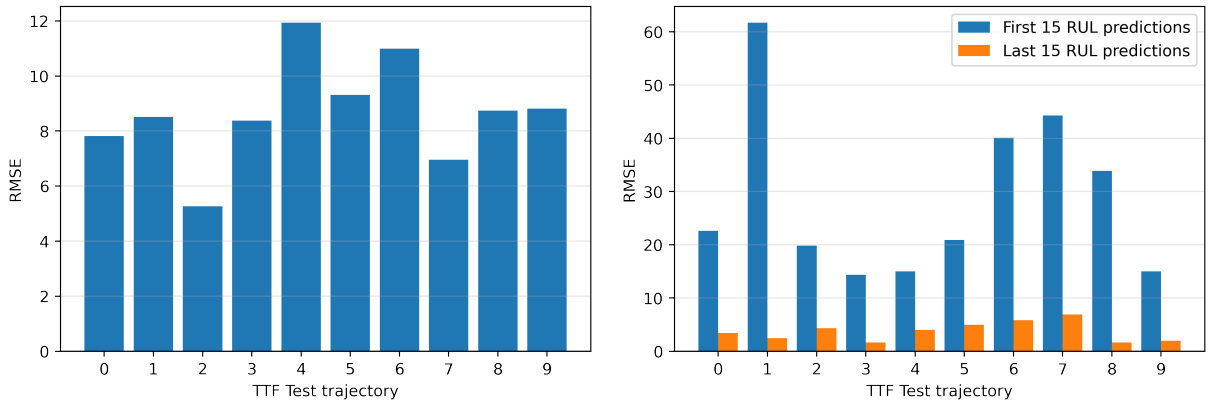
Discussion about the choice of the hyper-parameters

As presented in Section 4.2.1, for sequence-to-sequence prediction, three main hyper-parameters have a significant impact on the performance of the prediction, namely the input window length Δ_{input}, the output window length Δ_{pred} and the overlapping δ. In Table 4.4, various combinations of these three hyper-parameters are tested, with mean RMSE and standard deviation obtained in each case.

The following observations can be formulated:

- In all cases, the use of overlapping δ improves the performance of RUL prediction, all other hyper-parameters being equal. This therefore confirms the statement made in Section 4.2.1.
- The more $\Delta_{input} > \Delta_{pred}$, the better the prediction.
- It is essential to avoid the case where $\Delta_{input} < \Delta_{pred}$, i.e. trying to predict a longer sequence than the one provided as input, which would be equivalent to extrapolating excessively on the basis of limited information.

However, the choice of these parameters is never only based on the optimization of prediction performance. In reality, it is above all governed by the constraints of the system under study, therefore involving expert knowledge. In particular, the choice of Δ_{input} depends on the initial lifetime of the system where it is acceptable to have no RUL prediction. In addition, the Mean Time To Failure (MTTF) of the system (if it is known) should be taken into account when choosing Δ_{input} and Δ_{pred}. If the MTTF is small, relatively short windows should be used, whereas if it is large, input and output windows can be extended. Finally, it is absolutely



(a) RMSE for each complete TTF test trajectory.

(b) RMSE for each complete TTF test trajectory, computed on the first and last 15 predictions, demonstrating the clear improvement in the RUL prediction when approaching the EOL.

Figure 4.15: Evaluation of RUL prediction performance using RMSE.

possible to train several prediction models so as to be able to make predictions based on short input windows (Δ_{input} small) at the beginning of life of the system (i.e. when few HI values are known), and then switching to a model using longer input windows (Δ_{input} larger) when more data is available.

Input window length Δ_{input}	Output window length Δ_{output}	Overlapping δ	Mean RMSE	STD
8h	3h	0	14.89	2.4
8h	3h	1.5h	8.44	1.65
8h	8h	0	36.46	5.04
8h	8h	4h	17.18	2.97
3h	3h	0	26.2	13.13
3h	3h	1.5h	19.02	9.66
3h	8h	0	41.2	12.3
3h	8h	1.5h	35.68	12.06

Table 4.4: Impact assessment of hyper-parameters Δ_{input}, Δ_{output} and δ on RUL prediction (all other hyper-parameters being equal to the values indicated in Table 4.2).

In addition to these three time hyper-parameters, there are hyper-parameters that concern the model itself. For the stacked-LSTM model that is used here (see Figure 4.12), the key parameters are the number of LSTM layers (denoted as L) and the size of the hidden state (denoted as $|h_{t_k}|$). Choosing these hyper-parameters is formally equivalent to the question of model selection among a family of model (Bengio, 2012). Hyper-parameter tuning can be achieved by manual grid searching, by selecting hyper-parameters quoted in research papers (Brownlee, 2017) or by implementing optimization algorithms (Goodfellow et al., 2016). The results of a manual grid tuning of the number of layers and the size of the hidden state are shown in Table 4.5.

Regardless of the tuning method chosen, the crucial point is to find a trade-off between

Number of layers L	Hidden state size $\lvert h_{t_k} \rvert$	Mean RMSE	Standard deviation
	60	10.46	3.40
1	120	10.15	3.11
	240	9.31	2.73
	60	8.92	2.02
3	120	9.50	2.25
	240	**8.44**	**1.65**
	60	10.83	2.87
6	120	9.30	2.37
	240	10.63	2.70

Table 4.5: Manual grid optimization of the number of layers and hidden state size for the stacked-LSTM model used for RUL prediction (all other hyper-parameters being equal to the values indicated in Table 4.2). Best combination performance is in bold.

optimization (that is, finding the hyper-parameter configuration that yields to the best results on the available test data) and generalization (that is, expecting the same level of performance when applied to unseen real-life data). This is known as the bias–variance trade-off. In the specific application case of predictive maintenance, on the basis of the work carried out during this PhD, it is recommended to stick to models of limited complexity to give priority to the generalization capabilities and avoid over-fitting. This recommendation is particularly appropriate in cases (very often encountered in reality) where few training data are available.

4.5 Reliability-based assessment

A reliability-based assessment method is proposed in order to provide an evaluation of the RUL predictions against the laws of reliability. The aim is to demonstrate that, although the approaches are black-box, they conform to the established laws of reliability. The remainder of this part is structured as follows. The foundations of reliability are presented in Section 4.5.1. The proposed reliability-based assessment procedure is described in Section 4.5.2. Finally, it is applied to the Dassault Aviation experimental use case, and the results obtained are discussed in Section 4.5.3.

4.5.1 Fundamentals of reliability

The most commonly used function in fatigue analysis and reliability engineering is the reliability function. This function gives the probability that an item of a fleet will operate for a given amount of time without failure.

Let us consider a continuous random variable T such that $T > 0$ is the time to failure of an item. Then, the reliability function can be expressed as (Rausand and Hoyland, 2003):

$$R(t) = P(T > t) \tag{4.17}$$

which is the probability that an item survives beyond time t.
In the same way, the failure function $F(t)$ is defined as:

$$F(t) = 1 - R(t) \tag{4.18}$$

The Probability Density Function (PDF) of the failure function is therefore defined as:

$$f(t) = \frac{dF(t)}{dt} = -\frac{dR(t)}{dt} \tag{4.19}$$

Alternatively, the failure distribution can be described by the failure rate, defined as the probability of failure in a given time interval $[t, t + \Delta t]$ when $\Delta t \to 0$.

$$\lambda(t) = \lim_{t \to 0} \frac{F(t + \Delta t) - F(t)}{R(t)\Delta t} \tag{4.20}$$

Depending on the failure rate $\lambda(t)$ (whether it is increasing, decreasing or constant), different reliability distributions can be used to characterize the reliability of an equipment (O'Connor et al., 2011). Such distributions include the exponential function for constant failure rate, the normal distribution for increasing failure rate, the Poisson process for constant failure rate in discrete cases, etc. The Weibull distribution is one of the most widely used distributions in reliability engineering, as it is a versatile distribution that can take on the characteristics of other types of distributions, and be used to model any used type of failure rate.

The 3-parameter Weibull PDF is given by (McCool, 2012):

$$f(t; \eta, \beta, \gamma) = \frac{\beta}{\eta}(\frac{t - \gamma}{\eta})^{\beta - 1} e^{-(\frac{t-\gamma}{\eta})^{\beta}} \tag{4.21}$$

where $f(t) >= 0$, $t > \gamma$, and $\beta, \eta > 0$.

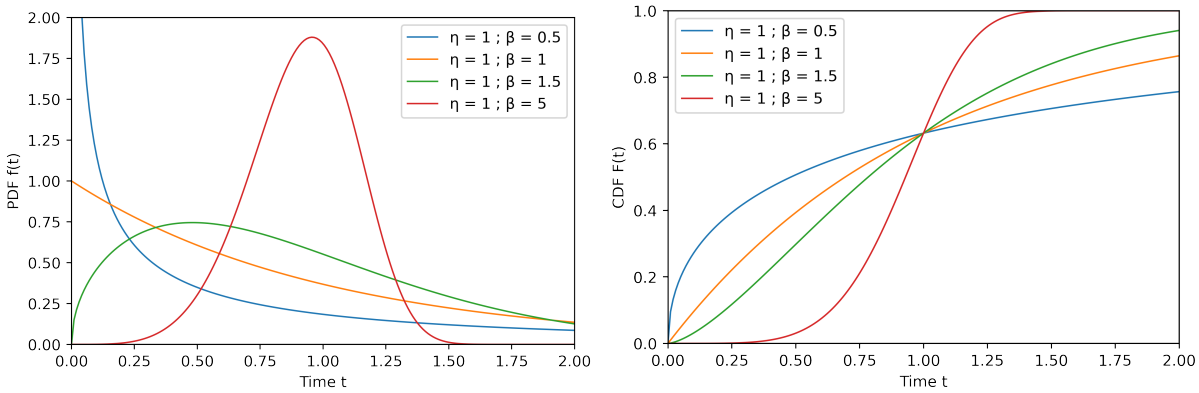
The three parameters are defined as follows:

- η is the scale parameter. It influences both mean and spread of the distribution. As η increases, all other things being equal, reliability increases at a given point in time. Moreover, slope of failure rate decreases as η increases.
- β is the shape parameter (also called the slope). Some values of this shape parameter reduce the distribution equations to those of other distributions. In particular:
  - $0 < \beta < 1$: $f(t)$ decreases monotonically, with $\lim_{t \to 0} f(t) = +\infty$ and $\lim_{t \to +\infty} f(t) = 0$
  - $\beta = 1$: $f(t) = \frac{1}{\eta} e^{-(\frac{t-\gamma}{\eta})}$. Therefore, in this case the Weibull distribution reduces to a 2-parameter exponential distribution.
  - $\beta > 1$: the Weibull PDF is positively skewed for $\beta < 2.6$, negatively skewed for $\beta > 3.7$ and similar to a normal distribution for $2.6 < \beta < 3.7$
- γ is the location parameter. As the name suggests, it locates the distribution along the horizontal axis. Changing the value of the parameter causes the distribution and its associated function to shift along the time axis. γ therefore has the same unit as t.

For the remainder of the work, it will be assumed that the degradation of the equipment starts at the origin of his life, i.e. location parameter $\gamma = 0$. In this case, the 2-parameter Weibull PDF is:

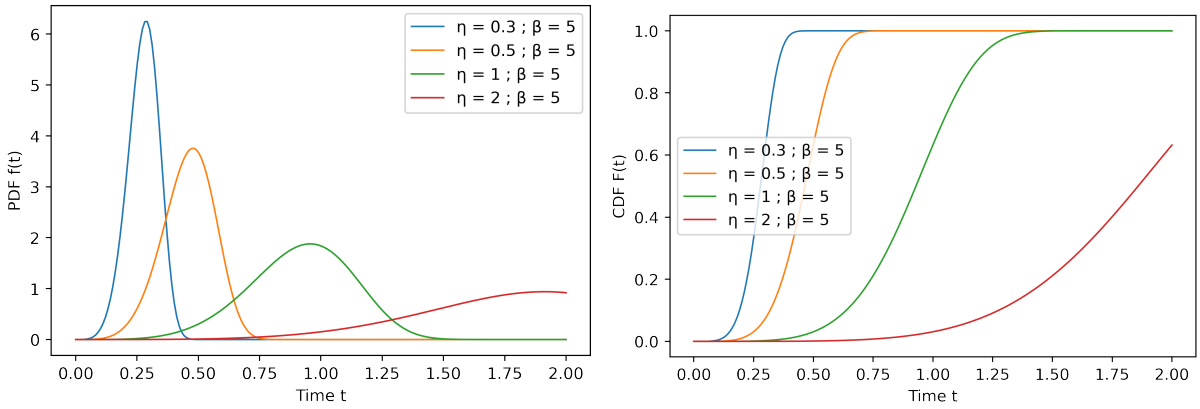
$$f(t; \eta, \beta) = \frac{\beta}{\eta}(\frac{t}{\eta})^{\beta - 1} e^{-(\frac{t}{\eta})^{\beta}} \tag{4.22}$$

where $t, \beta, \eta > 0$. Figures 4.16 and 4.17 display several PDF with associated CDF, for different combinations of parameters β and η, thereby demonstrating their respective impact on the shape of distribution.



(a) Probability Density Functions (PDF). (b) Cumulative Distribution Functions (CDF).

Figure 4.16: Several Weibull distributions for different values of the shape parameter β.



(a) Probability Density Functions (PDF). (b) Cumulative Distribution Functions (CDF).

Figure 4.17: Several Weibull distributions for different values of the scale parameter η.

The equation for the 2-parameter Weibull CDF is given by (Rinne, 2008):

$$F(t) = 1 - e^{-(\frac{t}{\eta})^{\beta}} \tag{4.23}$$

With this 2-parameter Weibull distribution, the failure rate is characterized by:

$$\lambda(t) = \frac{f(t)}{R(t)} = \frac{\beta}{\eta}(\frac{t}{\eta})^{\beta-1} \tag{4.24}$$

4.5.2 Reliability-based assessment methodology

Methodology overview

In the approach applied to Dassault Aviation experimental case, the HI is extracted and then predicted up to the EOL of the system using methods based on historical data specific to the system under study. That is, each prediction is performed at system-level, being essentially individualized, i.e. linked to the equipment studied.

An new assessment method is proposed here, which suggests evaluating the overall set of predictions made at individual scale, in order to verify their compliance with the well-established laws of reliability. Such an assessment thus aims demonstrating the robustness of the proposed methods, and providing greater credibility to the black-box AI-based models.

To that end, a set of N TTF trajectories are generated using the data-augmentation process presented in Chapter 2, with respect to a pre-selected Weibull distribution $F_{ref}(t)$. Then, these trajectories are used to obtain HI trajectories (Chapter 3) and then predicted RUL trajectories (Chapter 4). Therefore, by this global process, RUL predictions are obtained from raw sensor data. Since the RUL is essentially the remaining time before the EOL, a list of N EOL instants (i.e. scalars) can be reconstructed and used to identify the parameters of the corresponding approximated Weibull distribution. By doing so, the reference distribution F_{ref} can be compared with the approximated distribution obtained after prediction of the RUL $\widehat{F}_{pred}$.

This overall reliability-based assessment is summarized in Algorithm 9.

Algorithm 9: Overall reliability-based assessment process.

Input: A set $\mathbf{X}$ of N multi-dimensional time series $\mathbf{X}_i = \{X_{t_k}\}_{k=1}^{T_i}$ of duration T_i following a Weibull distribution F_{ref}, a selected distance d.

Output: A set of N EOL moments $\widehat{T}_i(t_{k=EOL-d})$ predicted starting from time step $t_{k=EOL-d}$ (i.e. at a distance d of the true EOL) following a distribution $\widehat{F_{RUL}}(t_{k=EOL-d})$

for *i = 1 to N* **do**

- $\mathbf{HI}_i \leftarrow m_{\Theta_m}(\mathbf{X}_i)$ where m_{Θ_m} represents the HI extraction model parameterized by Θ_m;
- $\mathbf{RUL}_i \leftarrow n_{\Theta_n}(\mathbf{HI}_i)$ where n_{Θ_n} represents the RUL prediction model parameterized by Θ_n;
- $\widehat{T}_i(t_{k=EOL-d}) \leftarrow \mathrm{RUL}_i(t_{k=EOL-d}) + k$ where $\mathrm{RUL}_i(t_k)$ is the k-th RUL value selected from trajectory $\mathbf{RUL}_i$;

end

$\widehat{F_{RUL}}(t_{k=EOL-d}) \leftarrow$ empirical CDF from all $\widehat{T}_i(t_{k=EOL-d})$;

Failure distribution parameters estimation

Assuming that a system has a Weibull lifetime distribution, it is possible to determine the values of parameters β and η that are the most likely to be the true parameter values given the available observed EOL $T_1, T_2, ..., T_N$ (Hallinan Jr, 1993, Nielsen, 2011). There are three main ways of doing this: using Maximum Likelihood Estimation (MLE) (Balakrishnan and Kateri, 2008), using Method-of-Moments Estimator (MME) (Cran, 1988) and finally using the median-rank regression estimator. Some additional estimation methods have also been proposed by Teimouri et al., 2013.

Since the median-rank regression method is relatively easy to implement and provides better results than MLE and MME for small sample size (Pobočíková and Sedliačková, 2014), which is the case here, it is this one that is chosen.

Consider N equipment, with associated EOL times $\{T_i\}_{i=1}^{N}$, the empirical failure distribution $\widehat{F}(t)$ can be estimated using the median rank estimation method defined as:

$$\widehat{F}(T_i) = \frac{n_i - 0.3}{N + 0.4} \tag{4.25}$$

where n_i is the cumulative number of faulty systems and T_i is the measured time to failure of the i-th element (Rinne, 2008).

It is assumed that $\widehat{F}(t)$ follows a Weibull distribution, so that:

$$\widehat{F}(t) = 1 - e^{-(\frac{t}{\widehat{\eta}})^{\widehat{\beta}}} \Leftrightarrow 1 - \widehat{F}(t) = e^{-(\frac{t}{\widehat{\eta}})^{\widehat{\beta}}} \tag{4.26}$$

Applying a double log transformation, a linear model is obtained:

$$\log(-\log(1 - \widehat{F}(t))) = \widehat{\beta}\log(t) - \widehat{\beta}\log(\widehat{\eta}) \tag{4.27}$$

Then, a least squares based linear regression can be applied to obtain estimated parameters $\widehat{\eta}$ and $\widehat{\beta}$ of the corresponding Weibull distribution.

Kolmogorov-Smirnov test

The Kolmogorov-Smirnov test (Stephens, 1992) compares the observed distribution of a statistical sample with a theoretical distribution. The hypothesis tested is:
"The CDF of the random variable T, denoted as F, is equal to the reference CDF F_0" with a risk of error α.

The aim is thus to obtain an estimate of the CDF from the observed sample, denoted as $\widehat{F}$, and then compare it with the CDF of the theoretical distribution, denoted as F_0. The idea behind this is that, if the tested hypothesis is true, then the empirical CDF $\widehat{F}$ of the sample must be close to the CDF F_0, because both mean and variance of the empirical CDF are unbiased estimators.

The empirical CDF is defined by:

$$\widehat{F}(t) = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}_{T_i<t} \tag{4.28}$$

The fit of the CDF $\widehat{F}$ to F_0 is quantified using a specific distance known as the Kolmogorov-Smirnov distance, which is intuitively the maximum absolute difference between $\widehat{F}$ and F_0 for each T_i according to the formula (Massey, 1951):

$$D_{KS}(F_0, \widehat{F}) = \max_{i=1,\dots,N} \left\{ \left| F(T_i) - \frac{i}{N} \right|, \left| F(T_i) - \frac{i-1}{N} \right| \right\} \tag{4.29}$$

The distance $D_{KS}(F_0, \widehat{F})$ is compared with a critical value $D_{a,N}$ provided by the Kolmogorov-Smirnov table (Berger and Zhou, 2014). This table can be found in Appendix D. The hypothesis is accepted with an α risk of error if $D_{KS}(F_0, \widehat{F}) < D_{\alpha,n}$.

In the experimental use case of Dassault Aviation, the Kolmogorov–Smirnov test is applied to ensure that the distribution $\widehat{F}_{pred}(t)$ follows the pre-selected distribution $F_{ref}(t)$.

4.5.3 Experimental results on Dassault Aviation use case

A set of 20 TTF trajectories is generated using the degraded data augmentation method described in Section 2.2, with respect to a predefined Weibull failure probability distribution $F_{ref}(t)$, with $\eta = 27$ and $\beta = 3.5$. The 20 TTF trajectories are tested through the already trained autoencoder (i.e. the weights are frozen) in order to obtain the corresponding HI trajectories, according to the procedure described in Section 3.4.2. From these 20 HI trajectories, 10 are used to train the RUL prediction model and 10 are available as a test set.

It has been stated in Section 4.4.2 that the accuracy of the RUL prediction improves as the EOL approaches, and is less reliable at the very beginning of the life of the system. For this reason, several assessments are carried out at different distances from the true EOL (distance d in Algorithm 9). That is, from all the available RUL trajectories, the EOL values T_i are retrieved from the RUL trajectories respectively at $t_{k=EOL-15h}$, $t_{k=EOL-10h}$, $t_{k=EOL-5h}$, $t_{k=EOL-3h}$. For instance, Figure 4.18 depicts the four points in time when RUL values are retrieved to perform the test, from the same complete RUL trajectory as in Figure 4.14.

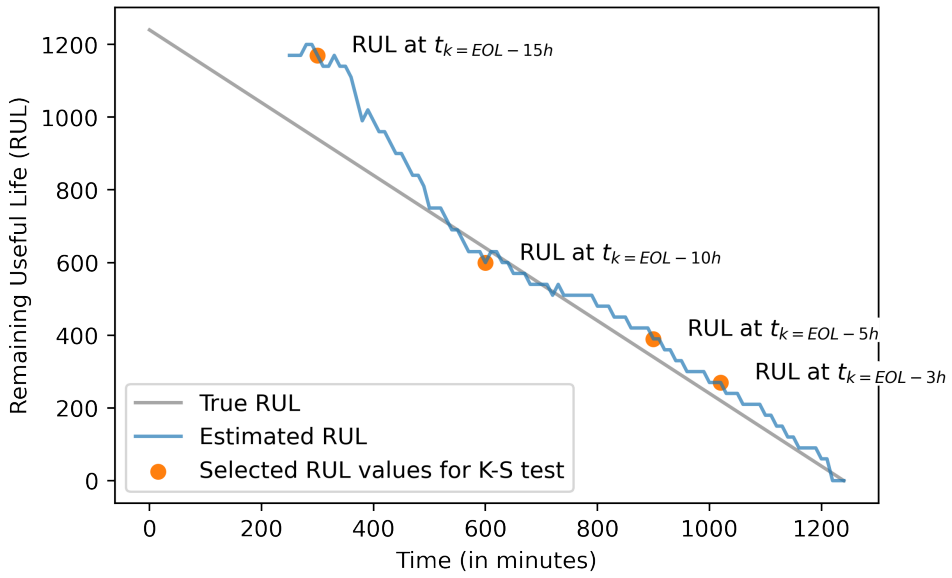


Figure 4.18: Collecting RUL values at four different time moments to perform several Kolmogorov-Smirnov tests.

From the collected RUL values, the associated EOL times T_i are deduced, thus leading to four different observed samples. In each case, the CDF is estimated using the procedure described

above, so that a Kolmogorov-Smirnov test can be handled.

It should be noted that the number N of available T_i in each observed sample varies, since not all trajectories are of the same duration and consequently some are too short to match the desired prediction distance from the EOL.

The results of the four tests performed are presented in Table 4.6.

	N TTF available	$\widehat{\eta}$	$\widehat{\beta}$	K-S Distance $D_{KS}(\widehat{F}_{pred}, F_{ref})$	Critical limit $D_{0.05,N}$	Hypothesis accepted? (i.e. $D_{KS}(\widehat{F}_{pred}, F_{ref}) < D_{0.05,N}$)
Reference Weibull distribution for data generation	20	27	3.5	-	-	-
Distribution $\widehat{F}_{pred}$ at $t_{k=EOL-3h}$	10	24.8	5	0.125	0.410	Yes
Distribution $\widehat{F}_{pred}$ at $t_{k=EOL-5h}$	8	22.8	3.9	0.24	0.457	Yes
Distribution $\widehat{F}_{pred}$ at $t_{k=EOL-10h}$	5	23.1	3.9	0.4	0.565	Yes
Distribution $\widehat{F}_{pred}$ at $t_{k=EOL-15h}$	4	22.1	3.4	0.5	0.624	Yes

Table 4.6: Parameters of the Weibull distribution(Reference vs. estimated from predicted RUL values) at different moments, along with Kolmogorov-Smirnov test result.

The hypothesis is verified in all cases. Moreover, the distance $D_{KS}(\widehat{F}_{pred}, F_{ref})$ is smaller and smaller as the EOL is approached, demonstrating that the prediction is increasingly robust. This corroborates the statements asserted in Section 4.4.2. It is a very important feature, because the nearer the EOL of a system, the greater the need to monitor its SOH in order to ensure optimal maintenance.

This reliability-based assessment procedure thus consolidates the proposed approach of the PhD. Although the prognostics is performed at system-level, on the basis of historical data collected from each individual equipment, it is demonstrated that by combining all individual predictions, the established laws of reliability are respected. Of course, the prediction approach remains black-box, but the results obtained are validated by recognized traditional methods, thus providing greater credibility in the AI-based models.

4.6 Conclusion

In this chapter, a two-step approach for RUL prediction has been proposed. Firstly, it consists of predicting the HI in a sequential manner, i.e. to predict overlapping fixed-size HI windows until the EOL of the system is reached, followed by deduction of the RUL in a recursive manner. Two methods for detecting the EOL are proposed, namely by exceeding a pre-selected threshold and by recognizing a reference pattern. They are then applied to the C-MAPSS reference dataset. By applying this two-stage approach to the Dassault Aviation application case, it is demonstrated that this proposal is well adapted to industrial realities. In particular, it ensures an increase in prediction performance as EOL approaches, which is extremely valuable from an industrial point of view.

The physics-based data augmentation mechanism presented in Chapter 2 generates the HI trajectories needed to train the HI prediction models. In doing so, the use of RUL-labeled data is completely avoided.

Finally, a reliability-based assessment is also proposed, in order to demonstrate that RUL prediction performed at system-level, using individual historical data are consistent with reliability laws. It is applied to the Dassault Aviation experimental case and lends credibility to the

proposed approach. Although AI-based architectures are black-box, this additional validation guarantees compliance with the established system reliability principles.

Chapter 5

General conclusion

5.1 Conclusion

At the beginning of this PhD manuscript, two main research challenges were highlighted:

- Firstly, research challenge 1 aimed at attempting to limit dependence on measured RUL-labeled data, since such data are generally unavailable in industrial applications.

- Secondly, research challenge 2 was focused on leveraging the existing system knowledge and physics of degradation in order to strengthen the AI-based prognostics.

In Chapter 1, the general context of prognostics has been introduced, and more specifically the use of AI-based methods for prognostics has been presented, with their potential advantages and drawbacks. A global approach has been proposed to overcome the two main research challenges identified, with the objective of being particularly applicable in real industrial situations. Throughout the following three chapters of this thesis, the stages of the proposed prognostics framework have been described in detail. In Chapter 2, a novel hybrid data augmentation strategy has been presented, where a nominal model is first built using data-driven system identification methods. Then, a physics-based degradation model has been injected in order to generate data under degradation. Thereby, this approach simultaneously augments the nominal data (i.e. when the system is healthy) and generates new degradation-enriched data. Then, in Chapter 3, several approaches based on autoencoders have been proposed for extracting HI from the raw data collected by sensors. These approaches were first applied to the reference C-MAPSS dataset, then to the experimental Dassault Aviation case (a temperature control system in the cockpit of the Falcon 6X). The preferred solution is an unsupervised indirect HI extraction approach using the reconstruction error, so that only nominal (i.e. healthy) data is used during training. Finally, the third stage of the proposed approach, which consists of long-term HI forecasting in order to predict the RUL, has been described in Chapter 4. As in the previous chapter, different methods have been explored through two studies applied to the C-MAPSS academic dataset. Then, the experimental use case of Dassault Aviation has been addressed and discussed. This dual source of data (C-MAPSS and Dassault Aviation) makes it possible to combine an academic application that is well-recognized in the PHM community with an industrial experimentation closer to field reality. In addition, a reliability-based assessment has been presented and applied, offering validation of the black-box AI-based approach with respect to established laws of reliability.

5.2 Contributions

Following the two research orientations mentioned above, several proposals have been made in this thesis, leading to various contributions:

- The use of data-driven system identification drastically reduces the amount of training data required for data augmentation task. Compared to generative data augmentation methods (GANs), this makes it possible to achieve nominal data augmentation in a realistic industrial context where very few measured data are available, thus addressing research challenge 1.
- Hybrid data augmentation, achieved by injecting a physics-based degradation model inside of the identified nominal model, allows to exploit both *a priori* knowledge about the system and about the Physics-of-Failure (PoF), in order to generate complete Time to Failure (TTF) trajectories without any use of measured degradation data. This approach, by integrating explicit knowledge directly into the training data which is then used to train neural structures, is related to PINNs (Pan et al., 2022). This therefore simultaneously fulfills research challenges 1 and 2.
- The fully unsupervised approach for extracting Health Index (HI) from sensor data using the reconstruction error of an autoencoder allows HI to be obtained only on the basis of nominal data (i.e. when the system is healthy). In addition, the long-term prediction of HI using LSTM-based structures provides a means of deducing RUL without using RUL-labeled data. Therefore, these two proposals prevent any use of RUL-labeled data in the whole prognostics approach, thus addressing research challenge 1.
- The proposed reliability-based assessment allows RUL prediction results to be assessed against established laws of reliability. The validation of the predefined Weibull distribution hypothesis ensures the integrity of the AI-based approaches used, even though they are black-box.
- Last but not least, a dual data source was used to validate the proposed approach. Applications were first carried out on the reference dataset C-MAPSS, which is widely used in the PHM community. This ensures the reproducibility of the obtained results from an academic point of view. Then, the overall prognostics framework has been applied to a real-life industrial case in collaboration with Dassault Aviation. This experiment confers a particular interest to the work presented by confronting it to realistic industrial problems.

5.3 Perspectives

5.3.1 Academic research perspectives

In addition to the two research challenges mentioned above, some other scientific questions appeared during the development of this PhD work. They constitute promising perspectives for further work on the subject.

- *Expansion of a priori knowledge in AI-based prognostics:* AI-based structures are black-box approaches that use complex computational models which have proven to be very efficient, but may on rare occasions deliver unexpected outlier predictions. As handled in this PhD work, the integration of *a priori* knowledge about the system inside of the training data

helps to focus the search space, making neural networks more consistent with the laws of physics. In future works, this knowledge integration could be extended. For instance, the Physics-of-Failure (PoF) could be embedded as linear equations based on physics laws inside of the neural network, through weighted connections between the input layer and the first hidden layer (Ma et al., 2023). In the same spirit, physics-augmented features could be employed to improve the prediction performances (Chao et al., 2022). Another possible extension would be to add one or more physics-based losses to the standard loss function of the learning algorithm (R. Guo et al., 2023). This would regularize the training on the basis of physics-based constraints, thus guiding the network towards a model compliant with the laws of physics.

- *Confidence intervals for AI-based decision support:* Another very valuable area of research in prognostics concerns the estimation of confidence intervals for neural network approaches. Generally, a RUL prediction is associated with a confidence interval to assist the maintenance decision process. The calculation of uncertainty in neural networks is a field of study in its own right (Gawlikowski et al., 2023). A few solutions applied to prognostics have been explored (K. Liu et al., 2020, Mazaev et al., 2021, C. Chen et al., 2023). Nevertheless, a PhD work could legitimately be devoted to this entire subject.

- *Generalization of the degraded data augmentation:* In this PhD work, degradation was injected into one component of the system, namely the actuator (i.e. the valve) as a gradual clogging process, creating friction. In future developments, it could be envisaged to generalize injected degradation in order to encompass additional failure modes. This could be achieved by injecting new degradation mechanisms into the actuator, but also by extending the degradation process to other components of the system. In particular, sensor drift could be considered. This increase in the complexity of the generated degradation would make it possible to assess the robustness of the proposed approach by extending it to a wider variety of situations.

5.3.2 Industrial perspectives

The certification of the Falcon 6X issued on 22 August 2023 by the EASA (European Aviation Safety Agency) and the FAA (Federal Aviation Administration) paves the way for its commercial launch. Moreover, the first commercial models are currently being finished. In this perspective, several industrial projections are possible:

- First of all, the system identification phase could be improved by collecting new data when the system is in its final condition, all adjustments having been made. Implementing a design of experiment on a flight could make it possible to collect sufficiently excited data to apply closed-loop identification methods and thus offer results potentially closer to the real system. In this way, the nominal data augmentation could be even more faithful to reality.

- The maintenance of aeronautical systems is obviously a complex and critical task, given the safety issues involved. While it is obviously not envisaged to apply the novel proposed prognostics approach immediately, it would be particularly interesting to test it as a "duplicate" of the current maintenance strategy. By monitoring the air distribution system of Falcon 6X aircraft during commercial operation using the prognostics method proposed in this thesis, while keeping effective the maintenance currently applied, it would be possible

to compare the two approaches and assess the advantages of the proposed approach, as well as its limitations.

- Finally, the global approach proposed in this PhD work could be extended to other controlled aircraft systems equipped with sensors, such as the landing gear. In this perspective, data augmentation could of course be achieved through the use of data-driven system identification approaches, but models could also be provided directly by the various Original Equipment Manufacturers (OEMs), and/or developed on a physics-based approach.

Appendix A

Presentation of the C-MAPSS Dataset

The Commercial Modular Aero-Propulsion System Simulation (C-MAPSS), as presented in Frederick et al., 2007, is a numerical model, developed by NASA, that aims to simulate the functioning of a 90,000 lb thrust class commercial aircraft turbofan engine. In this annex, the simplified operation of a turbojet engine is first presented (Section A.1), then the C-MAPSS simulator is introduced (Section A.2) and finally the dataset generated by this simulator is described (Section A.3). Moreover, a description of the data pre-processing and selection operations prior to the use of the dataset in publications (Hervé de Beaulieu et al., 2022b; Hervé de Beaulieu et al., 2022c) is described in Section A.4.

A.1 Simplified operation of a turbojet engine

A jet engine, also called gas turbine, is a means of propulsion which consists of producing a powerful thrust to push the aircraft forward. A working fluid is accelerated by the engine and the reaction to this acceleration produces a force on the engine (Newton's third law of action and reaction).
To go into a little bit more detail, air is sucked at the front, using a fan. This incoming air continues to the core of the engine where the high pressure compressor, made of many blades attached to a shaft spinning at high speed, squeezes the air, thus resulting in an increase in its pressure. The compressed air is then sprayed with fuel and an electric spark lights the mixture in the combustor. By this combustion, where the heat can reach $2700°C$, gases expand and blast out through the rear turbine, causing the turbines blades to rotate. These blades are mounted on the same shaft as the compressor and allow it to rotate and compress the air at the inlet. Finally, this accelerated airflow exits at the nozzle at the back of the engine, creating the thrust. The narrowing of the motor housing at the outlet also contributes to the acceleration of the jet. All of the above-mentioned components are shown in Figure A.1.

Turbofan engines are a specific type of turbojet where the core engine is surrounded by a large fan at the front and an additional turbine at the rear. It is the most advanced variant of the standard gas turbine engine and powers the vast majority of modern aircraft.
Thanks to its larger fan, the incoming air is divided into two flows. The first part continues to the core of the engine, following the process described in the previous paragraph, while the second part bypasses the core of the engine, going through a duct that surrounds the core to the back of the engine, thus remaining cooler, helping to quiet the engine and adding thrust to it. This operation can be seen in Figure A.2. Therefore, the turbofan allows most of the air to

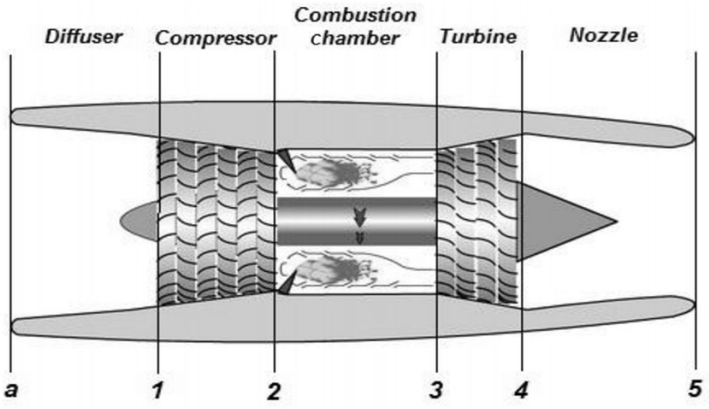


Figure A.1: Simplified sketch of a turbojet aircraft engine (from Najjar and AbuEisheh, 2016).

flow around the outside of the engine, thereby increasing the total air mass flow and reducing a little the velocity, for a very similar total energy input. The ratio of air going around the engine to air going through the core is called the bypass ratio. Since thrust is a function of the mass flow through the engine and the exit velocity of the gas, increasing the air mass flow by a large amount while decreasing its velocity by a small amount is a good way to increase thrust overall without using much more fuel.

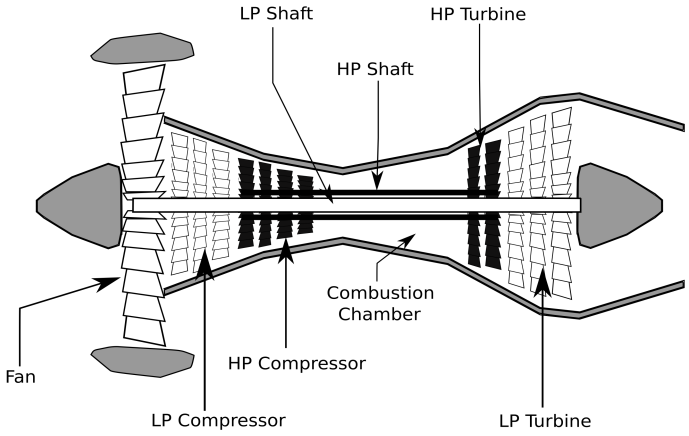


Figure A.2: Simplified diagram of a turbofan aircraft engine (from Hazan et al., 2010).

A.2 The C-MAPSS simulator

The numerical model of the C-MAPSS turbofan engine has been implemented in the MATLAB/SIMULINK environment, and provides a graphical user interface to allow the user to customize the parameters, closed-loop controllers, operating conditions of the model, etc.
The detailed lists of input and output variables of the system are provided in Tables A.1 and A.2.

As any turbofan, the C-MAPSS turbofan is composed of an inlet fan, a compressor, itself

Index	Description	Variable name	Units
1	Fuel flow	Wf	pph
2	Fan efficiency modifier	fan_eff_mod	%
3	Fan flow modifier	fan_flow_mod	%
4	Fan pressure ratio modifier	fan_PR_mod	%
5	LPC efficiency modifier	LPC_eff_mod	%
6	LPC flow modifier	LPC_flow_mod	%
7	LPC pressure ratio modifier	LPC_PR_mod	%
8	HPC efficiency modifier	HPC_eff_mod	%
9	HPC flow modifier	HPC_flow_mod	%
10	HPC pressure ratio modifier	HPC_PR_mod	%
11	HPT efficiency modifier	HPT_eff_mod	%
12	HPT flow modifier	HPT_flow_mod	%
13	LPT efficiency modifier	LPT_eff_mod	%
14	LPT flow modifier	LPT_flow_mod	%

Table A.1: C-MAPSS simulator input variables.

consisting of a Low Pressure Compressor (LPC) and a High Pressure Compressor (HPC), a combustor, an output turbine made up of a High Pressure Turbine (HPT), and a Low Pressure Turbine (LPT). These elements can be visualized in Figure A.3. This engine simulator allows faults to be injected in any of the five rotating components (fan, HPC, LPC, HPT, LPT) by varying the health parameters 2 to 14 (see Table A.1). It provides output responses for 27 readable variables (see Table A.2).
The user can define different operational conditions profiles in order to simulate a number of flight environments at altitudes ranging from sea level to $40,000ft$, Mach numbers from 0 to 0.90, and sea-level temperatures from -60 to $10°F$.
Finally, C-MAPSS can be operated either in open loop or in closed loop (with the engine and its control system) configurations.

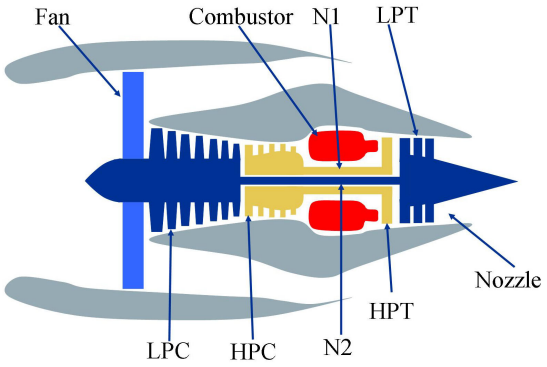


Figure A.3: Simplified diagram of the C-MAPSS turbofan engine (from Frederick et al., 2007).

Index	Description	Variable name	Units
1	Physical fan speed	Nf	RPM
2	Physical core speed	Nc	RPM
3	Engine pressure ratio (P50/P2)	EPR	–
4	Total pressure at fan outlet	P21	psia
5	Total temperature at fan outlet	T21	R
6	Total pressure at LPC outlet	P24	psia
7	Total temperature at LPC outlet	T24	R
8	Total pressure at HPC outlet	P30	psia
9	Total temperature at HPC outlet	T30	R
10	Total pressure at burner outlet	P40	psia
11	Total temperature at burner outlet	T40	R
12	Total pressure at HPT outlet	P45	psia
13	Total temperature at HPT outlet	T48	R
14	Total pressure at LPT outlet	P50	psia
15	Total temperature at LPT outlet	T50	R
16	Fan flow	W21	pps
17	Net thrust	Fn	lbf
18	Gross thrust	Fg	lbf
19	Fan stall margin	SmFan	%
20	LPC stall margin	SmLPC	%
21	HPC stall margin	SmHPC	%
22	Corrected fan speed	NfR	RPM
23	Corrected core speed	NcR	RPM
24	Total pressure in bypass-duct	P15	Psia
25	Percent corrected fan speed	PCNfR	%
26	Static pressure at HPC outlet	Ps30	psia
27	Ratio of fuel flow to Ps30	Phi	pph/psia

Table A.2: C-MAPSS simulator output variables.

A.3 The C-MAPSS Dataset

The C-MAPSS dataset contains a number of TTF turbofan engines trajectories. This dataset has been obtained by using the C-MAPSS simulator, in the closed-loop configuration. The entire process for obtaining data from the simulator is described in Saxena et al., 2008.
For each simulation case, an initial level of wear is given, considered as normal for a nominal behavior. Then, faults are initiated at a random time during the simulation, following an exponential law whose distribution is given below.

$$f(t) = \begin{cases} \lambda e^{-\lambda t} \text{ if } t \geqslant 0 \\ 0 \text{ if } t < 0 \end{cases} \tag{A.1}$$

where $\lambda > 0$ is the parameter of the distribution (usually called the rate parameter). The engine health is chosen to be the minimum health margin of the rotating equipment, where the health margin is a function of efficiency and flow for this particular component. When this health indicator reaches zero, the simulated engine is considered to have reached its EOL, which means that its RUL is 0. From this process, the RUL can be obtained recursively at each operating cycle.

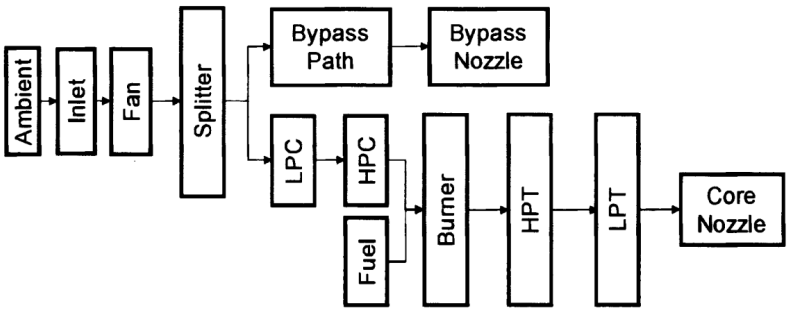


Figure A.4: Schematic diagram of the different modules used in the simulation process (from Saxena et al., 2008).

Out of the twenty-seven output variables available in the simulator, only twenty-one are selected to appear in the dataset, as raw training data. They are listed in Table A.4, along with their name assigned in the simulator and the one assigned to them in the dataset (which are not the same). Among the five rotating elements on which a fault or a deterioration can be simulated, only two are concerned in this dataset, namely the HPC and the fan.
The C-MAPSS dataset actually consists of four distinct subsets referred to as FD001, FD002, FD003 and FD004, each containing a training set composed of complete degradation sequences of a number of turbines, and a test set. For each of the four subsets, both turbine populations (training and test) are assumed to belong to the same distribution. Details about the four datasets are provided in Table A.3. The fault modes in the datasets vary between one (HPC degradation) in FD001 and FD002 and two (HPC degradation and fan degradation) in FD003 and FD004, while the operating conditions vary between one (sea level) in FD001 and FD003, to six, based on different combinations of altitude (0 to 42000 feet), throttle resolver angle (20 to 100) and Mach (0 to 0.84) in FD002 and FD004. Therefore, the prediction complexity is increasing from FD001 to FD004. In the present study, where work was carried out without the help of RUL labels, only FD001 dataset was used

	C-MAPSS subsets			
	FD001	FD002	FD003	FD004
Engine units for training	100	260	100	249
Engine units for testing	100	259	100	248
Operating conditions	1	6	1	6
Fault modes	1	1	2	2

Table A.3: Specific features of the C-MAPSS subsets

Each turbine i of each of the four datasets is arranged in an T_i-by-26 matrix, i.e. it is a multivariate sequence of length T_i, corresponding to the complete life of the engine. Each time step between 1 and T_i is a single operating time cycle. When the T_i^{th} cycle is reached (the last data entry for a given turbine i), it means that the turbine is out of usage. At every time step corresponds a series of 26 variables. The first variable represents the engine number, the second is the number of the current cycle (i.e. time step), third to fifth variables represent three operational settings and the variables left are the sensor readings, i.e. the training variables. The three

C-MAPSS Simulator Variable Symbol	C-MAPSS Dataset Sensor Number	Description	Unit
T2	s1	Total temperature at fan inlet	$^{\circ}R$
T24	s2	Total temperature at LPC outlet	$^{\circ}R$
T30	s3	Total temperature at HPC outlet	$^{\circ}R$
T50	s4	Total temperature at LPT outlet	$^{\circ}R$
P2	s5	Pressure at fan inlet	*psia*
P15	s6	Total pressure in bypass-duct	*psia*
P30	s7	Total pressure at HPC outlet	*psia*
Nf	s8	Physical fan speed	*rpm*
Nc	s9	Physical core speed	*rpm*
epr	s10	Engine pressure ratio (P50/P2)	$--$
Ps30	s11	Static pressure at HPC outlet	*psia*
phi	s12	Ratio of fuel flow to Ps30	*pps/psi*
NRf	s13	Corrected fan speed	*rpm*
NRc	s14	Corrected core speed	*rpm*
BPR	s15	Bypass Ratio	$--$
farB	s16	Burner fuel-air ratio	$--$
htBleed	s17	Bleed Enthalpy	$--$
Nf_dmd	s18	Demanded fan speed	*rpm*
PCNfR_dmd	s19	Demanded corrected fan speed	*rpm*
W31	s20	HPT coolant bleed	*lbm/s*
W32	s21	LPT coolant bleed	*lbm/s*

Table A.4: The selection of the simulator output variables used as training variables in the C-MAPSS dataset

operational settings are parameters that were used by the team who developed the dataset to calculate the HI, in order to be able to define the EOL of the turbine. To those data are attached a label, that is the RUL, corresponding to the number of cycles left, before the failure of the system.

A.4 C-MAPSS Dataset observation and selection

Among all the 21 sensor readings available, some are very correlated, and some always remain constant. A plot of the 21 sensor readings for one given turbine of dataset FD001 can be found in Figure A.5. In particular, in Ramasso and Saxena, 2014, a number of techniques were reviewed, such as PCA, in order to find the best combination of sensors to describe the behavior of the degradation of the turbine. Based on the analysis conducted by T. Wang et al., 2008, it was noted that the most meaningful and richest subset of sensors is: 2, 3, 4, 7, 11, 12, 15.
This sensor selection was used for all the studies carried out on C-MAPSS presented in this thesis. It can be visualized (after normalization) for a random turbine from FD001 in Figure A.6.

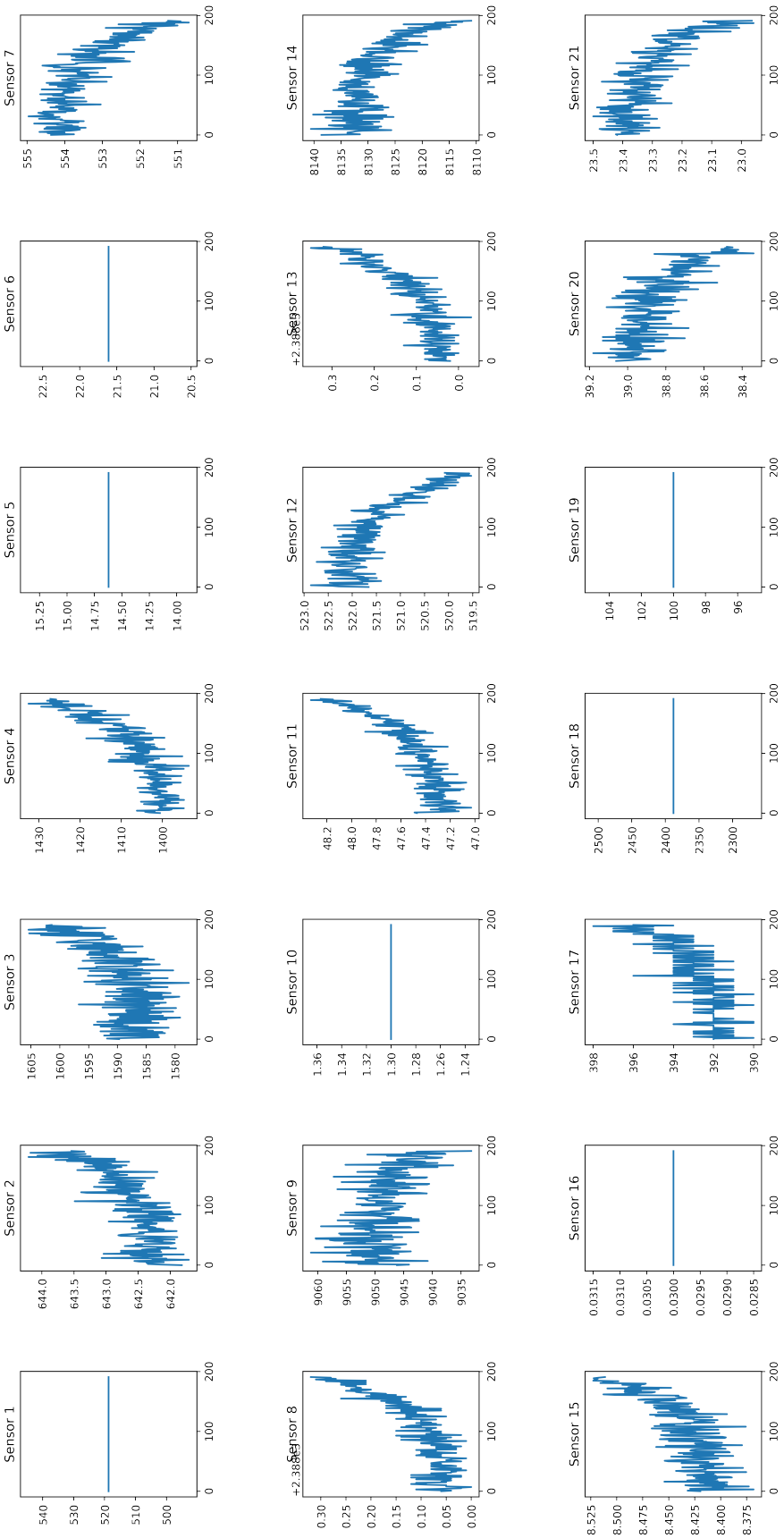


Figure A.5: Overview of the 21 sensor readings for one turbine from the FD001 set of C-MAPSS.

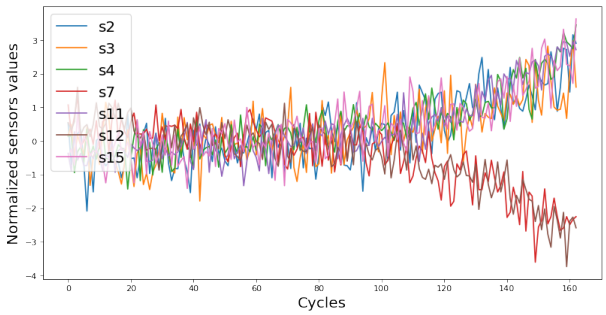


Figure A.6: Selected sensors readings for the complete lifetime of a turbine from FD001 set.

Data preparation for RUL prediction

In order to ensure an equal contribution from all sensor readings (usually called features in the Machine Learning (ML) community), it is necessary to perform some operation of data normalization. Therefore, all the data are normalized together, using standard scaling.

$$Norm(x^s) = \frac{x^s - \mu^s}{\sigma^s} \tag{A.2}$$

where s is the selected sensor, μ^s and σ^s are the mean and the standard deviation for the selected sensor.

Appendix B

CONTSID: a Matlab toolbox for continuous-time model identification

B.1 Overview of the CONTSID toolbox

The CONtinuous-Time System IDentification (CONTSID) toolbox provides MATLAB functions for estimating continuous-time black-box models of dynamical systems from measured data. It was first released in 1999 (Gamier and Mensler, 1999).

The toolbox includes tools for standard identification of linear continuous-time models such as simple process, transfer functions and state-space models. Although standard parametric estimation techniques such as subspace and Prediction Error Method (PEM) are provided, most learning algorithms exploit the Instrumental Variable (IV)-based estimation method which receives a special focus.

The CONTSID includes many demonstration programs to illustrate its use and can be freely downloaded at `www.cran.univ-lorraine.fr/contsid`.

This appendix presents a brief overview of the main features of the latest release of the CONTSID toolbox and outlines its use for low-order process model identification.

B.2 Main CONTSID toolbox estimation commands

The CONTSID toolbox contains a range of parametric model estimation methods for common linear and non-linear model structures. The main commands for standard linear model identification and for more advanced identification are respectively summarized in Tables B.1 and B.2.

B.3 The CONTSID Graphical User Interface

The CONTSID now integrates a Graphical User Interface (GUI) (Garnier et al., 2021 that allows the user to to analyze experimental data, identify and evaluate linear models in an intuitive way. Figures B.1, B.2 and B.3 give a brief overview of the operations that can be carried out with this GUI.

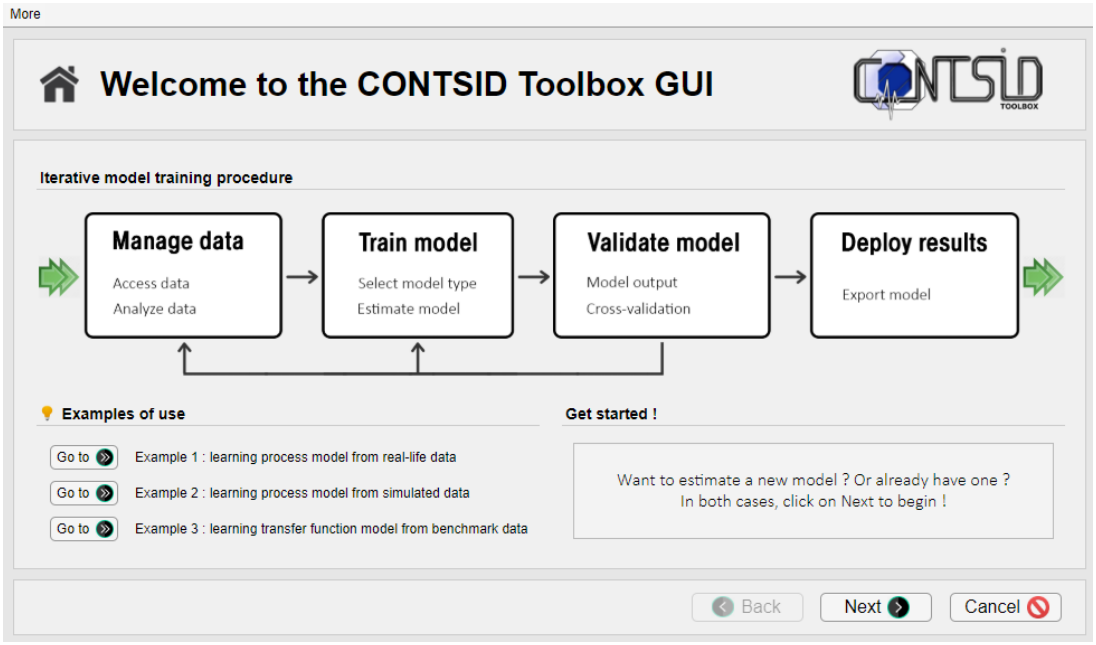


Figure B.1: Home page of the CONTSID GUI.

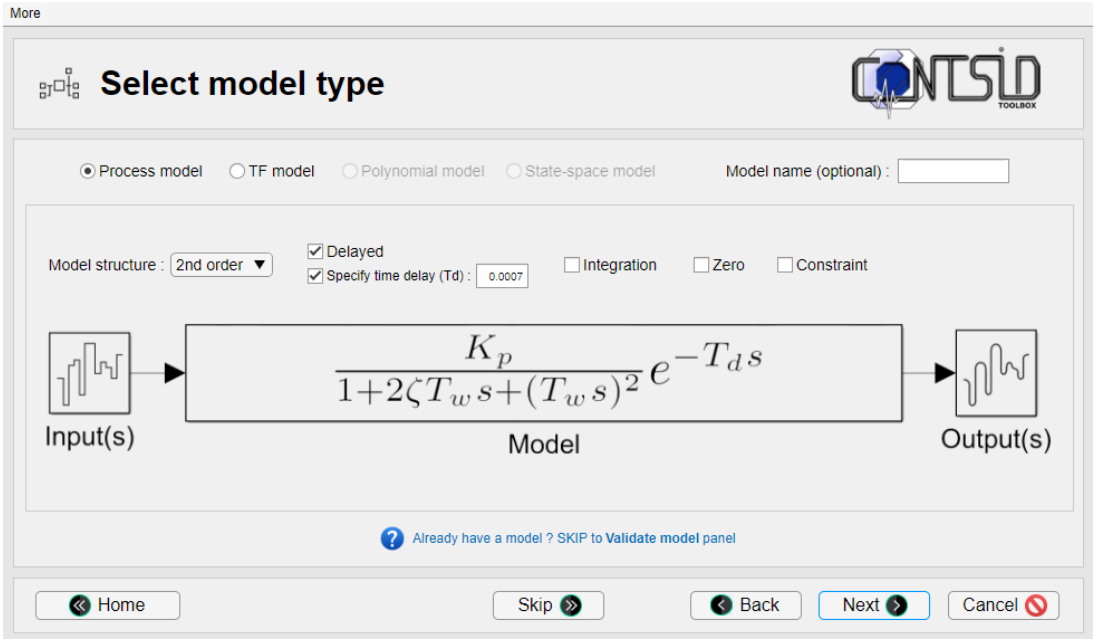


Figure B.2: Selection of the model type using the CONTSID GUI.

Model type	Estimation commands
Transfer function models	`tfsrivc`
Process models	`procsrivc`
Input/output polynomial models	`lssvf` (CARX models) `ivsvf` (CARX models) `coe` (COE models) `srivc` (COE models) `rivc` (CBJ models)
State-space models	`sidgpmf` `ssivgpmf`

Table B.1: Main CONTSID toolbox commands for standard linear model identification

Model Type	Estimation commands
Polynomial models in closed loop	`clsrivc` `cl2srivc`
EIV models	`focils`
Hammerstein models	`hsrivc`
Hammerstein-Wiener models	`hwsrivc`
LPV input/output models	`lpvsrivc`

Table B.2: Main CONTSID toolbox commands for more advanced identification

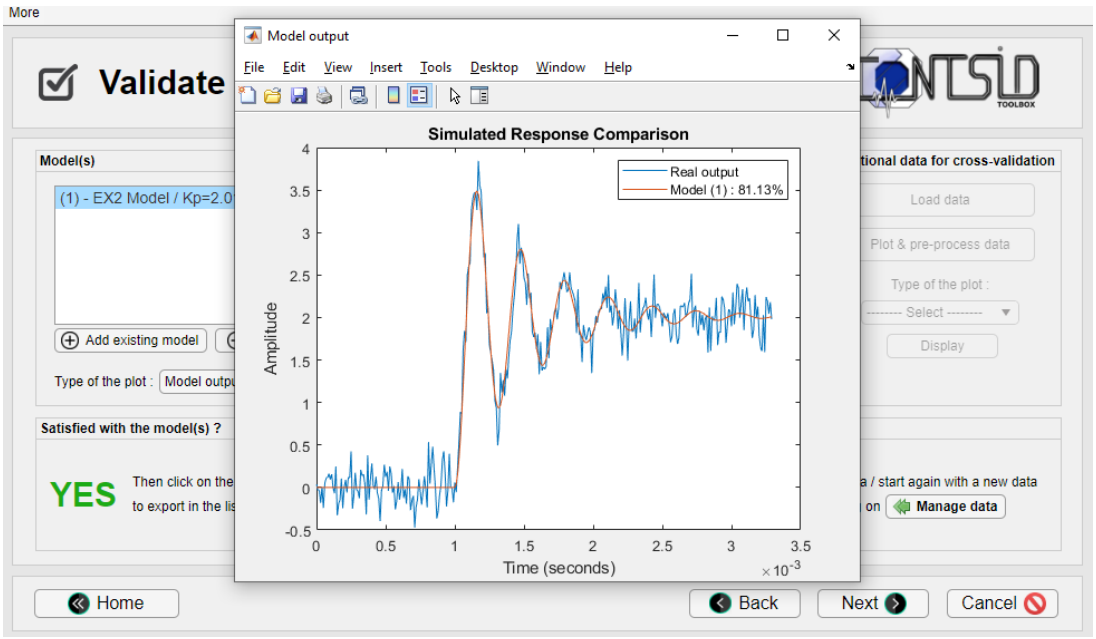


Figure B.3: Validation of an estimated model using the CONTSID GUI.

Appendix C

State of the art of the main LSTM-based architectures

This appendix presents the fundamental principles of LSTM-based recurrent networks. It is structured as follows. Section C.1 describes the functioning of a standard LSTM cell as well as the basic vanilla architecture. Section then C.2 introduces the stacked-LSTM architecture. Section C.3 finally explains the structure of bidirectional stacked-LSTM networks. Note that throughout this appendix, t refers to the time step.

C.1 Vanilla LSTM Model

Some very basic knowledge about RNN is provided in Section 1.2.1. LSTMs are simply a more advanced RNN model, where the standard cells are designed to remember information for long periods of time thanks to logic gates that allow the structure to forget useless information and add meaningful information. The internal operation of an LSTM cell can be seen in Figure C.1, with the associated variables.

The core element of an LSTM-based network is the cell state which acts as a conveyor belt, running through the entire chain of neurons and containing the information that is transmitted from cell to cell. Each LSTM cell receives three inputs: the cell state of the previous cell C_{t-1}, the hidden state of the previous cell h_{t-1}, and the current input x_t. Internally, each cell contains

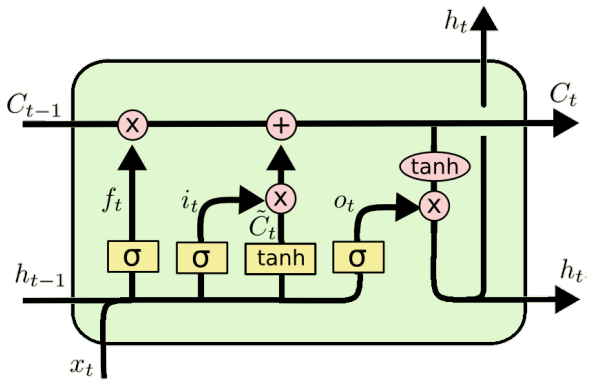


Figure C.1: Details of variables within an LSTM cell (from Olah, 2015)

a forget gate, an input gate and an output gate, which are defined hereafter.

- The forget gate aims to remove unnecessary information from the cell state by looking at the previous hidden state h_{t-1} and current input x_t. To that end, both h_{t-1} and x_t are passed through a sigmoid layer which outputs a number between 0 and 1 for each value in the cell state C_{t-1}. A value of 1 means that the information is completely retained while a value of 0 indicates that it is completely deleted. The cell state is then updated via an element-wise product.

- The input gate allows new information to be added inside of the cell state. Similarly to the forget gate, h_{t-1} and x_t are passed through a sigmoid layer, while a tanh layer creates a vector of new candidate values $\widetilde{C_t}$, that could be added to the cell state. Finally, i_t and $\widetilde{C_t}$ are combined thanks to an element-wise product whose result is added to the cell state.

- The output is obtained by filtering the cell state C_t. A sigmoid layer first decides which information from the cell state will be output, similarly to the two previous gates, while the cell state is passed through a tanh function, in order to normalize the values between -1 and 1. Finally, they are both combined via an element-wise product to give the cell output.

The operation of a basic LSTM cell can thus be summarized mathematically as follows (all employed variables being displayed in Figure C.1):

- Forget gate:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{C.1}$$

- Input gate:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i \tag{C.2}$$

$$\widetilde{C_t} = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{C.3}$$

- Cell state update:

$$C_t = f_t \odot C_{t-1} + i_t \odot \widetilde{C_t} \tag{C.4}$$

- Output gate:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{C.5}$$

$$h_t = o_t \odot \tanh(C_t) \tag{C.6}$$

where $\odot$ is an element-wise product.

Therefore, the main structure hyper-parameters of an LSTM network are the size of the hidden state h_t and the cell state C_t.

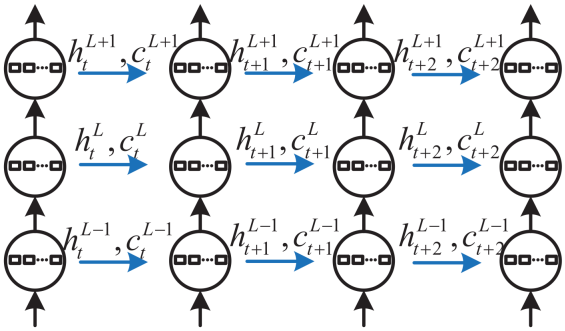


Figure C.2: An unrolled stacked-LSTM network (from Yu et al., 2019)

C.2 Stacked-LSTM Model

To increase the complexity and depth of an LSTM network, several layers can be stacked, as proposed by Graves et al., 2007.

In this case, the recurrent connections in the time dimension are performed only within each layer, and each of the stacked layers serves as input to the deeper layers. Therefore, the output of the $(L-1)^{th}$ layer is denoted as h_t^{L-1} and becomes the input to the next layer L. Such a stacked-LSTM structure is depicted in Figure C.2

Using the formalism shown in Figure C.2, the mathematical relations for the L^{th} layer become:

$$f_t^L = \sigma(W_f^L \cdot [h_{t-1}^L, x_t^{L-1}] + b_f^L) \tag{C.7}$$

$$i_t^L = \sigma(W_i^L \cdot [h_{t-1}^L, x_t^{L-1}] + b_i^L) \tag{C.8}$$

$$\widetilde{C_t^L} = \tanh(W_C^L \cdot [h_{t-1}^L, x_t^{L-1}] + b_C^L) \tag{C.9}$$

$$C_t^L = f_t^L \odot C_{t-1}^L + i_t^L \odot \widetilde{C_t}^L \tag{C.10}$$

$$o_t^L = \sigma(W_o^L \cdot [h_{t-1}^L, x_t^{L-1}] + b_o^L) \tag{C.11}$$

$$h_t^L = o_t^L \odot \tanh(C_t^L) \tag{C.12}$$

C.3 Bidirectional stacked-LSTM Model

Bidirectional LSTM, proposed by Graves et al., 2005 and inspired by bidirectional RNN (Schuster and Paliwal, 1997) consists of training the network in both time directions simultaneously, through the use of two separate hidden layers, namely a forward layer and a backward layer, thus offering a wider use of the time context. These two layers can be reproduced several times to obtain a stacked-bidirectional-LSTM network. This type of structure is widely used as it may significantly improve the predictions, but it is also more costly in terms of computation. Figure C.3 gives an overview of this kind of structure.

Based on the nomenclature of Figure C.3, thje following mathematical expressions can be inferred.

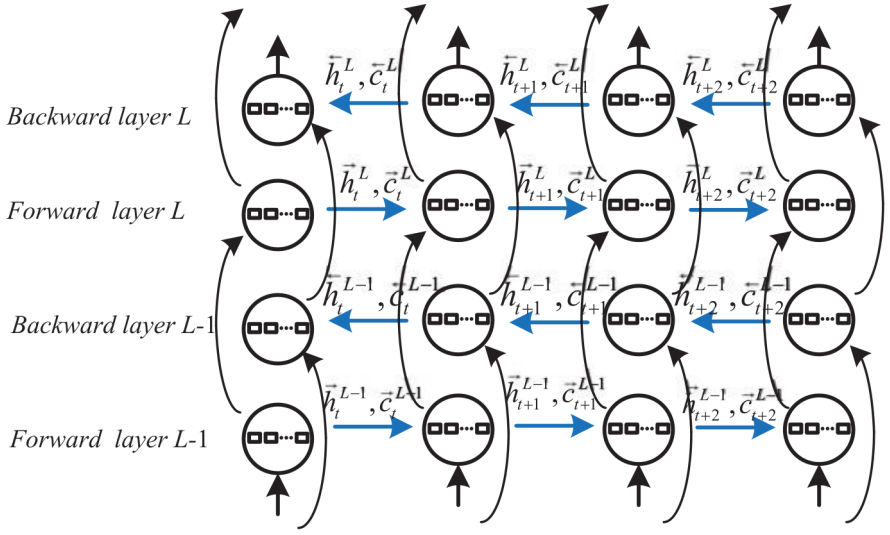


Figure C.3: An unrolled bidirectional LSTM network (from Yu et al., 2019)

For the forward layers, the mathematical relations are the same as in basic stacked-LSTMs, computing the pairs $(\overrightarrow{h_t^L}, \overrightarrow{C_t^L})$ for $t = 1$ to T, as follows:

$$\overrightarrow{f_t^L} = \sigma(W_{\overrightarrow{f}}^L \cdot [h_{t-1}^L, x_t^{L-1}] + b_{\overrightarrow{f}}^L) \tag{C.13}$$

$$\overrightarrow{i_t^L} = \sigma(W_{\overrightarrow{i}}^L \cdot [h_{t-1}^L, x_t^{L-1}] + b_{\overrightarrow{i}}^L) \tag{C.14}$$

$$\overrightarrow{\widetilde{C_t^L}} = \tanh(W_{\overrightarrow{C}}^L \cdot [h_{t-1}^L, x_t^{L-1}] + b_{\overrightarrow{C}}^L) \tag{C.15}$$

$$\overrightarrow{C_t^L} = \overrightarrow{f_t^L} \odot \overrightarrow{C_{t-1}^L} + \overrightarrow{i_t^L} \odot \overrightarrow{\widetilde{C_t^L}} \tag{C.16}$$

$$\overrightarrow{o_t^L} = \sigma(W_{\overrightarrow{o}}^L \cdot [h_{t-1}^L, x_t^{L-1}] + b_{\overrightarrow{o}}^L) \tag{C.17}$$

$$\overrightarrow{h_t^L} = \overrightarrow{o_t^L} \odot \tanh(\overrightarrow{C_t^L}) \tag{C.18}$$

For the backward layers, the relations are obtained for the pairs $(\overleftarrow{h_t^L}, \overleftarrow{C_t^L})$ for $t = T$ to 1, as follows:

$$\overleftarrow{f_t^L} = \sigma(W_{\overleftarrow{f}}^L \cdot [h_{t+1}^L, x_t^{L-1}] + b_{\overleftarrow{f}}^L) \tag{C.19}$$

$$\overleftarrow{i_t^L} = \sigma(W_{\overleftarrow{i}}^L \cdot [h_{t+1}^L, x_t^{L-1}] + b_{\overleftarrow{i}}^L) \tag{C.20}$$

$$\overleftarrow{\widetilde{C_t^L}} = \tanh(W_{\overleftarrow{C}}^L \cdot [h_{t+1}^L, x_t^{L-1}] + b_{\overleftarrow{C}}^L) \tag{C.21}$$

$$\overleftarrow{C_t^L} = \overleftarrow{f_t^L} \odot \overleftarrow{C_{t+1}^L} + \overleftarrow{i_t^L} \odot \overleftarrow{\widetilde{C_t^L}} \tag{C.22}$$

$$\overleftarrow{o_t^L} = \sigma(W_{\overleftarrow{o}}^L \cdot [h_{t+1}^L, x_t^{L-1}] + b_{\overleftarrow{o}}^L) \tag{C.23}$$

$$\overleftarrow{h_t^L} = \overleftarrow{o_t^L} \odot \tanh(\overleftarrow{C_t^L}) \tag{C.24}$$

Finally, the outputs are obtained through:

$$y_t = W_y \cdot [\overrightarrow{ht}, \overleftarrow{ht}] + b_y) \tag{C.25}$$

Appendix D

Kolmogorov-Smirnov One-Sided Test Table

n	0.1	0.05	0.025	0.01	0.005
1	0.9000	0.9500	0.9750	0.9900	0.9950
2	0.6838	0.7764	0.8419	0.9000	0.9293
3	0.5648	0.6360	0.7076	0.7846	0.8290
4	0.4927	0.5652	0.6239	0.6889	0.7342
5	0.4470	0.5094	0.5633	0.6272	0.6685
6	0.4104	0.4680	0.5193	0.5774	0.6166
7	0.3815	0.4361	0.4834	0.5384	0.5758
8	0.3583	0.4096	0.4543	0.5065	0.5418
9	0.3391	0.3875	0.4300	0.4796	0.5133
10	0.3226	0.3687	0.4092	0.4566	0.4889
11	0.3083	0.3524	0.3912	0.4367	0.4677
12	0.2958	0.3382	0.3754	0.4192	0.4490
13	0.2847	0.3255	0.3614	0.4036	0.4325
14	0.2748	0.3142	0.3489	0.3897	0.4176
15	0.2659	0.3040	0.3376	0.3771	0.4042
16	0.2578	0.2947	0.3273	0.3657	0.3920
17	0.2504	0.2863	0.3180	0.3553	0.3809
18	0.2436	0.2785	0.3094	0.3457	0.3706
19	0.2373	0.2714	0.3014	0.3369	0.3612
20	0.2316	0.2647	0.2941	0.3287	0.3524
21	0.2262	0.2586	0.2872	0.3210	0.3443
22	0.2212	0.2528	0.2809	0.3139	0.3367
23	0.2165	0.2475	0.2749	0.3073	0.3295
24	0.2120	0.2424	0.2693	0.3010	0.3229
25	0.2079	0.2377	0.2640	0.2952	0.3166
26	0.2040	0.2332	0.2591	0.2896	0.3106
27	0.2003	0.2290	0.2544	0.2844	0.3050
28	0.1968	0.2250	0.2499	0.2794	0.2997

29	0.1935	0.2212	0.2457	0.2747	0.2947
30	0.1903	0.2176	0.2417	0.2702	0.2899
31	0.1873	0.2141	0.2379	0.2660	0.2853
32	0.1844	0.2108	0.2342	0.2619	0.2809
33	0.1817	0.2077	0.2308	0.2580	0.2768
34	0.1791	0.2047	0.2274	0.2543	0.2728
35	0.1766	0.2018	0.2242	0.2507	0.2690
36	0.1742	0.1991	0.2212	0.2473	0.2653
37	0.1719	0.1965	0.2183	0.2440	0.2618
38	0.1697	0.1939	0.2154	0.2409	0.2584
39	0.1675	0.1915	0.2127	0.2379	0.2552
40	0.1655	0.1891	0.2101	0.2349	0.2521
> 40	$1.07/\sqrt{n}$	$1.22/\sqrt{n}$	$1.36/\sqrt{n}$	$1.52/\sqrt{n}$	$1.63/\sqrt{n}$

Bibliography

An, D., Kim, N. H., & Choi, J.-H. (2015). Practical options for selecting data-driven or physics-based prognostics algorithms with reviews. *Reliability engineering & system safety*, *133*, 223–236.

An, Q., Tao, Z., Xu, X., El Mansori, M., & Chen, M. (2020). A data-driven model for milling tool remaining useful life prediction with convolutional and stacked LSTM network. *Measurement*, *154*, 107461.

Arrhenius, S. (1967). On the reaction velocity of the inversion of cane sugar by acids. In *Selected readings in chemical kinetics* (pp. 31–35). Elsevier.

Atamuradov, V., Medjaher, K., Dersin, P., Lamoureux, B., & Zerhouni, N. (2017). Prognostics and health management for maintenance practitioners-review, implementation and tools evaluation. *International Journal of Prognostics and Health Management*, *8*(3), 1–31.

Babu, G. S., Zhao, P., & Li, X.-L. (2016). Deep convolutional neural network based regression approach for estimation of remaining useful life. *International Conference on Database Systems for Advanced Applications*, 214–228.

Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Bai, S., Kolter, J. Z., & Koltun, V. (2018). An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*.

Balakrishnan, N., & Kateri, M. (2008). On the maximum likelihood estimation of parameters of weibull distribution based on complete and censored data. *Statistics & Probability Letters*, *78*(17), 2971–2975.

Bank, D., Koenigstein, N., & Giryes, R. (2020). Autoencoders. *arXiv preprint arXiv:2003.05991*.

Bengio, Y. (2012). Practical recommendations for gradient-based training of deep architectures. In *Neural networks: Tricks of the trade: Second edition* (pp. 437–478). Springer.

Bengio, Y., Courville, A., & Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(8), 1798–1828.

Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, *5*(2), 157–166.

Berger, V. W., & Zhou, Y. (2014). Kolmogorov–smirnov test: Overview. *Wiley statsref: Statistics reference online*.

Bergmeir, C., Hyndman, R. J., & Benítez, J. M. (2016). Bagging exponential smoothing methods using stl decomposition and box–cox transformation. *International journal of forecasting*, *32*(2), 303–312.

Berndt, D. J., & Clifford, J. (1994). Using dynamic time warping to find patterns in time series. *Proceedings of the 3rd international conference on knowledge discovery and data mining*, 359–370.

Bohlin, T. P. (1991). *Interactive system identification: Prospects and pitfalls*. Springer Berlin, Heidelberg.

Bohlin, T. P. (2006). *Practical grey-box process identification: Theory and applications*. Springer Science & Business Media.

Box, G. E. (1976). Science and statistics. *Journal of the American Statistical Association*, *71*(356), 791–799.

Box, G. E., & MacGregor, J. F. (1976). Parameter estimation with closed-loop operating data. *Technometrics*, *18*(4), 371–380.

Brownlee, J. (2017). *Long short-term memory networks with python: Develop sequence prediction models with deep learning*. Machine Learning Mastery.

Chao, M. A., Kulkarni, C., Goebel, K., & Fink, O. (2019). Hybrid deep fault detection and isolation: Combining deep neural networks and system performance models. *arXiv preprint arXiv:1908.01529*.

Chao, M. A., Kulkarni, C., Goebel, K., & Fink, O. (2022). Fusing physics-based and deep learning models for prognostics. *Reliability Engineering & System Safety*, *217*, 107961.

Chaoub, A., Cerisara, C., Voisin, A., & Iung, B. (2022). Deep learning representation pre-training for industry 4.0. *PHM Society European Conference*, *7*(1), 571–573.

Chelouati, M., Jha, M. S., Galeotta, M., & Theilliol, D. (2021). Remaining useful life prediction for liquid propulsion rocket engine combustion chamber. *2021 5th International Conference on Control and Fault-Tolerant Systems (SysTol)*, 225–230.

Chen, C., Tao, G., Shi, J., Shen, M., & Zhu, Z. H. (2023). A lithium-ion battery degradation prediction model with uncertainty quantification for its predictive maintenance. *IEEE Transactions on Industrial Electronics*.

Chen, Y., Peng, G., Zhu, Z., & Li, S. (2020). A novel deep learning method based on attention mechanism for bearing remaining useful life prediction. *Applied Soft Computing*, *86*, 105919.

Chen, Z., Wu, M., Zhao, R., Guretno, F., Yan, R., & Li, X. (2020). Machine remaining useful life prediction via an attention-based deep learning approach. *IEEE Transactions on Industrial Electronics*, *68*(3), 2521–2531.

Cheng, S., & Pecht, M. (2009). A fusion prognostics method for remaining useful life prediction of electronic products. *2009 IEEE International Conference on Automation Science and Engineering*, 102–107.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Choudhury, M. S., Thornhill, N. F., & Shah, S. L. (2004). A data-driven model for valve stiction. *IFAC Proceedings Volumes*, *37*(1), 245–250.

Cran, G. (1988). Moment estimators for the 3-parameter weibull distribution. *IEEE Transactions on reliability*, *37*(4), 360–363.

Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B., & Bharath, A. A. (2018). Generative adversarial networks: An overview. *IEEE signal processing magazine*, *35*(1), 53–65.

Cubillo, A., Perinpanayagam, S., & Esperon-Miguez, M. (2016). A review of physics-based models in prognostics: Application to gears and bearings of rotating machinery. *Advances in Mechanical Engineering*, *8*(8), 1687814016664660.

da Costa, P. R. d. O., Akcay, A., Zhang, Y., & Kaymak, U. (2019). Attention and long short-term memory network for remaining useful lifetime predictions of turbofan engine degradation. *International Journal of Prognostics and Health Management*, *10*, 034.

Daigle, M., Saha, B., & Goebel, K. (2012). A comparison of filter-based approaches for model-based prognostics. *2012 ieee aerospace conference*, 1–10.

Daigle, M. J., & Goebel, K. (2011). A model-based prognostics approach applied to pneumatic valves. *International Journal of Prognostics and Health Management Volume 2 (color)*, *84*.

Daigle, M. J., & Goebel, K. (2012). Model-based prognostics with concurrent damage progression processes. *IEEE Transactions on Systems, man, and cybernetics: systems*, *43*(3), 535–546.

Du, S., Li, T., Yang, Y., & Horng, S.-J. (2020). Multivariate time series forecasting via attention-based encoder–decoder framework. *Neurocomputing*, *388*, 269–279.

Du, S., Lv, J., & Xi, L. (2012). Degradation process prediction for rotational machinery based on hybrid intelligent model. *Robotics and Computer-Integrated Manufacturing*, *28*(2), 190–207.

Dumoulin, V., & Visin, F. (2016). A guide to convolution arithmetic for deep learning. *arXiv preprint arXiv:1603.07285*.

Dwarampudi, M., & Reddy, N. (2019). Effects of padding on LSTMs and CNNs. *arXiv preprint arXiv:1903.07288*.

Ellefsen, A. L., Æsøy, V., Ushakov, S., & Zhang, H. (2019). A comprehensive survey of prognostics and health management based on deep learning for autonomous ships. *IEEE Transactions on Reliability*, *68*(2), 720–740.

Eltoft, T. (2002). Data augmentation using a combination of independent component analysis and non-linear time-series prediction. *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No. 02CH37290)*, *1*, 448–453.

Eykhoff, P. (1984). Identification theory: Practical implications and limitations. *Measurement*, *2*(2), 75–85.

Fallahi, F., Bakir, I., Yildirim, M., & Ye, Z. (2022). A chance-constrained optimization framework for wind farms to manage fleet-level availability in condition based maintenance and operations. *Renewable and Sustainable Energy Reviews*, *168*, 112789.

Fields, T., Hsieh, G., & Chenou, J. (2019). Mitigating drift in time series data with noise augmentation. *2019 International Conference on Computational Science and Computational Intelligence (CSCI)*, 227–230.

Fink, O., Wang, Q., Svensen, M., Dersin, P., Lee, W.-J., & Ducoffe, M. (2020). Potential, challenges and future directions for deep learning in prognostics and health management applications. *Engineering Applications of Artificial Intelligence*, *92*, 103678.

Forssell, U., & Ljung, L. (1999). Closed-loop identification revisited. *Automatica*, *35*(7), 1215–1241.

Forsyth, D. A., Mundy, J. L., di Gesú, V., Cipolla, R., LeCun, Y., Haffner, P., Bottou, L., & Bengio, Y. (1999). Object recognition with gradient-based learning. *Shape, contour and grouping in computer vision*, 319–345.

Frederick, D. K., DeCastro, J. A., & Litt, J. S. (2007). *User's guide for the commercial modular aero-propulsion system simulation (c-mapss)* (tech. rep.).

Gamier, H., & Mensler, M. (1999). Contsid—a continuous-time system identification toolbox for matlab®. *1999 European Control Conference (ECC)*, 3322–3327.

Garnier, H. (2015). Direct continuous-time approaches to system identification. overview and benefits for practical applications. *European Journal of control*, *24*, 50–62.

Garnier, H., Gilson, M., Muller, H., & Chen, F. (2021). A new graphical user interface for the contsid toolbox for matlab. *IFAC-PapersOnLine*, *54*(7), 397–402.

Garnier, H., Tran-Anh, K., & Bhujwalla, Y. (2018). Low-complexity process model identification of business jet aircraft bleed air system feedback control loops from routine operation data - confidential report.

Garnier, H., Wang, L., & Young, P. C. (2008). Direct identification of continuous-time models from sampled data: Issues, basic solutions and relevance. In *Identification of continuous-time models from sampled data* (pp. 1–29). Springer.

Gawlikowski, J., Tassi, C. R. N., Ali, M., Lee, J., Humt, M., Feng, J., Kruspe, A., Triebel, R., Jung, P., Roscher, R., et al. (2023). A survey of uncertainty in deep neural networks. *Artificial Intelligence Review*, 1–77.

Gay, A. (2023). *Pronostic de défaillance basé sur les données pour la prise de décision en maintenance: Exploitation du principe d'augmentation de données avec intégration de connaissances à priori pour faire face aux problématiques du small data set* (Doctoral dissertation). Université de Lorraine.

Gay, A., Voisin, A., Iung, B., Do, P., Bonidal, R., & Khelassi, A. (2022). Data augmentation-based prognostics for predictive maintenance of industrial system. *CIRP Annals*, *71*(1), 409–412.

Gensler, A., Henze, J., Sick, B., & Raabe, N. (2016). Deep learning for solar power forecasting—an approach using autoencoder and LSTM neural networks. *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 002858–002865.

Gers, F. A., Schmidhuber, J., & Cummins, F. (1999). Learning to forget: Continual prediction with LSTM. *Neural Computation*, *12*, 2451–2471.

Gilson, M., & Van den Hof, P. (2005). Instrumental variable methods for closed-loop system identification. *Automatica*, *41*(2), 241–249.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT press.

Goodwin, G. C., Graebe, S. F., & Salgado, M. E. (2001). *Control system design*. Prentice-Hall.

Gouriveau, R., Medjaher, K., & Zerhouni, N. (2017). *Du concept de phm à la maintenance prédictive 1: Surveillance et pronostic* (Vol. 3). ISTE Group.

Graves, A., Fernández, S., & Schmidhuber, J. (2005). Bidirectional lstm networks for improved phoneme classification and recognition. *International conference on artificial neural networks*, 799–804.

Graves, A., Fernández, S., & Schmidhuber, J. (2007). Multi-dimensional recurrent neural networks. *International conference on artificial neural networks*, 549–558.

Guo, J., Li, Z., & Li, M. (2019). A review on prognostics methods for engineering systems. *IEEE Transactions on Reliability*, *69*(3), 1110–1129.

Guo, R., Huang, T., Li, M., Zhang, H., & Eldar, Y. C. (2023). Physics-embedded machine learning for electromagnetic data imaging: Examining three types of data-driven imaging methods. *IEEE Signal Processing Magazine*, *40*(2), 18–31.

Hallinan Jr, A. J. (1993). A review of the weibull distribution. *Journal of Quality Technology*, *25*(2), 85–93.

Hanachi, H., Yu, W., Kim, I. Y., Liu, J., & Mechefske, C. K. (2019). Hybrid data-driven physics-based model fusion framework for tool wear prediction. *The International Journal of Advanced Manufacturing Technology*, *101*, 2861–2872.

Hazan, A., Verleysen, M., Cottrell, M., & Lacaille, J. (2010). Trajectory clustering for vibration detection in aircraft engines. *Advances in Data Mining. Applications and Theoretical Aspects: 10th Industrial Conference, ICDM 2010, Berlin, Germany, July 12-14, 2010. Proceedings 10*, 362–375.

He, Q., & Wang, J. (2010). Valve stiction modeling: First-principles vs data-drive approaches. *Proceedings of the 2010 American control conference*, 3777–3782.

Heckert, N. A., Filliben, J. J., Croarkin, C. M., Hembree, B., Guthrie, W. F., Tobias, P., & Prinz, J. (2002). *Handbook 151: Nist/sematech e-handbook of statistical methods*. N. Alan Heckert, James J. Filliben, C M. Croarkin, B Hembree, William F ...

Hervé de Beaulieu, M., Jha, M. S., Garnier, H., & Cerbah, F. (2021). Data-driven Health Index prognostics using Deep Learning (Poster). *Annual PhD students conference CRAN*.

Hervé de Beaulieu, M., Jha, M. S., Garnier, H., & Cerbah, F. (2022a). Data-driven system identification and unsupervised health index prognostics using deep learning. application to predictive maintenance of business aircraft (poster). *HCERES Day CRAN (Haut Conseil de l'Evaluation de la Recherche et de l'Enseignement Supérieur)*.

Hervé de Beaulieu, M., Jha, M. S., Garnier, H., & Cerbah, F. (2022b). Long range health index estimation based unsupervised RUL prediction using encoder-decoders. *11th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes*.

Hervé de Beaulieu, M., Jha, M. S., Garnier, H., & Cerbah, F. (2022c). Unsupervised prognostics based on deep virtual health index prediction. *7th PHM Society European Conference*, 193–199.

Hervé de Beaulieu, M., Jha, M. S., Garnier, H., & Cerbah, F. (2023). End-to-end remaining useful life prediction using physics-informed data augmentation. *Currently in submission process to Reliability Engineering & System Safety*.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780.

Horch, A. (2000). *Condition monitoring of control loops* (Doctoral dissertation). Signaler, sensorer och system.

Hu, C., Youn, B. D., Wang, P., & Yoon, J. T. (2012). Ensemble of data-driven prognostic algorithms for robust prediction of remaining useful life. *Reliability Engineering & System Safety*, *103*, 120–135.

Hu, Y., Palmé, T., & Fink, O. (2016). Deep health indicator extraction: A method based on auto-encoders and extreme learning machines. *PHM 2016, Denver, USA*, 446–452.

Huang, B., & Kadali, R. (2008). *Dynamic modeling, predictive control and performance monitoring: A data-driven subspace approach*. Springer.

Huang, N. E., Shen, Z., Long, S. R., Wu, M. C., Shih, H. H., Zheng, Q., Yen, N.-C., Tung, C. C., & Liu, H. H. (1998). The empirical mode decomposition and the hilbert spectrum for nonlinear and non-stationary time series analysis. *Proceedings of the Royal Society of London. Series A: mathematical, physical and engineering sciences*, *454*(1971), 903–995.

Isaksson, A. J. (2013). Some aspects of industrial system identification. *IFAC Proceedings Volumes*, *46*(32), 153–159.

Iung, B., & Levrat, E. (2014). Advanced maintenance services for promoting sustainability. *Procedía CIrP*, *22*, 15–22.

Iwana, B. K., & Uchida, S. (2021a). An empirical survey of data augmentation for time series classification with neural networks. *Plos one*, *16*(7), e0254841.

Iwana, B. K., & Uchida, S. (2021b). Time series data augmentation for neural networks by time warping with a discriminative teacher. *2020 25th International Conference on Pattern Recognition (ICPR)*, 3558–3565.

Jardine, A. K., Lin, D., & Banjevic, D. (2006). A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical Systems and Signal Processing*, *20*(7), 1483–1510.

Javed, K., Gouriveau, R., & Zerhouni, N. (2017). State of the art and taxonomy of prognostics approaches, trends of prognostics applications and open issues towards maturity at dif-

ferent technology readiness levels. *Mechanical Systems and Signal Processing*, *94*, 214–236.

Jha, M. S., Bressel, M., Ould-Bouamama, B., & Dauphin-Tanguy, G. (2016). Particle filter based hybrid prognostics of proton exchange membrane fuel cell in bond graph framework. *Computers & Chemical Engineering*, *95*, 216–230.

Jha, M. S., Dauphin-Tanguy, G., & Ould-Bouamama, B. (2016). Particle filter based hybrid prognostics for health monitoring of uncertain systems in bond graph framework. *Mechanical Systems and Signal Processing*, *75*, 301–329.

Jouin, M., Gouriveau, R., Hissel, D., Péra, M.-C., & Zerhouni, N. (2013). Prognostics and health management of pemfc–state of the art and remaining challenges. *International Journal of Hydrogen Energy*, *38*(35), 15307–15317.

Kalgren, P., Byington, C., Roemer, M., & Watson, M. (2006). Defining phm, a lexical evolution of maintenance and logistics. *2006 IEEE autotestcon*, 353–358.

Kanso, S., Jha, M. S., Galeotta, M., & Theilliol, D. (2022). Remaining useful life prediction with uncertainty quantification of liquid propulsion rocket engine combustion chamber. *IFAC-PapersOnLine*, *55*(6), 96–101.

Khorasgani, H., Kulkarni, C., Biswas, G., Goebel, K., et al. (2013). Degradation modeling and remaining useful life prediction of electrolytic capacitors under thermal overstress condition using particle filters. *Annual Conference of the PHM Society*, *5*(1).

Kumar, S., Torres, M., Chan, Y., & Pecht, M. (2008). A hybrid prognostics methodology for electronic products. *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*, 3479–3485.

LeCun, Y., Bengio, Y., et al. (1995). Convolutional networks for images, speech, and time series. *The Handbook of Brain Theory and Neural Networks*, *3361*(10), 1995.

Lee, J., Wu, F., Zhao, W., Ghaffari, M., Liao, L., & Siegel, D. (2014). Prognostics and health management design for rotary machinery systems—reviews, methodology and applications. *Mechanical systems and signal processing*, *42*(1-2), 314–334.

Lei, Y. (2016). *Intelligent fault diagnosis and remaining useful life prediction of rotating machinery*. Butterworth-Heinemann.

Lei, Y., Li, N., Guo, L., Li, N., Yan, T., & Lin, J. (2018). Machinery health prognostics: A systematic review from data acquisition to RUL prediction. *Mechanical systems and signal processing*, *104*, 799–834.

Li, H., & Wang, Y. (2013). Rolling bearing reliability estimation based on logistic regression model. *2013 International conference on quality, reliability, risk, maintenance, and safety engineering (QR2MSE)*, 1730–1733.

Li, N., Lei, Y., Lin, J., & Ding, S. X. (2015). An improved exponential model for predicting remaining useful life of rolling element bearings. *IEEE Transactions on Industrial Electronics*, *62*(12), 7762–7773.

Li, X., Ding, Q., & Sun, J.-Q. (2018). Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliability Engineering & System Safety*, *172*, 1–11.

Liao, L., & Köttig, F. (2014). Review of hybrid prognostics approaches for remaining useful life prediction of engineered systems, and an application to battery life prediction. *IEEE Transactions on Reliability*, *63*(1), 191–207.

Liu, C., Zhang, L., & Wu, C. (2019). Direct remaining useful life prediction for rolling bearing using temporal convolutional networks. *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2965–2971.

Liu, J., Wang, W., Ma, F., Yang, Y., & Yang, C. (2012). A data-model-fusion prognostic framework for dynamic system state forecasting. *Engineering Applications of Artificial Intelligence*, *25*(4), 814–823.

Liu, K., Shang, Y., Ouyang, Q., & Widanage, W. D. (2020). A data-driven approach with uncertainty quantification for predicting future capacities and remaining useful life of lithium-ion battery. *IEEE Transactions on Industrial Electronics*, *68*(4), 3170–3180.

Liu, Y., Zhao, G., Peng, X., & Hu, C. (2017). Lithium-ion battery remaining useful life prediction with long short-term memory recurrent neural network. *Annual Conference of the PHM Society*, *9*(1).

Ljung, L. (1999). *System identification. theory for the user* (2nd). Prentice Hall.

Ljung, L., & Glad, T. (2016). *Modeling and identification of dynamic systems*. Studentlitteratur.

Lu, Y., Rajora, M., Zou, P., & Liang, S. Y. (2017). Physics-embedded machine learning: Case study with electrochemical micro-machining. *Machines*, *5*(1), 4.

Ma, Z., Liao, H., Gao, J., Nie, S., & Geng, Y. (2023). Physics-informed machine learning for degradation modeling of an electro-hydrostatic actuator system. *Reliability Engineering & System Safety*, *229*, 108898.

MacGregor, J., & Fogal, D. (1995). Closed-loop identification: The role of the noise model and prefilters. *Journal of process control*, *5*(3), 163–171.

Malhotra, P., TV, V., Ramakrishnan, A., Anand, G., Vig, L., Agarwal, P., & Shroff, G. (2016). Multi-sensor prognostics using an unsupervised health index based on LSTM encoder-decoder. *arXiv preprint arXiv:1608.06154*.

Massey, F. J. (1951). The kolmogorov-smirnov test for goodness of fit. *Journal of the American statistical Association*, *46*(253), 68–78.

Mazaev, G., Crevecoeur, G., & Van Hoecke, S. (2021). Bayesian convolutional neural networks for remaining useful life prognostics of solenoid valves with uncertainty estimations. *IEEE Transactions on Industrial Informatics*, *17*(12), 8418–8428.

McCool, J. I. (2012). *Using the weibull distribution: Reliability, modeling, and inference* (Vol. 950). John Wiley & Sons.

Medjaher, K., & Zerhouni, N. (2013a). Framework for a hybrid prognostics. *Chemical Engineering Transactions*, *33*, 91–96.

Medjaher, K., & Zerhouni, N. (2013b). Hybrid prognostic method applied to mechatronic systems. *The International Journal of Advanced Manufacturing Technology*, *69*, 823–834.

Meng, H., & Li, Y.-F. (2019). A review on prognostics and health management (phm) methods of lithium-ion batteries. *Renewable and Sustainable Energy Reviews*, *116*, 109405.

Mobley, R. K. (2002). *An introduction to predictive maintenance*. Butterworth-Heinemann.

Müller, M. (2007). *Information retrieval for music and motion* (Vol. 2). Springer.

Najjar, Y., & AbuEisheh, H. (2016). Exergy analysis and greening performance carpets for turbojet engines. *Journal of Engineering Thermophysics*, *25*(2), 262–274.

Nascimento, R. G., Corbetta, M., Kulkarni, C. S., & Viana, F. A. (2021). Hybrid physics-informed neural networks for lithium-ion battery modeling and prognosis. *Journal of Power Sources*, *513*, 230526.

Nelles, O. (2001). *Nonlinear dynamic system identification*. Springer.

NF EN 13306. (2018). *Maintenance - terminologie de la maintenance*. Association française de normalisation (AFNOR).

Nguyen, K. T., Medjaher, K., & Tran, D. T. (2023). A review of artificial intelligence methods for engineering prognostics and health management with implementation guidelines. *Artificial Intelligence Review*, *56*(4), 3659–3709.

Nielsen, M. A. (2011). *Parameter estimation for the two-parameter weibull distribution*. Brigham Young University.

O'Connor, A. N., Modarres, M., & Mosleh, A. (2011). *Probability distributions used in reliability engineering*. DML International.

Olah, C. (2015). Understanding lstm networks–colah's blog. *Colah. github. io*.

Pan, Y., Jing, Y., Wu, T., & Kong, X. (2022). Knowledge-based data augmentation of small samples for oil condition prediction. *Reliability Engineering & System Safety*, *217*, 108114.

Pecht, M. G. (2010). A prognostics and health management roadmap for information and electronics-rich systems. *IEICE ESS Fundamentals Review*, *3*(4), 4_25–4_32.

Peretzki, D., Isaksson, A., Carvalho Bittencourt, A., & Forsman, K. (2011). *Data mining of historic data for process identification*. Linköping University Electronic Press.

Pintelon, R., & Schoukens, J. (2001). *System identification: A frequency domain approach*. IEEE Press.

Pobočíková, I., & Sedliačková, Z. (2014). Comparison of four methods for estimating the weibull distribution parameters. *Applied mathematical sciences*, *8*(83), 4137–4149.

Pugno, N., Ciavarella, M., Cornetti, P., & Carpinteri, A. (2006). A generalized paris' law for fatigue crack growth. *Journal of the Mechanics and Physics of Solids*, *54*(7), 1333–1349.

Qui, H., & Lee, J. (2004). Feature fusion and degradation using self-organizing map. *2004 International Conference on Machine Learning and Applications, 2004. Proceedings.*, 107–114.

Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, *378*, 686–707.

Ramasso, E., & Saxena, A. (2014). Performance benchmarking and analysis of prognostic methods for cmapss datasets. *International Journal of Prognostics and Health Management*, *5*(2), 1–15.

Rashid, K. M., & Louis, J. (2019). Window-warping: A time series data augmentation of imu data for construction equipment activity identification. *ISARC. Proceedings of the international symposium on automation and robotics in construction*, *36*, 651–657.

Rausand, M., & Hoyland, A. (2003). *System reliability theory: Models, statistical methods, and applications, 2nd edition*. Wiley Series in Probability; Statistics.

Rezamand, M., Kordestani, M., Carriveau, R., Ting, D. S.-K., Orchard, M. E., & Saif, M. (2020). Critical wind turbine components prognostics: A comprehensive review. *IEEE Transactions on Instrumentation and Measurement*, *69*(12), 9306–9328.

Rinne, H. (2008). *The weibull distribution: A handbook*. CRC press.

Sakoe, H., & Chiba, S. (1978). Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, *26*(1), 43–49.

Saon, S., Hiyama, T., et al. (2010). Predicting remaining useful life of rotating machinery based artificial neural network. *Computers & Mathematics with Applications*, *60*(4), 1078–1087.

Sawicki, A., & Zieliński, S. K. (2020). Augmentation of segmented motion capture data for improving generalization of deep neural networks. *International Conference On Computer Information Systems And Industrial Management*, 278–290.

Saxena, A., Celaya, J., Saha, B., Saha, S., & Goebel, K. (2010). Metrics for offline evaluation of prognostic performance. *International Journal of Prognostics and health management*, *1*(1), 4–23.

Saxena, A., Goebel, K., Simon, D., & Eklund, N. (2008). Damage propagation modeling for aircraft engine run-to-failure simulation. *International Conference on Prognostics and Health Management*, 1–9.

Schuster, M., & Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, *45*(11), 2673–2681.

Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of big data*, *6*(1), 1–48.

Si, X.-S., Wang, W., Hu, C.-H., & Zhou, D.-H. (2011). Remaining useful life estimation–a review on the statistical data driven approaches. *European Journal of Operational Research*, *213*(1), 1–14.

Sikorska, J. Z., Hodkiewicz, M., & Ma, L. (2011). Prognostic modelling options for remaining useful life estimation by industry. *Mechanical systems and signal processing*, *25*(5), 1803–1836.

Siraskar, R. (2021). Reinforcement learning for control of valves. *Machine Learning with Applications*, *4*, 100030.

Sjöberg, J., Zhang, Q., Ljung, L., Benveniste, A., Delyon, B., Glorennec, P.-Y., Hjalmarsson, H., & Juditsky, A. (1995). Nonlinear black-box modeling in system identification: A unified overview. *Automatica*, *31*(12), 1691–1724.

Sobie, C., Freitas, C., & Nicolai, M. (2018). Simulation-driven machine learning: Bearing fault classification. *Mechanical Systems and Signal Processing*, *99*, 403–419.

Söderström, T., Ljung, L., & Gustavsson, I. (1978). A theoretical analysis of recursive identification methods. *Automatica*, *14*(3), 231–244.

Söderström, T., & Stoica, P. (1989). *System identification*. Prentice Hall.

Sohlberg, B., & Jacobsen, E. W. (2008). Grey box modelling–branches and experiences. *IFAC Proceedings Volumes*, *41*(2), 11415–11420.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, *15*(1), 1929–1958.

Stephens, M. A. (1992). Introduction to kolmogorov (1933) on the empirical determination of a distribution. In *Breakthroughs in statistics: Methodology and distribution* (pp. 93–105). Springer.

Sun, R., & Giles, C. L. (2001). Sequence learning: From recognition and prediction to sequential decision making. *IEEE Intelligent Systems*, *16*(4), 67–70.

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. *arXiv preprint arXiv:1409.3215*.

Teimouri, M., Hoseini, S. M., & Nadarajah, S. (2013). Comparison of estimation methods for the weibull distribution. *Statistics*, *47*(1), 93–109.

Tobon-Mejia, D. A., Medjaher, K., Zerhouni, N., & Tripot, G. (2012). A data-driven failure prognostics method based on mixture of gaussians hidden markov models. *IEEE Transactions on reliability*, *61*(2), 491–503.

Um, T. T., Pfister, F. M., Pichler, D., Endo, S., Lang, M., Hirche, S., Fietzek, U., & Kulić, D. (2017). Data augmentation of wearable sensor data for parkinson's disease monitoring using convolutional neural networks. *Proceedings of the 19th ACM international conference on multimodal interaction*, 216–220.

Vachtsevanos, G. (2006). *Intelligent fault diagnosis and prognosis for engineering systems* (Vol. 456). Wiley Online Library.

Van den Hof, P. (1998). Closed-loop issues in system identification. *Annual reviews in control*, *22*, 173–186.

Von Rueden, L., Mayer, S., Beckh, K., Georgiev, B., Giesselbach, S., Heese, R., Kirsch, B., Pfrommer, J., Pick, A., Ramamurthy, R., et al. (2021). Informed machine learning–a taxonomy and survey of integrating prior knowledge into learning systems. *IEEE Transactions on Knowledge and Data Engineering*, *35*(1), 614–633.

von Hahn, T., & Mechefske, C. K. (2022). Knowledge informed machine learning using a weibull-based loss function. *arXiv preprint arXiv:2201.01769*.

Wang, B., Lei, Y., Li, N., & Li, N. (2018). A hybrid prognostics approach for estimating remaining useful life of rolling element bearings. *IEEE Transactions on Reliability*, *69*(1), 401–412.

Wang, J., Wen, G., Yang, S., & Liu, Y. (2018). Remaining useful life estimation in prognostics using deep bidirectional LSTM neural network. *2018 Prognostics and System Health Management Conference*, 1037–1042.

Wang, K., Gou, C., Duan, Y., Lin, Y., Zheng, X., & Wang, F.-Y. (2017). Generative adversarial networks: Introduction and outlook. *IEEE/CAA Journal of Automatica Sinica*, *4*(4), 588–598.

Wang, L., Zhang, L., & Wang, X.-z. (2015). Reliability estimation and remaining useful lifetime prediction for bearing based on proportional hazard model. *Journal of Central South University*, *22*(12), 4625–4633.

Wang, P., Long, Z., & Wang, G. (2020). A hybrid prognostics approach for estimating remaining useful life of wind turbine bearings. *Energy Reports*, *6*, 173–182.

Wang, T., Yu, J., Siegel, D., & Lee, J. (2008). A similarity-based prognostics approach for remaining useful life estimation of engineered systems. *International Conference on Prognostics and Health Management*, 1–6.

Wen, P., Ye, Z.-S., Li, Y., Chen, S., & Zhao, S. (2023). Fusing models for prognostics and health management of lithium-ion batteries based on physics-informed neural networks. *arXiv preprint arXiv:2301.00776*.

Wenqiang, J., Jian, C., & Yi, C. (2019). Remaining useful life prediction for mechanical equipment based on temporal convolutional network. *2019 14th IEEE International Conference on Electronic Measurement & Instruments (ICEMI)*, 1192–1199.

Wu, S.-j., Gebraeel, N., Lawley, M. A., & Yih, Y. (2007). A neural network integrated decision support system for condition-based optimal predictive maintenance policy. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, *37*(2), 226–236.

Wu, Y., Yuan, M., Dong, S., Lin, L., & Liu, Y. (2018). Remaining useful life estimation of engineered systems using vanilla LSTM neural networks. *Neurocomputing*, *275*, 167–179.

Xiang, S., Qin, Y., Zhu, C., Wang, Y., & Chen, H. (2020). LSTM networks based on attention ordered neurons for gear remaining life prediction. *ISA Transactions*, *106*, 343–354.

Xing, X., Wang, J., Yang, Z., & Gyasi, P. (2020). Identification of multiple first-order continuous-time dynamic models from special segments in historical data. *IEEE Access*, *8*, 44879–44888.

Xiong, J., Fink, O., Zhou, J., & Ma, Y. (2023). Controlled physics-informed data generation for deep learning-based remaining useful life prediction under unseen operation conditions. *Mechanical Systems and Signal Processing*, *197*, 110359.

Xu, Y., Kohtz, S., Boakye, J., Gardoni, P., & Wang, P. (2022). Physics-informed machine learning for reliability and systems safety applications: State of the art and challenges. *Reliability Engineering & System Safety*, 108900.

Yeomans, J., Thwaites, S., Robertson, W. S., Booth, D., Ng, B., & Thewlis, D. (2019). Simulating time-series data for improved deep neural network performance. *IEEE Access*, *7*, 131248–131255.

Yoon, J., Jarrett, D., & Van der Schaar, M. (2019). Time-series generative adversarial networks. *Advances in neural information processing systems*, *32*.

Young, P. C. (2011). *Recursive estimation and time-series analysis. an introduction for the student and practitioner*. Springer-Verlag.

Young, P. C. (2008). The refined instrumental variable method. *Journal Européen des Systemes Automatisés*, *42*(2-3), 149–179.

Yu, Y., Si, X., Hu, C., & Zhang, J. (2019). A review of recurrent neural networks: LSTM cells and network architectures. *Neural computation*, *31*(7), 1235–1270.

Yuan, M., Wu, Y., & Lin, L. (2016). Fault diagnosis and remaining useful life estimation of aero engine using LSTM neural network. *2016 IEEE International Conference on Aircraft Utility Systems (AUS)*, 135–140.

Yucesan, Y. A., & Viana, F. A. (2022). A hybrid physics-informed neural network for main bearing fatigue prognosis under grease quality variation. *Mechanical Systems and Signal Processing*, *171*, 108875.

Zhang, A., Lipton, Z. C., Li, M., & Smola, A. J. (2021). Dive into deep learning. *arXiv preprint arXiv:2106.11342*.

Zhang, H., Zhang, Q., Shao, S., Niu, T., & Yang, X. (2020). Attention-based LSTM network for rotatory machine remaining useful life prediction. *IEEE Access*, *8*, 132188–132199.

Zhang, Z., Wang, Y., & Wang, K. (2013). Fault diagnosis and prognosis using wavelet packet decomposition, fourier transform and artificial neural network. *Journal of Intelligent Manufacturing*, *24*, 1213–1227.

Zheng, S., Ristovski, K., Farahat, A., & Gupta, C. (2017). Long short-term memory network for remaining useful life estimation. *2017 IEEE International Conference on Prognostics and Health Management (ICPHM)*, 88–95.

Zhou, D., Li, Z., Zhu, J., Zhang, H., & Hou, L. (2020). State of health monitoring and remaining useful life prediction of lithium-ion batteries based on temporal convolutional network. *IEEE Access*, *8*, 53307–53320.

Zhou, G.-B., Wu, J., Zhang, C.-L., & Zhou, Z.-H. (2016). Minimal gated unit for recurrent neural networks. *International Journal of Automation and Computing*, *13*(3), 226–234.

Résumé

Le prognostic d'état de santé est un enjeu majeur dans le domaine de la maintenance prévisionnelle, et a été l'objet de nombreuses études ces dernières années, cherchant notamment à utiliser l'Intelligence Artificielle (IA) pour améliorer les performances de prédiction. Néanmoins, peu d'approches réalistes, prenant en compte les contraintes industrielles réelles, et notamment le manque de données sous dégradation, ont été proposées à ce jour.

L'objectif des travaux de cette thèse est de proposer une approche de prognostic à base d'IA la plus réaliste possible, en prenant notamment en compte la problématique de l'absence de données de dégradation, et en utilisant les connaissances *a priori* disponibles. Une stratégie globale de prognostic en l'absence de données mesurées sous dégradation est proposée. Elle se divise en trois grandes étapes. Tout d'abord, une phase d'augmentation de données hybride basée sur l'utilisation de l'identification de systèmes couplée à l'injection d'un modèle de dégradation physique permet de générer à la fois des données nominales et des données sous dégradation. Ensuite, une méthode d'extraction non-supervisée de l'indicateur d'état de santé, utilisant l'erreur de reconstruction d'un autoencodeur, permet d'obtenir un indicateur d'état de santé à partir des données de capteur mesurées sur le système. Enfin, un processus de prédiction à long terme de l'indicateur d'état de santé permet d'obtenir une prédiction de la durée de vie résiduelle. Certaines étapes sont d'abord validées sur un jeu de données académique (C-MAPSS), puis la méthode globale est appliquée à un cas industriel réel grâce au partenariat avec l'entreprise Dassault Aviation.

Les travaux réalisés mettent en évidence la nécessité de proposer des approches réalistes d'un point de vue industriel, tenant compte des contraintes pratiques, et les résultats obtenus constituent une validation de l'apport des méthodes hybrides (modèles IA intégrant les connaissances disponibles *a priori*) pour évoluer vers un concept de maintenance prévisionnelle applicable en milieu industriel.

Mots-clés: Prognostic de défaillance, Apprentissage profond, Identification de système, Indicateur d'état de santé, Durée de vie résiduelle, Aéronautique.

Abstract

State of Health (SOH) prognostics is a major challenge in the predictive maintenance domain, and has been the subject of numerous studies in recent years, with particular emphasis on the use of Artificial Intelligence (AI) to improve prediction performance. However, few realistic approaches have been proposed so far that take into account the real industrial constraints, and in particular the lack of data under degradation.

The aim of this PhD work is to propose an AI-based prognostics approach as realistic as possible, addressing in particular the problem of the absence of degradation data, and leveraging the available *a priori* knowledge. A global prognostics approach in the absence of measured degradation data is proposed. It is divided into three main stages. First of all, a hybrid data augmentation phase based on system identification coupled with the injection of a physics-based degradation model is used to generate both nominal data and degradation data. Next, an unsupervised Health Index (HI) extraction method, using the reconstruction error of an autoencoder, is used to obtain a HI from the sensor data collected on the system. Finally, a long-term HI prediction process leads to Remaining Useful Life (RUL) predictions. Some stages are first validated on an academic dataset (C-MAPSS), then the overall method is applied to a real industrial case thanks to a partnership with Dassault Aviation.

The research conducted highlights the need for approaches that are realistic from an industrial point of view, taking account of real-life constraints, and the results obtained open up new opportunities for the practical use of AI in predictive maintenance.

Keywords: Prognostics, Deep Learning (DL), System Identification, Health Index (HI), Remaining Useful Life (RUL), Aeronautics.