ABSTRACT

Title of Thesis: A FRAMEWORK FOR REMAINING USEFUL LIFE PREDICTION AND OPTIMIZATION FOR COMPLEX ENGINEERING SYSTEMS

Matthew J. Weiner, Master of Science, 2024

Thesis directed by: Professor Shapour Azarm
Center for Risk and Reliability, Mechanical Engineering

Associate Professor Katrina M. Groth
Center for Risk and Reliability, Mechanical Engineering

Remaining useful life (RUL) prediction plays a crucial role in maintaining the operational efficiency, reliability, and performance of complex engineering systems. Recent efforts have primarily focused on individual components or subsystems, neglecting the intricate relationships between components and their impact on system-level RUL (SRUL). The existing gap in predictive methodologies has prompted the need for an integrated approach to address the complex nature of these systems, while optimizing the performance with respect to these predictive indicators.

This thesis introduces a novel methodology for predicting and optimizing SRUL, and demonstrates how the predicted SRUL can be used to optimize system operation. The approach incorporates various types of data, including condition monitoring sensor data and component reliability data. The methodology leverages probabilistic deep learning (PDL) techniques to predict component RUL distributions based on sensor data and component reliability data when sensor data is not available. Furthermore, an equation node-based Bayesian network (BN) is employed to capture the complex causal relationships between components and predict the SRUL. Finally, the

system operation is optimized using a multi-objective genetic algorithm (MOGA), where SRUL is treated as a constraint and also as an objective function, and the other objective relates to mission completion time. The validation process includes a thorough examination of the component-level methodology using the C-MAPSS data set. The practical application of the proposed methodology in this thesis is through a case study involving an unmanned surface vessel (USV), which incorporates all aspects of the methodology, including system-level validation through qualitative metrics. Evaluation metrics are employed to quantify and qualify both component and system-level results, as well as the results from the optimizer, providing a comprehensive understanding of the proposed approach's performance.

There are several main contributions of this thesis. These include a new deep learning structure for component-level PHM, one that utilizes a hybrid-loss function for a multi-layer long short-term memory (LSTM) regression model to predict RUL with a given confidence interval while also considering the complex interactions among components. Another contribution is the development of a new framework for computing SRUL from these predicted component RULs, in which a Bayesian network is used to perform logic operations and determine the SRUL. These contributions advance the field of PHM, but also provide a practical application in engineering. The ability to accurately predict and manage the RUL of components within a system has profound implications for maintenance scheduling, cost reduction, and overall system reliability. The integration of the proposed method with an optimization algorithm closes the loop, offering a comprehensive solution for offline planning and SRUL prediction and optimization. The results of this research can be used to enhance the efficiency and reliability of engineering systems, leading to more informed decision-making.

A FRAMEWORK FOR REMAINING USEFUL LIFE PREDICTION AND OPTIMIZATION FOR COMPLEX ENGINEERING SYSTEMS

by

Matthew J. Weiner

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Master of Science
2024

Advisory Committee:
Professor Shapour Azarm, Chair/Advisor
Associate Professor Katrina M. Groth, Co-Advisor
Assistant Professor Yunfei Zhao

Foreword

There are three publications used throughout this thesis, all three I, Matthew J. Weiner, made substantial contributions to the relevant aspects of the jointly authored work included in the thesis.

The first reference is [1][1], which is referenced and used in Chapter 3.

The second reference is [2][2]. From this paper, I wrote multiple sections, including the section which is adapted to this thesis in Section 2.1.1. This journal paper has been peer reviewed and is accepted and awaiting publication.

The last reference is [3][3]. This paper has not been accepted for this journal, however, it is currently under review. The citation was created given the paper is accepted. I am the main author for this paper.

[1] Indranil Hazra, Arko Chattejee, Joseph Southgate, Matthew Weiner, Katrina Groth, and Shapour Azarm. "A Reliability-Based Optimization Framework for Planning Operational Profiles for Unmanned Systems". In: *Journal of Mechanical Design* (Oct. 2023), pp. 1–12. ISSN: 1050-0472. DOI: 10.1115/1.4063661.

[2] Indranil Hazra, Matthew J. Weiner, Ruochen Yang, Arko Chatterjee, Jospeh Southgate, Katrina M. Groth, and Shapour Azarm. "Prognostics and Health Management of Unmanned Surface Vessels: Past, Present, and Future". In: *Journal of Computing and Information Science in Engineering* (2024).

[3] Matthew J. Weiner, Ruochen Yang, Katrina M. Groth, and Shapour Azarm. "Probabilistic Deep Learning with Bayesian Networks for Predicting Complex Engineering Systems' Remaining Useful Life". In: *IEEE Transactions on Reliability* (Submitted in 2024).

Acknowledgments

I would like to thank my advisor Dr. Shapour Azarm, and co-advisor Dr. Katrina Groth for their support and guidance during my thesis work. This work would not have been possible without their exceptional work ethic in giving great feedback and motivation in a timely fashion, despite their busy schedules. I will always be grateful for this support as it has spurred success in my professional career. I would also like to thank the graduate program for awarding my the Dean's Fellowship, which has in part supported me financially during my thesis work. I would also like to thank Dr. Yunfei Zhao for being a part of my thesis committee and for providing feedback for improving the thesis.

I would also like to acknowledge the U.S. Department of the Navy, Office of Naval Research, who funded, in part, the work presented herein under Grant No. N000142212459. This support does not constitute an endorsement by the funding agency of the opinions expressed in the thesis.

Finally, I am thankful for my parents who showed continuous love and motivation to me throughout my thesis work, and have given me the ability to be successful as a person in this life.

Table of Contents

List of Tables

List of Figures

List of Abbreviations and Nomenclature

BN Bayesian network

C-MAPSS commercial modular aero-propulsion system simulation

cdf cumulative distribution function

CES complex engineering system

CLA component-level analysis

CWB coverage-width balance

ECCS engine cooling and control system

EOL end of life

FMEA failure modes and effects analysis

GA genetic algorithm

IoT internet of things

LSTM long short-term memory

MOGA multi-objective genetic algorithm

MPC model predictive control

MSE mean square error

MTTF mean-time-to-failure

NLL negative log-likelihood

NMPIW normalized mean prediction interval width

NN neural network

NRMSE normalized root mean square error

NSGA-II non-dominated sorting genetic algorithm II

pdf probability density function

PDL probabilistic deep learning

PHM prognostics and health management

PICP prediction interval coverage percentage

PRA probabilistic risk assessment

QRA quantitative risk assessment

RUL remaining useful life

SF scoring function

SIPPRA systematic integration of PHM and PRA for risk and reliability analysis

SLA system-level analysis

SMS-EMOA S-metric selection evolutionary multi objective algorithm

SNR signal-to-noise-ratio

SRUL system-level remaining useful life

USV unmanned surface vessel

Chapter 1: Introduction

1.1 Motivation

The foundation of the U.S. economy, security, and overall well-being relies on complex engineering system (CES). A CES refers to an intricate network of interdependent components and processes designed to perform specific functions or tasks, often found in industries such as maritime, energy, and aerospace, as seen in Figure 1.1 [4]. These systems exhibit dynamic characteristics, intricate physics, and involve numerous interconnected machine and human elements, operating across multiple levels and scales.

If a failure were to occur in one of these systems, this may directly lead to the failure of the mission or have catastrophic impacts and pose significant risks to the general public's health and safety. Also, in commercial settings, failures can contribute to machine downtime which leads to a loss of revenue for the customer. Accurate and timely prediction of these failures, as well as optimal operation, are essential steps to avoid or mitigate these failures and maximize the CESes' lifetimes.

Specific pitfalls of failures in CESes include system downtime, reactive maintenance cost and timing, and potential loss of a mission window. Unplanned failures can cost organizations significant amounts of money and time. An example of a CES is an unmanned surface vessel (USV), which is equipped with diverse sensors and payloads to be successful in military and civilian missions. However, for a USV, the expanding system and mission complexity elevates the possibility of different failures. These failures are mission critical, because the USV cannot be repaired during its operation [5].

Also, reactive maintenance is generally more expensive than proactive maintenance. This is

Figure 1.1: From left to right, unmanned surface vessel, a hydrogen renewable energy production system, and a nuclear power plant. All considered as complex engineering systems.

due to CESes often having long lead times for replacement parts. Finally, unreliable systems can lead to loss of ability to complete a mission, including loss of a mission window.

To alleviate these prior mentioned pitfalls, the goal of this thesis is to develop a systematic method for determining the system's health state to provide timely and accurate support for decision-making, and optimize the operation and health indicators of the CES. To achieve this, remaining useful life (RUL) will be used as an indicator of system health. Accurate RUL predictions allow for proactive maintenance, minimizing unplanned downtime and associated costs, as well as enhancing customer satisfaction, crucial for maintaining a positive reputation and repeat business.

1.2 Background

This section provides an overview of the problems relating to the reliability and maintainability of complex engineering systems, methods to alleviate these problems, with an overview into the RUL prediction and RUL-based optimization.

1.2.1 Reliability of Complex Engineering Systems

CESes are a crucial part of various critical infrastructures, encompassing physically dispersed hardware, software, and human components that are interlinked [6]. Traditionally, reliability engineers analyzed CESes using methods such as quantitative risk assessment (QRA) and qualitative techniques like failure modes and effects analysis (FMEA) [7]. However, these techniques are used for static risk analysis, not dynamic [8].

The fourth industrial revolution, Industry 4.0, is shifting this focus from expert knowledge to data driven techniques. Industry 4.0 is based upon innovations in data, information, analytics, automation, and emerging technologies, with a primary focus on improving reliability of processes through data [9]. This data-driven focus gives reliability engineers many opportunities to improve previous reliability analysis of CESes [10]. The advancements in low-cost sensors and powerful high-speed processors, open up new possibilities for gaining these data-driven insights; they create opportunities to enhance our understanding of complex system behaviors by leveraging extensive data sets. This new information can contribute to improved safety and reliability assessments, optimized logistics, cost reduction, and more efficient resource utilization. However, there are significant challenges in efficiently generating and using large amounts of data. Hazra et al. [11] tackled these issues by developing a diagnosis model for an unmanned CES, which is a step towards prognostics. To address these challenges in reliability engineering, a sub field of reliability engineering has emerged: prognostics and health management (PHM).

PHM is a field dedicated to predicting and managing the health and performance of components within various systems [12]. It utilizes technologies made common from Industry 4.0, such as sensors and data analytics, to monitor the condition of individual components in near real-time [12]. However, it's crucial to note that while PHM has proven effective at the component level, it often lacks comprehensive system-level coverage, particularly in the context of complex engineering systems [6]. Many current applications focus on isolated components, such as bearings and batteries [13, 14]. Thus, there is a need to extend PHM's reach to provide a more holistic understanding of entire interconnected systems [6]. Section 2.1.1 will provide a more in-depth review of the current state of PHM, its current challenges, and how this thesis attempts to fill the gaps.

The reliability field faces inherent challenges in accurately predicting and managing the health of components and systems. One goal of the field is to mitigate the pitfalls discussed in Section 1, including machine downtime, reactive maintenance, and loss of mission. One notable hurdle is the dynamic nature of operating conditions, making it challenging to develop universally applicable models [6, 15]. In response to these challenges, the concept of predicting RUL emerged as a pivotal solution. RUL estimation involves forecasting the time until a component or system is expected to fail, enabling proactive maintenance and minimizing unexpected downtime [16]. Another significant development in addressing reliability challenges is the optimization of system operation to maximize lifetime. By adjusting operating parameters based on near real-time health assessments, this approach aims to extend the overall system lifespan while ensuring efficient performance.

1.2.2 Remaining Useful Life Prediction Problem Overview

In the field of PHM, the RUL prediction problem revolves around forecasting the remaining operational lifespan of machinery or equipment based on historical performance data [17, 18]. This problem is crucial for optimizing maintenance schedules, reducing downtime, and maximizing the utilization of assets.

The fundamental premise of the RUL prediction problem involves receiving a data set containing the operational history of a system, where each data set is associated with a specific component or with the system as a whole. This data set typically includes operational parameters, sensor readings, and may include annotations denoting failure events. The goal is to develop predictive models that can estimate the remaining time until a particular component or system fails [19].

The problem assumes that the degradation of mechanical components is gradual and can be tracked over time [17]. The predictive models need to consider various factors such as operating conditions, environmental influences, and usage patterns to accurately forecast the RUL. The models should be capable of adapting to different types of machinery and diverse operating contexts [17]. The RUL prediction problem can be further categorized into two main types: component-level RUL prediction and system-level RUL prediction [6].

Component-level RUL Prediction: In this problem, the focus is on predicting the remaining lifespan of individual components within a machine or system. For example, in the aviation industry, predicting the RUL of aircraft engine components, like a turbine, is very common; the commercial modular aero-propulsion system simulation (C-MAPSS) data set is one common example [20].

System-level RUL Prediction: Contrary to the component-level approach, this problem involves predicting the remaining lifespan of the entire system or machine. This is particularly relevant in scenarios where the failure of any single component or combination of components can lead to the overall failure of the entire system. An example is predicting the RUL of the entire aircraft or a USV.

1.2.3 RUL-based Operational Optimization Problem Overview

In the RUL optimization problem, the primary goal is to optimize the utilization of a system or asset by maximizing its RUL while completing the mission as quickly as possible: a bi-objective study. This problem arises in various reliability engineering applications, including maximizing RUL while maintaining mission success, or following a maintenance plan [21]. Initially, information about the current state and health of the system is provided, and the objective is to determine the optimal operational and maintenance strategies to prolong the RUL.

The problem involves managing the operational decisions over time to ensure that the system functions efficiently and effectively for as long as possible. Unlike traditional optimization problems that focus on minimizing costs or meeting specific deadlines, the RUL maximization problem places emphasis on extending the useful lifespan of the system, given the mission must succeed, which is crucial for minimizing downtime, reducing replacement costs, and enhancing overall reliability [22]. Two common variations of the RUL maximization problem can be identified based on the nature of the objectives: single or multi-objective problems.

Single-objective optimization problems: In this scenario, the optimization objective is focused on optimizing a single-objective such as maximizing the RUL of the system. Decision variables may include the scheduling of maintenance tasks, adjusting operational parameters, and implementing preventive measures to achieve the desired outcome.

Multi-objective optimization problems: The multi-objective or specifically the bi-objective version of the problem, as will be presented in this thesis, considers two conflicting objectives simultaneously. These objectives may include RUL, maintenance costs, and system performance. The challenge is to find a set of solutions that represent a trade-off among these conflicting objectives, known as the Pareto front [23].

As an example, a bi-objective optimization problem is written in the following form [24]:

$$
\begin{aligned}
\text{maximize} \quad & f_1(\boldsymbol{x}, \boldsymbol{p}) \\
\text{maximize} \quad & f_2(\boldsymbol{x}, \boldsymbol{p}) \\
\text{subject to} \quad & g_i(\boldsymbol{x}, \boldsymbol{p}) \leq 0, \quad i = 1, \ldots, m \\
& h_j(\boldsymbol{x}, \boldsymbol{p}) = 0, \quad j = 1, \ldots, n
\end{aligned}
\tag{1.1}
$$

where $\boldsymbol{x}$ represents the input vector of design (or decision) variables and refer to those that the decision maker has control over and are assumed to have no uncertainty. The input vector $\boldsymbol{p}$ refers to parameters which are fixed during an optimization run but have uncertainty. f_i represents the objective functions which are maximized. Finally, g_i and h_i are inequality and equality constraints, respectively. For a bi-objective maximization problem, as an example, one objective function can be the RUL, and the other competing objective can be the speed of mission completion.

1.3 Research Questions, Objectives and Tasks

This thesis is aimed to answer the following research questions.

- **Research Question 1:** How can multiple component RULs be combined into a system-level remaining useful life (SRUL) while preserving dependencies and considering uncertainties?
  - *Significance and Contribution*: Current methods focus on RUL prediction for a single component or assume independence among components for multi-component systems. Hence, there is a need for predicting SRUL for an interdependent multi-component system.
- **Research Question 2:** How can the operation of an interdependent multi-component system be optimized to maximize SRUL, and other objectives, subject to constraints?
  - *Significance and Contribution*: The application of an integrated approach with SRUL prediction and optimization can be used to enhance operational efficiency and reliability by enabling proactive maintenance and resource optimization, minimizing downtime.
- **Research Question 3:** Is this methodology possible to run in near real-time?

- *Significance and Contribution*: There is no evidence of this complete framework in the literature. Our initial evaluations show that there is potential for this framework for near real-time application of the proposed framework, particularly for the SRUL prediction, thus, it is the first algorithm to meet this criterion.

To address these research questions and technical gaps in predicting and maximizing system-level RUL, a single research objective was written and a series of tasks that are required to accomplish the objective were defined. The outcome of this thesis is a novel framework for system-level RUL prediction and maximization. In order to build a credible result, the framework is validated using a series of case studies at three levels: component RUL prediction, system RUL prediction, and RUL maximization as part of a bi-objective optimization model. Therefore, three research tasks were created, respectively. The research objective and tasks are listed below. Additionally, the relationship between each of the tasks is shown in Figure 1.2

Research Objective: Establish a causal framework and model which fuses historical failure rates and condition monitoring data to predict and maximize RUL at the component and system levels with uncertainty.

- **Task 1:** Construct a causal model of component-level RUL prediction.
  - *Task 1.1*: Conduct a literature review of RUL prediction methods for single components to identify common models and gaps.
  - *Task 1.2*: Create a framework which can predict RUL of a single component with aleatory and epistemic uncertainties.
  - *Task 1.3*: Generate or obtain data needed to accomplish Task 1.2.
  - *Task 1.4*: Develop the models needed to accomplish Task 1.2
  - *Task 1.5*: Validate the models created in Task 1.4 using a case study from the dataset from Task 1.3.
- **Task 2:** Develop a causal model for integrating component RULs into a single SRUL.
  - *Task 2.1*: Conduct a literature review of system RUL prediction methods to identify common models and gaps.

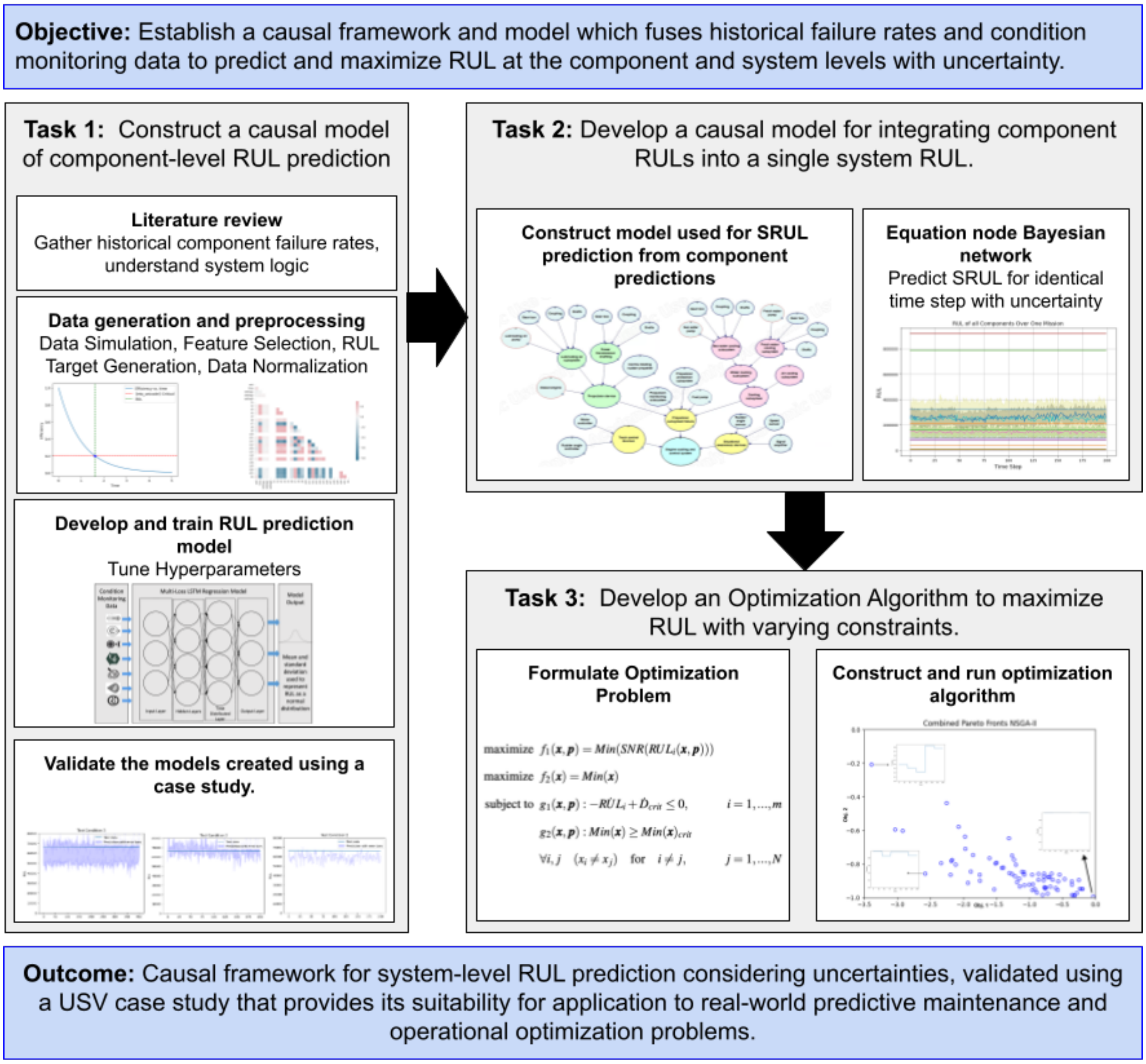


Figure 1.2: An overview of the research for this thesis.

- *Task 2.2*: Create a framework which can integrate component RULs into a single system RUL prediction.
- *Task 2.3*: Develop the models needed to accomplish Task 2.2
- *Task 2.4*: Validate the models created in Task 2.3 using a case study from the dataset from Task 1.3.

- **Task 3:** Develop an optimization algorithm to maximize RUL with varying constraints.
  - *Task 3.1*: Conduct a literature review of RUL maximization techniques for multi and

single objective optimization problems.

- *Task 3.2*: Create a framework which is capable of maximizing RUL and another objective, while satisfying the constraints.
- *Task 3.3*: Develop the models needed to accomplish Task 3.2
- *Task 3.4*: Validate the models created in Task 3.3 using a case study from the data set from Task 1.3.

1.4 Thesis Structure

The remainder of this thesis is structured as follows. Chapter 2 provides background information on RUL and SRUL prediction, and RUL-based optimization methods, details of the conducted literature review, and a brief introduction of Bayesian network (BN), Deep Learning, and Genetic Algorithms. Chapter 3 describes a methodology for how RUL and SRUL can be predicted using historical data, condition monitoring data, a Bayesian network, and probabilistic deep learning (PDL). It then validates the component-level analysis and system-level analysis using two case studies, and finally uses quality metrics to give meaning to the results. Chapter 4 proposes a methodology for RUL-based optimization using a Genetic Algorithm and uses a case study on a USV to verify the method. Lastly, Chapter 5 concludes with a brief summary of the thesis with highlights of the results, key contributions made, and some possible directions for future research related to this thesis.

Chapter 2: Literature Review

Portions of this chapter are reproduced from [2][1] and an additional paper which is still under review.

2.1 Review of Research Fields of Interest

The literature review in this chapter is broken up according to the two main problems which it attempts to review. The first section, Section 2.1, aims to understand how current literature solves problems within the fields concerning this thesis, including system health prognostics for unmanned and complex engineering systems in Section 2.1.1, RUL and SRUL prediction in Sections 2.1.1.2 and 2.1.1.3, and finally the area of RUL-based optimization is reviewed in Section 2.1.2. These areas are reviewed, because this thesis proposes an approach for SRUL prediction and optimization for CESes, which falls under the PHM umbrella.

2.1.1 System Health Prognostics Methods

System health prognosis plays a crucial role in the process of PHM. Prognosis refers to the capability of predicting the future health state or performance degradation of a system or component. The information obtained from system health prognosis can facilitate proactive maintenance and optimize operational decisions, minimize downtime, and maximize system availability. Prognosis is typically achieved through the analysis of historical data, real-time monitoring, and the application of advanced techniques such as machine learning and statistical modeling to forecast the

[1] Indranil Hazra, Matthew J. Weiner, Ruochen Yang, Arko Chatterjee, Jospeh Southgate, Katrina M. Groth, and Shapour Azarm. "Prognostics and Health Management of Unmanned Surface Vessels: Past, Present, and Future". In: *Journal of Computing and Information Science in Engineering* (2024).

RUL or estimate the probability of failure [25]. It is used to predict the degradation and RUL of components, subsystems, or systems. This process comprises several steps, each employing various methodologies to achieve the desired outcomes (see [6] for an overview); these steps include health indication and RUL estimation. These steps build off of the information collected from other system health monitoring steps, such as system health diagnosis or anomaly detection. In the context of unmanned CESes, prognostics allows informed decisions on operations and helps to determine the optimal times to perform maintenance, thereby improving reliability and safety. Through reviewing the related literature, this section discusses the key aspects of prognostics.

The methodology for the literature review in this section is the same of that from [2]. The search resulted in a total of 309 articles. All these papers were read and those relevant to this thesis were included, as well as supplementary sources.

2.1.1.1 System Health Indication

The first step towards the health prognostic is to identify meaningful health indicators that effectively reflect the health state of the studied system. These indicators should serve as quantifiable measures to quantify the rate of degradation of components, subsystems, and systems within the CES [6].

Examples of methods used for identifying the state of these health indicators include unsupervised deep learning [26], kernel-based synthesis of multiple sensor signals [27], Hotelling's and residuals metrics using principal component analysis (PCA) with statistical process control (SPC) [28], and data-level fusion models maximizing signal-to-noise-ratio (SNR) [29]. Additionally, Guo and Utne [30] utilized a Bayesian belief network to develop risk indicators based on various factors. These approaches leverage historical data and offline processing, making them computationally efficient. However, since they operate offline, they can only detect faults that have already occurred. Consequently, real-time monitoring and immediate preventive measures are not possible.

Real-time health indication methods are used to monitor the health of machinery in real-time, allowing for preventive measures to be taken immediately and adapt to change according to operating conditions. There are various methods for determining health indicators, including physics-based methods [31], binary differential evolution algorithms [32], and multiple health indicator-based methods [33]. These methods enable informed decisions on the optimal time to perform maintenance for reliable and safe operations. It is also important to predict when faults will occur to estimate RUL. Different possible classifications of the system, such as healthy, degraded, and failure, require different techniques to estimate RUL [34]. These methods enable real-time monitoring of machinery, but real-time processing may be computationally expensive and may not recognize all possible faults, leading to an inability to capture some forms of degradation. The various operational state indications obtained from these methods will impact the algorithms used for estimating the RUL, which are discussed in the following section.

2.1.1.2 RUL Estimation Techniques

The importance of optimizing operational downtime, system reliability and safety, and operational efficiencies will continue to increase as system complexity and maintenance costs grow. Therefore, the ability to reliably predict the RUL of the said system receives a great deal of attention [35]. There are also environmental impacts that the reliable prediction of RUL for components can create. For example, RUL prediction will aid in the decarbonization of the marine transport industry through an optimized system operation [36]. One of the greatest challenges for RUL prediction of CESes, such as marine or aero equipment is handling the impact of dynamic environmental conditions [37]. The failure modes of equipment may change based on the temperature, salinity, pH, etc. of the water, or humidity of the air. These harsh conditions can cause corrosion of critical structures. The variability of conditions, the complexity of the corrosion mechanism, and the difficulty of condition monitoring make accurate RUL prediction quite difficult [38]. This section summarizes techniques used to estimate the RUL for systems, subsystems, and components related to CESes, as well as the challenges which come with these techniques.

The methods for RUL prediction can be categorized into physics model-based methods, data-driven methods, and hybrid methods. In physics model-based methods, the failure process, i.e., degradation, is described by mathematical models derived from the physical principles. Typical failure models such as those in Paris' law [39] and Forman law [40] have been applied to predict the RUL of various systems due to crack growth. With the help of data assimilation methods, such as Kalman and particle filters, the physics-based methods are proven to provide good RUL estimation results. For example, the methods in [41] use a particle filter-based algorithm for RUL prediction. Similarly, Bazilevs et al. [42] use a structural health monitoring system and an advanced fluid-structure interaction simulation module to perform structural maintenance and repair, and predict remaining fatigue life. Physics-based methods can be very difficult and complex for making accurate predictions. There are many factors that are difficult to capture in equation forms, such as complex interactions or non linear behaviors like creep and crack growth [43]. For these reasons, data-driven approaches are preferred when data are available to capture and predict intricate failure phenomena within CESes.

In contrast to physics-based methods, data-driven techniques exploit historical data collected from the system or its components to predict the RUL using methods that rely on advanced data analysis and machine learning algorithms. By processing and analyzing the historical data, these techniques aim to identify patterns, trends, and correlations that can provide insights into the health and performance of the CES. These data-driven approaches involve many steps. First, the data is pre-processed to ensure its quality, which may involve cleaning, normalization, and feature extraction. Next, the data is used to build or train the model specific to the algorithm. Finally, during the operational phase, the model is tested on new data which it was not trained on, and gives a corresponding prediction.

Several data-driven methods have been developed for RUL prediction of CESes and their components. Initial methods focused on simple models, while subsequent works aimed to address the challenges and improve these approaches. Simple data-driven methods, such as statistical models, have been applied for RUL estimation in complex systems. For instance, Zhai and Ye [44] utilized an adaptive Wiener process model; however, as the complexity of degradation processes increased, the limitations of this simple model became apparent. To overcome these limitations, researchers have employed more advanced techniques such as semi-Markov decision processes [45], and hidden semi-Markov models [46]. These methods consider dynamic operating conditions and the degradation process, leading to more accurate predictions compared to other data-driven methods. Furthermore, the advantage of these approaches lies in the limited number of equations, making them particularly suitable for real-time RUL prediction. In addition, more sophisticated models have been proposed to capture the nonlinear behavior of degradation processes, such as Zhang et al. [47] who introduced a method for modeling stochastic degradation processes. By considering the complexities of degradation, these models enhance the accuracy of predictions. Moreover, to address the uncertainties in the prediction, BahooToroody et al. [48] utilized Bayesian inference and Markov chain Monte Carlo simulation. This approach enables the quantification of uncertainty, providing a more complete understanding of ship reliability and RUL. These methods are relatively simple to implement and have low computational complexity but require large amounts of data for significant results.

The use of machine learning, in particular deep learning, techniques in predicting RUL is effective with its ability to handle complex data and capture intricate relationships between features, which can lead to more accurate predictions compared to traditional statistical and physics-based methods. These methods can consist of supervised or semi-supervised learning. Supervised learning involves using labeled data to train a model to make predictions; several studies have utilized this technique for RUL. Semi-supervised techniques, however, leverage both labeled and unlabeled data to train the model. There are many different types of deep learning methods, however, the most simple in terms of architecture are simple neural networks. Ruiz-Tagle Palazuelos, Droguett, and Pascual [49] and Zhao et al. [50] utilize a neural network (NN) within their architecture for RUL estimation from sensor data. Building off of NNs, Zhang et al. [51] and Mitici et al. [52] use convolutional NNs, and the authors in [53–55] use another type of NN, a long-short term memory (LSTM) architecture, for RUL prediction on marine engines, an integral part of a USV.

Bae and Xi [56] introduce the concept of using LSTM networks to learn the unit's physical health over time instead of relying on the traditional labeled time of a test unit, which enables more accurate RUL estimation, especially when the time steps are inconsistent among training units. However, one issue with these types of deep learning methods is that they neglect epistemic uncertainty, meaning their predictions are a single point value without a confidence interval.

Nguyen, Medjaher, and Gogu [57] propose a novel method for using a probabilistic LSTM NN for component-level RUL prediction and using system logic to go from component level to system level. This is an example of epistemic uncertainty being included. Finally, aleatoric uncertainty must also be accounted for. BahooToroody et al. [58] and Abaei et al. [59], propose hierarchical Bayesian inference-based reliability frameworks for RUL prediction with confidence intervals of conventional ships and machine systems, respectively. The models are robust, as they account for uncertainties in population variability and unknown parameters, which are examples of aleatoric uncertainty. In these cases, training data is used from run-to-failure tests where the time to failure is known, meaning the data is labeled, however, some methods utilize partially or unlabeled learning data. For example, Ellefsen et al. [60] presents a novel semi-supervised deep learning methodology where the model is pre-trained without labels and the results suggest the approach is more effective for changing operating conditions and fault modes than fully supervised learning.

Alternatively, some have explored hybrid methods which combine the strengths of different approaches discussed to overcome the limitations previously discussed of individual methods for RUL prediction. Cai et al. [61] and Nielsen and Sørensen [62] incorporate BNs and Markov-decision processes for RUL prediction. This method uses Markov processes for modeling sequential decision-making and system dynamics and handles the uncertainty by using BNs. In another study, Khorasgani, Biswas, and Sankararaman [63] propose a hybrid approach consisting of a particle filtering algorithm to estimate degradation parameters and an inverse-FORM approach for system-level RUL prediction. Finally, Zhicai, Dongfeng, and Xinfa [64] utilize support vector machine and probability-based methods such as Markov chain to estimate RUL for USV components.

An in-depth summary of these methods is shown in Table 2.1. As shown in the table, a majority of the recent novel methodologies are focusing on RUL prediction through mostly deep learning and statistical methods, which are able to handle the complex relationships within components and sub-systems in a CES. With this extensive amount of literature, there are still several potential areas for growth. The next section discusses these gaps, specially the ones which will be addressed in this thesis.

Table 2.1: Overview of prognosis methods for PHM of CESes.

Prognosis	**Method**	**Reference**	**Significance**
Health Indication	Offline	[26–30]	• Use of readily available historical data • Computationally less expensive as they don't require real-time processing • Limited to detecting faults that have already occurred • No real-time monitoring and immediate preventive measures
Health Indication	Real-time	[31–34]	• Real-time monitoring of machinery health • Immediate preventive measures with adaptability to changing conditions • Real-time processing may be computationally expensive, leading to low-resolution updates • Inability to recognize all possible faults, limiting capture of some forms of degradation
RUL	Physics-based	[41, 42]	• Accurate RUL prediction with precise design and expert experience • Model complexity and time-consuming computations
RUL	Markov processes	[45, 46, 48]	• Simple implementation and interpretation • Capture of dynamic operating conditions and degradation process • Flexibility and ability to quantify uncertainty in RUL prediction • Requirement of a large amount of data and potential difficulty in validation
RUL	Wiener processes	[44, 47]	• Low computational complexity for real-time applications • Requirement of large amounts of data for training • Potential limitation in capturing complex interactions and dependencies in the system
RUL	Semi-supervised	[60]	• More effective for changing operating conditions and fault modes than fully supervised learning
RUL	Neural networks	[49–56]	• High accuracy when trained with large and diverse data • Low complexity allowing for real-time predictions • Neglect of epistemic and aleatoric uncertainties, limiting interpretability
RUL	Probabilistic deep learning	[57–59]	• Same strengths as neural networks • Coverage of epistemic and aleatoric uncertainties, providing confidence bounds for predictions
RUL	Hybrid	[61–64]	• Potential improvement in prediction accuracy by combining multiple approaches • Addressing limitations of individual models (e.g., capturing complex interactions and generalization to new conditions)

2.1.1.3 Gaps Within RUL Prediction Methods

As previously discussed, many papers predict RUL of single components using deep learning, such as [65–67]. However, these methods focus on RUL estimation offline. Although their models have demonstrated utility, they fall short in providing near real-time predictions, limiting their applicability for timely maintenance planning and mission success.

The larger shortcoming of all these component RUL methodologies is that they fail to apply for CESes, which have many components [6]. This limits the ability of PHM to support system health holistically, potentially leading to sub optimal maintenance decisions and unexpected downtime.

To solve this issue, some attempts have been made to define how to predict the RUL at the system level. Moradi and Groth [6] and Moradi et al. [68] proposed integrating PHM and probabilistic risk assessment (PRA), into a single approach: systematic integration of PHM and PRA for risk and reliability analysis (SIPPRA), for risk monitoring of CESes. However, the approach stops at diagnosis, and does not provide health prognosis. Nguyen, Medjaher, and Gogu [57] attempted to address the challenge of predicting the RUL in multi-component systems. They introduced a methodology employing PDL method for component-level RUL predictions and evaluated their approach with various metrics. However, their model did not consider interdependencies between components, a notable limitation. Also, they only tested the component level of their approach, since the C-MAPSS data set used is for a single component. Rodrigues [69] also tackled the problem of system RUL prediction in CESes, relying on system-level performance indicators derived from component-level data. However, this method does not include confidence intervals for its predictions. Mitici et al. [52] presented a maintenance planning scheme for CESes, emphasizing probabilistic system RUL predictions using a convolutional NN. While their approach optimized engine replacement timing and reduced maintenance costs, it still overlooked component interactions, a critical aspect in CESes. Tamssaouet et al. [70] performed further research into the current state-of-the-art of system-level RUL prediction. They identified the three lacking areas as the consideration of the interdependencies of degradation on mission performance, the consideration uncertainty and its propagation from component to system-level, and the calculation of system RUL using more than just series and parallel connections. They also state that digital twins are a promising method for data generation and creation of these system level prognostic models. All these issues and methods are addressed in this thesis.

To summarize, existing methods of system-level RUL prediction still have challenges in providing accurate system RUL for CESes, including neglecting interdependencies between components and relying on limited performance indicators. These make it difficult to provide timely and accurate support for decision-making. This thesis will attempt to create a method which addresses all these issues simultaneously, instead of only one or two.

2.1.2 RUL-Based Optimization

2.1.2.1 Background and Method of Review

At its essence, RUL-based optimization falls into the category of PHM-based optimization, which leverages offline and/or online (real-time) health information to enhance decision-making, improve resource allocation, and optimize system performance by considering the dynamic health status of components or processes [71]. While RUL-based optimization aims to directly maximize the lifetime, there are other techniques such as reliability-based optimization and maintenance optimization which aim to minimize the probability of a failure at any time, or maximize availability. Examples of such reliability-based applications include optimization of tunnel boring machines [72], healthcare systems [73], microelectromechanical systems [74], and structural systems [75]. Examples of maintenance optimization include determining the best maintenance strategy [76], optimal grouping of components for maintenance [77], and optimized inventory management strategy [78]. While these approaches are closely related, they are different from RUL-based optimization.

The Web of Science and Google Scholar are the two primary sources used to identify relevant literature in this section. The following search was used to identify the relevant literature: (”lifetime maximization” OR ”remaining useful life optimization”) ”optimization” AND -”sensor network*” ”mechanical”. This search allowed for relevant lifetime maximization, or RUL-based optimization literature on mechanical system, not network systems. The symbol “*” represents an incomplete word that can be replaced by any number of letters to complete the word. For example, the word “network*” can include network and networks. Boolean operators such as AND and OR are used to combine different key phrases. This resulted in 138 papers, those which under further investigation were relevant were included in this section.

2.1.2.2 Introduction to Operational Planning and RUL-Based Optimization for CESes

Operational planning serves as a strategy for numerous industries, guiding the efficient allocation of resources, scheduling of tasks, and overall optimization of processes [79]. It ensures the functionality of organizations, contributing significantly to productivity, cost-effectiveness, and overall performance. With CESes becoming more prevalent, there arises a critical need to enhance operational planning methodologies. RUL specifically has emerged as an important factor, as discussed previously, for anticipating lifespan of equipment or when maintenance or replacements should occur. By incorporating RUL into operational planning, industries can move beyond traditional reactive approaches, transitioning to proactive strategies that maximize resource utilization and minimize downtime. This literature review aims to bridge the gap of limited research into RUL-based optimization for CESes, by examining existing research. This literature on RUL-based operational planning encompasses diverse optimization techniques applied to single or multi-component systems, and single or multi-objective problems, using several different optimization algorithms.

2.1.2.3 Single-Component, Single-Objective RUL Maximization

The first combination of these topics include single component, single objective RUL maximization research. This is the most simple problem researched in this review, and has the sole goal of maximizing the lifetime of the component. Examples of such single component research include maximizing the lifetime of lead-acid batteries; Abdallah, Gehin, and Dieulot [80] used a genetic algorithm (GA) to optimize the number of connected photovoltaic panels and extend battery life, and Bashir et al. [81] solved a similar problem using MATLAB's $fmincon$ algorithm. Liu et al. [82] propose a temperature-suppression charging strategy for super capacitors, formulating a convex optimization problem and showcasing the effectiveness of the strategy in extending overall system lifetime. Herr et al. [83] address multi-stack fuel cell systems, utilizing mixed integer linear programming for optimal scheduling to maximize system lifetime. Although these methods are all reported to be successful, they do not consider multiple components or objectives, which is one of the goals of this thesis.

2.1.2.4 Multi-Component RUL-Based Optimization

Looking at multiple component RUL-based optimization, there is a heavy focus on energy systems currently in the literature. However, while these papers study complex systems, they still treat multiple components as a "single system." For example, Heidari et al. [84] propose a novel approach for optimizing the lifetime of internet of things (IoT) devices with low battery capacities by employing offloading strategies and deep reinforcement learning. While these IoT devices are multi-component systems, only a single RUL is considered. Similarly, Zhi et al. [85] propose a novel approach to optimize the lifetime of new energy storage modules based on battery capacity, charging and discharge currents, and employing a new artificial fish swarm algorithm enhanced with deep learning to improve optimization efficiency. The study develops a single life model which considers the entire system as one, determining the system life, which is then maximized. Moreover, Zheng et al. [86] maximize satellite lifetime by employing a multilevel BN reliability model, using mass, power, and cost, as configuring components for model/type selection and redundancy determination. Lastly, Fu, Yuan, and Zhu [87] address a similar problem for degrading components in systems with hot and cold degradation termination modes. It introduces degradation models for components and a system-lifetime maximization model. The paper utilizes a continuous nonlinear optimization algorithm and a heuristic combinatorial optimization method, to solve the problem. Although proven effective, this paper treats the entire system as a single entity and objective to optimize, while the goal of this thesis is to analyze the individual components.

2.1.2.5 Multi-Objective RUL Maximization

These reviewed articles present different approaches to multi-objective RUL-based optimization. Similar to the last subsections, much of the current-state-of-the-art focuses on energy storage systems and renewable energy components. D'Arpino D'Arpino and Cancian [88] employ dynamic programming to coordinate battery energy management with electric vehicle charging events, demonstrating significant reductions in degradation rates over a 10-year horizon. Abdelghany et al. [89] introduce a model predictive controller for grid-connected wind farms with hydrogen-based energy storage systems, optimizing lifetime by considering various operating modes and minimizing costs associated with frequent switching. Abou El-Ela et al. [90] propose a multi-objective equilibrium optimization method for integrating renewable energy resources and battery energy storage units in distribution networks, considering factors such as battery cycle to failure, operation costs, and environmental impacts. Pour et al. [91] focus on autonomous racing vehicles, introducing a health-aware control approach that optimizes lap time while maximizing battery RUL through a two-layered methodology. Meanwhile, Mabed and Dedu [92] present a bi-criteria optimization model considering short-term objectives and RUL for system lifespan extension. Finally, García-Triviño et al. [93] formulate a multi-objective optimization problem for a grid-connected hybrid renewable energy system, effectively balancing operating costs, system efficiency, and device degradation over a 25-year period. While these studies make significant contributions, there remains a potential gap in research that specifically addresses multi-objective RUL maximization for a multi-component system, focusing on nuanced aspects of the components' degradation and operational efficiency. Further exploration in this direction could enhance our understanding of how to extend the lifespan of CESes.

There are few examples of research which perform this RUL-based optimization for CESes, involving a multi-objective optimization problem. Again, most of the literature focuses solely on renewable energy systems. Moreno et al. [94] introduce an innovative approach to enhance the lifetime of solar central receivers in concentrated solar power plants by employing linear actuators in the longitudinal supports of the tubes. Through the application of a genetic algorithm, the study optimizes two key objectives: minimizing the equivalent elastic stress and maximizing the creep rupture time. Ahmadi et al. [95] address the challenges of managing the operation of distribution systems in hybrid electric vehicles. Specifically, the study focuses on optimizing tap positions of voltage regulators and charging and discharging powers for energy storage devices, considering multiple objectives such as voltage profile improvement, device lifetime maximization, and energy loss minimization. These papers offer a robust solution to their intended problems, however, they still leave a gap for this thesis. These studies consider only a single system RUL, however, they do not consider subsystem RULs which our problem considers. Also, application wise, they do not consider an unmanned system such as a USV, which is considered in this thesis.

2.1.2.6 *Real-time Optimization and Future Directions*

One additional sector which is included in the literature is the idea of real-time optimization. Real-time optimization attempts to update the systems operation in real time in response to the conditions of the system [96]. One example of RUL-based real-time optimization is by Ferrara et al. [22] who explore the application of a model predictive control (MPC) concept for optimizing fuel consumption and extending the system lifetime of a heavy-duty fuel cell electric vehicle. The online control scheme proposed in the study demonstrates a significant reduction in the average fuel cell power change rate. While the results are promising, the study acknowledges the need for further enhancements, such as the introduction of a speed prediction model. This is a major downfall in real-time optimization, because often times the objective function is computationally expensive. However, our study uses a surrogate model. Although it is not a main objective of this thesis, Chapters 3 and 4 will analyze the algorithm's ability for real-time use.

2.1.2.7 *Gaps Within RUL-Based Optimization Methods*

The novelty of this thesis lies in the system RUL prediction methodology. Adding the optimization framework to this methodology allows for the loop to be closed. By integrating the prediction and optimization frameworks, a cohesive framework is created which can leverage health information for effective decision-making and resource allocation. The next section will review techniques which will be used in this thesis to address these problems identified in Section 2.1.

2.2 Review of Technical Methods Applied

For completeness, this part of the literature review provides a quick background of a few existing methods which this thesis will use and/or extend in the next two chapters. First, BNs are reviewed, followed by deep learning methods which will be used for both SRUL and RUL prediction, respectively. Finally, genetic algorithms are reviewed which will be used for the RUL-based optimization.

2.2.1 Bayesian Networks

This section is taken from Section 2.1 in [3][2].

BNs are a commonly used modeling method in the field of reliability engineering. They offer a systematic approach for capturing knowledge from experts, integrating various data sources, and updating beliefs based on new evidence. They also facilitate risk assessment and decision-making by providing insights into the probabilities of different failure scenarios and their consequences [97]. BNs have many successful applications in CESes, including reliability analysis, risk assessment [98–100], condition monitoring [68, 101], system health diagnosis and fault detection [102–104], and prognosis [105]. Advantages of using BNs in the context of CESes include their ability to model and analyze complex relationships between components, handle uncertainties, and provide probabilistic predictions [106].

In the context of RUL estimation, traditional BNs, which primarily deal with probabilities (and thus take numbers in the range of [0,1]), may not be inherently suited for estimating variables such as RUL that can take any positive real value and are not confined to the range of [0,1]. This is where an equation-node based BN approach can be utilized, which, to the authors' knowledge, has not been used in a previous RUL estimation study. This approach involves the inclusion of equation nodes in the BN, allowing for the representation of complex relationships and the estimation of RUL values. By incorporating relevant equations and data-driven models into the BN structure, it becomes possible to estimate RUL while considering the interdependencies between system components. In this study, an equation-node based BN is constructed using system description, historical failure information and engineering knowledge, and is applied to provide a bridge from component-level RUL prediction to system-level.

[2] Matthew J. Weiner, Ruochen Yang, Katrina M. Groth, and Shapour Azarm. "Probabilistic Deep Learning with Bayesian Networks for Predicting Complex Engineering Systems' Remaining Useful Life". In: *IEEE Transactions on Reliability* (Submitted in 2024).

2.2.2 Deep Learning

This section is taken from Section 2.2 in [3][3].

Machine learning methods, specifically deep learning, have been widely adopted for PHM applications in scenarios where data is available. Deep learning methods have been implemented for all parts of the PHM process, including diagnosis through fault detection [107], fault classification [108], health state classification [109], and prognosis through estimating RUL [110] and developing predictive maintenance strategies [111]. This is due to their ability to handle large amounts of data and extract complex patterns, making it ideal for analyzing complex systems. These methods are also highly adaptable to changes in system behavior and scalable in cases of large-scale data sets.

Deep learning in PHM encompasses various approaches, including supervised learning with labeled data, unsupervised learning relying on raw data patterns, and semi-supervised learning falling in between. These methods share common requirements of training/testing data, prediction objectives, and architectural choices. Notable architectures in PHM include convolutional NNs [112], recurrent NNs [113], long short-term memory (LSTM) NNs [114], autoencoders (AEs) [53], generative adversarial networks [115], and Bayesian NNs [68, 116, 117].

One issue with many of these traditional deep learning methods is that they predict only a point value, with no degree of uncertainty. However, in engineering applications such as PHM uncertainties from input data, measurement error, and prediction models can largely lower the estimation accuracy. Therefore, uncertainty propagation is very important to ensure that the decision is made with due consideration of uncertainty. There are two main types of uncertainty to consider: aleatoric and epistemic. Aleatoric uncertainty refers to the uncertainty of the input data in training [118]. This uncertainty can exist in scenarios such as randomness of environmental conditions and uncertainty in historical failure rates [119]. Epistemic uncertainty refers to the state of knowledge uncertainties, including the uncertainty in the prediction of the deep learning method [118]. This

[3] Matthew J. Weiner, Ruochen Yang, Katrina M. Groth, and Shapour Azarm. “Probabilistic Deep Learning with Bayesian Networks for Predicting Complex Engineering Systems’ Remaining Useful Life”. In: *IEEE Transactions on Reliability* (Submitted in 2024).

includes uncertainty in the parameter estimation and model choice [119].

In order to incorporate epistemic uncertainty into deep learning, the structure of the method needs to be altered. From this dilemma, probabilistic deep learning (PDL) has gained popularity. PDL methods aim to provide not only point estimates but also quantify the uncertainty associated with the predictions. This is particularly relevant in engineering applications such as PHM, where decision-making often requires a clear understanding of the level of confidence or uncertainty associated with the predictions. In the context of RUL estimation for CESes, PDL methods have been successfully applied. For example, approaches such as Bayesian convolutional NNs [120], Bayesian LSTMs [121], variational AEs [122] allow for a more robust decision-making process by considering the variability and confidence intervals associated with the RUL estimates which is a valuable tool for decision-makers. By incorporating uncertainty estimation, these methods offer a more comprehensive assessment of system health and enable proactive maintenance strategies based on the level of confidence in the predicted remaining life.

2.2.3 Genetic Algorithm

Genetic algorithms are optimization techniques inspired by the process of natural selection. These algorithms utilize principles such as selection, crossover, and mutation to evolve a population of potential solutions toward an optimal or near-optimal solution [123]. In the context of PHM, GAs offer a versatile and adaptive approach to address complex optimization problems associated with system health monitoring, fault diagnosis, and prognosis.

Single-objective GAs focus on optimizing a singular criterion. In PHM, this may involve optimizing condition monitoring placement [124], maximizing reliability [125], or optimizing maintenance scheduling [126]. However, in PHM, many optimization problems involve conflicting objectives, such as maximizing system reliability while minimizing mission success time [11], or optimizing inspection programs minimizing costs and risk level [127]. However, multi-objective genetic algorithm (MOGA), such as the non-dominated sorting genetic algorithm II (NSGA-II), excel in handling multiple, often competing, objectives simultaneously [128].

NSGA-II is a widely used MOGA known for its efficiency in solving complex optimization problems. It employs non-dominated sorting and crowding distance to maintain diversity in the population and identify Pareto-optimal solutions [129]. NSGA-II has been successfully applied to various PHM problems, including reliability-based design optimization of roller bearings [130], optimizing maintenance scheduling for drilling rigs based on cost, time, person-hours and environmental health [131], and choosing optimal warranty strategies based on cost and availability [132].

2.3 Summary

The literature review described the working and shortcomings of prognostic and optimization methods in the context of remaining useful life prediction. Although many approaches predict the remaining useful life of simple components, few methods are able to integrate this component information into the system-level. When one considers aleatory and epistemic uncertainties, as well as interdependencies, there exists a gap in the literature. If this information was systematically predicted, it would enable lifetime maximization of the system, enabling reduced maintenance costs. The next chapter describes a methodology developed to address this gap.

Chapter 3: Integration of Probabilistic Deep Learning and Bayesian Networks for System Remaining Useful Life Prediction

This chapter is focused on creating and developing the methodology for system RUL prediction. This includes an introduction and problem definition, followed by an in-depth description of the methodology, and finally multiple case studies used to validate the developed models.

3.1 Introduction

CESes, which are intricate networks of interdependent components, are a crucial part of various critical infrastructures, encompassing physically dispersed hardware, software, and human components that are interlinked. If a failure were to occur in one of these systems, this may directly lead to the failure of the mission or have catastrophic impacts and pose significant risks to the general public's health and safety. Accurate and timely prediction of these failures is an essential step to avoid or mitigate these failures. Therefore, it is important to propose a systematic method for determining the system's health state to provide timely and accurate support for decision-making. To achieve this, RUL is often used as an indicator of system health.

In Chapter 2, an in depth literature review was performed in the area of remaining useful life prediction of complex engineering systems. This literature review identified several gaps and unsolved problems. The literature lacks a systematic method capable of predicting the RUL of a system and its components with aleatory and epistemic uncertainties, while taking interdependencies into account.

To solve this issue, this thesis proposes a systematic method for performing system health prognostics considering uncertainty in both input data and predictions. The methodology predicts system RUL by taking into account component interactions and multiple health indicators utilizing deep learning. A set of metrics which are easily comparable across all RUL data sets is also proposed. Finally, the methodology is validated for online use, filling all shortcomings of the discussed literature. Using a case study of a USV, this chapter demonstrates the application of the proposed methodology for prediction of the system RUL for CESes with interdependent components while capturing uncertainties. A complete validation of the analysis results are provided.

Specifically, there are several main contributions of this chapter: (1) A new deep learning structure for component-level PHM is developed; it utilizes a hybrid-loss function multi-layer LSTM regression model in order to predict RUL with a given confidence interval while also taking into account component interactions. (2) The development of a new framework for computing system RUL from these predicted component RULs; in which BNs are used to perform logic operations and determine the system RUL from component RULs. The entire proposed method can be applied in near real-time, allowing system RUL prediction and thus supporting timely decision-making. (3) A new set of quality metrics is proposed for comparing probabilistic deep learning applications to RUL for different data sets.

This chapter builds off of background information on current methods for RUL prediction of multi-component systems, as well as methods used in the approach including BNs and deep learning described in Sections 2.2.1 and 2.2.2. The structure of this chapter is as follows. First, Section 3.2 defines the assumptions needed to achieve the goals of this thesis, and formulates a corresponding problem statement. This is followed by a description of the general approach used to predict the RUL with uncertainty of CESes using these methods in Section 3.3. Next, a case study on the C-MAPSS data set is performed in order to validate the component-level analysis (CLA) in Section 3.4, followed by a validation of the system-level analysis (SLA) using a simulation of an engine cooling and control system (ECCS) in a USV in Section 3.5 with each respective results. Finally, this is followed by a discussion of the results in Section 3.6 and a conclusion in Section 3.7.

3.2 Problem Definition

3.2.1 RUL Prediction Problem Definition

For the RUL prediction problem as described in Section 1.2.2, there are several assumptions which need to be made about the data input to and output to the problem.

1. The predicted distribution of each component's RUL follows a non-negative truncated normal distribution. This assumption is made because the normal distribution's symmetry and well-understood properties make it a convenient choice for mathematical modeling and statistical analysis and also simplifies calculations. The assumption is valid, as it follows the idea that RUL cannot be negative. However, it is important to note, this framework can be altered for use of a lognormal or weibull distribution as well.

2. The collected condition monitoring data captures the dependencies between components. This assumption is made because it allows for the formation of relationships between components' sensor data and multiple RUL. The assumption is valid, based on the probabilistic and stochastic nature of the components degradation.

3. In the scenario, right censored data is available and it is assumed that the real-world degradation rate of each component will remain consistent with the rate observed during testing, under the assumption that operating conditions will remain similar. This assumption is made in order to construct pseudo end of life (EOL) times. This assumption is based upon the pseudo-failure time method from [133].

4. RUL is assumed to decrease linearly with time. This assumption is made in order to create targets for training and validation of predictive models using the pseudo EOL time. This assumption has been used in several RUL prediction manuscripts [134].

5. If only historical failure data is known for a specific component, the EOL can be found through the assumption of an appropriate reliability distribution. With this assumption, the EOL can be denoted as the mean-time-to-failure (MTTF). This assumption allows for

system-level RUL predictions when condition monitoring data is not available for all components. This assumption has been used in several RUL prediction manuscripts [135].

6. Since it is assumed the RUL of the components which are connected both follow a normal distribution, then the system RUL must also follow a normal distribution [136]. This assumptions allows for the integration of multiple component RUL distributions to predict a single system-level RUL distribution.

These assumptions will help formulate the problem formally as described in the following section.

3.2.2 Remaining Useful Life Prediction Problem Formulation

Using the assumptions made in Section 3.2.1, the final formulations can be made for the RUL prediction problem descried in Section 1.2.2. This problem formulation will be broken into three separate categories: model inputs, component-level outputs, and system-level outputs.

The problem that this study aims to solve is a discrete-time system health prognosis problem with unit time steps based on the sampling rate of the condition monitoring data available. The input data to train and test the models will include condition monitoring data from various sensors at a constant sampling rate, known historical failure rates, engineering knowledge, and a system description. The condition monitoring data for each component, c, can be described in the following form:

$$\boldsymbol{CMD_c} = f(\boldsymbol{x}, \boldsymbol{p}, t) \tag{3.1}$$

where Condition Monitoring Data, $\boldsymbol{CMD_c}$, is a vector of all sensors for the component collected at a specific time t, based on known input parameters, $\boldsymbol{x}$, and unknown parameters $\boldsymbol{p}$. The aleatory uncertainty in this problem lies in this environmental uncertainty. Based on assumption 2 in Section 3.2.1, this data contains the relationships and interactions between different components. These relationships need to be extracted by the model in order to capture interdependencies. One specific sensor value from the $\boldsymbol{CMD_c}$ vector will represent the efficiency of each component. This efficiency will be used with assumption 3 from Section 3.2.1 as the rate of degradation of the component. The aleatory uncertainty in this problem lies in this environmental uncertainty. Based on assumption 2 in Section 3.2.1, this data contains the relationships and interactions between different components. These relationships need to be extracted by the model in order to capture interdependencies. One specific sensor value from the $\boldsymbol{CMD_c}$ vector will represent the efficiency of each component. This efficiency will be used with assumption 3 from Section 3.2.1 as the rate of degradation of the component.

Using this input data, the component-level model will be used to predict μ_{RUL_c}, the expected RUL of each component in the system. If there is condition monitoring data available for a component, then the component's predicted, RUL_c, will contain uncertainty, in the form of a truncated non-negative normal distribution (based on assumption 1 from Section 3.2.1). If there is no condition monitoring data available, a historical failure rate, λ_c, will be used to estimate the EOL using the MTTF (from assumption 5 in Section 3.2.1).

$$\boldsymbol{\mu}_{RUL,c}, \boldsymbol{\sigma}_{RUL,c} = \begin{cases} f(\boldsymbol{CMD}_c, t) & \text{if CMD is available} \\ f(\lambda_c, t) & \text{if CMD is not available} \end{cases} \tag{3.2}$$

Equation 3.2 formulates this in equation form, where $\boldsymbol{\mu}_{RUL,}\boldsymbol{,c}$ and $\boldsymbol{\sigma}_{RUL}\boldsymbol{,c}$ are the predicted vector of the mean and standard deviation of the component RUL function for all timesteps. This completes the component-level formulation.

Moving to the system-level formulation, the only item which must be formulated is the component interactions with the system. These interactions will decide how the SRUL relates to the constituent components. The two interactions which will be studied in this methodology are series and parallel connections. This formulation can be written as follows:

$$\boldsymbol{\mu}_{SRUL}, \boldsymbol{\sigma}_{SRUL} = \begin{cases} f(\mu_{\mathbf{RUL_i}}, \sigma_{\mathbf{RUL_i}}) & \text{if connected in series} \\ f(\mu_{\mathbf{RUL_j}}, \sigma_{\mathbf{RUL_j}}) & \text{if connected in parallel} \end{cases} \tag{3.3}$$

where i and j represent the number of components connected in series or parallel, respectively.

Note in Equations 3.1, 3.2, 3.3, $f()$ denotes that the equation is a function of those input variables.

This concludes the problem overview, definition, and formulation of the RUL prediction problem tackled in this thesis.

3.3 Methodology

To solve the problem described and formulated in Sections 1.2.2, 3.2.2, and 3.2.1 this study proposes a mathematical and computational architecture which enables the fusion of data and models from PRA and PHM. Following the SIPPRA framework [1, 6], the proposed method consists of a training and testing methodology to be completed at the component and system-level.

Figure 3.1 shows the complete methodology for the proposed RUL prediction algorithm. As shown, the approach is broken down into two main steps: training steps and testing and integration steps. During the training steps, information about the problem, such as system description, components' historical reliability data from previous work, and engineering knowledge are collected and used to develop models used in the framework. The specific output of the training steps will be fully trained probabilistic LSTM model and a complete equation node-based Bayesian network. These models are constructed using known system logic and historical component failure rates, along with failure data which is used to train the models.

The testing and integration steps of the methodology are broken into two levels: component level and system level. In the component level, the PHM model, previously developed using PDL in the training steps, is used to predict the RUL of each component. In the system level, the BN integrates each components RUL and calculates the system RUL.

A detailed process of how each of these steps are completed is presented separated into two levels: component-level and system-level. For this thesis, component refers to any mechanical component or subsystem which is a part of the larger overall system. System refers to the entire overall system being evaluated.

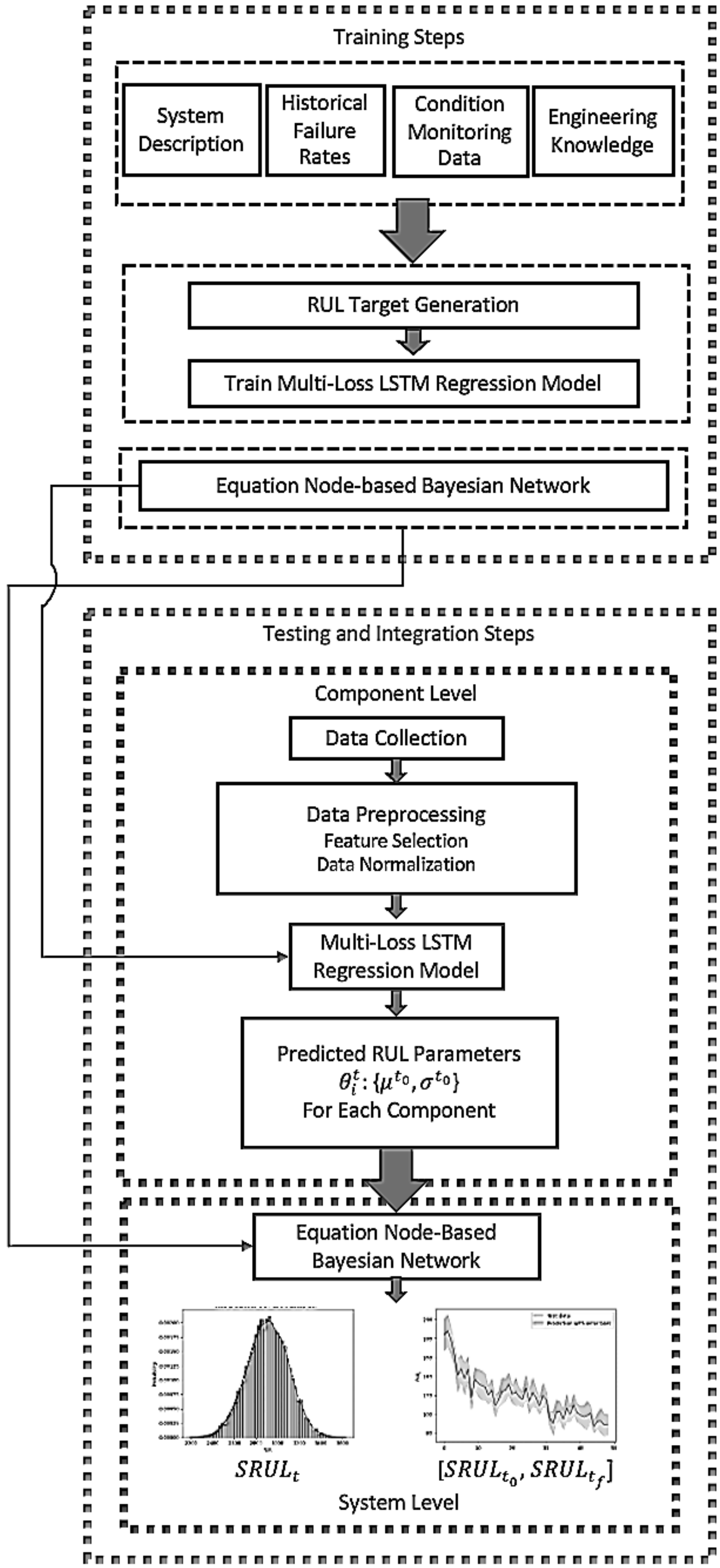


Figure 3.1: Proposed approach for RUL prediction algorithm.

3.3.1 Component-Level Analysis

The component-level analysis is focused around collection of operational data and the use of historical failure data for the construction of PHM models for prognostics of individual components. The two types of data used in this framework will be historical failure data, from past qualified literature and databases, and operational sensor data; depending on the data available, different steps will be taken. The goal of this portion of the methodology is to provide a point estimate, along with uncertainty bounds, of the RUL of each component which has operational data available. In order for this to be accomplished, the data must first be collected and then pre-processed. Next, the data is used to train a probabilistic deep learning model. The PDL model used in this framework will be a LSTM regression model, due to its ability to handle sequential data, like RUL. These steps will be explained in-depth in the following sub-sections.

3.3.1.1 Data Collection

In an ideal scenario, complete failures with condition monitoring data is available. However, this is not always plausible, especially for CESes where there are very limited complete failure history records. For this reason, the methodology also allows for use of historical failure data to be used in place of real-time operational sensor data with failures when not available. The operational data used in the current methodology is sensor data, such as temperature, pressure, etc., collected during operation, along with the failure time or time of termination of the test. Operational sensor data can be obtained from real experimental tests or from physics-based simulations with built in stochasticity. Historical data can be collected from various sources, including reliability handbooks and academic journals.

3.3.1.2 Data pre-processing

There are two different categories of failure data, including complete or censored. In the scenario that complete failure data is available, RUL targets can be constructed easily using Equation 3.4, where EOL is the end of life, or the time in which the failure occurs or usable lifetime ends, and t is the current time step.

$$RUL(t) = EOL - t \tag{3.4}$$

In the case where complete failure data is not available, censored operational data is used. Among all types of censored data, the most common is right censored, which is where the test is terminated before failure occurs [137]. In order to create a RUL target to use this data for training, efficiencies can be used to denote failure [61]. That is, if a threshold for the efficiency of a component is predefined at a critical value $\eta_{Critical}$, the time which this efficiency is reached can be denoted as the EOL. In this case, RUL can be calculated at each time step using Equation 3.4, as well. The last scenario for right censored data is that this failure threshold was not reached during testing. In this scenario, we assume that the degradation rate of each component will remain consistent with the rate observed during testing, under the assumption that operating conditions will remain similar. Using a line of best fit for all data sets with the same conditions, the time at which η_{crit} can be predicted with some amount of uncertainty. Figure 3.2 shows an example of the visualization of this process. $\eta_{Critical}$ is defined prior to the study, and is used to determine when EOL occurs and calculate RUL based off of that.

In cases where efficiencies are not available, reliability can be used in place of efficiency using the same method, and the EOL can be denoted as the time in which a critical reliability, R_{crit}, is reached. This method would require historical failure data which can be used to construct failure rates or reliability distribution. If a reliability distribution is available, the EOL can be calculated as the time in which the reliability is expected to reach the critical value. If only historical failure data are known, the EOL can be found through the assumption of an appropriate reliability distribution. With this assumption, the EOL can be denoted as the MTTF.

Next, feature extraction is performed in order to remove unneeded features. To do so, a heat map from seaborn library in Python is used [138]. This heat map computes the pairwise correlation between all features in the dataset, including RUL. The resulting matrix is a square matrix where the off-diagonal elements represent the correlation between two different columns. The correlation

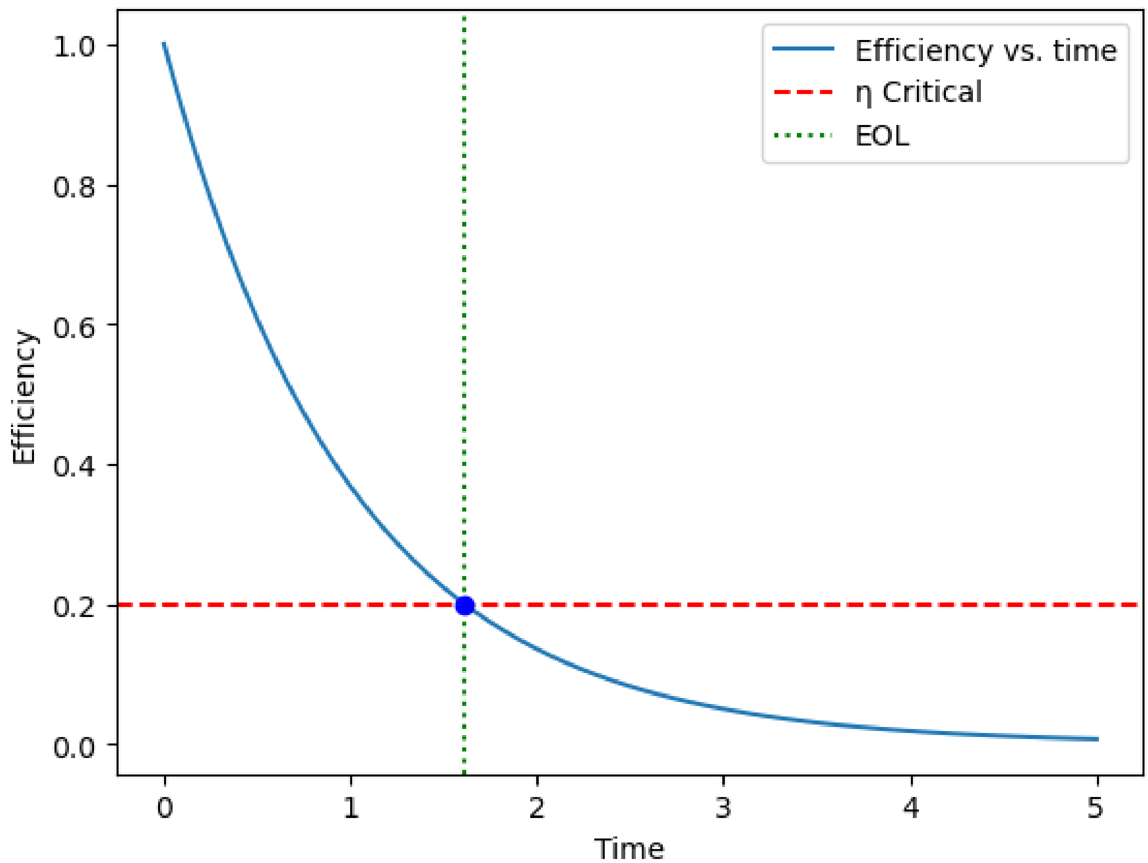


Figure 3.2: Visualization of EOL calculation using efficiency.

is represented by the Pearson's correlation coefficient, $\rho_{X,Y}$.

$$\rho_{X,Y} = \frac{\mathrm{cov}(X,Y)}{\sigma_X \sigma_Y} \tag{3.5}$$

where $\mathrm{cov}(X,Y)$ is the covariance between variables X and Y, and σ_X and σ_Y are the standard deviations of variables X and Y, respectively.

The Pearson correlation coefficient is a measure of the linear relationship between two variables, and it ranges from -1 to 1. A correlation coefficient of -1 indicates a perfect negative linear relationship, a coefficient of 0 indicates no linear relationship, and a coefficient of 1 indicates a perfect positive linear relationship. Therefore, any features which have a correlation value close to zero can be removed. The cutoff will depend upon the problem, for example, in cases of large amounts of features it may be advantageous to remove more features in order to reduce the computational expense of the model.

Also, it is required to normalize the sensor data. This ensures that all input variables have similar scales, preventing certain features from dominating the learning process and enabling the model to converge faster and perform better. Max-min normalization is used which scales the features of a data set to a fixed range of [0, 1]. The min-max scaling equation for a feature column X is as follows:

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{3.6}$$

where X_{min} and X_{max} are the minimum and maximum values of X, respectively, and X_{norm} is the normalized value of X.

3.3.1.3 Multi-Loss Deep LSTM Regression Model

Once the data pre-processing steps are complete, the PDL model can be constructed and trained. The model used in this methodology is a multi-loss deep LSTM regression model. This model is chosen, because LSTM neural networks are effective in predicting sequential data, such as RUL. Also, the multi-loss function allows the training to penalize error in the mean as well as a large uncertainty. The predicted mean and standard deviation allow for uncertainty estimations. The model consists of three LSTM layers followed by a time-distributed layer and two fully connected layers for mean and standard deviation predictions. The model uses a combination of mean square error (MSE) and negative log-likelihood (NLL) loss as its loss function and is trained using the Adam optimizer. The combination of negative log-likelihood loss and mean square error loss is calculated to account for the model's probabilistic output. The model uses the PyTorch library [139], and is trained using different hyper parameters for each case study.

Figure 3.3 shows a visualization of the data flow into and out of the PDL model. As shown, operational data is input into the model and a mean and standard deviation is output which is used to construct a normal distribution to model RUL for each time step of data available.

The input layer receives and processes incoming sequential data in the form of $(number features) \times (timesteps)$. Next, the LSTM layers include specialized memory cells to

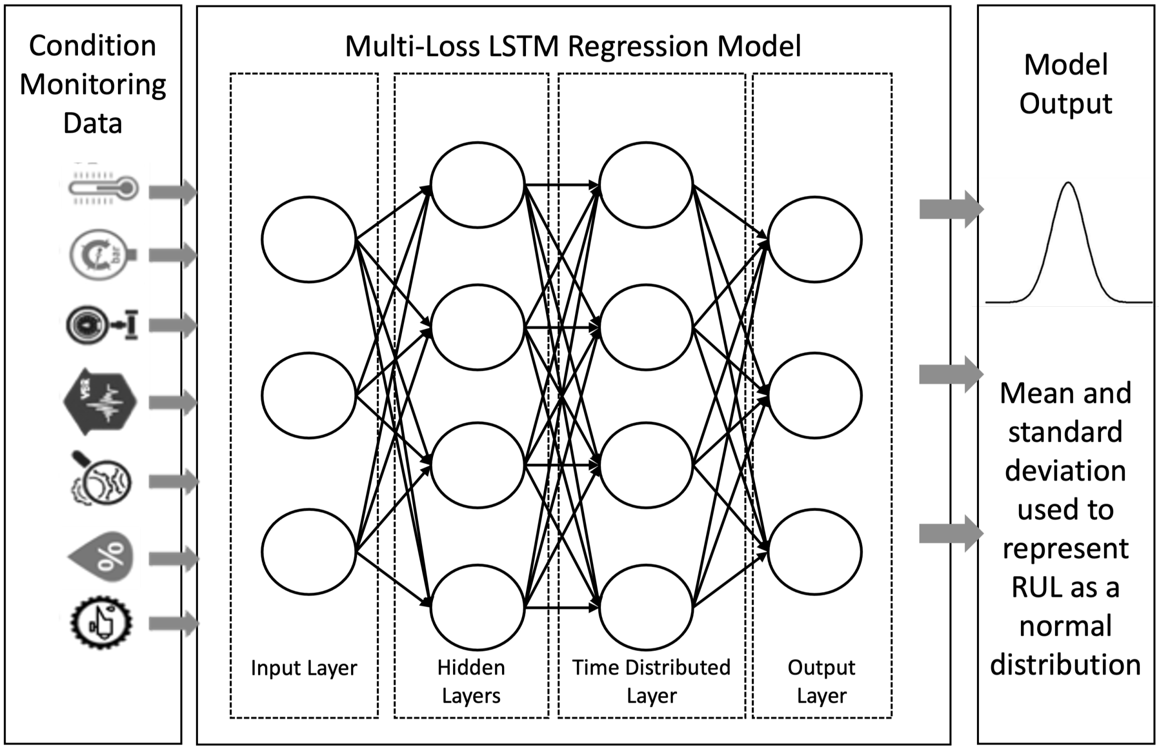


Figure 3.3: Visualization of LSTM model architecture and data flow.

capture and store information over long sequences, allowing it to effectively model and learn from sequential data with the ability to retain important context. These layers feed into a time distributed layer used to apply a specific operation to each time step of a sequence independently, in this case mapping a normal distribution to the data. This allows for parallelization of computations across time steps while maintaining temporal dependencies. Finally, the output layer uses a specific activation function to create a meaningful output, the predicted mean and standard deviation.

The LSTM layers calculate three gates (input (i), forget (f), output (o)) and a candidate hidden state at each time step. The equations for these gates and the candidate hidden state are:

Input gate:

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \tag{3.7}$$

Forget gate:

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \tag{3.8}$$

Output gate:

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \tag{3.9}$$

Candidate hidden state:

$$g_t = \tanh(W_g x_t + U_g h_{t-1} + b_g) \tag{3.10}$$

Current hidden state:

$$h_t = o_t \times \tanh(c_t) \tag{3.11}$$

Current cell state:

$$c_t = f_t \times c_{t-1} + i_t \times g_t \tag{3.12}$$

where σ represents the sigmoid function, W represents the corresponding weight matrix of each input for each state. U represents the corresponding weight matrix of each previous state for each state. Finally, b represents the bias terms that are added to the weighted inputs.

The LSTM model outputs a mean and standard deviation for each input sequence. These are calculated using fully connected layers with softplus activation functions, in order to ensure a non-negative mean and standard deviation.

$$\mu_{RUL,c} = W_{mean} h_{last} + b_{mean} \tag{3.13}$$

Here, $\mu_{RUL,c}$ is the predicted mean of the RUL distribution for component, c.

$$\sigma_{RUL,c} = W_{std} h_{last} + b_{std} \tag{3.14}$$

$\sigma_{RUL,c}$ is the predicted standard deviation of the RUL distribution for component c.

It is important to note that these equations are based on the underlying assumption that the predicted component RUL is normally distributed. However, the proposed methodology is very flexible, and several other distributions can be assumed depending on different cases, such as weibull or lognormal. The normal distribution is a widely used continuous probability distribution for modeling lifetimes including RUL. A truncated non-negative normal distribution, as used in this methodology, does not allow negative values, which, in the context of RUL prediction, is effective in aligning with the physical constraint that RUL cannot be negative.

As discussed, the loss function used consists of both NLL and MSE loss. The NLL loss function encourages the model to produce low-variance predictions by penalizing the model if it has high variance in its predictions, encouraging the model to produce more confident predictions. Given a true value y and a predicted mean and standard deviation (μ_i, σ_i), the NLL loss is:

$$\Lambda = -0.5 \times ln(2\pi\sigma_i^2) - \frac{(y-\mu_i)^2}{2\sigma_i^2} \tag{3.15}$$

$$nllloss = -E(\sum_{i=1}^{N} \Lambda) \tag{3.16}$$

where N is the total number of data points in the sample. The MSE loss function encourages the model to make accurate predictions of the mean and can be represented by the following equation:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3.17}$$

where n is the number of data points, y_i is the true value of the i-th data point, and $\hat{y}_i$ is the predicted value of the i-th data point.

Combining these two loss functions together, the following final loss function can be constructed:

$$Loss = (1-\lambda) \times MSE + \lambda \times NLL \tag{3.18}$$

The final loss is a combination of the two in order for the model to equally penalize large variances and differences in the mean. The value λ can be optimized based on certain quality metrics, explained in the next section.

The optimizer used in this code is *Adam*, based on [140], which updates the model weights based on the gradient of the loss function with respect to the weights.

Update rule:

$$w_{t+1} = w_t - \alpha \times \frac{m_t}{\sqrt{v_t} + \varepsilon} \tag{3.19}$$

$$m_t = \beta_1 \times m_{t-1} + (1 - \beta_1) \times g_t \tag{3.20}$$

$$v_t = \beta_2 \times v_{t-1} + (1 - \beta_2) \times g_t^2 \tag{3.21}$$

where w_t are the weights, α is the learning rate, β_1 and β_2 are hyper parameters, m_t and v_t are the first and second moment estimates, g_t is the gradient, and ε is a small constant to avoid division by zero.

Finally, in order to account for inter-dependencies between different components, condition monitoring data from all connected components is used to train each component. This allows the PDL model to account for relationships between other components performance and its own.

3.3.1.4 Component-Level Quality Metrics

The output of the deep learning method used gives a point prediction as well as a confidence interval. In order to properly assess the quality of the prediction, point-based and uncertainty-based validation metrics are used in this study as an interpreter of the accuracy of the prediction results.

Previous studies usually used metrics such as MSE and a scoring function (SF), such as [57, 141]. However, due to the specific nature of the equations, both these metrics can only be used to compare methods within the same data test-set. Therefore, this methodology proposes a normalized root mean square error (NRMSE) metric which can compare models' accuracy for different data sets.

$$NRMSE = \frac{RMSE}{\bar{y}} \tag{3.22}$$

where $RMSE$ is the root mean square error and $\bar{y}$ is the mean of the test values.

In order to compare the measure of the uncertainty prediction, two commonly-used metrics, prediction interval coverage percentage (PICP) and normalized mean prediction interval width (NMPIW), are used in this study. The equations for each are taken from [142]. The first, PICP, evaluates the percentage of true values which fall between the mean prediction with uncertainty bounds based on a certain value of confidence. This thesis used 90% confidence to be consistent with existing literature [143]. The equation for PICP is given as:

$$PICP = \frac{1}{n}\sum_{k=1}^{n} I(RUL_k \in [\widehat{RUL}_k^{LB}, \widehat{RUL}_k^{UB}]) \tag{3.23}$$

where n is the total number of data points, RUL_k is the RUL at point k, and $\widehat{RUL}_k^{LB}$ and $\widehat{RUL}_k^{UB}$ are the lower and upper bounds of the prediction interval, respectively.

The function I is defined as a piece-wise function, which can be written as:

$$I(RUL_k \in [\widehat{RUL}_k^{LB}, \widehat{RUL}_k^{UB}]) = \begin{matrix} 1, & if RUL_k \in [\widehat{RUL}_k^{LB}, \widehat{RUL}_k^{UB}] \\ 0, & otherwise \end{matrix} \tag{3.24}$$

The PICP equation then takes the average of the values of I over all the data points to compute the overall prediction interval coverage probability.

Next, the NMPIW is used to evaluate the width of prediction intervals. The smaller this value, the more tight of a predicted bounds. The NMPIW can be calculated as follows:

$$NMPIW = \frac{1}{n \times (RUL_{max} - RUL_{min})} \sum_{k=1}^{n} (\widehat{RUL}_k^{UB}(\alpha) - \widehat{RUL}_k^{LB}(\alpha)) \tag{3.25}$$

In Equation 3.25, RUL_{max} and RUL_{max} are the maximum and minimum values of the RUL variable in the data set, and n is the number of data points. The terms $\widehat{RUL}_k^{UB}(\alpha)$ and $\widehat{RUL}_k^{LB}(\alpha)$ are the upper and lower bounds on the predicted RUL at each data point k, which depend on the value of the confidence parameter α.

The equations for PICP and NMPIW have a natural trade off. A high PICP has little significance if there is a high NMPIW, because this means the uncertainty is so large that it covers a large amount of the domain. Therefore, in order to quantify this trade off, this thesis proposes a metric called the coverage-width balance (CWB), which can be given by the following equation,

$$CWB = PICP - NMPIW \tag{3.26}$$

where a value closer to 1 shows an ideal scenario where the model's uncertainty is accurate, but not large. A low value denotes that the uncertainty width devalues the PICP.

3.3.2 System-Level Analysis

The goal of the system-level analysis is to predict a SRUL distribution, which can then be used for decision-making. To achieve this, information from the component-level analysis is integrated with the system functional logic. In this part of the architecture, an equation node-based Bayesian network is developed using system logic and engineering knowledge of the system. The BN incorporates all the components and subsystems included in the system at hand. Due to the complex nature of CESes, the BN does not contain a node for every single sub-component, due to the computational expense which would come along with this. In order to have this methodology give a reliable prediction in real time, nodes related to one subsystem are aggregated into one single node.

3.3.2.1 Equation Node-Based Bayesian Network

Unlike most BNs, where each node represents a probability of that node event occurring, equation-based BNs use an equation to generate a value for each node. In this methodology, each node's value represents the RUL of that subsystem or component. In order to determine the system RUL, component interactions must be known. The two interaction considered in this study are parallel and series connections. Parallel systems fail when all components of said system fail, therefore the system RUL can be shown by the following.

$$SRUL = MAX(RUL_i); \in 1, 2, ..., N \tag{3.27}$$

Systems connected in series fail when any of the components fail, therefore the SRUL can be shown by Equation 3.28.

$$SRUL = MIN(RUL_i); \in 1, 2, ..., N \tag{3.28}$$

As an artifact of this methodology, the SRUL may resemble or exactly replicate the RUL of a single component if that component's mean RUL is significantly greater or less than other components which it is connected to, depending on the failure logic equation which the equation-node BN represents.

A goal of this methodology is to not only equate the SRUL, but do so in real time. Advantageously, an equation node-based BN can be processed very quickly. In order to allow the system-level code to be integrated with the component level, GeNIe Modeler by BayesFusion [144], is used to create the BN which is updated at each time step in python using pySMILE. This allows for seamless integration of the two levels of analysis in real time.

3.3.2.2 SRUL Distribution Calculation

The equations constructed in Section 3.3.2.1 only consider the RUL if it was a single mean point prediction, ignoring the possible uncertainty in prediction.

To consider the possible uncertainties involved in the RUL prediction, Equations 3.27 and 3.28 need to be extended to solve this issue. In this study, the RUL is assigned a probability density function (pdf) in order to represent the uncertainty in the SRUL prediction. Since it is assumed the RUL of the components which are connected both follow a normal distribution, then the SRUL must also follow a normal distribution. The assumption that RUL follows a normal distribution can be considered valid despite the constraint that it cannot be negative, because it is often employed as a simplifying approximation in various engineering and statistical models. Although a negative RUL is not possible, if it is included within the predicted confidence interval, it still provides key information that the system is at a critical state. Moreover, the normal distribution's symmetry and well-understood properties make it a convenient choice for mathematical modeling and statistical analysis, simplifying calculations.

Determining the exact distribution of the MAX/MIN for two normal random variables has been studied in the literature by [136]. This methodology will extend this approach for estimating the

SRUL distribution from component-level.

First, the pdf of the ith component's RUL, f_i, can be defined as,

$$f_i(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{3.29}$$

where x represents the RUL, μ and σ are the mean and standard deviation of the predicted distribution, and $f(x)$ is the pdf of the predicted distribution. For parallel systems the resultant pdf can be determined using,

$$P(x) = \sum_{i=1}^{n} f_i(-x) \tag{3.30}$$

where n represents the total number of components in the system connected in parallel, and $P(x)$ represents the pdf for the SRUL of the parallel system. Now, substituting in the normal pdf for f_i, the mean for the resultant SRUL normal pdf, $\mu_{Parallel}$, can be determined using the following equation,

$$\mu_{Parallel} = \mu_1 \times \Phi(\frac{\mu_1 - \mu_2}{\theta}) + \mu_2 \times \Phi(\frac{\mu_2 - \mu_1}{\theta}) + \theta \times \phi(\frac{\mu_1 - \mu_2}{\theta}) \tag{3.31}$$

where μ_1 and μ_2 are the means of the respective components' RUL distributions, σ_1 and σ_2 are the standard deviations of the respective components' RUL distributions, and $\phi()$ and $\Phi()$ are the pdf and the cumulative distribution function (cdf) of the standard normal distribution, respectively. Also, the variance can be determined using the following equation,

$$\begin{split}\sigma^2_{Parallel} = (\sigma_1^2 + \mu_1^2) \times \Phi(\frac{\mu_1 - \mu_2}{\theta}) + (\sigma_2^2 + \mu_2^2) \times \Phi(\frac{\mu_2 - \mu_1}{\theta}) \\ + (\mu_1 + \mu_2) \times \theta \times \phi(\frac{\mu_1 - \mu_2}{\theta})\end{split} \tag{3.32}$$

where

$$\theta = \sqrt{\sigma_1^2 + \sigma_2^2} \tag{3.33}$$

This equation can include a third term which is dependent on the correlation between the two distributions, ρ, however, because the component's RUL distributions are assumed independent of each other, the correlation coefficient is zero and this term disappears.

For systems with more than two components connected, these equations can be used multiple times. For example, in a system with 3 components, Equations 3.31 and 3.32 are used for two of the three components, then the resultant mean and standard deviations are used with the third to determine the SRUL of the three component system.

Similar equations can be written for components connected in series. Following Equation 3.28, the resultant pdf can be determined using,

$$S(x) = \sum_{i=1}^{n} f_i(x) \tag{3.34}$$

where n represents the total number of components in the system connected in series, and $S(x)$ represents the pdf for the SRUL of the series system.

Now, substituting in the normal pdf for f_i, the mean can be determined using the following equation,

$$\mu_{Series} = \mu_1 \times \Phi(\frac{\mu_2 - \mu_1}{\theta}) + \mu_2 \times \Phi(\frac{\mu_1 - \mu_2}{\theta}) + \theta \times \phi(\frac{\mu_2 - \mu_1}{\theta}) \tag{3.35}$$

and the variance can be determined using the following equation.

$$\begin{aligned}\sigma_{Series}^2 = (\sigma_1^2 + \mu_1^2) \times \Phi(\frac{\mu_2 - \mu_1}{\theta}) + (\sigma_2^2 + \mu_2^2) \times \Phi(\frac{\mu_1 - \mu_2}{\theta}) \\ - (\mu_1 + \mu_2) \times \theta \times \phi(\frac{\mu_2 - \mu_1}{\theta})\end{aligned} \tag{3.36}$$

Although these relationships are simple and have long been known in the statistics literature [145–148], this application of the knowledge for SRUL prediction has not been studied to the authors' knowledge.

3.3.2.3 System-level Metrics

For this thesis, since there is no common data set, it is difficult to compare different methodologies against each other fairly. Additionally, problems with CESes involve multiple facets of prediction, which often require more nuanced methods for comparison.

Due to the need for multiple metrics to evaluate the system-level analysis framework, metrics proposed in [149] will be discussed. In the paper, they propose multiple system-level analysis metrics which can be used to compare different techniques. Table 3.1 shows a list of these metrics along with their example qualitative ranges and phase which they refer to.

The next sections of the chapter will use various test data to validate the component-level analysis step of the framework.

3.4 Component-level analysis Validation

In this section, the Turbofan Engine Degradation Simulation provided by NASA Ames Prognostics Data Repository, known as the C-MAPSS, is used to validate the proposed component level analysis. This data set has been commonly used in the PHM field, and is comprised of 4 subsets: FD001, FD002, FD003 and FD004, the first of which is the most commonly used [57, 143, 150, 151]. C-MAPSS uses a realistic engine control system, and subjects all engines to a single fault mode. Further details of this data set can be found in [20].

3.4.1 Data Pre-processing

The C-MAPSS data set is condition monitoring data labeled with a time to failure. There are 21 sensors and three settings which record measurements across the entire lifetime of the components, including various temperatures, pressures, fan speed, flow rates, and more. Before the model can

Health monitoring phase	Metric	Example Qualitative Ranges
Gather system failure information	Age of system failure information	Outdated – Up-to-date
	Cost-effectiveness of information	Inefficient – Economical
Construct risk assessment framework	Ease of model modification	Rigid – Fully adaptable
	Framework coverage of known failure modes	Single failure – Comprehensive
	Framework explain-ability	Black box – Fully explainable
	Framework trace-ability	Black box – Fully explainable
Collect system data	Availability of data collection sources	Unavailable – Available
	Data collection costs	Expensive – None
	Data coverage	Limited – Full
	Required data storage capacity	Prohibitive – None/Minimal
Pre-process data	Processing equipment requirements	Significant burden–None
	Processing explainability	Black box–Fully explainable
Perform (Sub)system-level assessment	Time required for assessment	Slow - Fast
Evaluate and adjust system management	Expertise required for model use	None – Specific training
	Model output interpretability	Not meaningful – Meaningful

Table 3.1: A subset of the relevant qualitative metrics for evaluating system-level analysis, adapted from [149].

be trained, the raw data must be pre-processed following the methodology from Section 3.3.1.

RUL Target Generation The first step, which is to construct RUL targets for training, is completed using Equation 3.4. As discussed in the methodology section and shown through Equation 3.4, traditionally, RUL is assumed to decrease linearly with time. Therefore, this assumes that system health also degrades linearly with time. The actuality for many engineering applications, however, is that components degrade some at the beginning of their life, levels out, and then the rate of degradation increases as it approached the end-of life. In order to address this concern, [152, 153] suggest employing a piece-wise linear target function for RUL estimation. This approach imposes a fixed upper limit on the RUL, beyond which linear degradation begins after a

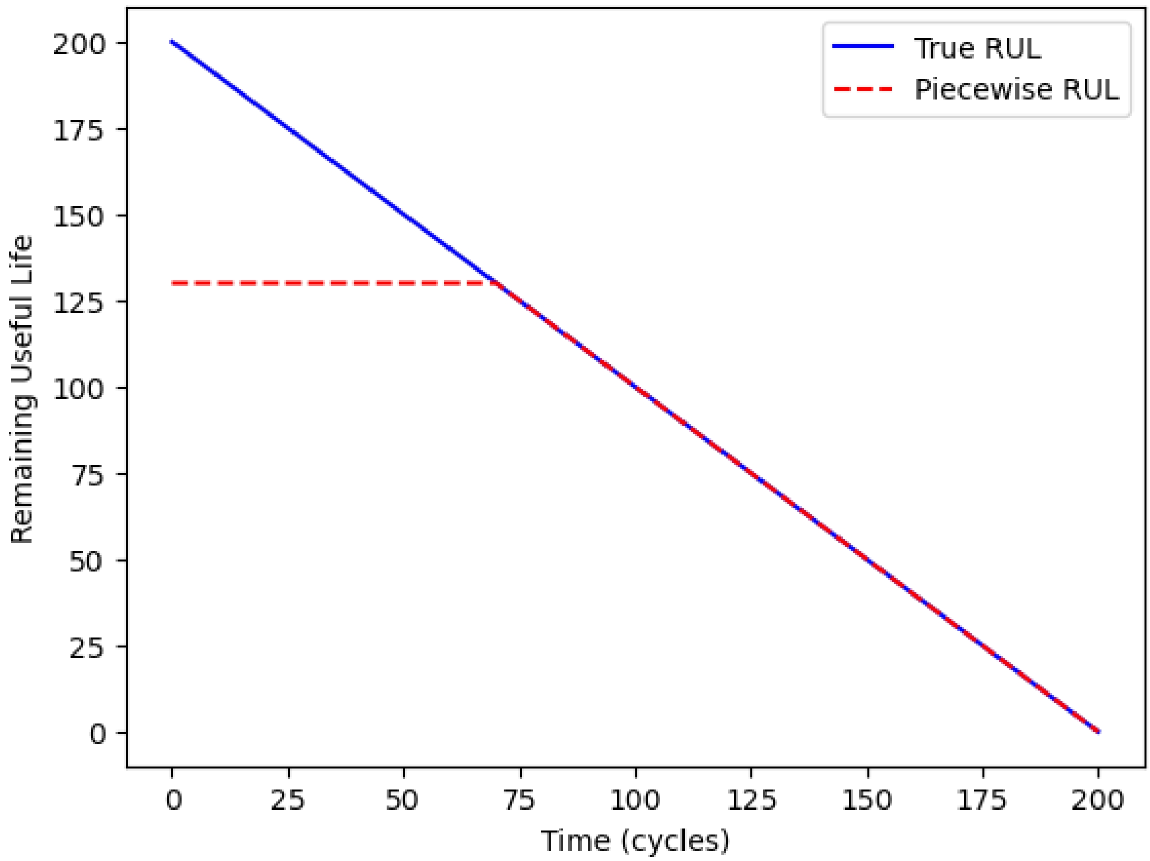


Figure 3.4: Piece-wise RUL target function for C-MAPSS data set.

specific level of usage. In the case of the C-MAPSS data set, they defined the maximum limit as 130 time cycles as shown in Figure 3.4. This piece-wise function was used for model training and used to calculate the quality metrics for testing.

Feature Selection Figure 3.5 shows a heat map constructed using the 21 sensors. The color scale shows the correlation value. From this figure, ”unit” ”setting3”, ”s1”, ”s5”, ”s10”, ”s16”, ”s18”, ”s19” were all removed, because they show no correlation based on Equation 3.5. As a result, time, 2 settings and 15 sensors were selected.

Data Normalization Following feature selection, all measurements were normalized according to Equation 3.6.

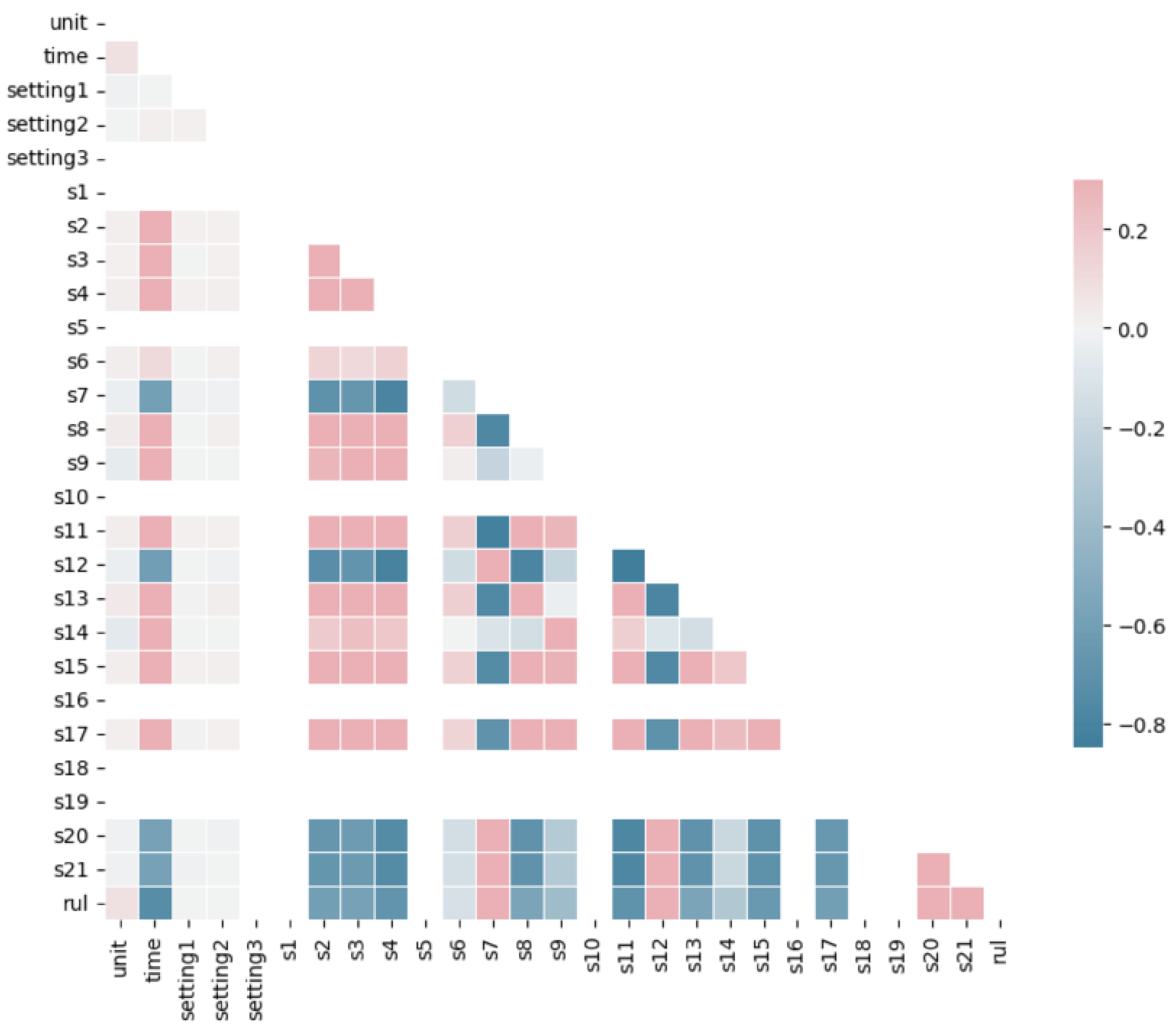


Figure 3.5: Heat map showing variable correlation for C-MAPSS data set FD001.

3.4.2 PDL Model Development and Training

The multi-loss deep LSTM regression model presented in Section 3.3.1.3 is constructed using the PyTorch library of Python. The hyper parameters of the model can be shown in Table 3.2. The hyper parameters were chosen using a Bayesian optimization technique in which the quality metric RMSE was minimized. The optimizer was implemented using the Ray Tune library [154]. The optimizer ran for 20 iterations.

The model was trained using the 100 training conditions provided in the data set. Once the model was trained, it was tested on the testing data sets. The results of this testing is presented in the following section.

λ	Learning rate	Number of epochs	Batch size	Number of LSTM layers	Number of hidden layers
0.7	0.005	245	64	3	140

Table 3.2: Hyper parameter values for deep learning model case study using C-MAPSS dataset.

3.4.3 Quality Metrics for C-MAPSS Case Study

In order to compare results to other methods in the C-MAPSS case study, four metrics are used in this study; two for point prediction and two for uncertainty quantification. These metrics are used widely in papers such as [57, 143, 150, 151]. In addition to the two metrics for uncertainty quantification shown in Section 3.3.1.4, two other popular point-prediction metrics that are specifically used for the C-MAPSS data set are considered in the current validation process. The first commonly used metric is a SF. This popular metric allows late predictions to be punished more harshly.

$$SF = \sum_{k=1}^{M} s_k; s_k = \left\{ \begin{array}{ll} e^{\frac{-d_k}{13}-1}, & d_k < 0 \\ e^{\frac{-d_k}{10}-1}, & d_k \geq 0 \end{array} \right\} \tag{3.37}$$

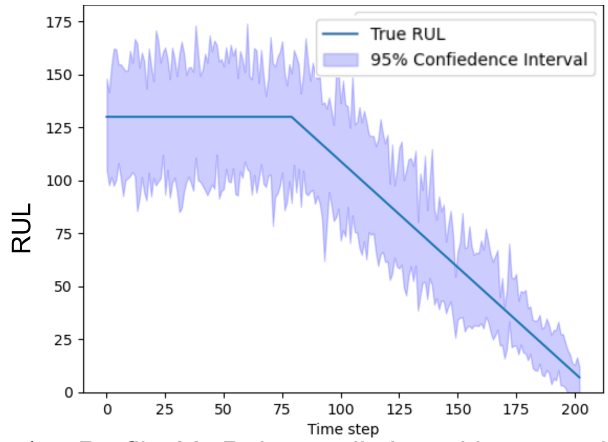
Here, d_k is the difference between the predicted and true mean value, k is the current time step being evaluated, and M is the total number of time steps.

The second commonly used metric is accuracy. This measures the percentage of predictions falling within a certain interval from the correct value. For the C-MAPSS data set that interval is [-13, 10].

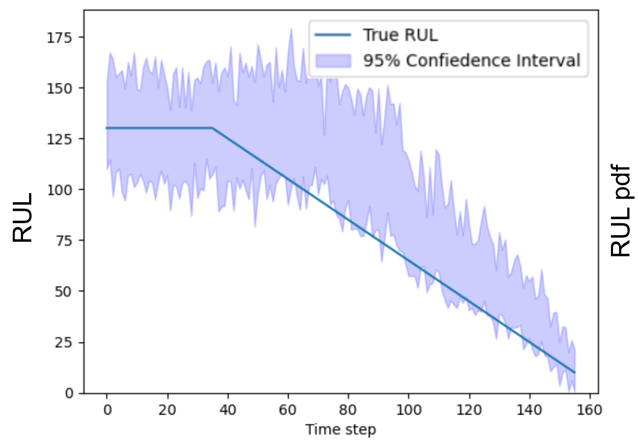
$$Accuracy = \frac{100}{M} \sum_{k=1}^{M} I(d_k); I(d_k) = \left\{ \begin{array}{ll} 1, & d_k \in [-13, 10] \\ 0, & otherwise \end{array} \right\} \tag{3.38}$$

3.4.4 Validation Results Using C-MAPSS Data Set

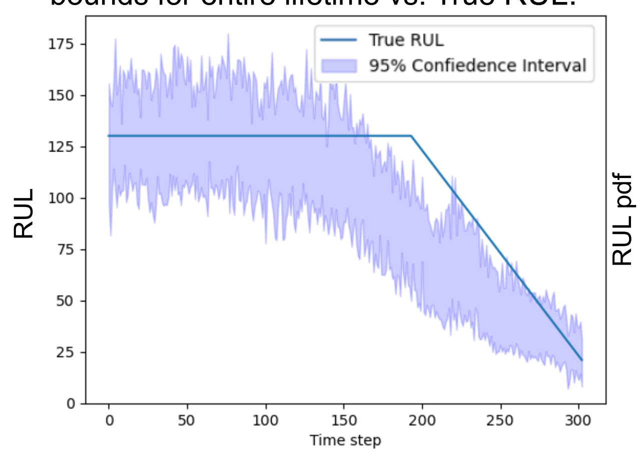
Using the metrics from Sections 3.3.1.4 and 3.4.3, the proposed multi-loss LSTM regression model's performance in both point-wise prediction and uncertainty will be compared to those reported in existing literature in this section.



a) Profile 33: Point prediction with uncertainty bounds for entire lifetime vs. True RUL



b) Profile 41: Point prediction with uncertainty bounds for entire lifetime vs. True RUL.



c) Profile 48: Point prediction with uncertainty bounds for entire lifetime vs. True RUL.

Figure 3.6: RUL prediction with 95% confidence interval for 3 test profiles.

It's crucial to note that the quality metrics provided solely consider the predictions made for the final data point in each test. This approach is chosen because the engine's performance might not deteriorate consistently throughout the entire data set. The engine could encounter an anomaly and start degrading at any point during its operation. Consequently, the accuracy of predictions made earlier in the mission timeline will likely exhibit more error, while the accuracy of the final prediction should be the most reliable since it corresponds to the moment when failure has occurred.

Figure 3.6 shows the predicted RUL along with its 95% confidence bounds for all time steps for a selected engine test profile. Each of the three chosen profiles highlights an important result. First, profile I.D. 33 shows a scenario where the maximum value of the true RUL being equal to 130 seems accurate. The model shows a predicted degradation in RUL around time step 75 in reaction to the sensor values, similar to the true RUL. Also, this engine I.D. shows a large uncertainty in the predicted RUL towards the beginning of the test case, but a smaller uncertainty as more information becomes available and ultimately high confidence in the final value. Lastly, this test case shows a downside to using the normal distribution.

Looking at the second row, profile 41 shows a scenario where the value of 130 for the maximum true RUL may not have been correct, because the model appears the predict degradation starts at around time step 80, instead of time step 35. Similarly, there is an increase in confidence of the predictions as more data becomes available throughout the test case. Lastly, profile 48 is the opposite of the last profile. Here, degradation appears to begin earlier than the true RUL denotes. In this case, the maximum RUL should likely be greater, allowing for degradation to begin earlier. Again there is an increase in confidence of the predictions as more data becomes available.

In order to validate these results, the proposed approach is compared with previous approaches presented in the literature. A summary of the validation results can be found in Table 3.3. Within Table 3.3, the first three columns from the left include the point prediction quality metrics. Each of these three metrics weight different aspects of the results differently. For example, SF punishes late predictions more than early predictions while MSE punishes equally. Based on the papers shown in Table 3.3, [57] showed the most success on both MSE, and SF, this means it was both

Approach	RMSE	PICP	NMPIW	CWB	SF	Accuracy	System-level Coverage
Lognormal LSTM [57]	**12.20**	0.950	0.316	**0.634**	**243.8**	**72%**	Yes
IESGP [143]	14.70	**0.995**	0.540	0.455	331.9	-	No
LSTM BS [150]	14.48	0.960	0.377	0.583	481.1	-	No
CNN [153]	18.45	-	-	-	-	-	No
RNP [151]	-	0.870	-	-	-	-	No
TSCG [155]	17.43	-	-	-	468.5	-	No
ESN-MVE [156]	-	0.890	0.670	0.22	-	-	No
PGRU model [157]	12.39	0.870	**0.309**	0.561	-	-	No
DLSTM [158]	18.33	-	-	-	655	-	No
BiLSTM-ED [159]	14.73	-	-	-	273.0	57%	No
DBNBP-IPF [160]	-	-	-	-	543.0	51%	No
Multi-loss LSTM (This Thesis)	17.01	0.910	0.376	0.536	422.8	64.5 %	**Yes**

Table 3.3: Comparison on point prediction and uncertainty of the multi-loss LSTM results from this thesis with other state-of-the-art methods.

accurate and conservative.

With respect to the proposed approach, the RMSE metric performed worse relative to the SF and the accuracy. This shows the model performed better than the average in most scenarios, however, there are outlier cases which raise the RMSE. In fact, if RMSE was computed using the median error, the RMSE would have been 12.40 which compares well with the current literature. Also, our SF performed well which shows that the model predicted failure earlier rather than later more often. This conservative nature of prediction is important in scenarios with regarding failure.

With respect to the evaluation of uncertainty in the prediction, the final three columns from the left in Table 3.3 are used. As shown, the model proposed in [143] shows the greatest PICP, while [157] has the lowest NMPIW. However, it is important to note that a strong PICP should be ignored if the NMPIW is high, because this shows that the uncertainty is very large so most predictions will encapsulate the true value. For example, while [143] has the best PICP, the NMPIW is very high which explains that the models predictions have large uncertainties which makes the test values fall between the 90% confidence range. For these reasons, the proposed CWB metric is used to measure this trade off. As shown, [57] has neither the best PICP or NMPIW metric, but has the best CWB. It is of note that the proposed approach is the only method compared which enables system-level coverage, not just for a singular component.

With respect to the proposed multi-loss LSTM, the performance was up to par with the existing literature. The PICP metric calculated for the last point in all test sets is up to par with the existing literature, while still having a relatively low NMPIW. As shown in Table 3.3, our PICP was above 90%, as well as having a NMPIW below 40%, with a CWB over 50%. When predicting SRUL, uncertainty management is extremely important, because a large uncertainty in the predicted components' RULs can translate to extremely large uncertainties in the predicted SRUL. Therefore, developing a model with strong uncertainty measures was an important step in validating the CLA.

As a whole, the results of the C-MAPSS case study validates the component-level analysis portion of this proposed SRUL prediction algorithm. As shown through the comparisons, many have used probabilistic deep learning for RUL prediction. The novelty of the approach lays in the integration of this method with the system-level approach. Our methods simple output, a normal distribution, combined with its flexible training methodology, allowing for different values of λ based on different applications, make it a preferred method for SRUL prediction as compared to the existing approaches. Due to C-MAPSS data set only supplying data for a single component, it is not possible to test the system-level. In order to validate the entire approach, a case study on an unmanned surface vessel will be conducted in the following section.

3.5 USV Case Study

To demonstrate the applicability of the proposed architecture, an ECCS of a USV is studied. The case study uses a simulation model, developed in [11], to create operational data for the ECCS subsystem with stochasticity for both input and output data.

3.5.1 System Description

A USV consists of five major subsystems, the ECCS, the power system, the navigation system, the data acquisition system, and the communication system [5]. The ECCS is a vital subsystem in a USV responsible for regulating and maintaining the optimal temperature of the marine diesel engine [161]. It comprises several key elements, including track control devices, situational awareness devices, and the propulsion subsystem. The propulsion subsystem encompasses various components such as the marine diesel engine, lube oil pump, seawater pump, and freshwater pump. These components are equipped with sensors that monitor their performance and provide essential data. In this particular case study, the USV is expected to navigate through the North Pacific Ocean for surveillance purposes, covering a distance of approximately 2,400 nautical miles. The duration of this sailing mission is estimated to be around 200 hours. The simulation incorporates environmental uncertainty and assumes that no human intervention occurs through the entirety of the mission.

Within this subsystem, there are any interdependence's between components. For example, the marine diesel engine relies on cooling and control components for propulsion, and as it operates it generates significant heat, which can harm performance and component health. To counter this, the ECCS components need to rely on each other. The ECCS comprises four cooling subsystems: air, freshwater, lube oil, and seawater. These subsystems extract heat from the engine, with freshwater also cooling the lube oil. The air and freshwater are then cooled by seawater, which ultimately dissipates the heat into the environment. This highlights the interconnectedness and mutual dependence of these components in maintaining optimal engine performance. These relationships make this a great case study of the developed approach, as any degradation in one component will cause others to suffer. Figure 3.7 shows a block flow diagram of the ECCS subsystem. As shown in Figure 3.7, a multitude of sensors including temperature, pressure, shaft speed and more, monitor the performance of various components at different locations. From these sensors, the condition monitoring data can be generated.

3.5.2 Data Generation

In order to test the model using the ECCS, a physics-based simulation of the subsystem, developed in [11] using the Simulink environment provided by MathWorks®[162] is used. This simulation takes in an input, a given throttle profile relating to the operating operation of the USV, and includes uncertainty in environmental profiles in order to generate condition monitoring data of 56 sensors measuring values such as temperatures and pressures belonging to 4 main components in the ECCS: diesel engine, lube-oil pump, freshwater pump, and seawater pump.

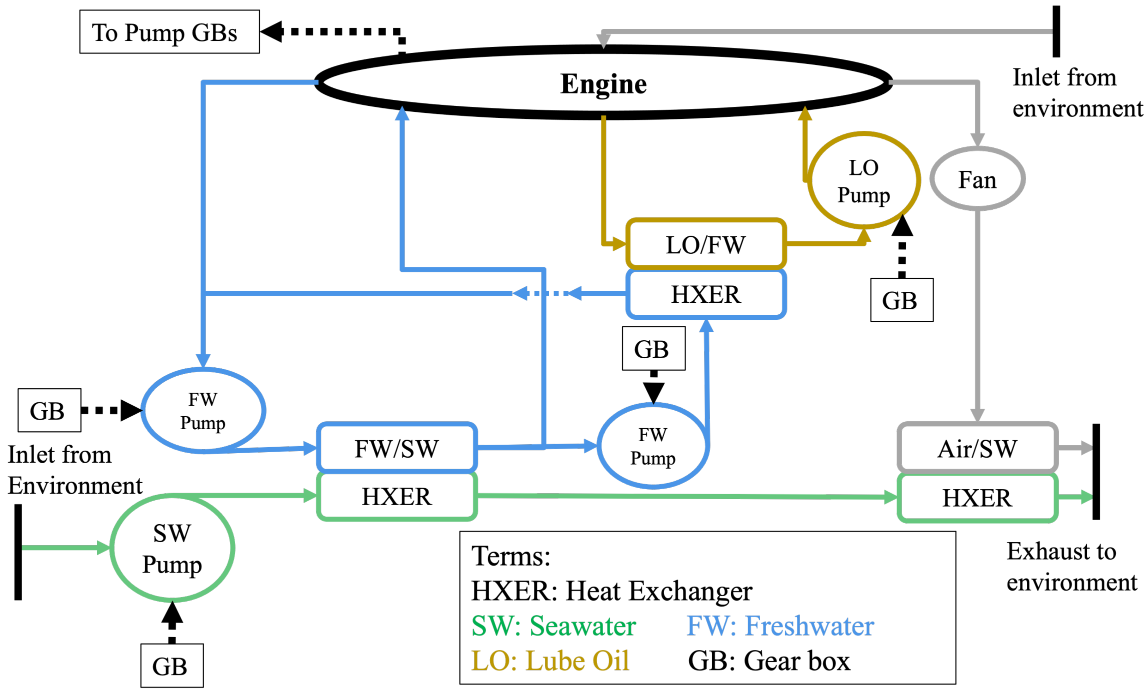


Figure 3.7: Block diagram of ECCS.

In order to mimic the USVs operation through the data generation process, 200 hours of data is generated for each input profile, assuming the vessel traveling from San Diego to Hawaii, or one mission. As the goal of this approach is to provide health prognosis during real operation and to correctly respond to anomalies, the amount of data generated and used to train the models is a critical decision, as it drastically affects the performance during testing. An extensive and balanced data set was generated using the ECCS simulation including all possible conditions experienced due to environmental uncertainties. To generate the training data, a Monte Carlo simulation is used to randomly generate input profiles including air temperature, sea temperature, and air pressure profile. Each component in the simulation is also randomly given different starting health states with none lower than 20% to ensure that no component is failed before the mission starts. 36 GBs of data and 96,000 total missions were simulated. 20% of the data was randomly selected for testing and the rest was used for training.

3.5.3 Component-level Analysis

3.5.3.1 Data Pre-processing

The simulation consists of 56 sensors which monitor a variety of measurements such as: inlet and outlet temperatures, flow rates, and pressures, shaft speeds, and filter openings, among others. The simulated data comes from an accurate operational environment with built in stochasticity, thus noise contamination exist in the data set. Many of these measurements come on different magnitudes, thus, to train a PDL model, it is necessary to pre-process the data following the methodology in Section 3.3.1.2.

RUL Target Generation The operational data obtained from the simulation is right-censored. As discussed in Section 3.3.1.2, with right-censored data RUL targets need to be constructed based off of available efficiency or reliability values. The simulation outputs sensors which measure the efficiency of each component for all time steps. For each component, an η_{crit} is defined as 20%. This number is defined based on the design of the simulation model and denotes the efficiency in which failure is declared. As mentioned, the simulation starts with each component at different health states meaning their $\eta_{t=0}$ is not the same for each trial in order to include diverse data, however, no components start failed. Therefore, the EOL can be denoted as the time in which the efficiency crosses this threshold. However, because the mission-time is limited to 200 hours, the efficiency of the components may not cross this threshold during the simulation. For these components, the procedures from Section 3.3.1.2 for right censored data in which this failure threshold was not reached, must be used.

For the components in the ECCS subsystem, excluding the diesel engine and lube-oil, freshwater, and seawater pumps, condition monitoring data is not available. Therefore, historical failure rates must be used to construct the target. These failure rates for the ECCS components which don't have simulation data are taken from NPRD [163] OREDA [164], and NSWC [165] and are shown in Table 3.4. Using these failures rates, MTTF is calculated assuming an exponential distribution for each component. Using Equation 3.4, the MTTFs are used as the EOL in order to

Failure event	Failure rate (h^{-1})
Air cooling subsystem failure	8.526×10^{-5}
Contra–rotating rudder propeller failure	8.933×10^{-4}
Coupling failure	1.086×10^{-6}
Diesel engine failure	1.537×10^{-4}
Freshwater pump failure	2.388×10^{-5}
Fuel pump failure	1.429×10^{-4}
Gear box failure	1.268×10^{-6}
lube oil pump failure	2.403×10^{-5}
Motor controller failure	4.753×10^{-6}
Propulsion monitoring subsystem failure	1.738×10^{-4}
Propulsion protection subsystem failure	1.738×10^{-4}
Rudder angle controller failure	3.102×10^{-6}
Rudder angle sensor failure	6.239×10^{-6}
Seawater pump failure	1.002×10^{-5}
Shafts failure	1.248×10^{-5}
Signal amplifier failure	1.095×10^{-5}
Speed sensor failure	1.248×10^{-5}

Table 3.4: Failure rates for the ECCS components [163–165].

construct RUL targets for the remaining components.

Feature Selection In order to reduce computation time and improve efficiency of the PDL model, feature selection is performed. Before this step, there are 56 sensors monitoring a variety of measurements. Following the methodology from Section 3.3.1.2, a heat map was constructed for the diesel engine and lube-oil, freshwater and seawater pumps' RUL and all sensor values. In order to capture inter dependencies between components, all features from the entire system are used for all components.

For this case study, any sensors with the absolute value of their correlation values under 0.1 were removed.

Data Normalization To complete the data pre-processing portion of the methodology, all condition monitoring data was normalized using Equation 3.29.

	Diesel Engine	Lube-Oil Pump	Freshwater Pump	Seawater Pump
Number of Features	45	38	42	47
λ	0.7	0.7	0.7	0.7
Learning rate	.01	.01	.01	.01
Number epochs	20	20	40	40
Number LSTM layers	3	3	3	3
Dropout	0.2	0.2	0.2	0.2

Table 3.5: Number of features and hyper parameter values for ECCS components' models.

3.5.3.2 Probabilistic Deep Learning Model Development and Training

In order to complete the testing part of the methodology, a separate PDL model for each component in the ECCS case study was constructed. The models were trained using the data generated from the 96,000 simulated missions. The computing platform used was a desktop computer with an Intel® Core™ i9-12900KF processor.

Due to the large size of the data set, the model's architecture was made to be less complex in order to reduce computational expense. The number of epochs was decreased for computational sake, and the learning rate was increased in order to expedite learning in this shorter amount of epochs. Table 3.5 shows the hyper parameters chosen for each component's PDL model. Comparing number of features in the model with its hyper parameters in Table 3.5, there was no noticeable trend between the size of the model and the length of training epochs, versus the amount of features in which the model was trained with. However, the trend of increasing amount of epochs was positively correlated to the mean remaining useful life of the component's target function.

3.5.4 System-level analysis

In order to create the equation node-based BN, the ECCS model is adapted from [11] and Equations 3.27 and 3.28 from Section 3.3.2 are utilized. The failure rates for components which do not have condition monitoring data are taken from Table 3.4.

Figure 3.8 depicts the equation node-based BN used for the ECCS case study. Within the BN, there are 37 nodes. The nodes, "Lubricating Oil pump", "Diesel Engine", "Sea water Pump", and "Fresh water pump" have condition monitoring data available and their RULs will be predicted

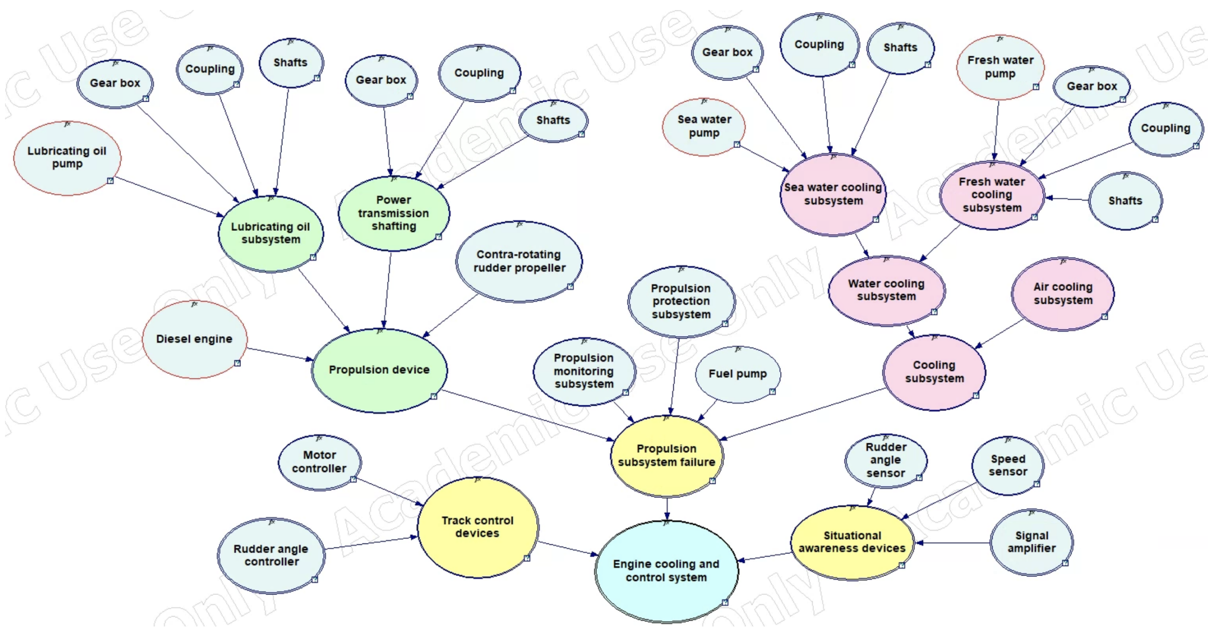


Figure 3.8: Bayesian network of engine cooling and control subsystem.

using the component level analysis methodology. The other nodes RULs are estimated using their historic failure rates given in Table 3.4. The BN is updated at the same sampling rate was the condition monitoring data, every hour. At each time step, the BN updates the nodes based on the information provided by the deep learning model. This information is used to determine the SRUL for each time step, but also keep a memory of all points from that entire test case in order to plot the SRUL over the course of the entire mission.

3.5.5 Results

In this section, the results for the ECCS case study will be presented. A discussion of key findings and performance according to the proposed component-level quality metrics will be presented. Due to this data set not being used by any other literature sources for development of a prognostic model, the results can not be directly compared to existing literature in a quantitative manner. However, the system-level approach will be measured using the qualitative metrics discussed in Section 3.3.2.3.

	Diesel Engine	Freshwater Pump	Seawater Pump	Lube-Oil Pump
RMSE	521.9	28371.3	18350.6	59023
NRMSE	0.109	0.130	0.122	0.227
PICP	0.944	0.962	0.970	0.995
NMPIW	0.553	0.516	0.645	0.666
CWB	0.391	0.446	0.325	0.329

Table 3.6: Average quality metrics across all test cases for each component with condition monitoring data.

3.5.5.1 Component-Level Results

Table 3.6 shows a summary of the quality metrics used to characterize the results of each approach for component-level analysis. As shown by the table, the value of the RMSE changes significantly for each component. This is due to the average RUL being much higher for those components which causes a large difference in RMSE even when the percent in which the two predictions are off by are the same. This illustrates that RMSE is not a strong quality metric to use when comparing approach, because it is dependent on the scale data set. However, the proposed NRMSE is used to get around this flaw by normalizing the RMSE by the mean. The NRMSE shows there is an average of about 10-15% error for the mean prediction between the four components with condition monition data. Also, the larger the RUL value which is being predicted, which can be shown in the trend of RMSE, the larger the error in the point predictions, or NRMSE. This shows that the model is more accurate when the components are not healthy and in danger of failing, as compared to when the RUL is high and there is no danger of failure.

The performance of the uncertainty metrics of the model can be judged using the PICP and NMPIW results from Table 3.6. The CWB metric captures the trade off between PICP and NMPIW. As shown by the high values for the NMPIW for all components, the large number of features causes a larger uncertainty in the prediction. The prediction interval coverage percentage is very high for all components, including the diesel engine, and freshwater, seawater and lube oil pumps (PICP = 0.944; 0.962; 0.970; 0.995;), respectively, with a good enough score of the width of 95% RUL prediction interval (NMPIW = 0.553; 0.516; 0.645; 0.666), respectively.

Metric	Qualitative Value	Reasoning
Age of system failure information	Up-to-date	This methodology used a physics-based approach for data generation, therefore, the data is as up to date as the physics-of-failure modeling. As for historical failure rates used, the data comes from OREDA [164].
Cost-effectiveness of information	Economical	All information used for this approach was available prior or generated via a digital twin. Because no new tests were run, this approach is highly economical.
Ease of model modification	Semi-adaptable	In order to allow the model to respond to new failures, no new time/effort/material is needed in the system-level portion of the analysis, because the approach is data-driven. However, historical failure rates are needed for components which do not have condition monitoring data available, therefore, these may need to be updated if more information becomes available. Also, the BN nodes need to be update manually. For this reason, semi-adaptable is given.
Framework coverage of known failure modes	Comprehensive	The model is applicable for the comprehensive amount of system failures/accidents/degradation.
Framework explainability	Fully explainable	While the component-level is a black box, the system-level is fully explainable and obtained using known statistical methods which have been explained in Section 3.3.2.
Framework traceability	Fully explainable	The structure and nodes for the Bayesian network are based on system knowledge. The BN clearly shows how data is used and connected to the problem. The failure rates used are used from well-trusted sources. Finally, the input data is taken from the component-level analysis.
Availability of data collection sources	Available	This case study uses a digital twin to generate data, therefore, all sensors and other data sources are fully available.
Data collection costs	Minimal	The data collected for this case study comes from a digital twin, therefore the time needed to generate the data is minimal compared to real time.
Data coverage	Full	Data was generated for all scenarios, including components starting at different health states, environmental conditions, etc..)
Required data storage capacity	Prohibitive	To ensure full data coverage, about 30 GBs of data was used to train a model for as many scenarios as possible.
Processing equipment requirements	Minimal	Computing hardware is needed in order to process the data collected from the simulation and make it usable for training and testing. The only computational step needed is to normalize the data which is computationally inexpensive.
Processing explainability	Fully explainable	The processing procedure is simple and fully explained in Section 3.5.3.1
Time required for assessment	Fast	While training the models is time intensive, testing the models with input data is very fast and can be assessed faster than the data sampling rate.
Expertise required for model use	Moderate	Knowledge of the system is needed in order to create the structure of the Bayesian network. The other steps require no expertise, only knowledge related to software use.
Model output interpretability	Meaningful	The final output is the SRUL predicted distribution which can be directly used in determining the time for the next maintenance.

Table 3.7: Results of qualitative metrics for evaluating system-level analysis methodology.

3.5.5.2 System-Level Results

Table 3.7 shows a comprehensive evaluation of the system-level approach with respect to the metrics discussed in Section 3.3.2.3. The table lists the metric along with the chosen qualitative value and the reasoning. The horizontal lines separate the metrics in the following groups: system information, constructing framework, data collection, data pre-processing, system-level analysis, and system management. The metrics are specially discussed for the approaches application to the ECCS case study

To summarize, the methodology yields up-to-date information on system failures and is cost-effective. It is semi-adaptable, as it can respond to new failures but may require updates for components without condition monitoring data. The framework is comprehensive, fully explainable, and traceable, with available data sources and minimal processing equipment requirements. However, it has prohibitive data storage needs. It offers a fast assessment, requires moderate expertise for model use, and provides meaningful output in the form of SRUL predicted distributions for maintenance planning.

Figure 3.9 shows one example of how the system-level approach is used to combine several component RULs into a single SRUL through the entirety of one test mission. The top graph shows all component-level RULs, including those with and without condition monitoring data. As shown, different components have RULs of different magnitudes, making components with large RULs hardly contribute to the SRUL. The bottom graph shows the subsequent SRUL prediction mean and 90% confidence interval. The predicted SRUL decreases by about 200 hours through the 200 hour mission, which is logical in the case of no faults. This figure also shows an important artifact of the model; if one RUL is significantly lower or higher, depending on the system logic, than the others, then it will dominate the SRUL calculation.

3.6 Discussion

3.6.1 Impact of λ on Component Level RUL Analysis Results

As shown in Equation 3.18, the choice of λ has great influence on the PDL model training process during the component RUL analysis. In this section, this influence is further investigated and discussed through a case study.

In order to understand the impact of the parameter and to choose its optimal value, a small grid

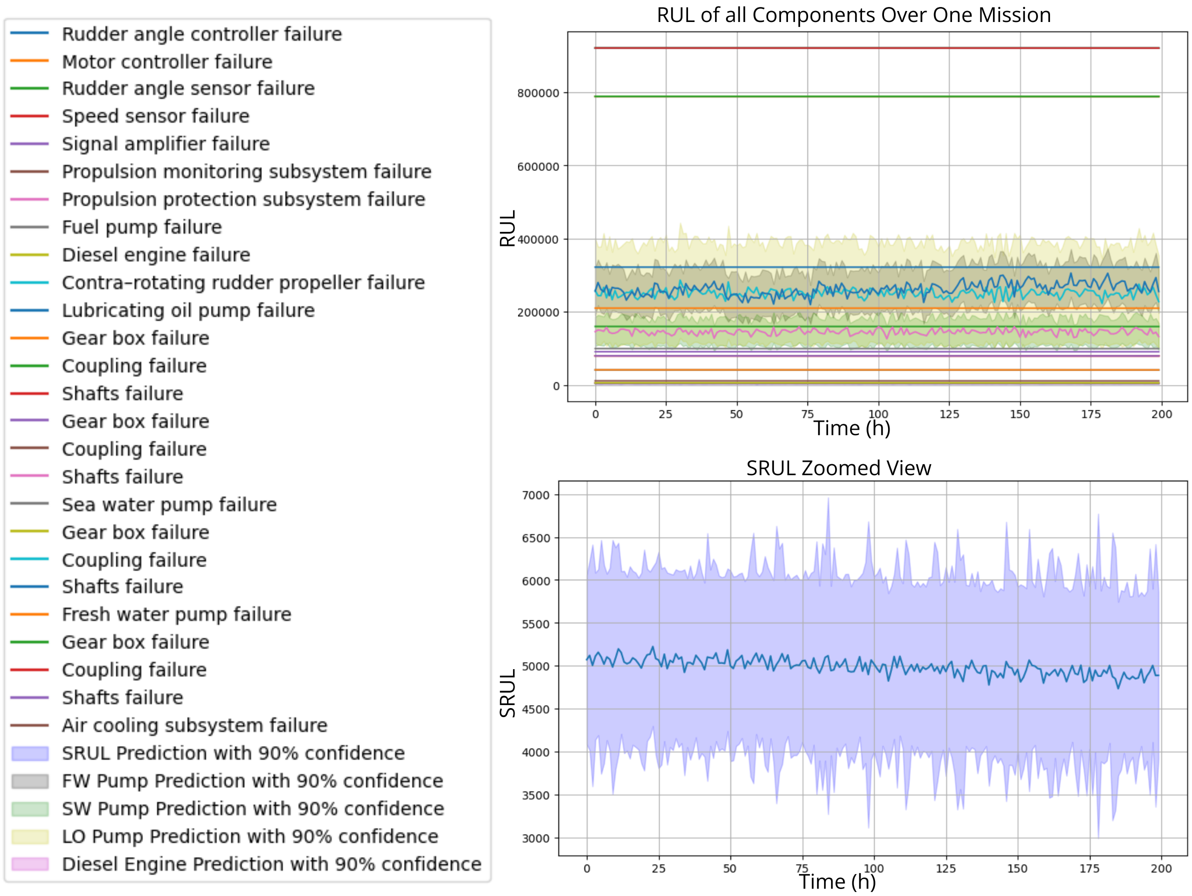


Figure 3.9: Component RUL predictions and calculated SRUL over one test mission.

search was performed using the proposed metrics. All other hyper parameters were held constant and λ was changed between 0 and 1 at an interval of 0.1. The constant hyper parameters included a dropout of 0.25, hidden size of 200, 100 epochs, 0.001 learning rate and 2 layers. These were not the same hyper parameters chosen for the case study, however, a less time-expensive architecture. Table 3.8 shows the results of this grid search for the same C-MAPSS data set discussed in Section 3.4. Due to the built in stochastic nature of training deep learning models, the values are the average taken from several training runs.

The value λ takes into account the MSE and NLL losses during training. While MSE takes into account only the point prediction, NLL takes into account both the point prediction and uncertainty. For this reason, it is expected λ will be greater than 0.5.

As shown in Table 3.8, varying λ from 0 to 1 has a significant impact on our quality metrics. When λ is set to 0, the model predominantly emphasizes minimizing the point prediction error,

λ	RMSE	NRMSE	PICP	NMPIW	CWB
0.0	21.8	0.292	0.091	0.034	0.057
0.1	23.1	0.310	0.667	0.249	0.418
0.2	22.2	0.298	0.899	0.495	0.404
0.3	21.7	0.291	0.768	0.374	0.394
0.4	21.2	0.284	0.838	0.409	0.429
0.5	21.5	0.289	0.788	0.369	0.419
0.6	20.4	0.273	0.848	0.396	0.452
0.7	21.1	0.284	0.797	0.365	0.432
0.8	22.1	0.298	0.910	0.501	0.409
0.9	22.2	0.298	0.910	0.467	0.443
1.0	21.1	0.284	0.899	0.450	0.449

Table 3.8: Effect of weight parameter for Loss Function, λ, on Quality Metrics for the C-MAPSS data set.

which is reflected in a low RMSE and NRMSE. However, this extreme focus on point prediction comes at the cost of neglecting uncertainty estimation, resulting in a very small prediction standard deviation. Consequently, this leads to poor scores in PICP and NMPIW. Conversely, as λ approaches 1, the model places more emphasis on learning the uncertainty associated with predictions. This leads to better calibration of prediction intervals, resulting in higher scores for PICP and NMPIW. However, this may come at the expense of slightly higher RMSE.

The significance of tuning λ lies in its ability to strike a balance between prediction accuracy and uncertainty quantification. Therefore, the characteristics of the studied system and its operations should be considered during the choice of λ value. For example, in the ECCS case study small uncertainty is valued more, because there are many components and many large uncertainties will lead to a very large bounds for the SRUL. This case a larger value of λ is preferred.

3.6.2 Utilization for Near Real-time RUL Prediction

Some of the key areas where this methodology can be applied include defense, aerospace, shipping, energy, and healthcare. The methodology allows improvements in system reliability and performance through enabling proactive maintenance and resource optimization. However, there are several factors which must be validated before the model can be used in near real-time applications with confidence. These factors include how fast the information is updated, and the

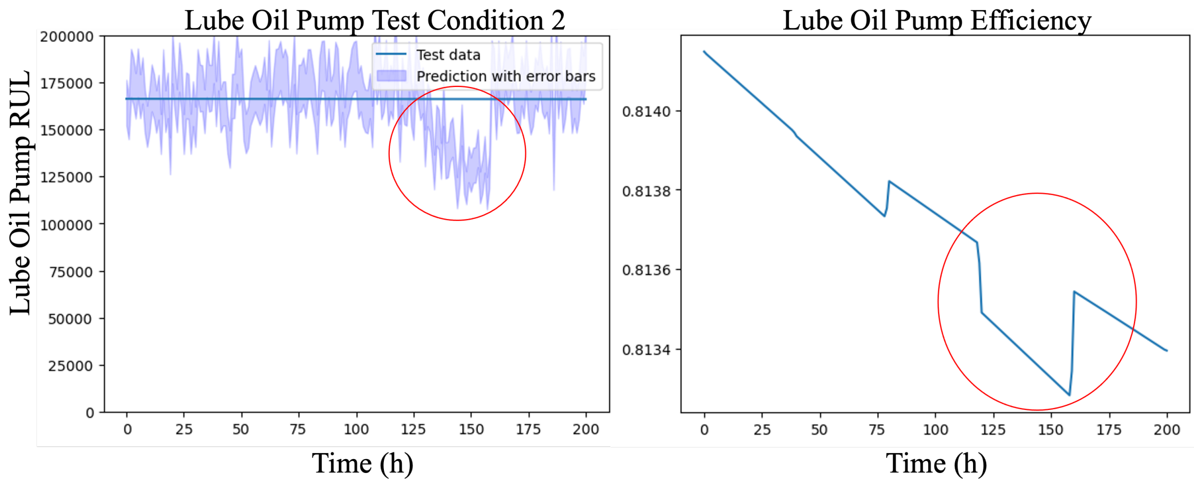


Figure 3.10: Fault Prognosis Validation - Lube Oil Pump

frequency at which it is received, and its ability to react to unforeseen faults. This section discusses these factors with respect to the proposed model to determine its applicability in the near real-time setting.

The proposed approach's time complexity depends mostly on the PDL model, which depends on several factors, including the architecture of the model, the size of the input data, and the hardware used for inference. When testing a model, this means a forward pass is run. According to the PyTorch documentation from [139], the time complexity for a bidirectional forward pass is given by $O(4h(5+h))$, where h is the input size. Because this time complexity is only squared, there is very little computational expense as there is no need to sample at frequencies greater than 1 Hz anyways for most health monitoring applications. This allows for information to be updated as quickly as every second, which is crucial for near real-time use.

An important step to validate the approach for near real-time use is to ensure that when faced with unforeseen failures or sudden decreases in efficiency, the models respond accordingly. In order to test this, several test cases were altered to include random and sudden faults. Figure 3.10 shows an example case where the efficiency of the lube oil pump was dropped suddenly in order to simulate a sudden deterioration in health. As shown in the left side of the figure, the model in turn predicts a lower RUL. This test case was not included in the training data which gives confidence that the model is able to extrapolate properly in unfamiliar conditions.

3.7 Concluding Remarks

In this chapter, a new methodology to predict the remaining useful life with uncertainty of a multi-component dependent system was presented. This architecture combines operation data-driven PRA and PHM for CESes. It uses operational condition monitoring data of components as the input to a multi-loss LSTM regression model which predicts the RUL distribution, using the assumption that the component's RUL follows a non-negative truncated normal distribution. Then, the system's architecture is used to create an equation node-based Bayesian network to equate each component's' RUL into a system RUL. This scheme is expected to be used for decision-making, such as optimized maintenance planning.

The performance of the component-level analysis, which consisted of the deep learning model, was measured and validated versus current literature using the C-MAPSS data set. The performance of the model was quantified by both point-prediction accuracy, using the mean of the predicted distribution, as well as the uncertainty accuracy, using the spread of the distribution. The performance was compared to the literature in order to validate it for system-level use.

Finally, the entire methodology was verified and tested through a case study on the engine cooling and control subsystem from an unmanned surface vessel. The results were able to give the most conservative SRUL and the architecture could be implemented on various systems as long as data from operation, maintenance and system design exists. The performance of the methodology was described using quantitative and qualitative metrics. The model offers cost-effective, up-to-date, and comprehensive system failure information, with semi-adaptability to new failures and meaningful maintenance planning output, although it has prohibitive data storage requirements.

This study is an important step to creating a comprehensive health monitoring system for complex engineering systems. While this approach is only prognosis, with more data and computational models, the approach can incorporate health diagnosis and a decision-support system to optimize system performance by connecting to SIPPRA model [68]. An important note to be made is an artifact of the model is that if a certain component's RUL is much greater or much less, depending on system structure, than the other components, the SRUL will resemble this component's RUL. However, the authors believe this is a reasonable artifact, because it is consistent with reliability calculations with similar scenarios.

This RUL prediction algorithm is only the beginning of the plans for this thesis. This information is vital to the operation of the system at hand, however, what is being done with it? In the future, the goal is to implement a decision-support system. This system will use the predicted RUL distribution among many other variables such as environmental conditions, reliability, cost, and others, in order to optimize operation of the system and maintenance planning at the component and system-level. This decision support system will enable action items based on the output of the RUL prediction algorithm, closing the loop.

3.8 Summary

This chapter described in detail a methodology for RUL prediction at the component and system-level while considering uncertainties and interdependencies. This chapter also validated the methodology at the component-level using a case study of the C-MAPSS data set, and also validated the entire methodology, component and system-level, using a case study of an ECCS from a USV. The next chapter describes a framework to optimize this predicted SRUL, closing the loop, and validates it using the ECCS of a USV case study.

Chapter 4: A RUL-Based Optimization Framework for Planning Operational Profiles with an Application to Unmanned Systems

This chapter is focused on creating and developing the methodology for RUL optimization. This includes an introduction and problem definition, followed by an in-depth description of the methodology, and finally a case study used to validate the developed model.

4.1 Introduction

Chapter 3 explained the criticality of CESes, and why health prognostic methods were crucial in preventing failures. However, in the case a failure, fuel leakage, or any other mission-threatening event does occur, operating the system to maximize lifetime until maintenance can be performed is critical. Operating the CES in a way that extends the lifetime can prevent catastrophic failures or loss of life. A concrete example would include a spacecraft, on a long-duration mission, equipped with a complex propulsion system, and a critical component, the fuel pump, begins to show signs of malfunction. This framework would generate an operational profile, where it uses minimal fuel for basic course adjustments allowing it to complete its primary mission or at least remain functional until a potential maintenance mission can be launched. This scenario highlights the importance and need for a systematic method of optimizing the operational profile to maximize lifetime, which is adaptable and resourceful, to overcome unexpected challenges during extended missions.

This chapter aims to propose a framework that integrates RUL prediction with an optimization algorithm, enabling utilization of health information for operational decision-making. Additionally, the study seeks to validate the method by comparing results to another optimization technique.

This methodology maximizes the predicted RUL from Chapter 3 utilizing a genetic algorithm (GA) for offline operational planning. The study uses the S-metric selection evolutionary multi objective algorithm (SMS-EMOA) [166] technique to cross verify the results of the GA. Finally, the applicability for near real-time optimization using MPC is explored, addressing all shortcomings of the discussed literature. This chapter demonstrates the application of the approach using an ECCS case study from a USV.

Moreover, the contributions of this chapter is not the application itself, but the addition of the optimization framework to the SRUL prediction method from Chapter 3. This optimization framework considers maximization of the RUL in a bi-objective optimization problem, enabling utilization of health information for operational decision-making.

The structure of this chapter is as follows: this chapter builds off of background information on current methods for RUL-based optimization, as well as methods used in the approach including a GA described in Section 2.2.3. First, Section 4.2 defines the assumptions needed to achieve the goals of this chapter, and formulates a corresponding optimization problem. This is followed by a description of the general algorithm used to optimize RUL in Section 4.3. Next, a case study using a simulation of an ECCS in a USV is considered in Section 4.4 using the proposed GA and SMS-EMOA methods, with the respective results. Finally, this is followed by a discussion of the results in Section 4.5, which includes an application of near real-time use and sensitivity analysis, and a conclusion in Section 4.6.

4.2 Problem Definition

4.2.1 RUL-based Operational Optimization Problem Definition

For the RUL-based operational optimization problem as described in Section 1.2.3, there are several assumptions which need to be made about the data input to and output from the final optimization model.

1. It is assumed that the system's operation can only be changed at N discrete time intervals. The operation level at each time interval will be the design variables for the optimization

problem, such as the throttle percentage of the engine for an interval in a USV. This arbitrary value N is chosen based on the problem, and is assumed to scale down the computational complexity. Obviously, if the throttle was optimized at each time step, the problem would be more accurate, but would take significantly longer to achieve convergence.

2. All systems begin the mission with no degradation in any component. This assumption is made in order to plan an operational profile applicable to the most general case.

These assumptions will help formulate the problem formally described in the following section.

4.2.2 RUL-based Operational Optimization Problem Formulation

In order to formulate the RUL-based operational optimization problem, each objective and constraint must be fully defined. This section describes the general objective and constraint functions considered in the bi-objective optimization problem which this thesis considers.

Objective 1 (with SRUL):

$$f_1(\mathbf{x},\mathbf{p}) = SRUL(RUL_i(\mathbf{x},\mathbf{p})) \tag{4.1}$$

This objective relates to the SRUL, which is aimed to be maximized. Here, let SRUL be a function of the i components' RULs. Each component's RUL is a function of certain parameters, $\mathbf{x}$, and uncertain parameters, $\mathbf{p}$. There are no bounds for this objective. This objective is referred to 'with SRUL', because it may not be plausible depending on the specific problem. For example, if not all components in the system have $\boldsymbol{CMD_c}$ available, then the SRUL may be dictated by components whose RULs are not dependent on the $\boldsymbol{CMD_c}$. In other words, the optimizer would not find a solution, because changing the inputs would not affect the SRUL prediction. If this is the case, an alternative to objective 1 is presented next.

Objective 1 (with RUL_i):

$$f_1(\boldsymbol{x},\boldsymbol{p}) = Min(SNR(RUL_i(\boldsymbol{x},\boldsymbol{p}))) \tag{4.2}$$

This objective relates to the component RULs which have condition monitoring data available. *SNR* represents the signal-to-noise-ratio, and its minimum value is aimed to be maximized. Each components' RUL, which has $\boldsymbol{CMD_c}$ available, is a function of certain parameters, $\boldsymbol{x}$, and uncertain parameters, $\boldsymbol{p}$. There are no bounds for this objective. This objective is used if the objective 1 with SRUL is not plausible, because it maximizes the limiting RUL of the system, which has $\boldsymbol{CMD_c}$ available.

Objective 2:

$$f_2(\boldsymbol{x}) = Min(\boldsymbol{x}) \tag{4.3}$$

This objective takes into account the operation of the system. $\boldsymbol{x}$ represents the throttle percentage of the system. Therefore, a higher throttle leads to a greater system output. This objective is conflicting with objective 1, because with greater operation, the system is expected to deteriorate at a greater rate, leading to a smaller RUL.

Constraint 1:

$$g_1(\boldsymbol{x},\boldsymbol{p}) : -R\dot{U}L_i + \dot{D}_{crit} \leq 0, i = 1,...,m \tag{4.4}$$

This constraint is aimed to limit the rate of degradation of each component. Let $R\dot{U}L_i$ be the degradation rate for a system with m components, i. In normal operation, this rate should be 1, because intuitively, the RUL will decrease proportionally to time. Therefore $\dot{D}_{crit}$ is used as a threshold to reduce the ability for the system to be operated at full throttle and cause a higher rate of degradation.

Constraint 2:

$$g_2(\boldsymbol{x},\boldsymbol{p}) : Min(\boldsymbol{x}) \geq Min(\boldsymbol{x})_{crit} \tag{4.5}$$

This constraint is created to avoid the possibility that the operation is set to the lowest possible value for the entire mission, which would most likely lead to the greatest lifetime. Let $Min(\boldsymbol{x})_{crit}$ be the minimum operational level allowed over the course of the mission. Let $Min(\boldsymbol{x})$ be the minimum operation during the mission. This value will be chosen based on the application, depending on the maximum allowable mission completion time.

Table 4.1: Environment profile uncertainties.

Environment	Upper bound		Lower bound	
	Start	Finish	Start	Finish
Air Temperature (°C)	22	34	10	20
Seawater Temperature (°C)	25	28	19	22
Air pressure (MPa)	0.102	0.102	0.1006	0.1006
Sea current (m/s)	1	1	0.5	0.5

Constraint 3:

$$\forall i, j \quad (x_i \neq x_j) \quad \text{for} \quad i \neq j, j = 1, ..., N \tag{4.6}$$

The final constraint states that all optimized design variables, the throttle percent x_i, can not all be equal to each other. For this problem, the mission is split between N different operation levels. This constraint is applied for computational ability of the simulation used to model the USV.

Uncertainty:

The final aspect of the optimization problem is the uncertainty in parameters in the optimization problem. These include the vector of environmental conditions, $\boldsymbol{p}$, which have a set range. The ranges of these variables is given in Table 4.1.

4.3 Methodology

This methodology is aimed towards solving the optimization problem described, defined and formulated in Sections 1.2.3, 4.2.1, and 4.2.2, respectively. In this bi-objective study, where the RUL objective is maximized, the NSGA-II algorithm is used [167]. Figure 4.1 shows an updated framework from Chapter 3 with the addition of the optimization process on the right side. The optimization methodology begins assuming that the component and system-level RUL prediction methodologies have been completed. The framework is split into 3 main sections: inputs, optimization algorithm, and output or optimization results.

4.3.1 Inputs

The first step in the methodology is to collect the required inputs. The inputs to the optimization algorithm come directly from the methodology described in Chapter 3. These include the data

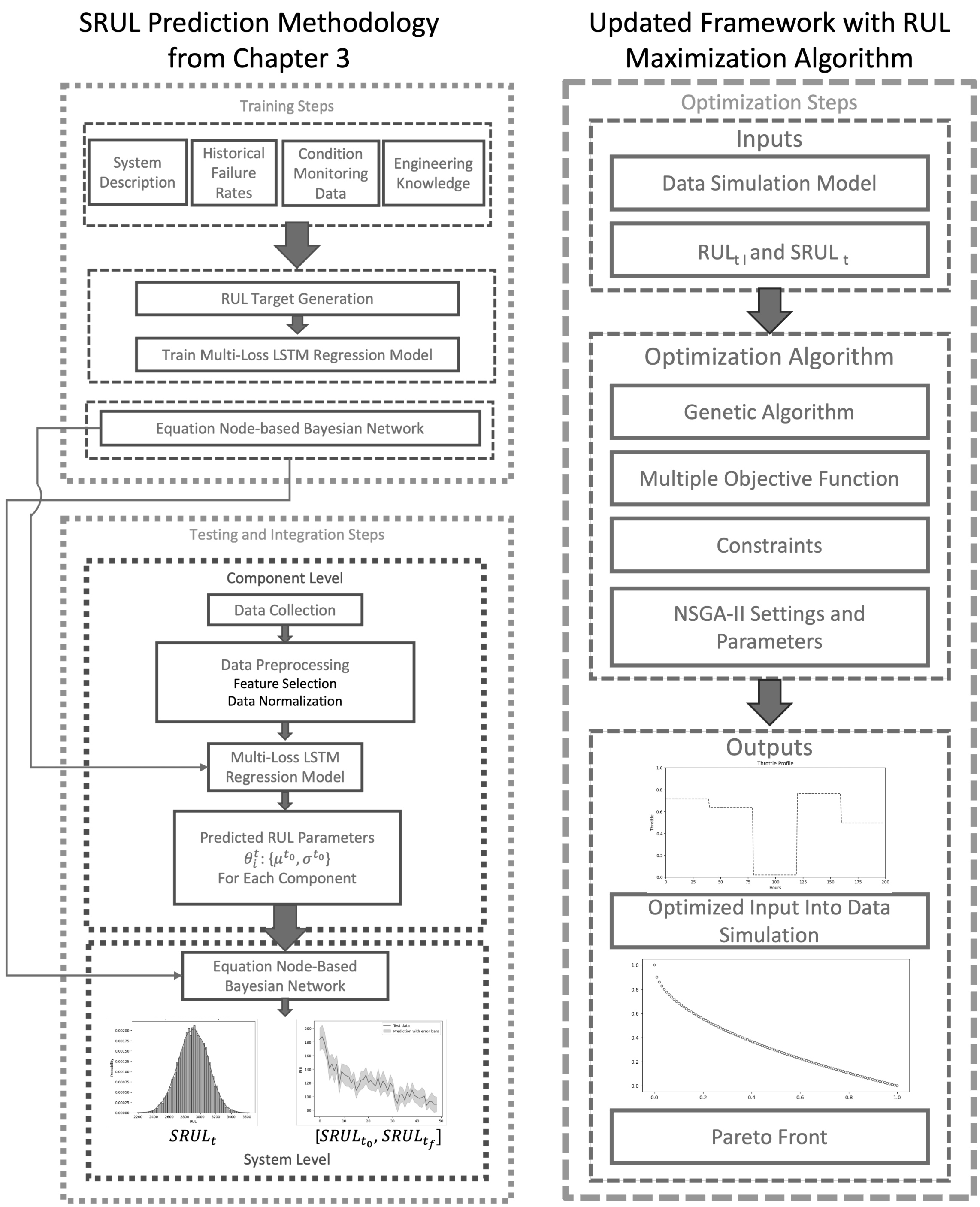


Figure 4.1: Complete Methodology for SRUL Prediction and Optimization.

simulation or generation model, the RUL prediction models and the SRUL prediction model.

The data simulation or generation model is a key aspect of the optimization algorithm. This model will be used as a function call during each iteration of the optimizer, therefore, computational time must be minimal. For this reason, this methodology uses a surrogate in the place of the data generation model. A surrogate model refers to a simplified or approximate representation of a more complex and computationally expensive simulation model, and are commonly used to reduce the computational cost associated with optimizing complex problems [168].

The next inputs into the optimization algorithm include the trained RUL prediction models from Chapter 3. Like the data simulation model, these models need to be computationally inexpensive. As shown in Section 3.6.2, the time complexity of one forward pass is $O(4h(5+h)$, where h is the input size, which can be simplified as $O(h^2)$. According to [169], this can be reduced to $O(h)$ through the inclusion of a population limit function in the selection process. Therefore, this model will be efficient in the optimization algorithm.

Finally, the last input in the framework is the SRUL prediction model. Based on the methodology explained in Section 3.3.2, the SRUL is calculated using an equation node BN which uses the set equations. Therefore, this model is time efficient in its calculation, and will be effective in the optimization algorithm.

4.3.2 Optimization Algorithm

The next section of the methodology includes the optimization algorithm. This section utilizes a NSGA-II approach for a bi-objective optimization problem formulated in Chapter 4.2.2. The rest of this section describes the specifics of the algorithm used to solve the formulated problem. This algorithm is well known in the literature, developed by [167].

4.3.2.1 NSGA-II: Algorithm & Parameters

The first subsection to discuss is the NSGA-II algorithm. NSGA-II is an optimization algorithm used to solve multiple objective problems, where the goal is to find a set of solutions that are not dominated by any other solution in the search space. NSGA-II was developed in [167] as an

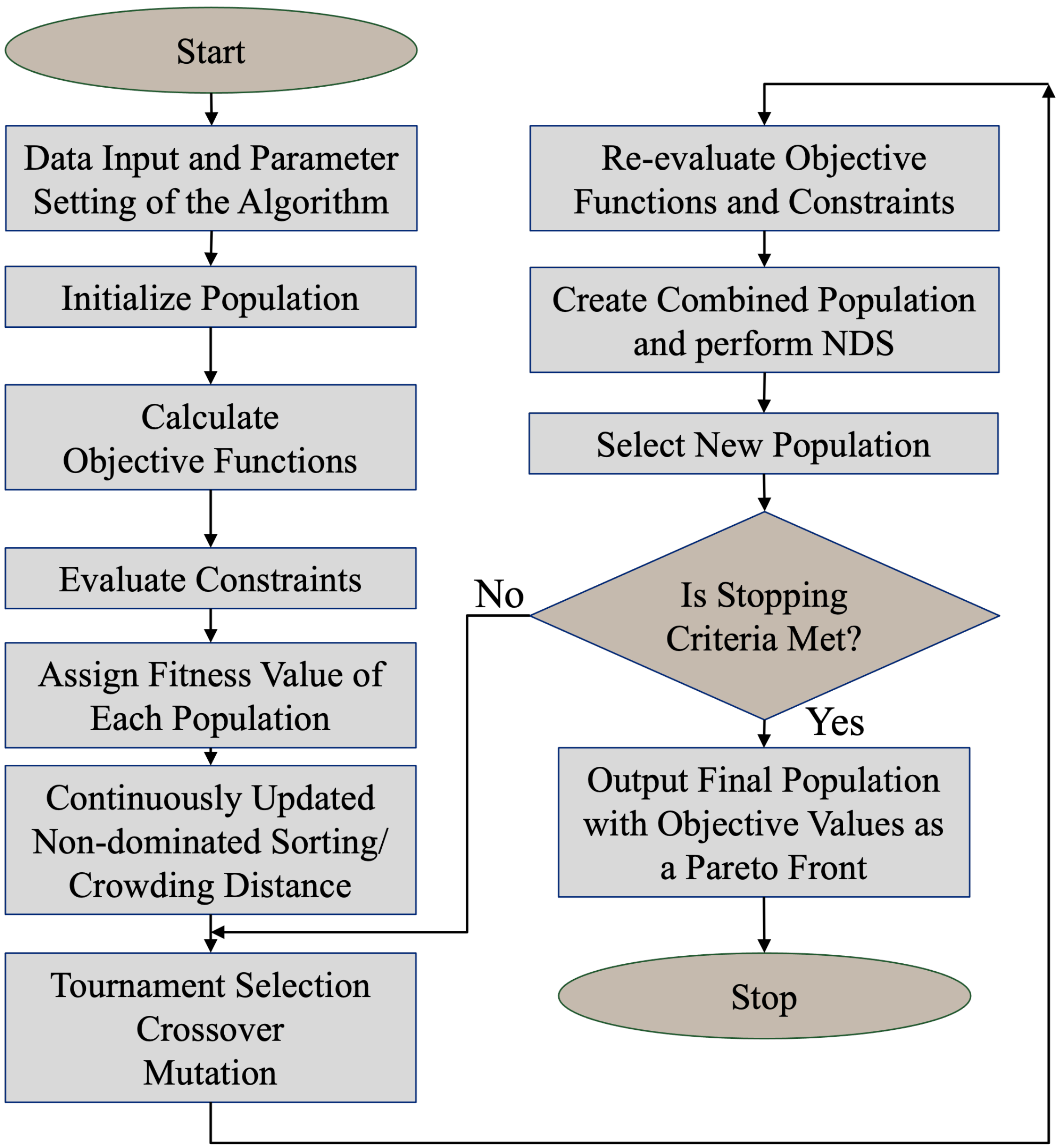


Figure 4.2: Flowchart for overall NSGA-II algorithm, as described by [167].

```
initialize P' = {1}, set i = 2
while ( i ≤ N)
        set j = 1 and K = |P'| (total number in P')
        set 'i is NOT dominated'
        while ( j ≤ K AND 'i is NOT dominated')
                compare i with P'(j)
                if P'(j) dominates i
                        set 'i is dominated'
                if i dominates P'(j)
                        P' = P' \ {P'(j)}*
                j = j + 1
        if i is NOT dominated
                update P' = P' U {i}
        i = i +1 (move to next point in P)
Declare P' as the non-dominated set
Return X_f
```

Note: A \ B means the set that contains all those elements of A that are not in B

Figure 4.3: Psuedocode of continuously updated non-dominated sorting algorithm, from [167].

extension of the original NSGA. Figure 4.2 shows a flowchart overview of the optimization algorithm described by [167]. The first step includes the method of choosing the specific parameters of the algorithm. However, these decisions are problem dependent, and an understanding of the parameters' purposes is needed for decision-making. Therefore, the steps of the algorithm will be described briefly.

Initialization: The first step is to randomly create an initial population of solutions. Each solution is represented as a set of decision variables. In this step, the only parameter which must be chosen is the population size, N. This methodology uses a rule-of-thumb starting point of $N \geq (10 \text{ to } 20) \times (\# \, of \ variables)$. Similar to hyper parameters in deep learning, these values can be changed during the optimization process to improve results. With each member of this initial population, the objectives and constraints are evaluated in order to determine a fitness value used in the next steps.

Sorting: Next, the initial population is sorted into fronts based on their fitness value. Fronts represent layers of non-dominated solutions, where the first front contains the non-inferior solutions. This specific methodology utilizes the continuously updated non-dominated sorting method [170]. Figure 4.3 shows the pseudo code for the continuously updated non-dominated sorting

(NDS) method. In this step, the crowding distance is also calculated for each individual within each front. This measures how close an individual is to its neighbors in order to maintain diversity in the subsequent selection.

Selection: The selection process, which is based on non-dominated sorting and crowding distance, employs a binary tournament selection to choose parents for crossover, and is used to select individuals for reproduction.

Crossover and Mutation: Crossover and mutation are applied to the selected individuals to create offspring. Crossover combines genetic information from two parents, and mutation introduces small random changes. This methodology uses the simulated binary crossover [171] operator for recombination. The respective operator is defined as follows:

$$\begin{aligned}
\beta_q &= \begin{cases} (2u)^{1/(1+\eta)}, & \text{if } u \leq 0.5 \\ (1/(2(1-u)))^{1/(1+\eta)}, & \text{otherwise} \end{cases} \\
c_{1q} &= 0.5\left[(1+\beta_q)u_1 + (1-\beta_q)u_2\right] \\
c_{2q} &= 0.5\left[(1-\beta_q)u_1 + (1+\beta_q)u_2\right]
\end{aligned} \tag{4.7}$$

where β_q is a scaling factor used in the crossover process. It determines the shape of the probability distribution function for selecting the offspring solutions. u, u_1, and u_2 are random numbers, c_{1q} and c_{2q} are two example offspring solutions generated through crossover, and η is the distribution index. In this step, there are two parameters which must be chosen. First being the probability that this crossover is applied during the evolution process. This value is chosen as 0.9, which is a common starting value [172]. Also, the value for η must be chosen. A large value of η gives a higher probability for creating near parent solutions and a small value of η allows distant solutions to be selected as children solutions [171].

The next step of the process is to perform mutation; this implementation uses polynomial mutation. The mutation operator is defined as:

$$y_q = x_q + \delta_q(U_q - L_q) \tag{4.8}$$

where y_q is the mutated variable, x_q is the current variable value, δ_q is a mutation strength, and U_q and L_q are the upper and lower bounds, respectively. The distribution index, η, is applied to introduce diversity, and plays a similar role as in crossover.

Termination: These steps are repeated with the new population replacing the initial population. If the stopping criteria is met, the algorithm stops, if not in continues. This algorithm uses the stopping criteria as the number of generations, which must be chosen based on the application.

This NSGA-II algorithm is implemented using the Python library `pymoo` [173]. In order to validate the results, this algorithm is run 10 times for the proposed NSGA-II approach, and all non-inferior solutions are used to evaluate the consistency.

4.3.2.2 Constraint Handling

In *Pymoo*, constraints are handled as a default using feasibility first (parameter-less approach). This process works in the following manner: prior to evaluating the fitness of a population member, you must first check if the solution is feasible. If that member is invalid, a penalty is applied to the function value [174]. The equation below demonstrates this method.

$$F(\boldsymbol{x},\boldsymbol{p}) = \begin{cases} f(\boldsymbol{x},\boldsymbol{p}), & \text{if } g_i(\boldsymbol{x},\boldsymbol{p}) \leq 0, \forall i \in I \\ f_{max}(\boldsymbol{x},\boldsymbol{p}) + \sum_{i=1}^{I}[g_i(\boldsymbol{x},\boldsymbol{p})], & \text{otherwise} \end{cases} \tag{4.9}$$

Here $f(\boldsymbol{x},\boldsymbol{p})$ represents the fitness prior to the addition of the constraint, and $F(\boldsymbol{x},\boldsymbol{p})$ is the penalized fitness function.

4.3.2.3 Fitness Function

In order to combine the bi-objective problem, with constraints, into a single fitness value, this implementation uses a weighted fitness function. This combined fitness function builds upon

Equation 4.9.

$$Fitness = \sum_{n=1}^{N} f_n \cdot w_n + \sum_{i=1}^{I} g_i \cdot w_i + \sum_{j=1}^{J} h_j \cdot w_j \tag{4.10}$$

4.3.2.4 Verification

Before implementing the described methodology on a case study, the algorithm was applied to a few complex, non-convex constrained bi-objective optimization problems. The NSGA-II methodology, as well as the verification process, SMS-EMOA, were tested on well known test problems ZDT [175], OSY [176], and TNK [177]. The results showed convergence to the theoretical Pareto front, which verifies that these algorithms are working as intended. Now that this process of the methodology has been verified, the next section will detail the case study using this thesis' methodology.

4.4 Case Study

4.4.1 Model Development

The optimization algorithm is applied to a similar simulation of an ECCS of a USV as detailed in Section 3.5 from [11], with a few changes. The main change involves the inclusion of external wave forces acting on the USV during travel. Also, the ECCS was expanded to include an additional freshwater, seawater, and lube-oil pump. This change means that there are now 7 components in the simulation which have condition monitoring data: a diesel engine and two freshwater, seawater, and lube-oil pumps.

All aspects of the data generation are the same as from Section 3.5. To summarize, the USV is assumed to complete a 200 hour mission from San Diego, California to Honolulu, Hawaii, while exposed to uncertain environmental and wave conditions. The USV is assumed to have 5 stages of operation, meaning over the 200 hour mission, the operation state, throttle, can be 5 different values, all for equal time.

In order to reduce the time needed for a function call of the simulation, a surrogate model of the ECCS simulation was developed using a K-nearest neighbors regression model from [178]. This surrogate was trained using simulated data, and obtained an R^2 of 1 of the training data, and a 0.961 on testing data using a 80/20 training testing split.

Using the updated USV simulation, a RUL prediction model was trained for each of the 7 components with $\boldsymbol{CMD_c}$ available. These were trained using the same methodology as Section 3.5.3. Also, the system-level BN was constructed using the same methodology as discussed in Section 3.5.4.

4.4.2 Bi-objective Problem Formulation

The final problem formulation which this section will solve is given below. In this problem, the alternative objective 1 with RUL_i is used. This is because the ECCS model only has 7 components with $\boldsymbol{CMD_c}$, and 25 components without. The SRUL was unaffected by changing the input conditions, because the lowest component-level RUL did not have $\boldsymbol{CMD_c}$ available, calling for the need of the alternative objective 1 , as discussed in Section 4.2.2.

$$
\begin{aligned}
\text{maximize} \quad & [f_1(\boldsymbol{x},\boldsymbol{p}) = Min(SNR(RUL_i(\boldsymbol{x},\boldsymbol{p})))] \\
\text{maximize} \quad & [f_2(\boldsymbol{x}) = Min(\boldsymbol{x})] \\
\text{subject to} \quad & g_1(\boldsymbol{x},\boldsymbol{p}) : -R\dot{U}L_i + 1.1 \leq 0, \qquad i = 1,\ldots,m \\
& g_2(\boldsymbol{x},\boldsymbol{p}) : Min(\boldsymbol{x}) \geq 0.2 \\
& \forall i,j \quad (x_i \neq x_j) \quad \text{for} \quad i \neq j, \qquad j = 1,\ldots,5
\end{aligned}
\tag{4.11}
$$

Note that $\dot{D}_{crit}$ is set to be 1.1, N = 5, and $Min(\boldsymbol{x})$ = 0.2.

4.4.3 Results

Using the two optimization methods, NSGA-II and SMS-EMOA, the optimization problem formulated in the last section was solved. In order to assess the convergence of the optimization methods, each method was ran a total of 10 times, collecting the final non-inferior front from

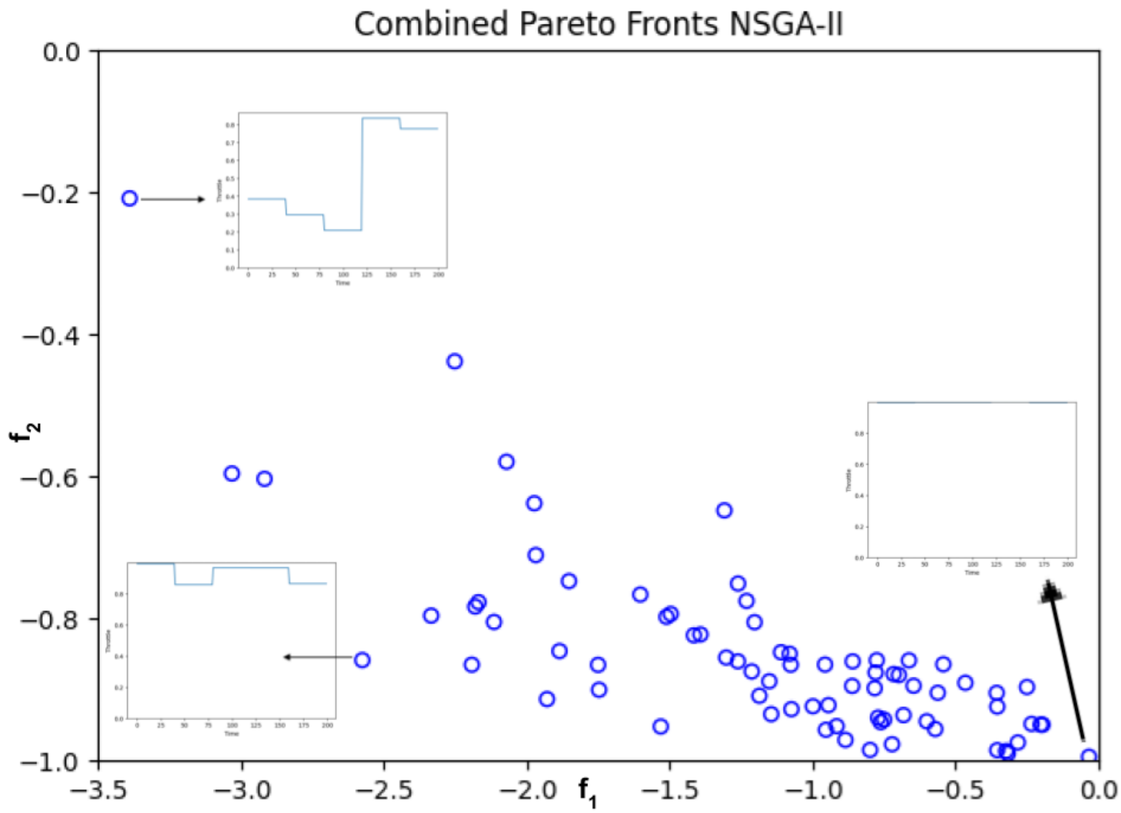


Figure 4.4: Combined non-inferior fronts from 10 NSGA-II runs.

each run. Finally, the non-inferior fronts were combined into a single population and plotted for analysis. Figure 4.4 shows the combined 10 non-inferior fronts for the NSGA-II methodology, and Figure 4.5 shows the combined 10 non-inferior fronts for the SMS-EMOA method for comparison. Also, each figure shows 3 resultant throttle profiles to the corresponding point in the objective space. These 3 were selected for visualisation based on the following criteria: one point with the minimum RUL objective value, one point with the minimum speed objective value, and one non-inferior point in between those. Table 4.2 shows these throttle values for the NSGA-II method, and Table 4.3 shows these throttle values for the SMS-EMOA method.

To obtain these results, the following parameters were used. The NSGA-II methodology used an initial population size of 100, 20 offspring, and a stopping criteria of 100 gen. The SMS-EMOA method had only one parameter, generations, which was set to 20. The parameters for the NSGA-II method were chosen using the steps described in the methodology section, while the SMS-EMOA method's number of generations was chosen to have a similar amount of time per optimization run.

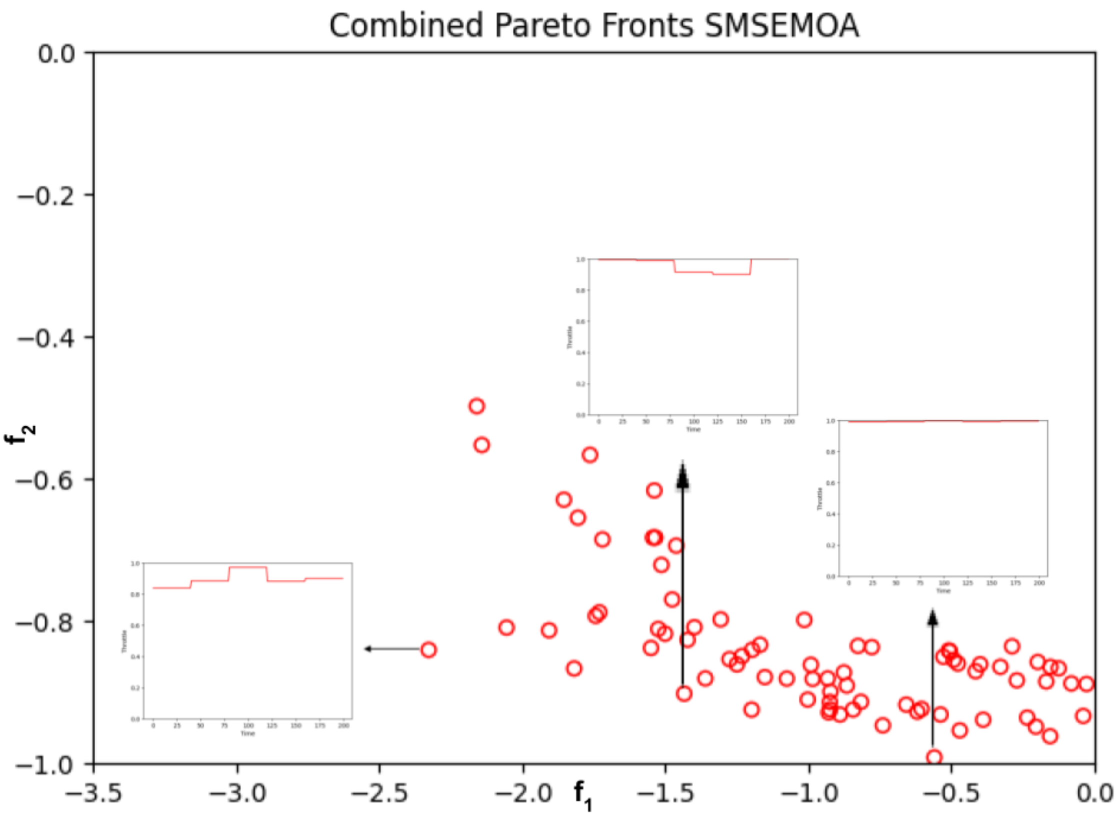


Figure 4.5: Combined non-inferior fronts from 10 SMS-EMOA runs.

	Highest RUL	Non-inferior Point In Between	Highest Speed
Throttle 1	0.384	0.993	0.985
Throttle 2	0.295	0.858	0.977
Throttle 3	0.208	0.968	0.992
Throttle 4	0.835	0.967	0.999
Throttle 5	0.776	0.865	0.994

Table 4.2: Throttle values from optimizer for 3 points shown in Figure 4.4.

	Highest RUL	Non-inferior Point In Between	Highest Speed
Throttle 1	0.840	0.996	0.991
Throttle 2	0.886	0.990	0.993
Throttle 3	0.972	0.917	0.997
Throttle 4	0.883	0.902	0.993
Throttle 5	0.902	0.999	0.996

Table 4.3: Throttle values from optimizer for 3 points shown in Figure 4.5.

The NSGA-II method took an average of 5 hours and 21 minutes per run. SMS-EMOA took an average of 5 hours and 7 minutes. As these times are similar, the results are used to validate that the optimizers did converge.

The x-axis of Figures 4.4 and 4.5 represents objective 1, and the y-axis is objective 2. Both plots use the same bounds in order to show their shared trend; objective 1 has a range of [-3.5, 0], and objective 2 has a range of [-1, 0]. Both methods have a similar density of points, however, the NSGA-II method has a greater spread, as it has points on its non-inferior front which are outside those from the SMS-EMOA method.

Both figures show a few of the non-inferior points in the design space, or the corresponding throttle profile. As expected, as points in the objective space move from the bottom right to the top left, the minimum throttle value decreases, causing a greater average SNR of RUL. The throttle corresponding to 100% for the entire simulation led to the smallest average SNR of RUL.

Figure 4.6 shows the outline of the best and worst non inferior points from the optimizer for each method. As the best and worst points of each method are very similar, this verifies that the optimization was the same, and that both methods worked as intended. The figure also clearly shows that the NSGA-II method has a greater spread.

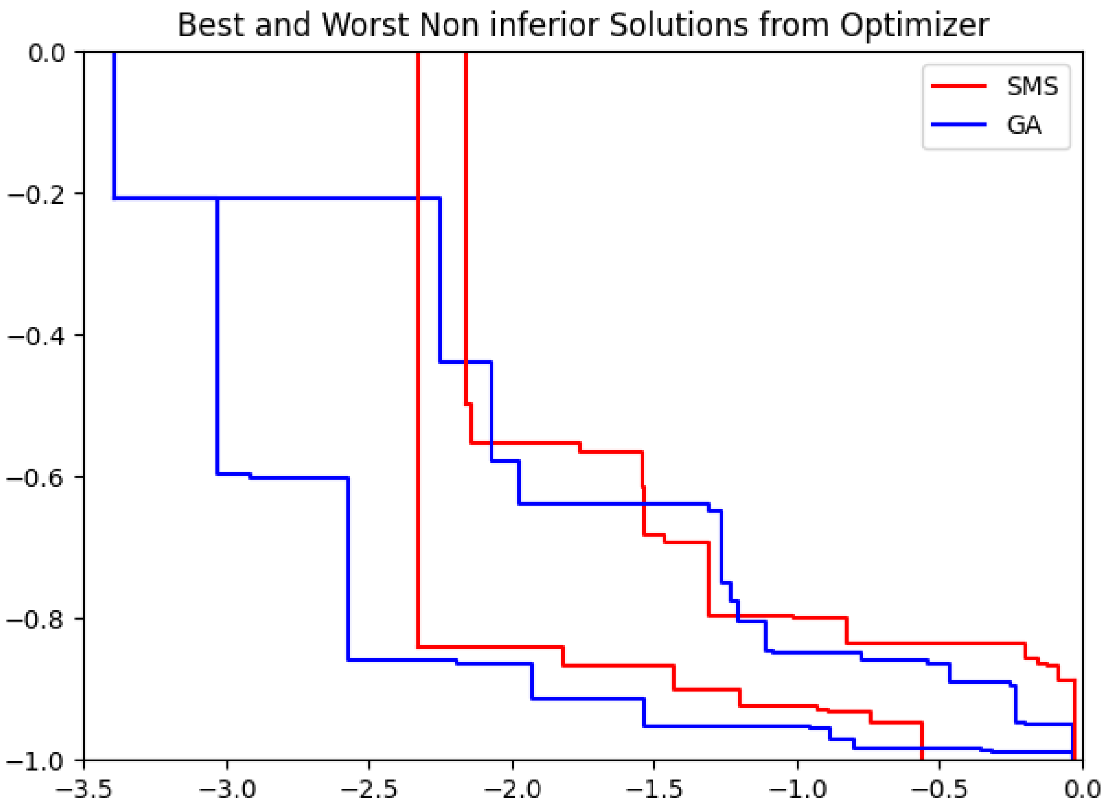


Figure 4.6: Best and Worst Non inferior Solutions from Optimizer

4.5 Discussion of Optimization Results

For the obtained results in Section 4.4.3 it is essential to discuss key aspects, including the nuances of the results and the differences between the two methods.

One expected outcome before the study was performed was that the throttles leading to the highest average RUL would be all low values. While both methods did find this pattern, neither method found points corresponding to the lowest possible throttle values, where constraint 2 was active. This may have been to to the use of the L_∞ norm, the minimum, rather than the L_1 norm which uses the sum of the inputs. Using the L_∞ norm led to results where the highest RUL were the throttle which had the largest difference between their minimum and maximum throttles, rather than minimizing the throttle at all time steps. These results are application based, and prior to setting up the problem should be determined.

The second item to discuss is the reasoning for difference in results between the NSGA-II method, and the comparison SMS-EMOA method. While NSGA-II demonstrates a greater spread, especially for RUL objective, its consistency appears comparatively inferior. This discrepancy may stem from the inherent characteristics of NSGA-II, which might prioritize exploration over exploitation, potentially leading to a broader but less refined spread of solutions. Conversely, SMS-EMOA, with its emphasis on maintaining a more consistent spread, might exhibit a narrower but more consistent distribution of non-inferior solutions. The trade off between exploration and exploitation strategies in both algorithms likely contributes to the observed distinctions, highlighting the importance of understanding algorithmic behaviors in complex optimization scenarios. There are many other factors which could have caused these results, such as the parameters chosen. Both optimization methods are stochastic processes, and their behavior can be random and difficult to analyze.

4.6 Concluding Remarks

In this chapter, the RUL prediction methodology from Chapter 3 was translated into impact through the development of an RUL-based optimization framework. This framework consisted of maximizing RUL within a bi objective study solved using a NSGA-II method, and compared to a SMS-EMOA method. Both methods were applied to a case study using an ECCS simulation from a USV, and their results were compared. This scheme closes the loop of the RUL prediction methodology, making it usable for prognosis and correction.

The performance of optimization algorithms were compared using a similar amount of time for each run, and a total of 10 separate runs' non-inferior points were combined into a single set. The results showed similarities in the layout of their non-inferior points, but also subtle differences which could be explained by the differences in the algorithms.

This chapter is an important step to creating a comprehensive health monitoring system for complex engineering systems. It combines the prognostic model developed in Chapter 3 with an optimization method, moving towards a complete integrated approach for health monitoring for complex engineering systems.

4.7 Summary

This chapter described in detail a methodology for RUL maximization at the component and system-level using a NSGA-II method. This chapter also validated the methodology using a case study of an ECCS from a USV.

The next chapter concludes the thesis, presenting a summary, contributions to the literature, and future work.

Chapter 5: Conclusion

5.1 Summary

This thesis presents a novel framework for system-level RUL prediction and optimization, considering uncertainties and component dependencies. The framework utilizes deep learning, a Bayesian network, and a genetic algorithm, as well as techniques from PHM and PRA connected via the SIPPRA framework. This research establishes the ability to optimize maintenance scheduling and decision-making from the predicted SRUL. The thesis presents an in-depth literature review which helpeed identify the gaps in the PHM field, specifically RUL prediction of CESes. This review also identified the need for a causal framework and model which fuses historical failure rates and condition monitoring data to predict and maximize RUL at the component- and system-levels with uncertainty. This need shaped the main objective of this thesis.

The proposed framework consists of two main parts: SRUL prediction and SRUL optimization. The SRUL prediction framework consisted of component-level analysis, where condition monitoring and historical data were fused using a multi-loss LSTM NN to predict RUL at component-level, and system-level analysis, where system knowledge was used to create an equation node Bayesian network to calculate the SRUL. The optimization framework solved a bi-objective optimization problem where component and system RULs are maximized with respect to the operation of the system.

In order to validate the proposed methodology, this thesis used three different case studies. The first case study evaluated the component-level RUL prediction framework using the C-MAPSS data, comparing against other methods in the literature. Next, a simulation of an ECCS of a USV was used to validate the entire SRUL prediction framework. Finally, a surrogate of the ECCS simulation was used to validate the optimization framework by maximizing component RULs in a bi-objective optimization problem.

The results from the component-level RUL prediction framework using the C-MAPSS dataset show that our approach is competitive with existing literature for this dataset. Additionally, all but one of these approaches are not applicable for SRUL prediction; however, the approach which is capable of SRUL prediction doesn't consider interdependencies between components. The ECCS case study validated the entire SRUL prediction framework. Finally, the optimization study showed convergence to a set of non-inferior solutions for the proposed bi-objective optimization problem.

5.2 Contributions to the Literature

The following are the main contributions this thesis makes to the literature.

1. A structured literature review and analysis of the gaps in the PHM field as applied to unmanned systems and SRUL prediction and maximization.

2. A new deep learning structure for component-level PHM, one that utilizes a hybrid-loss function multi-layer LSTM model to predict RUL with a given confidence interval, while also considering the complex interactions between components.

    - Developed PDL model trained using a mixture of MSE and NLL loss functions to predict RUL.
    - Developed methodology for data collection and pre-processing in order to train PDL model.
    - Validated the proposed methodology using a case study of the C-MAPSS dataset, and compared results to existing solutions.

3. For the first time in the literature, an equation node BN is used to integrate multiple component RULs into a single SRUL, which enabling SRUL prediction of complex engineering systems.

- The predicted component RULs are maximized using a genetic algorithm in a bi-objective study.

4. The thesis provides evidence for the methodology's ability to be used in near real-time applications.

This thesis carries significant real-world implications for the PHM field, particularly in the context of unmanned systems. The implications of these contributions extend beyond theoretical advancements, as the methodology proves applicable for near real-time applications. This work not only addresses current limitations in PHM but also provides a robust and versatile framework that can enhance the reliability and efficiency of predictive maintenance strategies in multi-component systems. As a result, this thesis contributes valuable tools and insights that have the potential to impact various industries relying on unmanned systems, including predictive maintenance and operational optimization.

5.3 Highlights of Results

The following are some highlights from the results of the three case studies included in this thesis

1. Case Study 1: C-MAPSS

   The results from the proposed component RUL prediction methodology using the C-MAPSS dataset were competitive against other methods in existing literature. Namely, the proposed methodology obtained an RMSE of 17.01, PICP of 0.910, and NMPIW of 0.376. These results gave confidence that this methodology is effective in predicting the RUL of a component. Additionally, only one of the approaches in existing literature is applicable for SRUL prediction. The proposed method overcomes the limitation of the only other system-level method (Lognormal LSTM approach), because the Lognormal LSTM approach does not consider dependencies in the system-level.

2. Case Study 2: ECCS of USV

Section 3.6.2 tested the RUL prediction model with unforeseen failures or sudden decreases in efficiency. An example case, where the efficiency of the lube oil pump was dropped suddenly in order to simulate a sudden deterioration in health, was shown and the model in turn predicted a lower RUL. This test case was not included in the training data which gives confidence that the model is able to extrapolate properly in unfamiliar conditions.

3. Case Study 3: Updated ECCS of USV

   The results of the bi-objective RUL-based optimization problem were obtained using NSGA-II and SMS-EMOA methodologies. Both methods were ran for comparable time frames, roughly 5 hours. The NSGA-II showcased a broader spread of solutions, particularly for the RUL objective, albeit with less consistency compared to SMS-EMOA. The results highlighted the trade off of the objectives, that a greater speed led to a lower RUL, and vice versa.

5.4 Limitations and Possible Future Directions

This section describes the limitations of the proposed SRUL prediction and optimization methodology, and possible future developments to improve them.

Limitations:

1. **Assumed RUL Distribution:** The proposed methodology assumes a parametric distribution for the RUL, such as truncated normal, log-normal or weibull. In cases where real time-to-failure data is not available, such as the USV case study, it is impossible to know which distribution would best fit the data, therefore, this is a weak assumption, but one that must be made.

2. **SRUL Resemblance:** The proposed methodology uses a BN to integrate component RULs into a system RUL, using the system logic. Based on the logic, it is possible that the SRUL will almost exactly equal one component's distribution, if that component's mean is significantly higher or lower than the other components. This is an artifact of the model, and simply a limitation.

3. **SRUL Optimization:** The proposed methodology planned to use SRUL as an objective to maximize in the optimization study. However, it became apparent that SRUL cannot be an objective if a majority of the components in the system do not have condition monitoring data available. This is because SRUL is dominated by the outlier RULs, and if this dominating RUL is not affected by the condition monitoring data, the optimizer's iterations will not affect the objective. For this reason, the minimum RUL of those components with condition monitoring data available is used as the objective.

Possible Future Directions: The following are some of the possible future improvements that can be made to the proposed SRUL prediction and optimization methodology.

1. **Near Real-time Optimization:** In future work, transitioning the bi-objective throttle profile optimization for maximizing speed and SRUL into a fault-triggered RUL maximization problem for near real-time health optimization is an exciting avenue. This involves integrating a reliable health prognostics model, as developed in Chapter 3, with a near real-time optimization algorithm capable of dynamically adjusting throttle profiles in response to faults. Model Predictive Control (MPC) is a common real-time optimization technique. This thesis investigated the ability of the models created in this thesis to be used in such a problem, and found it to be feasible. However, the solution will always be rather obvious; the system will always operate at the lowest possible speed, given the constraints, leading to the greatest RUL. A more complex problem will be included in the future work, however, the preliminary investigation gives confidence that this framework can be tweaked for near real-time use.

2. **Non Parametric PDL Model:** Currently, the methodology has only been investigated for parametric distributions of RUL. However, non-parametric distributions may be more accurate in certain applications.

3. **Limited System Logic:** The proposed methodology currently analyzes two system logic connections: series and parallel. The framework could be extended for other connections, including k-out-of-n systems or stand-by components.

4. **Maintenance Planning:** The current optimization application uses the operation of the system as the decision variables. This can be adjusted to optimize the maintenance planning based on the component and system RULs.

Bibliography

[1] Indranil Hazra, Arko Chattejee, Joseph Southgate, Matthew Weiner, Katrina Groth, and Shapour Azarm. "A Reliability-Based Optimization Framework for Planning Operational Profiles for Unmanned Systems". In: *Journal of Mechanical Design* (Oct. 2023), pp. 1–12. ISSN: 1050-0472. DOI: 10.1115/1.4063661.

[2] Indranil Hazra, Matthew J. Weiner, Ruochen Yang, Arko Chatterjee, Jospeh Southgate, Katrina M. Groth, and Shapour Azarm. "Prognostics and Health Management of Unmanned Surface Vessels: Past, Present, and Future". In: *Journal of Computing and Information Science in Engineering* (2024).

[3] Matthew J. Weiner, Ruochen Yang, Katrina M. Groth, and Shapour Azarm. "Probabilistic Deep Learning with Bayesian Networks for Predicting Complex Engineering Systems' Remaining Useful Life". In: *IEEE Transactions on Reliability* (Submitted in 2024).

[4] Ramin Moradi, Andrés Ruiz-Tagle Palazuelos, Enrique Lopez Droguett, and Katrina M Groth. "Toward a framework for risk monitoring of complex engineering systems with online operational data: A deep learning-based solution". In: *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability* (2020), p. 1748006X221079964.

[5] Chao Gao, Yongjin Guo, Mingjun Zhong, Xiaofeng Liang, Hongdong Wang, and Hong Yi. "Reliability analysis based on dynamic Bayesian networks: a case study of an unmanned surface vessel". In: *Ocean Engineering* 240 (2021), p. 109970.

[6] Ramin Moradi and Katrina M Groth. "Modernizing risk assessment: A systematic integration of PRA and PHM techniques". In: *Reliability Engineering & System Safety* 204 (2020), p. 107194.

[7] Mohammad Modarres. *Risk analysis in engineering: techniques, tools, and trends*. CRC press, 2006.

[8] George E Apostolakis. "How useful is quantitative risk assessment?" In: *Risk Analysis: An International Journal* 24.3 (2004), pp. 515–520.

[9] Jay Lee, Behrad Bagheri, and Hung-An Kao. "A cyber-physical systems architecture for industry 4.0-based manufacturing systems". In: *Manufacturing letters* 3 (2015), pp. 18–23.

[10] Mohammad Ali Farsi and Enrico Zio. "Industry 4.0: Some challenges and opportunities for Reliability Engineering". In: *International Journal of Reliability, Risk and Safety: Theory and Application* 2.1 (2019), pp. 23–34.

[11] Indranil Hazra, Arko Chatterjee, Joseph Southgate, Matthew J. Weiner, Katrina M. Groth, and Shapour Azarm. "A reliability-based optimization framework for planning operational profiles for unmanned systems". In: *Proceedings of the ASME International Design Engineering Technical Conferences & Computers and Information in Engineering (IDETC-CIE) Conference, August 20-23, 2023, Boston MA*. 2023.

[12] Enrico Zio. "Prognostics and Health Management (PHM): Where are we and where do we (need to) go in theory and practice". In: *Reliability Engineering & System Safety* 218 (2022), p. 108119.

[13] Moussa Hamadache, Joon Ha Jung, Jungho Park, and Byeng D Youn. "A comprehensive review of artificial intelligence-based approaches for rolling element bearing PHM: Shallow and deep learning". In: *JMST Advances* 1 (2019), pp. 125–151.

[14] Huixing Meng and Yan-Fu Li. "A review on prognostics and health management (PHM) methods of lithium-ion batteries". In: *Renewable and Sustainable Energy Reviews* 116 (2019), p. 109405.

[15] Austin D Lewis and Katrina M Groth. "A Dynamic Bayesian network structure for joint diagnostics and prognostics of complex engineering systems". In: *Algorithms* 13.3 (2020), p. 64.

[16] Nam-Ho Kim, Dawn An, and Joo-Ho Choi. "Prognostics and health management of engineering systems". In: *Switzerland: Springer International Publishing* (2017).

[17] Yaguo Lei, Naipeng Li, Liang Guo, Ningbo Li, Tao Yan, and Jing Lin. "Machinery health prognostics: A systematic review from data acquisition to RUL prediction". In: *Mechanical systems and signal processing* 104 (2018), pp. 799–834.

[18] Yaguo Lei. *Intelligent fault diagnosis and remaining useful life prediction of rotating machinery*. Butterworth-Heinemann, 2016.

[19] Hehua Yan, Jiafu Wan, Chunhua Zhang, Shenglong Tang, Qingsong Hua, and Zhongren Wang. "Industrial big data analytics for prediction of remaining useful life based on deep learning". In: *IEEE access* 6 (2018), pp. 17190–17197.

[20] Abhinav Saxena and Kai Goebel. "Turbofan engine degradation simulation data set". In: *NASA AMES prognostics data repository* 18 (2008).

[21] Joseph H Saleh. "Durability choice and optimal design lifetime for complex engineering systems". In: *Journal of Engineering Design* 19.4 (2008), pp. 377–400.

[22] Alessandro Ferrara, Michael Okoli, Stefan Jakubek, and Christoph Hametner. "Energy management of heavy-duty fuel cell electric vehicles: Model predictive control for fuel consumption and lifetime optimization". In: *IFAC-PapersOnLine* 53.2 (2020), pp. 14205–14210.

[23] Jin Wu and Shapour Azarm. "Metrics for quality assessment of a multiobjective design optimization solution set". In: *J. Mech. Des.* 123.1 (2001), pp. 18–25.

[24] Jasbir Singh Arora. *Introduction to optimum design*. Elsevier, 2004.

[25] Xiao Han, Zili Wang, Min Xie, Yihai He, Yao Li, and Wenzhuo Wang. "Remaining useful life prediction and predictive maintenance strategies for multi-state manufacturing systems considering functional dependence". In: *Reliability Engineering & System Safety* 210 (2021), p. 107560.

[26] Ingeborg de Pater and Mihaela Mitici. "Developing health indicators and RUL prognostics for systems with few failure instances and varying operating conditions using a LSTM autoencoder". In: *Engineering Applications of Artificial Intelligence* 117 (2023), p. 105582.

[27] Changyue Song, Kaibo Liu, and Xi Zhang. "Integration of data-level fusion model and kernel methods for degradation modeling and prognostic analysis". In: *IEEE Transactions on Reliability* 67.2 (2017), pp. 640–650.

[28] Panagiotis Loukopoulos, George Zolkiewski, Ian Bennett, Suresh Sampath, Pericles Pilidis, X Li, and David Mba. "Abrupt fault remaining useful life estimation using measurements from a reciprocating compressor valve failure". In: *Mechanical Systems and Signal Processing* 121 (2019), pp. 359–372.

[29] Kaibo Liu, Abdallah Chehade, and Changyue Song. "Optimize the signal quality of the composite health index via data fusion for degradation modeling and prognostic analysis". In: *IEEE Transactions on Automation Science and Engineering* 14.3 (2015), pp. 1504–1514.

[30] Chuanqi Guo and Ingrid Bouwer Utne. "Development of risk indicators for losing navigational control of autonomous ships". In: *Ocean Engineering* 266 (2022), p. 113204.

[31] Houman Hanachi, Jie Liu, Avisekh Banerjee, Ying Chen, and Ashok Koul. "A physics-based modeling approach for performance monitoring in gas turbine engines". In: *IEEE Transactions on Reliability* 64.1 (2014), pp. 197–205.

[32] P Baraldi, G Bonfanti, and Enrico Zio. "Differential evolution-based multi-objective optimization for the definition of a health indicator for fault diagnostics and prognostics". In: *Mechanical Systems and Signal Processing* 102 (2018), pp. 382–400.

[33] Haihong Pan, Zhiqiang Lü, Huimin Wang, Haiyan Wei, and Lin Chen. "Novel battery state-of-health online estimation method using multiple health indicators and an extreme learning machine". In: *Energy* 160 (2018), pp. 466–477.

[34] Sinda Rebello, Hongyang Yu, and Lin Ma. "An integrated approach for system functional reliability assessment using Dynamic Bayesian Network and Hidden Markov Model". In: *Reliability Engineering & System Safety* 180 (2018), pp. 124–135.

[35] J Ahmadzadeh F. & Lundberg. "Remaining useful life estimation: review". In: *International Journal of System Assurance Engineering and Management* 5 (Dec. 2014), pp. 461–474.

[36] Wenshuo Tang, Darius Roman, Ross Dickie, Valentin Robu, and David Flynn. "Prognostics and health management for the optimization of marine hybrid energy systems". In: *Energies* 13.18 (2020), p. 4676.

[37] Carlos Ferreira and Gil Gonçalves. "Remaining Useful Life prediction and challenges: A literature review on the use of Machine Learning Methods". In: *Journal of Manufacturing Systems* 63 (2022), pp. 550–562.

[38] Muntazir Abbas and Mahmood Shafiee. "An overview of maintenance management strategies for corroded steel structures in extreme marine environments". In: *Marine Structures* 71 (2020), p. 102718.

[39] Paul Paris and Fazil Erdogan. "A critical analysis of crack propagation laws". In: (1963).

[40] RG Forman. "Study of fatigue crack initiation from flaws using fracture mechanics theory". In: *Engineering Fracture Mechanics* 4.2 (1972), pp. 333–345.

[41] Viethung Nguyen, Jeevanand Seshadrinath, Danwei Wang, Sivakumar Nadarajan, and Viswanathan Vaiyapuri. "Model-based diagnosis and RUL estimation of induction machines under interturn fault". In: *IEEE Transactions on Industry Applications* 53.3 (2017), pp. 2690–2701.

[42] Yuri Bazilevs, Alison L Marsden, F Lanza di Scalea, Amit Majumdar, and Mahidhar Tatineni. "Toward a computational steering framework for large-scale composite structures based on continually and dynamically injected sensor data". In: *Procedia Computer Science* 9 (2012), pp. 1149–1158.

[43] Nak Yong Ko, Gyeongsub Song, Hyun Taek Choi, and Joono Sur. "Fault detection and diagnosis of sensors and actuators for unmanned surface vehicles". In: *2021 21st International Conference on Control, Automation and Systems (ICCAS)*. IEEE. 2021, pp. 1451–1453.

[44] Qingqing Zhai and Zhi-Sheng Ye. "RUL prediction of deteriorating products using an adaptive Wiener process model". In: *IEEE Transactions on Industrial Informatics* 13.6 (2017), pp. 2911–2921.

[45] Diyin Tang, Jinrong Cao, and Jinsong Yu. "Remaining useful life prediction for engineering systems under dynamic operational conditions: A semi-Markov decision process-based approach". In: *Chinese Journal of Aeronautics* 32 (2019), pp. 627–638.

[46] Ying Du, Chaoqun Duan, and Tonghai Wu. "Lubricating oil deterioration modeling and remaining useful life prediction based on hidden semi-Markov modeling". In: *Proceedings of the Institution of Mechanical Engineers, Part J: Journal of Engineering Tribology* 236.5 (2022), pp. 916–923.

[47] Yuhan Zhang, Ying Yang, He Li, Xianchao Xiu, and Wanquan Liu. "A data-driven modeling method for stochastic nonlinear degradation process with application to RUL estimation". In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 52.6 (2021), pp. 3847–3858.

[48] Ahmad BahooToroody, Mohammad Mahdi Abaei, Osiris Valdez Banda, Jakub Montewka, and Pentti Kujala. "On reliability assessment of ship machinery system in different autonomy degree; A Bayesian-based approach". In: *Ocean Engineering* 254 (2022), p. 111252.

[49] Andres Ruiz-Tagle Palazuelos, Enrique López Droguett, and Rodrigo Pascual. "A novel deep capsule neural network for remaining useful life estimation". In: *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability* 234.1 (2020), pp. 151–167.

[50] Sen Zhao, Yong Zhang, Shang Wang, Beitong Zhou, and Cheng Cheng. "A recurrent neural network approach for remaining useful life prediction utilizing a novel trend features construction method". In: *Measurement* 146 (2019), pp. 279–288.

[51] Lijie Zhang, Bin Wang, Xiaoming Yuan, and Pengfei Liang. “Remaining useful life prediction via improved CNN, GRU and residual attention mechanism with soft thresholding”. In: *IEEE Sensors Journal* 22.15 (2022), pp. 15178–15190.

[52] Mihaela Mitici, Ingeborg de Pater, Anne Barros, and Zhiguo Zeng. “Dynamic predictive maintenance for multiple components using data-driven probabilistic RUL prognostics: The case of turbofan engines”. In: *Reliability Engineering & System Safety* 234 (2023), p. 109199.

[53] Peihua Han, André Listou Ellefsen, Guoyuan Li, Finn Tore Holmeset, and Houxiang Zhang. “Fault detection with LSTM-based variational autoencoder for maritime components”. In: *IEEE Sensors Journal* 21.19 (2021), pp. 21903–21912.

[54] Ke Yang, Yong-jian Wang, Yu-nan Yao, and Shi-dong Fan. “Remaining useful life prediction via long-short time memory neural network with novel partial least squares and genetic algorithm”. In: *Quality and Reliability Engineering International* 37.3 (2021), pp. 1080–1098.

[55] Magnus Gribbestad, Muhammad Umair Hassan, and Ibrahim A Hameed. “Transfer learning for Prognostics and health Management (PHM) of marine Air Compressors”. In: *Journal of Marine Science and Engineering* 9.1 (2021), p. 47.

[56] Jinwoo Bae and Zhimin Xi. “Learning of physical health timestep using the LSTM network for remaining useful life estimation”. In: *Reliability Engineering & System Safety* 226 (2022), p. 108717.

[57] Khanh TP Nguyen, Kamal Medjaher, and Christian Gogu. “Probabilistic deep learning methodology for uncertainty quantification of remaining useful lifetime of multi-component systems”. In: *Reliability Engineering & System Safety* 222 (2022), p. 108383.

[58] Ahmad BahooToroody, Mohammad Mahdi Abaei, Osiris Valdez Banda, Pentti Kujala, Filippo De Carlo, and Rouzbeh Abbassi. “Prognostic health management of repairable ship systems through different autonomy degree; From current condition to fully autonomous ship”. In: *Reliability Engineering & System Safety* 221 (2022), p. 108355.

[59] Mohammad Mahdi Abaei, Robert Hekkenberg, Ahmad BahooToroody, Osiris Valdez Banda, and Pieter van Gelder. “A probabilistic model to evaluate the resilience of unattended machinery plants in autonomous ships”. In: *Reliability Engineering & System Safety* 219 (2022), p. 108176.

[60] André Listou Ellefsen, Emil Bjørlykhaug, Vilmar Æsøy, Sergey Ushakov, and Houxiang Zhang. “Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture”. In: *Reliability Engineering & System Safety* 183 (2019), pp. 240–251.

[61] Baoping Cai, Xiaoyan Shao, Yonghong Liu, Xiangdi Kong, Haifeng Wang, Hongqi Xu, and Weifeng Ge. “Remaining useful life estimation of structure systems under the influence of multiple causes: Subsea pipelines as a case study”. In: *IEEE Transactions on Industrial Electronics* 67.7 (2019), pp. 5737–5747.

[62] Jannie S Nielsen and John D Sørensen. “Bayesian estimation of remaining useful life for wind turbine blades”. In: *Energies* 10.5 (2017), p. 664.

[63] Hamed Khorasgani, Gautam Biswas, and Shankar Sankararaman. “Methodologies for system-level remaining useful life prediction”. In: *Reliability Engineering & System Safety* 154 (2016), pp. 8–18.

[64] Zhou Zhicai, Liu Dongfeng, and Shi Xinfa. “Research on combination of data-driven and probability-based prognostics techniques for equipments”. In: *2014 Prognostics and System Health Management Conference (PHM-2014 Hunan)*. IEEE. 2014, pp. 323–326.

[65] Yongzhi Zhang, Rui Xiong, Hongwen He, and Michael G Pecht. “Lithium-ion battery remaining useful life prediction with Box–Cox transformation and Monte Carlo simulation”. In: *IEEE Transactions on Industrial Electronics* 66.2 (2018), pp. 1585–1597.

[66] Dong Wang and Kwok-Leung Tsui. “Brownian motion with adaptive drift for remaining useful life prediction: Revisited”. In: *Mechanical Systems and Signal Processing* 99 (2018), pp. 691–701.

[67] Yingjun Deng, Anne Barros, and Antoine Grall. “Degradation modeling based on a time-dependent ornstein-uhlenbeck process and residual useful lifetime estimation”. In: *IEEE Transactions on Reliability* 65.1 (2015), pp. 126–140.

[68] Ramin Moradi, Sergio Cofre-Martel, Enrique Lopez Droguett, Mohammad Modarres, and Katrina M Groth. “Integration of deep learning and Bayesian networks for condition and operation risk monitoring of complex engineering systems”. In: *Reliability Engineering & System Safety* 222 (2022), p. 108433.

[69] Leonardo Ramos Rodrigues. “Remaining useful life prediction for multiple-component systems based on a system-level performance indicator”. In: *IEEE/ASME Transactions on Mechatronics* 23.1 (2017), pp. 141–150.

[70] Ferhat Tamssaouet, Khanh Tp Nguyen, Kamal Medjaher, and Marcos Eduardo Orchard. “System-level failure prognostics: Literature review and main challenges”. In: *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability* 237.3 (2023), pp. 524–545.

[71] Dariusz Mazurkiewicz, Yi Ren, and Cheng Qian. “Novel Approach to Prognostics and Health Management to Combine Reliability and Process Optimisation”. In: *Advances in Reliability and Maintainability Methods and Engineering Applications: Essays in Honor of Professor Hong-Zhong Huang on his 60th Birthday*. Springer, 2023, pp. 559–580.

[72] Lintao Wang, Wei Sun, Yangyang Long, and Xu Yang. “Reliability-based performance optimization of tunnel boring machine considering geological uncertainties”. In: *IEEE Access* 6 (2018), pp. 19086–19098.

[73] Taha-Hossein Hejazi, Hossein Badri, and Kai Yang. “A reliability-based approach for performance optimization of service industries: An application to healthcare systems”. In: *European Journal of Operational Research* 273.3 (2019), pp. 1016–1025.

[74] Chwail Kim, Semyung Wang, Ilhan Hwang, and Jonghyun Lee. “Application of reliability-based topology optimization for microelectromechanical systems”. In: *Aiaa Journal* 45.12 (2007), pp. 2926–2934.

[75] Dan M Frangopol and Kurt Maute. “Life-cycle reliability-based optimization of civil and aerospace structures”. In: *Computers & structures* 81.7 (2003), pp. 397–410.

[76] M Assid, Ali Gharbi, and A Hajji. “Joint production, setup and preventive maintenance policies of unreliable two-product manufacturing systems”. In: *International Journal of Production Research* 53.15 (2015), pp. 4668–4683.

[77] Hai Canh Vu, Phuc Do, Anne Barros, and Christophe Bérenguer. “Maintenance grouping strategy for multi-component systems with dynamic contexts”. In: *Reliability Engineering & System Safety* 132 (2014), pp. 233–249.

[78] Wenbin Wang. “A stochastic model for joint spare parts inventory and planned maintenance optimisation”. In: *European Journal of Operational Research* 216.1 (2012), pp. 127–139.

[79] Michael Pinedo. *Planning and scheduling in manufacturing and services*. Springer, 2005.

[80] Rim Abdallah, Anne-Lise Gehin, and Jean-Yves Dieulot. “Battery Lifetime Optimization in a Solar Microgrid”. In: *2022 IEEE 17th International Conference on Control & Automation (ICCA)*. IEEE. 2022, pp. 154–159.

[81] Noman Bashir, Hira Shahzad Sardar, Mashood Nasir, Naveed Ul Hassan, and Hassan Abbas Khan. “Lifetime maximization of lead-acid batteries in small scale UPS and distributed generation systems”. In: *2017 IEEE Manchester PowerTech*. IEEE. 2017, pp. 1–6.

[82] Yongjie Liu, Zhiwu Huang, Hongtao Liao, Chengzhang Lyu, Yanhui Zhou, Yun Jiao, Heng Li, Chao Hu, and Jun Peng. “A temperature-suppression charging strategy for supercapacitor stack with lifetime maximization”. In: *IEEE Transactions on Industry Applications* 55.6 (2019), pp. 6173–6183.

[83] Nathalie Herr, Jean-Marc Nicod, Christophe Varnier, Louise Jardin, Antonella Sorrentino, Daniel Hissel, and Marie-Cécile Péra. “Decision process to manage useful life of multi-stacks fuel cell systems under service constraint”. In: *Renewable energy* 105 (2017), pp. 590–600.

[84] Arash Heidari, Nima Jafari Navimipour, Mohammad Ali Jabraeil Jamali, and Shahin Akbarpour. “A hybrid approach for latency and battery lifetime optimization in IoT devices through offloading and CNN learning”. In: *Sustainable Computing: Informatics and Systems* 39 (2023), p. 100899.

[85] Pengfei Zhi, Yongshuang Qi, Weiran Wang, Haiyang Qiu, Wanlu Zhu, and Ye Yang. “A lifetime optimization method of new energy storage module based on new artificial fish swarm algorithm”. In: *Advances in Mechanical Engineering* 13.9 (2021).

[86] Xiaohu Zheng, Wen Yao, Yingchun Xu, and Xiaoqian Chen. “Complex satellite lifetime optimization based on Bayesian network reliability compression inference algorithm”. In: *IEEE Access* 7 (2019), pp. 175166–175185.

[87] Yuqiang Fu, Tao Yuan, and Xiaoyan Zhu. “Importance-measure based methods for component reassignment problem of degrading components”. In: *Reliability Engineering & System Safety* 190 (2019), p. 106501.

[88] Matilde D’Arpino and Massimo Cancian. “Lifetime optimization for a grid-friendly dc fast charge station with second life batteries”. In: *Dynamic Systems and Control Conference*. Vol. 59162. American Society of Mechanical Engineers. 2019, V003T18A007.

[89] Muhammad Bakr Abdelghany, Muhammad Faisal Shehzad, Davide Liuzza, Valerio Mariani, and Luigi Glielmo. “Optimal operations for hydrogen-based energy storage systems in wind farms via model predictive control”. In: *International Journal of Hydrogen Energy* 46.57 (2021), pp. 29297–29313.

[90] Adel A Abou El-Ela, Ragab A El-Sehiemy, Abdullah M Shaheen, Walaa A Wahbi, and Mohamed T Mouwafi. “A multi-objective equilibrium optimization for optimal allocation of batteries in distribution systems with lifetime maximization”. In: *Journal of Energy Storage* 55 (2022), p. 105795.

[91] Fatemeh Karimi Pour, Didier Theilliol, Vicenç Puig, and Gabriela Cembrano. “Health-aware control design based on remaining useful life estimation for autonomous racing vehicle”. In: *ISA transactions* 113 (2021), pp. 196–209.

[92] Hakim Mabed and Eugen Dedu. “Short and long term optimization for micro-object conveying with air-jet modular distributed system”. In: *Journal of Parallel and Distributed Computing* 144 (2020), pp. 98–108.

[93] Pablo García-Triviño, Luis M Fernández-Ramírez, Antonio J Gil-Mena, Francisco Llorens-Iborra, Carlos Andrés García-Vázquez, and Francisco Jurado. “Optimized operation combining costs, efficiency and lifetime of a hybrid renewable energy system with energy storage by battery and hydrogen in grid-connected applications”. In: *International Journal of Hydrogen Energy* 41.48 (2016), pp. 23132–23144.

[94] Javier A Moreno, Andrés Montoya, Jorge López-Puente, and Domingo Santana. “Lifetime optimization of solar central receivers via linear actuators”. In: *International Journal of Mechanical Sciences* 226 (2022), p. 107388.

[95] Bahman Ahmadi, Juan S Giraldo, Gerwin Hoogsteen, Marco ET Gerards, and Johann L Hurink. “A multi-objective decentralized optimization for voltage regulators and energy storage devices in active distribution systems”. In: *International Journal of Electrical Power & Energy Systems* 153 (2023), p. 109330.

[96] Yang Wang and Stephen Boyd. “Fast model predictive control using online optimization”. In: *IEEE Transactions on control systems technology* 18.2 (2009), pp. 267–278.

[97] Helge Langseth and Luigi Portinale. “Bayesian networks in reliability”. In: *Reliability Engineering & System Safety* 92.1 (2007), pp. 92–108.

[98] Andres Ruiz-Tagle, Austin D Lewis, Colin A Schell, Ernest Lever, and Katrina M Groth. “BaNTERA: A Bayesian Network for Third-Party Excavation Risk Assessment”. In: *Reliability Engineering & System Safety* 223 (2022), p. 108507.

[99] Shinyoung Kwag, Abhinav Gupta, and Nam Dinh. “Probabilistic risk assessment based model validation method using Bayesian network”. In: *Reliability Engineering & System Safety* 169 (2018), pp. 380–393.

[100] Ruochen Yang, Jens Einar Bremnes, and Ingrid Bouwer Utne. “Online risk modeling of autonomous marine systems: Case study of autonomous operations under sea ice”. In: *Ocean Engineering* 281 (2023), p. 114765.

[101] Hongrui Wang, Alfredo Núñez, Zhigang Liu, Dongliang Zhang, and Rolf Dollevoet. "A Bayesian network approach for condition monitoring of high-speed railway catenaries". In: *IEEE Transactions on Intelligent Transportation Systems* 21.10 (2019), pp. 4037–4051.

[102] Austin D Lewis and Katrina M Groth. "A comparison of DBN model performance in SIPPRA health monitoring based on different data stream discretization methods". In: *Reliability Engineering & System Safety* (2023), p. 109206.

[103] Shen Chen, Zhen Qi, Dehuai Chen, Liangfu Guo, and Weiwen Peng. "Investigation of Bayesian network for reliability analysis and fault diagnosis of complex systems with real case applications". In: *Advances in Mechanical Engineering* 9.10 (2017), p. 1687814017728853.

[104] Atma Ram Sahu and Sanjay Kumar Palei. "Fault analysis of dragline subsystem using Bayesian network model". In: *Reliability Engineering & System Safety* 225 (2022), p. 108579.

[105] Morteza Soleimani, Felician Campean, and Daniel Neagu. "Integration of Hidden Markov Modelling and Bayesian Network for fault detection and prediction of complex engineered systems". In: *Reliability Engineering & System Safety* 215 (2021), p. 107808.

[106] Philippe Weber, Gabriela Medina-Oliva, Christophe Simon, and Benoıˆt Iung. "Overview on Bayesian networks applications for dependability, risk analysis and maintenance areas". In: *Engineering Applications of Artificial Intelligence* 25.4 (2012), pp. 671–682.

[107] Georg Helbing and Matthias Ritter. "Deep Learning for fault detection in wind turbines". In: *Renewable and Sustainable Energy Reviews* 98 (2018), pp. 189–198.

[108] Changchang Che, Huawei Wang, Qiang Fu, and Xiaomei Ni. "Combining multiple deep learning algorithms for prognostic and health management of aircraft". In: *Aerospace Science and Technology* 94 (2019), p. 105423.

[109] Prasanna Tamilselvan and Pingfeng Wang. "Failure diagnosis using deep belief learning based health state classification". In: *Reliability Engineering & System Safety* 115 (2013), pp. 124–135.

[110] Jason Deutsch and David He. "Using deep learning-based approach to predict remaining useful life of rotating components". In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 48.1 (2017), pp. 11–20.

[111] Khanh TP Nguyen and Kamal Medjaher. "A new dynamic predictive maintenance framework using deep learning for failure prognostics". In: *Reliability Engineering & System Safety* 188 (2019), pp. 251–262.

[112] Xiang Li, Qian Ding, and Jian-Qiao Sun. "Remaining useful life estimation in prognostics using deep convolution neural networks". In: *Reliability Engineering & System Safety* 172 (2018), pp. 1–11.

[113] Min Xia, Xi Zheng, Muhammad Imran, and Muhammad Shoaib. "Data-driven prognosis method using hybrid deep recurrent neural network". In: *Applied Soft Computing* 93 (2020), p. 106351.

[114] Shuai Zheng, Kosta Ristovski, Ahmed Farahat, and Chetan Gupta. "Long short-term memory network for remaining useful life estimation". In: *2017 IEEE international conference on prognostics and health management (ICPHM)*. IEEE. 2017, pp. 88–95.

[115] Sourajit Behera and Rajiv Misra. “Generative adversarial networks based remaining useful life estimation for IIoT”. In: *Computers & Electrical Engineering* 92 (2021), p. 107195.

[116] Taotao Zhou, Te Han, and Enrique Lopez Droguett. “Towards trustworthy machine fault diagnosis: A probabilistic Bayesian deep learning framework”. In: *Reliability Engineering & System Safety* 224 (2022), p. 108525.

[117] Taotao Zhou, Laibin Zhang, Te Han, Enrique Lopez Droguett, Ali Mosleh, and Felix TS Chan. “An uncertainty-informed framework for trustworthy fault diagnosis in safety-critical applications”. In: *Reliability Engineering & System Safety* 229 (2023), p. 108865.

[118] Eyke Hüllermeier and Willem Waegeman. “Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods”. In: *Machine Learning* 110 (2021), pp. 457–506.

[119] Armen Der Kiureghian and Ove Ditlevsen. “Aleatory or epistemic? Does it matter?” In: *Structural safety* 31.2 (2009), pp. 105–112.

[120] Rong Zhu, Yuan Chen, Weiwen Peng, and Zhi-Sheng Ye. “Bayesian deep-learning for RUL prediction: An active learning perspective”. In: *Reliability Engineering & System Safety* 228 (2022), p. 108758.

[121] Yan-Hui Lin and Gang-Hui Li. “A Bayesian deep learning framework for RUL prediction incorporating uncertainty quantification and calibration”. In: *IEEE Transactions on Industrial Informatics* 18.10 (2022), pp. 7274–7284.

[122] Ikram Remadna, Labib Sadek Terrissa, Zeina Al Masry, and Noureddine Zerhouni. “RUL Prediction Using a Fusion of Attention-Based Convolutional Variational AutoEncoder and Ensemble Learning Classifier”. In: *IEEE Transactions on Reliability* (2022).

[123] Mujahid Tabassum, Kuruvilla Mathew, et al. “A genetic algorithm analysis towards optimization solutions”. In: *International Journal of Digital Information and Wireless Communications (IJDIWC)* 4.1 (2014), pp. 124–142.

[124] Austin Downey, Chao Hu, and Simon Laflamme. “Optimal sensor placement within a hybrid dense sensor network using an adaptive genetic algorithm with learning gene pool”. In: *Structural Health Monitoring* 17.3 (2018), pp. 450–460.

[125] Lars Junghans and Nicholas Darde. “Hybrid single objective genetic algorithm coupled with the simulated annealing optimization method for building optimization”. In: *Energy and Buildings* 86 (2015), pp. 651–662.

[126] José F Villanueva, AI Sanchez, Sofía Carlos, and Sebastián Martorell. “Genetic algorithm-based optimization of testing and maintenance under uncertain unavailability and cost estimation: a survey of strategies for harmonizing evolution and accuracy”. In: *Reliability Engineering & System Safety* 93.12 (2008), pp. 1830–1841.

[127] Márcio das Chagas Moura, Isis Didier Lins, Enrique López Droguett, Rodrigo Ferreira Soares, and Rodrigo Pascual. “A multi-objective genetic algorithm for determining efficient risk-based inspection programs”. In: *Reliability Engineering & System Safety* 133 (2015), pp. 253–265.

[128] Gade Pandu Rangaiah and AB Petriciolet. "Multi-objective optimization in chemical engineering". In: *Developments and applications/edited by Gade Pandu Rangaiah, Adrián Bonilla-Petriciolet* (2013).

[129] Yusliza Yusoff, Mohd Salihin Ngadiman, and Azlan Mohd Zain. "Overview of NSGA-II for optimizing machining process parameters". In: *Procedia Engineering* 15 (2011), pp. 3978–3983.

[130] Ahmad Baklouti, Khalil Dammak, and Abdelkhalak El Hami. "Optimum reliable design of rolling element bearings using multi-objective optimization based on C-NSGA-II". In: *Reliability Engineering & System Safety* 223 (2022), p. 108508.

[131] Mojgan Mortezaei Farizhendy, Esmatullah Noorzai, and Mahmood Golabchi. "Implementing the NSGA-II genetic algorithm to select the optimal repair and maintenance method of jack-up drilling rigs in Iranian shipyards". In: *Ocean Engineering* 211 (2020), p. 107548.

[132] Xinyu Zhang and Chun Su. "Multi-objective optimization design of warranty strategy considering warranty cost and availability". In: *2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*. Vol. 1. IEEE. 2019, pp. 1221–1225.

[133] Mohammad Modarres, Mehdi Amiri, and Christopher Jackson. *Probabilistic physics of failure approach to reliability: modeling, accelerated testing, prognosis and reliability assessment*. Wiley Online Library, 2017.

[134] Xiao-Sheng Si, Wenbin Wang, Chang-Hua Hu, and Dong-Hua Zhou. "Remaining useful life estimation–a review on the statistical data driven approaches". In: *European journal of operational research* 213.1 (2011), pp. 1–14.

[135] Joanna Z Sikorska, Melinda Hodkiewicz, and Lin Ma. "Prognostic modelling options for remaining useful life estimation by industry". In: *Mechanical systems and signal processing* 25.5 (2011), pp. 1803–1836.

[136] Saralees Nadarajah and Samuel Kotz. "Exact distribution of the max/min of two Gaussian random variables". In: *IEEE Transactions on very large scale integration (VLSI) systems* 16.2 (2008), pp. 210–212.

[137] Mohammad Modarres, Mark P Kaminskiy, and Vasiliy Krivtsov. *Reliability engineering and risk analysis: a practical guide*. CRC press, 2016.

[138] Michael L. Waskom. "seaborn: statistical data visualization". In: *Journal of Open Source Software* 6.60 (2021), p. 3021. DOI: 10.21105/joss.03021.

[139] Adam Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035.

[140] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[141] Lu Liu, Xiao Song, and Zhetao Zhou. "Aircraft engine remaining useful life estimation via a double attention-based data-driven architecture". In: *Reliability Engineering & System Safety* 221 (2022), p. 108330.

[142] Abbas Khosravi, Saeid Nahavandi, Doug Creighton, and Amir F Atiya. "Lower upper bound estimation method for construction of neural network-based prediction intervals". In: *IEEE transactions on neural networks* 22.3 (2010), pp. 337–346.

[143] Chongdang Liu, Linxuan Zhang, Yuan Liao, Cheng Wu, and Gongzhuang Peng. "Multiple sensors based prognostics with prediction interval optimization via echo state Gaussian process". In: *IEEE Access* 7 (2019), pp. 112397–112409.

[144] BayesFusion,LLC. *GeNIe Modeler*. User Manual. Available online: https://support.bayesfusion.com/docs/GeNIe.pdf. 2022.

[145] AP Basu and JK Ghosh. "Identifiability of the multinormal and other distributions under competing risks model". In: *Journal of Multivariate Analysis* 8.3 (1978), pp. 413–429.

[146] HN Nagaraja. "On the independence of system life distribution and cause of failure". In: *Scandinavian Actuarial Journal* 1982.3-4 (1982), pp. 188–198.

[147] Herbert A David and Haikady N Nagaraja. *Order statistics*. John Wiley & Sons, 2004.

[148] Yung Liang Tong and YL Tong. *Fundamental properties and sampling distributions of the multivariate normal distribution*. Springer, 1990.

[149] Austin D Lewis and Katrina M Groth. "Metrics for evaluating the performance of complex engineering system health monitoring models". In: *Reliability Engineering & System Safety* 223 (2022), p. 108473.

[150] Yuan Liao, Linxuan Zhang, and Chongdang Liu. "Uncertainty prediction of remaining useful life using long short-term memory network based on bootstrap method". In: *2018 IEEE international conference on prognostics and health management (icphm)*. IEEE. 2018, pp. 1–8.

[151] Guozhen Gao, Zijun Que, and Zhengguo Xu. "Predicting remaining useful life with uncertainty using recurrent neural process". In: *2020 IEEE 20th International Conference on Software Quality, Reliability and Security Companion (QRS-C)*. IEEE. 2020, pp. 291–296.

[152] Felix O Heimes. "Recurrent neural networks for remaining useful life estimation". In: *2008 international conference on prognostics and health management*. IEEE. 2008, pp. 1–6.

[153] Giduthuri Sateesh Babu, Peilin Zhao, and Xiao-Li Li. "Deep convolutional neural network based regression approach for estimation of remaining useful life". In: *Database Systems for Advanced Applications: 21st International Conference, DASFAA 2016, Dallas, TX, USA, April 16-19, 2016, Proceedings, Part I 21*. Springer. 2016, pp. 214–228.

[154] Richard Liaw, Eric Liang, Robert Nishihara, Philipp Moritz, Joseph E Gonzalez, and Ion Stoica. "Tune: A Research Platform for Distributed Model Selection and Training". In: *arXiv preprint arXiv:1807.05118* (2018).

[155] Huyang Xu, Nasser Fard, and Yuanchen Fang. "Time series chain graph for modeling reliability covariates in degradation process". In: *Reliability Engineering & System Safety* 204 (2020), p. 107207.

[156] Marco Rigamonti, Piero Baraldi, Enrico Zio, Indranil Roychoudhury, Kai Goebel, and Scott Poll. "Ensemble of optimized echo state networks for remaining useful life prediction". In: *Neurocomputing* 281 (2018), pp. 121–138.

[157] Junqi Zeng and Zhenglin Liang. “A dynamic predictive maintenance approach using probabilistic deep learning for a fleet of multi-component systems”. In: *Reliability Engineering & System Safety* 238 (2023), p. 109456.

[158] Wenjun Sun, Rui Zhao, Ruqiang Yan, Siyu Shao, and Xuefeng Chen. “Convolutional discriminative feature learning for induction motor fault diagnosis”. In: *IEEE Transactions on Industrial Informatics* 13.3 (2017), pp. 1350–1359.

[159] Wennian Yu, II Yong Kim, and Chris Mechefske. “Remaining useful life estimation using a bidirectional recurrent neural network based autoencoder scheme”. In: *Mechanical Systems and Signal Processing* 129 (2019), pp. 764–780.

[160] Kaixiang Peng, Ruihua Jiao, Jie Dong, and Yanting Pi. “A deep belief network based health indicator construction and remaining useful life prediction using improved particle filter”. In: *Neurocomputing* 361 (2019), pp. 19–28.

[161] Ruochen Yang, Jørn Vatn, and Ingrid Bouwer Utne. “Dynamic maintenance planning for autonomous marine systems (AMS) and operations”. In: *Ocean Engineering* 278 (2023), p. 114492.

[162] The MathWorks Inc. *MATLAB version: 9.13.0 (R2022b), Natick, Massachusetts, United States. https://www.mathworks.com*. Natick, Massachusetts, United States, 2022.

[163] David Mahar, William Fields, and D.B Nicholls. *Nonelectronic Parts Reliability Data (NPRD)*. Reliability Information Analysis Center, Utica, NY, 2016.

[164] SINTEF. *Offshore Reliability Data Handbook: 6th Edition*. OREDA Participants, Det Norske Veritas (DNV), 2015.

[165] J C Chesley. *Handbook of Reliability Prediction Procedures for Mechanical Equipment*. Naval Surface Warfare Center (NSWC) Carderock Division, Logistics Engineering Technology Branch, 1998.

[166] Nicola Beume, Boris Naujoks, and Michael Emmerich. “SMS-EMOA: Multiobjective selection based on dominated hypervolume”. In: *European Journal of Operational Research* 181.3 (2007), pp. 1653–1669.

[167] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. “A fast and elitist multiobjective genetic algorithm: NSGA-II”. In: *IEEE transactions on evolutionary computation* 6.2 (2002), pp. 182–197.

[168] Slawomir Koziel, David Echeverría Ciaurri, and Leifur Leifsson. “Surrogate-based methods”. In: *Computational optimization, methods and algorithms* (2011), pp. 33–59.

[169] ZM Nopiah, MI Khairir, Shahrum Abdullah, MN Baharin, and A Arifin. “Time complexity analysis of the genetic algorithm clustering method”. In: *Proceedings of the 9th WSEAS international conference on signal processing, robotics and automation, ISPRA*. Vol. 10. 2010, pp. 171–176.

[170] Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and Tanaka Meyarivan. “A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II”. In: *Parallel Problem Solving from Nature PPSN VI: 6th International Conference Paris, France, September 18–20, 2000 Proceedings 6*. Springer. 2000, pp. 849–858.

[171] Kalyanmoy Deb, Karthik Sindhya, and Tatsuya Okabe. "Self-adaptive simulated binary crossover for real-parameter optimization". In: *Proceedings of the 9th annual conference on genetic and evolutionary computation*. 2007, pp. 1187–1194.

[172] Feifei Zheng, Zhexian Qi, Weiwei Bi, Tuqiao Zhang, Tingchao Yu, and Yu Shao. "Improved understanding on the searching behavior of NSGA-II operators using run-time measure metrics with application to water distribution system design problems". In: *Water Resources Management* 31 (2017), pp. 1121–1138.

[173] J. Blank and K. Deb. "pymoo: Multi-Objective Optimization in Python". In: *IEEE Access* 8 (2020), pp. 89497–89509.

[174] Andrej Dobnikar, Nigel C Steele, David W Pearson, Rudolf F Albrecht, Kalyanmoy Deb, and Samir Agrawal. "A niched-penalty approach for constraint handling in genetic algorithms". In: *Artificial Neural Nets and Genetic Algorithms: Proceedings of the International Conference in Portorož, Slovenia, 1999*. Springer. 1999, pp. 235–243.

[175] Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele. "Comparison of multiobjective evolutionary algorithms: Empirical results". In: *Evolutionary computation* 8.2 (2000), pp. 173–195.

[176] Andrzej Osyczka and Sourav Kundu. "A new method to solve generalized multicriteria optimization problems using the simple genetic algorithm". In: *Structural optimization* 10 (1995), pp. 94–99.

[177] Masahiro Tanaka, Hikaru Watanabe, Yasuyuki Furukawa, and Tetsuzo Tanino. "GA-based decision support system for multicriteria optimization". In: *1995 IEEE international conference on systems, man and cybernetics. Intelligent systems for the 21st century*. Vol. 2. IEEE. 1995, pp. 1556–1561.

[178] Elizabeth Jordan, Arko Chatterjee, Ruochen Yang, Shapour Azarm, and Katrina M. Groth. "A Fixed Sampling Based Method For Multi-Objective Robust Optimization Problems". In: *Proceedings of the ASME International Mechanical Engineering Congress & Exposition (IMECE), November 21-24, 2024, Portland, OR*. 2024.