Reliability and Maintenance for Engineering Systems: Fault Trees, Degradation Modelling and Maintenance Optimisation

Lisandro Arturo Jimenez Roa

Reliability and Maintenance for Engineering Systems: Fault Trees, Degradation Modelling and Maintenance Optimisation

DISSERTATION

to obtain
the degree of doctor at the University of Twente,
on the authority of the rector magnificus,
prof. dr. ir. A. Veldkamp,
on account of the decision of the Doctorate Board
to be publicly defended
on Friday 7 February 2025 at 14.45 hours

by

Lisandro Arturo Jimenez Roa

born on the 22nd of September, 1991
in Cali, Colombia

This dissertation has been approved by:

Promotors
prof. dr. M.I.A. Stoelinga
prof. dr. ir. T. Tinga

Co-promotor
prof. dr. T.M. Heskes

Cover design:	Lisandro Arturo Jimenez Roa
Printed by:	Ipskamp Printing
Lay-out:	Typeset in LaTeX, based on the Overleaf template by Steve Gunn, Sunil Patel, and Joshua Botha. Modifications by Lisandro Arturo Jimenez Roa
ISBN (print):	978-90-365-6406-9
ISBN (digital):	978-90-365-6407-6
URL:	https://doi.org/10.3990/1.9789036564076

Graduation Committee:

Chair / secretary: prof. dr. ir. B.R.H.M. Haverkort

Promotors: prof. dr. M.I.A. Stoelinga
University of Twente, EEMCS,
Formal Methods and Tools

prof. dr. ir. T. Tinga
University of Twente, ET,
Dynamics based Maintenance

Co-promotor: prof. dr. T.M. Heskes
Radboud Universiteit Nijmegen

Committee Members: prof. Dr. Ing. B. Rosic
University of Twente, ET,
Applied Mechanics & Data Analysis

prof. dr. ir. M. van Keulen
University of Twente, EEMCS,
Datamanagement & Biometrics

prof. dr. ir. G.J. van Houtum
Eindhoven University of Technology

prof. dr. ir. E. Zio
Mines Paris – PSL University, Politecnico di Milano

prof. dr. ir. P. Dersin
Luleå University of Technology

“*I do not think there is any thrill that can go through the human heart like that felt by the inventor [who] sees a creation of the brain unfolding to success . . .*”

Nikola Tesla

UNIVERSITY OF TWENTE

Abstract

Faculty of Electrical Engineering, Mathematics and Computer Science

Formal Methods and Tools

Doctor of Philosophy

Reliability and Maintenance for Engineering Systems: Fault Trees, Degradation Modelling and Maintenance Optimisation

by Lisandro Arturo Jimenez Roa

Modern infrastructures, machines, and manufacturing processes require effective management through sustainable policies under constrained resources, where determining *when* and *how* to intervene becomes crucial. The Prognostics and Health Management (PHM) paradigm provides a systematic framework for leveraging data collection and computational models, supporting the management of virtually *any* engineering component or system. This dissertation delves into three key aspects of PHM: Reliability Modelling, Markov Process-based Prognostics, and Maintenance Optimisation. Data-driven techniques are crucial in these areas, enhancing the automation of model development and deployment.

Part I centres on Reliability Modelling, specifically the automatic inference of Fault Tree (FT) models. Traditionally, graph-based models like FTs are manually constructed through iterative collaboration between system experts and FT modellers. However, this manual approach is prone to human error and may result in incomplete models. With the increasing availability of data, methodologies that attempt to automate this process, discover patterns and reduce dependency on manual intervention have gained significant relevance. Thus, in Part I of this dissertation, we focus on *how to obtain efficient and compact Fault Tree models from failure datasets in a robust and scalable manner*.

For this matter, we propose, for the first time, using Multi-Objective Evolutionary Algorithms (MOEAs) to automatically infer FT models and cast the optimisation as a multi-objective task. This resulted in the `FT-MOEA` algorithm (Chapter 2), focusing on three optimisation metrics, including FT size and accuracy-related metrics. `FT-MOEA` consistently produced compact FT structures, but faced scalability issues. To address this, we developed the `SymLearn` toolchain (Chapter 3), which uses a ‘divide-and-conquer’ approach by identifying modules and symmetries in the

failure dataset, breaking the inference problem into smaller tasks. Additionally, to improve robustness and scalability, the `FT-MOEA-CM` extension (Chapter 4) includes additional metrics from the confusion matrix. Our approaches in Part I of this dissertation contribute to automating FT model construction, revealing compact structures. These consistent structures can help uncover relationships between basic and intermediate events, providing valuable insights for asset managers to improve reliability modelling.

Part II focuses on Markov Process-based Prognostics, specifically the stochastic deterioration modelling of sewer mains. Sewer systems are critical to social welfare but pose significant challenges due to their extensive scale, slow degradation, and limited capacity to monitor the entire network. Accurate modelling of the deterioration profile is crucial for optimising inspections and maintenance, thereby enhancing the reliability and availability of the networks. Various deterioration models are discussed in the literature, ranging from physics-based to data-driven approaches, each with distinct advantages and limitations. In Part II of this dissertation, we address *how and to what extent it is possible to accurately model Multi-State Deterioration with applications in sewer mains.*

For this, we focus on *Markov chains*, widely used to model stochastic sequences through states and transitions. Since the 1990s, they have been applied to represent *damage severity levels* in sewer mains using inspection data from Closed Circuit Television cameras. Nonetheless, further evaluation of their assumptions and properties is required. We present a case study of a Dutch sewer network (Chapter 5), starting with Discrete-Time Markov Chains for deterioration modelling and examining two Markov chain structures. Given challenges such as interval-censored data, advanced analysis was necessary. In Chapter 6, we use the Turnbull estimator for non-parametric analysis to establish a ground truth. Although both homogeneous and inhomogeneous-time Markov chains are employed for sewer mains deterioration, no prior studies have compared their performance on the same dataset. Chapter 6 addresses this by demonstrating that inhomogeneous-time Markov chains are more versatile at capturing non-linear stochastic behaviour, while also highlighting issues like overfitting that reduce predictive accuracy. Part II provides a real-world case study, emphasising the need to critically evaluate modelling assumptions to enhance deterioration modelling of sewer mains using Markov chains.

Finally, Part III focuses on Maintenance Optimisation of sewer mains, where obtaining optimal maintenance policies for such large-scale systems is a complex task. This complexity arises, among others, from the system's scale and simplifications in the deterioration model. Among the various techniques available, Reinforcement Learning (RL) approaches remain largely unexplored for devising strategic maintenance actions in sewer mains. Thus, in Part III of this dissertation, we focus on *how to devise optimal maintenance strategies for components with Multi-State Deterioration such as sewer mains using Deep Reinforcement Learning.*

In Chapter 7, we frame the sequential decision-making problem using Deep Reinforcement Learning (DRL) for component-level maintenance of sewer mains. This framework considers damage severity levels, testing different deterioration model assumptions and evaluating their impact on maintenance policy. Our results show that agent-based policies outperformed heuristics by learning optimal sequences of maintenance actions. Part III provides evidence that DRL-based techniques offer a flexible framework with the potential to improve heuristics and support maintenance decision-making for sewer mains. However, training these models to achieve the desired behaviour remains a challenging task.

UNIVERSITEIT TWENTE

Samenvatting

Faculteit Elektrotechniek, Wiskunde en Informatica, Afdeling Informatica

Formal Methods and Tools

Doctor of Philosophy

Betrouwbaarheid en Onderhoud van Technische Systemen: Fault Trees, Degradatiemodellering en Onderhoudsoptimalisatie

door Lisandro Arturo Jimenez Roa

Moderne infrastructuren, machines en productieprocessen vereisen effectief beheer door duurzame beleidsmaatregelen onder beperkte middelen, waarbij het bepalen van *wanneer* en *hoe* in te grijpen cruciaal wordt. Het Prognostics and Health Management (PHM)-paradigma biedt een systematisch kader voor het benutten van gegevensverzameling en computationele modellen, ter ondersteuning van het beheer van vrijwel *elk* technisch onderdeel of systeem. Deze dissertatie verdiept zich in drie sleutelaspecten van PHM: Betrouwbaarheidsmodellering, Prognoses Gebaseerd op Markov-Processen, en Onderhoudsoptimalisatie. Datagestuurde technieken zijn cruciaal in deze gebieden en verbeteren de automatisering van modelontwikkeling en -implementatie.

Deel I richt zich op Betrouwbaarheidsmodellering, specifiek de automatische inferentie van Fault Tree (FT)-modellen. Traditioneel worden grafisch gebaseerde modellen zoals FTs handmatig geconstrueerd door iteratieve samenwerking tussen systeemexperts en FTs-modelleurs. Echter, deze handmatige aanpak is gevoelig voor menselijke fouten en kan resulteren in onvolledige modellen. Met de toenemende beschikbaarheid van gegevens hebben methoden die dit proces automatiseren, patronen ontdekken en afhankelijkheid van manuele interventie verminderen, aanzienlijke relevantie gekregen. In deel I van deze dissertatie richten we ons op *hoe efficiënte en compacte Fault Tree modellen uit foutendatasets te verkrijgen op een robuuste en schaalbare manier*.

Hiervoor stellen we, voor de eerste keer, voor om Multi-Objective Evolutionary Algorithms (MOEAs) te gebruiken om automatisch FT-modellen te infereren en de optimalisatie als een multi-objectieve taak te beschouwen. Dit resulteerde in het `FT-MOEA`-algoritme (Hoofdstuk 2), dat zich richt op drie optimalisatiemetrieken, inclusief FT-grootte en aan nauwkeurigheid gerelateerde metrieken. `FT-MOEA`

produceerde consistent compacte FT-structuren, maar kende schaalbaarheidsproblemen. Om dit aan te pakken, hebben we de SymLearn-toolchain ontwikkeld (Hoofdstuk 3), die een ‘verdeel-en-heers’-benadering gebruikt door modules en symmetrieën in de faalgegevensset te identificeren, en het inferentieprobleem opdeelt in kleinere taken. Bovendien, om de robuustheid en schaalbaarheid te verbeteren, bevat de FT-MOEA-CM-uitbreiding (Hoofdstuk 4) extra metrieken uit de verwarringsmatrix. Onze benaderingen in Deel I van deze dissertatie dragen bij aan het automatiseren van de constructie van FT-modellen, door compacte structuren te onthullen. Deze consistente structuren kunnen helpen relaties tussen basis- en tussengebeurtenissen te onthullen, en bieden waardevolle inzichten voor vermogensbeheerders om betrouwbaarheidsmodellering te verbeteren.

Deel II richt zich op Prognoses Gebaseerd op Markov-Processen, specifiek de stochastische degradatiemodellering van rioolhoofdleidingen. Riolen zijn essentieel voor het sociale welzijn maar vormen significante uitdagingen vanwege hun uitgebreide schaal, langzame degradatie en beperkte capaciteit om het gehele netwerk te monitoren. Het nauwkeurig modelleren van het degradatieprofiel is cruciaal voor het optimaliseren van inspecties en onderhoud, en verbetert daarmee de betrouwbaarheid en beschikbaarheid van de netwerken. Verschillende degradatiemodellen worden besproken in de literatuur, variërend van op fysica gebaseerde tot datagedreven benaderingen, elk met eigen voor- en nadelen. In Deel II van deze dissertatie behandelen we *hoe en in welke mate het mogelijk is om Multi-State Deterioration nauwkeurig te modelleren met toepassingen in rioolhoofdleidingen.*

Voor dit doel richten we ons op Markov-ketens, die veel worden gebruikt om stochastische sequenties te modelleren door middel van staten en overgangen. Sinds de jaren ’90 zijn deze toegepast om schadeniveaus in rioolhoofdleidingen te representeren met behulp van inspectiegegevens van Closed Circuit Television-camera’s. Desondanks is verdere evaluatie van hun aannames en eigenschappen vereist. We presenteren een casestudie van een Nederlands rioolnetwerk (Hoofdstuk 5), beginnend met Discrete-Time Markov Chains voor degradatiemodellering en het onderzoeken van twee Markov-ketenstructuren. Gezien uitdagingen zoals intervalgecensureerde gegevens, was een meer geavanceerde analyse nodig. In Hoofdstuk 6 gebruiken we de Turnbull-schatter voor niet-parametrische analyse om een grondwaarheid vast te stellen. Hoewel zowel homogene als inhomogene tijd-Markov-ketens worden ingezet voor de degradatiemodellering van rioolhoofdleidingen, zijn eerdere studies die hun prestaties op dezelfde dataset hebben vergeleken, nog niet uitgevoerd. Hoofdstuk 6 adresseert dit door aan te tonen dat inhomogene-tijd-Markov-ketens veelzijdiger zijn bij het vastleggen van niet-lineair stochastisch gedrag, waarbij ook problemen zoals overfitting worden belicht die de voorspellende nauwkeurigheid verminderen. Deel II biedt een praktijkvoorbeeld en benadrukt het belang van het kritisch evalueren van modelaannames voor verbeterde degradatiemodellering van rioolhoofdleidingen met Markov-ketens.

Ten slotte richt Deel III zich op Onderhoudsoptimalisatie van rioolhoofdleidingen, waarbij het verkrijgen van optimaal onderhoudsbeleid voor dergelijke grootschalige systemen een complexe taak is. Deze complexiteit ontstaat, onder andere, uit de schaal van het systeem en vereenvoudigingen in het degradatiemodel. Onder de verschillende beschikbare technieken blijven Reinforcement Learning (RL)-benaderingen grotendeels onverkend voor het bedenken van strategische onderhoudsacties in rioolhoofdleidingen. Derhalve richten we ons in Deel III van deze dissertatie op *hoe optimale onderhoudsstrategieën te ontwikkelen voor componenten met Multi-State Deterioration zoals rioolhoofdleidingen met gebruik van Deep Reinforcement Learning*.

In Hoofdstuk 7 kaderen we het sequentiële besluitvormingsprobleem met gebruik van Deep Reinforcement Learning (DRL) voor componentniveau onderhoud van rioolhoofdleidingen. Dit kader beschouwt schadeniveaus, test verschillende aannames van het degradatiemodel en evalueert hun impact op het onderhoudsbeleid. Onze resultaten tonen aan dat op agenten gebaseerde beleidsmaatregelen heuristieken overtroffen door optimale onderhoudsactiesequenties te leren. Deel III biedt bewijs dat DRL-gebaseerde technieken een flexibel kader bieden met het potentieel om heuristieken te verbeteren en ondersteuning te bieden bij onderhoudsbesluitvorming voor rioolhoofdleidingen. Echter, het trainen van deze modellen om het gewenste gedrag te bereiken blijft een uitdagende taak.

Acknowledgements

This doctorate is the result of a long journey during which I had the opportunity to meet incredible people and visit amazing places. Here, I would like to express my gratitude to those who have been part of this beautiful journey, starting with my close-knit family.

Quiero dar un agradecimiento muy especial a mi madre, **Luz Hatty Roa** (mi luz), quien desde pequeño me ha impulsado y brindado su apoyo incondicional. A mi padre, **Lisardo A. Jimenez**, quien me ha enseñado la tenacidad y la fuerza, que ante la vida uno no se rinde, sin importar qué; también, un agradecimiento especial a mis hermanos, **Yennifer Jimenez Roa** y **Roger A. Jimenez Roa**; a mi sobrinito, **Nicola Pasqualin Jimenez**; y a mis cuñados, **Lorena M. Montoya** y **Edwin Plaza**.

Ik wil ook mijn dankbaarheid uiten aan mijn uitgebreide familie, in het bijzonder aan mijn partner **Niels van Huizen** voor zijn onvoorwaardelijke steun en het samen delen van deze prachtige fase in ons leven; en aan **Patrick**, **Helma**, en **Zanne van Huizen** voor alle mooie momenten.

To my alma mater, **Universidad del Valle** in my home city Cali-Colombia, of which I am immensely proud and to which I hope to one day give back as much as it has given me. I am particularly thankful to Professors **Ricardo Ramírez**, **Patricia Guerrero**, and **Doris Hinestroza**, who played pivotal roles in my career as a Civil Engineer. I also want to thank **João S. Almeida**, **Angelica Rosso**, and **Katrin Beyer** for the enriching experience at **École Polytechnique Fédérale de Lausanne** in Lausanne-Switzerland. My gratitude also extends to my friends **Laura Villegas** (mi flaquita) and **Raffael Tschui** (meme) for their friendship despite the long distance.

To the *"Pelicanoes"* **Andrea Vargas**, **Leopoldo Ríos**, **Lara Wöhler**, **Norwing Thunhorst**, **Daria Nemashkalo**, **Oleksandr Mialyk**, **Kees Lieverse**, **Daniel Foley**, and **Sara Dionisio Antonio** for the amazing and fun times we shared.

To the **University of Twente**, my other alma mater, for opening its doors and supporting my career. I particularly want to thank the **Construction Management and Engineering** group for hosting my Engineering Doctorate and allowing me to meet remarkable people, especially **Monik Peña Acosta** (Colombiana), **Tom Coenen**, **Angie Ruiz**, **Ramon ter Huurne**, **Tim van Ee**, **Ruth Sloot**, **Roland Kromanis**, and **Franziska Baack**.

I want to thank the **Formal Methods and Tools** group for making me feel at home during my PhD project and my adventure in Computer Science, special thanks to **Yeray Barrios**, **Stefano Nicoletti**, **Reza Soltani**, (not mathematician)

Yanni Dong, **Milan Lopuhaä**, **Edoardo Putti**, **Lukas Armborst**, **Arnd Hartmanns**, **Arend Rensink**, **Moritz Hahn**, **Djurre van der Wal**, **Tannaz Zameni**, **Tannaz Zameni**, **Marcus Gerhold**, **Nhat bui**, **Marijn Peppelman**, **Bram Kohlen**, **Bob Rubbens**, **Rom Langerak**, **Petra Bos**, **Vadim Zaytsev**, **Georgiana Caltais**, **Peter Lammich**, **Faizan Ahmed**, **Tom van Dijk**, and **Marieke Huisman**.

I want to thank the **Laboratory of Analysis of Systems for the Assessment of Reliability (LASAR)** for hosting my research visit at the **Politecnico di Milano** in Milan-Italy, where, despite the short time I spent there, I got to meet amazing people. Particular thanks to **Stefano Marchetti**, **Valentina Clavijo**, **Giovanni Floreale**, **Thomas Coscia**, **Nicolas Cárdenas**, **Sara Castellani**, **Federico Bassi**, **Piero Baraldi**, and **Enrico Zio**.

To the **PrimaVera** project for sponsoring my PhD and allowing me to pursue my passion. Special thanks to **Matthias Völk**, **Thiago D. Simão**, **Luc Keizers**, **Nubia Alves de Silveira**, **Natália Marinho**, **Thom Badings**, **Ragnar Eggertsson**, **Hajo Molegraaf**, and **Juseong Lee**.

Finally, a special thanks to my PhD supervisors **Mariëlle Stoelinga**, **Tiedo Tinga**, and **Tom Heskes** who supported and guided me through this 4-years research journey, thanks for all the lessons, advice, commitment, and patience.

Lisandro Arturo Jimenez Roa
Enschede, 10 December, 2024

Contents

Abstract vii

Samenvatting xi

Acknowledgements xv

1 Introduction 1
- 1.1 Research Context and Motivation . . . 1
- 1.2 Main concepts . . . 2
  - 1.2.1 Prognostics and Health Management . . . 2
  - 1.2.2 Reliability Modelling . . . 4
  - 1.2.3 Markov Process-based Prognostics . . . 8
  - 1.2.4 Maintenance Optimisation . . . 12
- 1.3 Research gaps . . . 16
- 1.4 Research questions . . . 19
- 1.5 Research methodology . . . 20
- 1.6 Thesis outline . . . 21
- 1.7 Main contributions . . . 21
- 1.8 List of publications . . . 23
- 1.9 References . . . 25

Part I: Data-driven Inference of Fault Tree models **33**
- I.1 Introduction . . . 33
- I.2 Nomenclature . . . 34
- I.3 Related work . . . 34
- I.4 Preliminaries . . . 36
  - I.4.1 Fault Trees . . . 36
  - I.4.2 Failure Dataset . . . 37
  - I.4.3 Inference of Fault Tree models . . . 38
- I.5 References . . . 38

2 Automatic Inference of Fault Tree Models via Multi-Objective Evolutionary Algorithms **41**
  - Abstract . . . 41

2.1 Introduction 42
2.2 Fault Tree Analysis 43
2.3 Multi-Objective Evolutionary Algorithms 44
2.3.1 Elitist Non-dominated Sorting Genetic Algorithms 44
2.3.2 Crowding-Distance 45
2.4 Methodology 46
2.4.1 The Failure Dataset 46
2.5 Inferring fault trees via multi-objective evolutionary algorithms (FT-MOEA) 47
2.5.1 Step 1 - Initialisation 47
2.5.2 Step 1.2 - Extraction of MCSs from the failure dataset (optional step) 48
2.5.3 Step 2 - Parent fault tree(s) 49
2.5.4 Step 3 - Genetic operators 49
2.5.5 Step 4 - Multi-objective Function 50
Metrics Calculation 50
Setups of the multi-objective functions 51
2.5.6 Step 5 - Convergence criterion 51
2.6 Experimental evaluation 51
2.6.1 The Monte Carlo method 51
2.6.2 Case studies 52
2.6.3 Key findings of the FT-MOEA algorithm 52
2.6.4 Parametric analysis 54
2.7 Discussion and Conclusions 57
2.8 References 58

3 Data-Driven Inference of Fault Tree Models Exploiting Symmetry and Modularisation **61**
Abstract 61
3.1 Introduction 61
3.2 Modules and Symmetries 63
3.2.1 Modules 63
3.2.2 Symmetries 64
3.3 Exploiting Modules and Symmetries in Fault Tree Inference 65
3.4 Experimental Evaluation 69
3.5 Conclusions 72
3.6 References 73

4 Fault Tree inference using Multi-Objective Evolutionary Algorithms and Confusion Matrix-based metrics **75**
Abstract 75
4.1 Introduction 76
4.2 Confusion Matrix-based metrics 77
4.3 FT-MOEA-CM's methodology 78

4.4 Experimental Evaluation . . . 79
4.5 Results . . . 80
4.5.1 Feature Assessment . . . 80
4.5.2 Comparing FT-MOEA and FT-MOEA-CM . . . 82
4.5.3 FT-MOEA-CM's Features: Parallelisation and Caching . . . 86
4.6 General discussion . . . 87
4.7 Conclusions . . . 89
4.8 References . . . 91

Part II: Multi-state deterioration modelling 93
II.1 Introduction . . . 93
II.2 Nomenclature . . . 94
II.3 Related work . . . 94
II.4 Preliminaries . . . 95
II.4.1 Markov Chains . . . 95
II.4.2 Case studies in sewer networks: overview . . . 99
II.4.3 Breda's case study . . . 100
II.4.4 Deterioration modelling in sewer mains using Markov Chains 103
II.5 References . . . 106

5 Deterioration Modelling of Sewer Pipes via Discrete-Time Markov Chains: A Large-Scale Case Study in the Netherlands 111
Abstract . . . 111
5.1 Introduction . . . 112
5.2 Homogeneous discrete-time Markov chain . . . 113
5.3 Methodology . . . 115
5.3.1 Data pre-processing . . . 115
5.3.2 Definition of cohorts . . . 115
5.3.3 Discretised table . . . 116
5.3.4 Calibration of the Discrete-Time Markov Chain . . . 116
5.4 Experimental Evaluation . . . 117
5.4.1 Case study . . . 117
5.4.2 Results . . . 117
Comparing Cohorts CMW and CR . . . 118
Comparing Cohorts PMW and PR . . . 119
Comparing Cohorts CdL and CdG . . . 119
Comparing Chains "Multi" and "Single" . . . 119
5.5 Discussion and Conclusions . . . 119
5.6 References . . . 120

6 Comparing Homogeneous and Inhomogeneous Time Markov Chains for Modelling Deterioration in Sewer Pipe Networks 123
Abstract . . . 123

6.1 Introduction 124
6.2 Methods and materials 125
6.2.1 Multi-state deterioration modelling for sewer networks using parametrised Markov chains 126
6.2.2 Model calibration 127
6.2.3 Non-parametric modelling 129
6.2.4 Goodness-of-fit metrics 129
6.3 Experimental setup and evaluation 130
6.3.1 Case study 130
6.3.2 Experimental setup 130
6.3.3 Results 130
6.4 Findings 130
6.4.1 Comparison between cohorts 130
6.4.2 Transition probabilities over time 131
6.4.3 Overfitting 132
6.4.4 Comparing inhomogeneous Markov chains 132
6.5 Conclusions and future research 132
6.6 References 135

Part III: Maintenance optimisation of multi-state components 137

III.1 Introduction 137
III.2 Nomenclature 138
III.3 Related work 139
III.4 Preliminaries 143
III.4.1 Markov Decision Process 143
III.4.2 Deep Reinforcement Learning 143
III.4.3 Contextual Markov Decision Process 145
III.4.4 Proximal Policy Optimisation 145

7 Maintenance Strategies for Sewer Pipes with Multi-State Deterioration and Deep Reinforcement Learning 151

Abstract 151
7.1 Introduction 152
7.2 Technical background 153
7.3 Methodology 153
7.4 Multi-state deterioration models 154
7.4.1 Case study 154
7.4.2 Parametrisation 154
7.4.3 Solving the Multi-State Deterioration Model 155
7.4.4 Parametric Multi-State Deterioration Models 155
7.5 Definition of Markov Decision Process for Maintenance Policy Optimisation of a Sewer Main considering pipe length deterioration . . 156

7.5.1 State space $\mathbb{S}$ . . . 157
Health vector (**h**) . . . 157
Stochastic prediction of severity levels . . . 158
7.5.2 Action space $\mathcal{A}$ . . . 158
7.5.3 Transition probability function **P** . . . 159
7.5.4 Reward function $\mathcal{R}$ . . . 160
7.6 Experimental setup . . . 161
7.6.1 Setup . . . 161
7.6.2 Comparison of maintenance strategies . . . 162
7.7 Results . . . 162
7.7.1 Implementation and hyper-parameter tuning . . . 162
7.7.2 Policy analysis: overview . . . 163
7.7.3 Policy analysis over episode . . . 166
7.8 Discussion and Conclusions . . . 167
7.9 References . . . 170

Discussion, Conclusions & Recommendations 173

8 Discussion 175
8.1 Reliability Modelling: Data-driven Inference of Fault Tree models . 175
8.2 Markov Process-based Prognostics: Multi-state deterioration modelling177
8.3 Maintenance Optimisation: Maintenance optimisation of multi-state components . . . 180
8.4 Moving towards comprehensive Prognostics and Health Management: Closing Thoughts . . . 182
8.5 References . . . 183

9 Conclusions & Recommendations 185
9.1 Conclusions . . . 185
9.2 Recommendations . . . 186
9.2.1 Automatic Inference of Fault Tree Models . . . 186
9.2.2 Multi-State Deterioration Modelling of Sewer Mains . . . 187
9.2.3 Strategic Maintenance Planning for Sewer Mains using Reinforcement Learning . . . 189
9.3 References . . . 190

Appendices 193

A Appendix: Introduction 195
A.1 Example of a Multi-State Deterioration model with two states . . . 195
A.2 Example of a Multi-State Deterioration model with three states . . 198

B Appendixes: `FT-MOEA` 201

B.1 Data-driven methods to infer FTs from data 201
B.2 Applying NSGA-II and Crowding-Distance to infer FTs 201
B.2.1 Applying NSGA-II to infer FTs 201
B.2.2 Crowding-Distance 205
B.3 Example of inferred fault trees 205
B.4 Details of Convergence of Metrics Over Generations 206
B.5 Varying parent Fault Trees (details) 209
B.6 Comparing the performance of m.o.f.'s for the case studies CSD, PT, and SMS' 210
B.7 Noise Effects in the Inference of Fault Trees with `FT-MOEA` 210
B.8 References 211

C Appendix: Multi-State Deterioration 215
C.1 Relations in reliability analysis 215

D Parameters of multi-state deterioration models 217

List of Abbreviations

AI	Artificial Intelligence. 9, 15
BE	Basic Event. 34, 54, 57, 88–90, 177
CBM	Condition-Based Maintenance. 2
CCTV	Closed Circuit Television. viii, xii, 11, 18, 94, 99, 101, 125
CM	Confusion Matrix. 76–78, 80
CMDP	Contextual Markov Decision Process. 138, 145, 189
CTMC	Continuous-Time Markov Chain. 98, 126, 134
DL	Deep Learning. 14
DNN	Deep Neural Network. 15, 138, 143–145
DRL	Deep Reinforcement Learning. viii, ix, xiii, 2, 15–17, 19, 21, 23, 137, 140, 142, 143, 145, 146, 152–154, 165, 167, 175, 181–183, 186, 189, 190
DTMC	Discrete-Time Markov Chain. viii, xix, 98, 99, 104, 111–117, 119, 120, 126, 130, 134, 188
EA	Evolutionary Algorithm. 44
FT	Fault Tree. vii, viii, xi, xii, xxi, xxii, 2, 6, 14, 16–21, 33–36, 38, 41–52, 54–58, 61–66, 68, 69, 72, 73, 75–85, 87–90, 175–177, 182, 185–187, 201, 203–205, 209–212
FTA	Fault Tree Analysis. 5, 17, 42, 43, 61, 62, 76, 175
HR	Hazard Rate Function. 97
HTMC	Homogeneous Time Markov Chain. 125, 126
IHTMC	Inhomogeneous Time Markov Chain. 97, 98, 125, 126, 134, 188

MCS	Minimal Cut Set. xviii, 14, 34, 43, 47–52, 56, 57, 63–69, 71, 72, 76, 83, 84, 87–90, 177
MDP	Markov Decision Process. 2, 14–16, 19, 21, 138–140, 143, 145, 156, 186, 189, 190
ML	Machine Learning. 9, 12–15, 18, 19, 94, 95, 99
MO	Maintenance Optimisation. 12–14
MOEA	Multi-Objective Evolutionary Algorithm. vii, xi, 6, 7, 21, 34, 43, 44, 47, 175–177, 182, 185, 187
MPO	Maintenance Policy Optimisation. 13, 19, 137, 140, 142, 152, 181
MSD	Multi-State Deterioration. viii, xii, xiii, xx–xxii, 2, 10, 11, 17, 19, 21, 93, 137, 152, 153, 155, 177, 185, 186, 195, 197–199, 215, 216
MSDM	Multi-State Deterioration Model. xx, 23, 138, 151, 153, 155, 156, 158, 159, 161, 162, 177, 178, 180–183, 217
MSM	Multi-State Modelling. 95, 125
NN	Neural Network. 16
PC	Principal Component. 81, 82
PCA	Principal Component Analysis. 77, 80, 81, 86
PDF	Probability Density Function. 97
PdM	Predictive Maintenance. 2
PHM	Prognostics and Health Management. vii, xi, 1–4, 8, 9, 16, 167, 175, 177, 182, 183
PoF	Physics of Failure. 5
PPO	Proximal Policy Optimisation. 138, 145, 146, 153, 154, 162, 168, 186
RL	Reinforcement Learning. viii, xiii, 14, 15, 19, 138, 139, 142–144, 152, 154, 156, 162, 181, 183, 189
RUL	Remaining Useful Life. 8
SF	Survival Function. 97

Chapter 1

Introduction

1.1 Research Context and Motivation

Modern societies rely on a variety of engineering systems that help maintain order, ensure safety, provide comfort, as well as promote growth and well-being. *Engineering systems* are specifically designed, built, and managed to address challenges by combining different technological parts. They cover many areas such as mechanical, electrical, civil, and computer engineering, and are crucial in building and maintaining our infrastructure, producing goods, managing transportation, and handling information technology. These systems are usually complex and have many interconnected components that depend on each other. Careful management and continuous improvement are key for them to work as intended.

According to fundamental principles of physics, everything naturally progresses towards a state of wear and breakdown through *deterioration* (McPherson, 2019), encompassing processes or events that affect the functionality of system components or the system as a whole. Keeping engineering systems operational and available under limited resources highlights the importance of approaches such as *Prognostics and Health Management (PHM)*, which will be later explained. PHM is gaining popularity in various industries because it improves sustainability by considering environmental, social, and economic factors, aiming to use resources efficiently and positively impact the environment. The main goal of PHM is to use models and data to spot unusual behaviours and problems, diagnose issues, and anticipate future performance. This aids in developing optimal strategies for effective system management.

Within the context of the *PrimaVera* project (https://primavera-project.com/), this thesis explores relevant aspects within PHM in terms of '*Reliability Modelling*', '*Markov Process-based Prognostics*', and '*Maintenance Optimisation*'. Before presenting the research gaps detailed in Section 1.3, I will first provide in the next section relevant literature and background of the main methods, models, and algorithms used in this thesis.

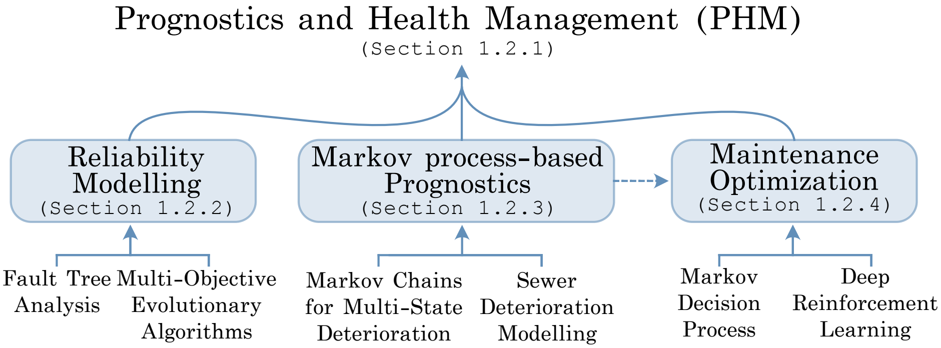

Prognostics and Health Management (PHM)
(Section 1.2.1)
Reliability Modelling
(Section 1.2.2)
Markov process-based Prognostics
(Section 1.2.3)
Maintenance Optimization
(Section 1.2.4)
Fault Tree Analysis
Multi-Objective Evolutionary Algorithms
Markov Chains for Multi-State Deterioration
Sewer Deterioration Modelling
Markov Decision Process
Deep Reinforcement Learning

Figure 1.1: Main theoretical concepts used in this thesis.

1.2 Main concepts

This section provides a concise overview of the main concepts and relevant literature for this thesis, though it should be noted that many details fall outside its scope. Prognostics and Health Management (PHM), detailed in Section 1.2.1, is the overarching concept connecting the research directions explored in this thesis. Figure 1.1 shows the main concepts discussed in this thesis and how they relate.

Section 1.2.2 introduces *Reliability Modelling*, focusing on *Fault Tree Analysis* and the use of *Multi-Objective Evolutionary Algorithms* to automatically infer the structure of Fault Tree models. Section 1.2.3 introduces *Markov Process-based Prognostics*, with emphasis on *Markov Chains* for Multi-State Deterioration with applications on *Sewer Deterioration Modelling*. Finally, Section 1.2.4 introduces *Maintenance Optimisation* with applications to sewer mains, where we use *Markov Decision Process* to cast the optimisation problem and approximate a solution via *Deep Reinforcement Learning*.

1.2.1 Prognostics and Health Management

Prognostics and Health Management (PHM) is *a holistic multi-disciplinary computation-based engineering paradigm that employs measurements, models, and software for the efficient and comprehensive management of system health* (Sheppard, Kaufman, and Wilmering, 2008; Lee, F. Wu, Zhao, et al., 2014; W. Zhang, D. Yang, and H. Wang, 2019; Fink, Q. Wang, Svensén, et al., 2020; Hu, Miao, Y. Si, et al., 2022; L. Zhang, Lin, Liu, et al., 2019; Zio, 2022).

Notice that in the literature, the terms Predictive Maintenance (PdM) and Condition-Based Maintenance (CBM) greatly overlap with PHM. However, the use of PdM and CBM may be ambiguous and, in some cases, interchangeable (Esteban, Zafra, and Ventura, 2022). For consistency, in this dissertation, we adopt PHM as a more general term.

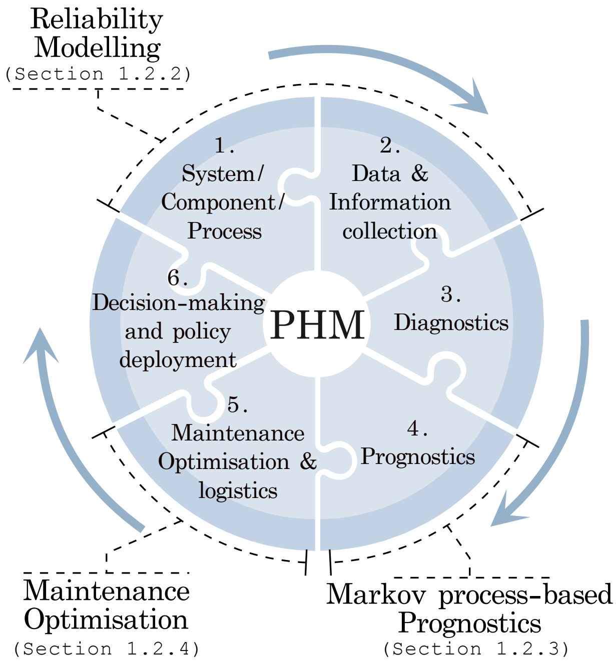


Figure 1.2: The main stages of the PHM paradigm.

The main stages in PHM are depicted in Figure 1.2. Stage 1 identifies the system, component, or process of interest. Stage 2 involves the design and implementation of infrastructure that enables the collection of data and relevant information, such as monitoring systems. Stage 3, known as *diagnostics*, addresses *what is wrong?* and focuses on the *current* system condition. Here, models detect, isolate, locate, quantify, and classify anomalies and failure modes. Stage 4, known as *prognostics*, answers *how long until an event or state is reached?* and focuses on the *future* condition of the system. Here, models attempt to characterise the system's future performance. Stage 5, referred to as *maintenance optimisation & logistics*, focuses on the set of algorithms that seek the optimal set of actions to *control* and meet functionality requirements. Lastly, Stage 6 focuses on the *decision-making* and *deployment* of the controlling *policy*.

Once Stage 6 is reached, the cycle restarts, even with new goals given former stages, e.g., evaluating the effectiveness of a maintenance action. The arrows in Figure 1.2 refer to the logical progression between stages rather than a strict sequence of steps. For instance, a non-nominal behaviour identified in Stage 3 may prompt a maintenance action (Stage 5), such as performing an inspection, thus skipping Stage 4—Prognostics. Additionally, PHM can be used as a *design* tool; for example, the requirements of using a certain type of prognostic model in Stage 4 will define part of the data that needs to be collected in Stage 2.

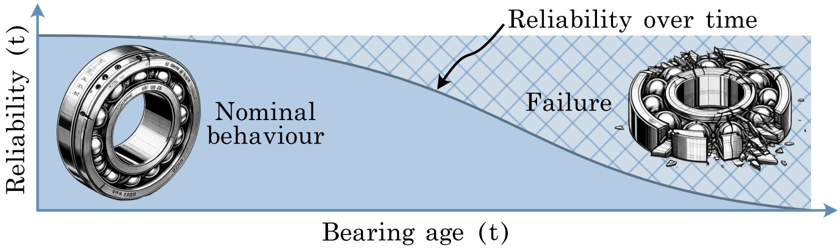


Figure 1.3: Reliability function of a bearing (example).

As mentioned earlier, this thesis focuses on relevant aspects within PHM. Figure 1.2 highlights that *Reliability Modelling* is mainly associated with Stages 1 and 2; *Markov Process-based Prognostics* with Stage 4, and *Maintenance Optimisation* with Stage 5.

1.2.2 Reliability Modelling

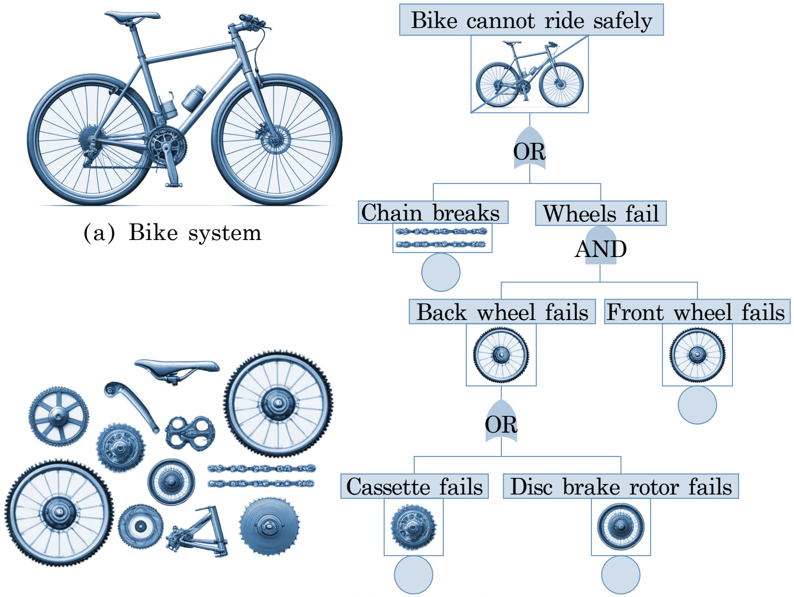
Reliability is the *"ability to perform as required, without failure, for a given time interval, under given conditions"* (IEC 60050:192-01-24), where *"failure is the loss of the ability to perform as required"* (IEC 60050:192-03-01).

Reliability modelling is the process of developing mathematical models that encapsulate *reliability functions* and *dependencies* within a system. The former quantifies the *probability* that a system or component will perform without failure over a specified period under defined conditions. The latter details the interactions between components that influence the overall system reliability (Assaf, 2018).

Reliability modelling provides stochastic-based outcomes (e.g., probabilities) useful for reliability assessment to enhance a system's lifespan, scheduling maintenance appropriately, and reducing the risk of failures (O'Connor and Kleyner, 2012). The application of reliability modelling spans various industries such as nuclear, aerospace, automotive, electronics, and manufacturing, where reliability is a critical factor (Modarres, Kaminskiy, and Krivtsov, 2016).

Figure 1.3 exemplifies the reliability function of a bearing as a function of its age. Initially, a new bearing has high reliability, representing *nominal behaviour.* As the bearing ages and wear occurs, reliability decreases due to the effects of deterioration. This means that the older (or more used) the bearing is, the *less likely* it is to perform nominally. Alternatively, the older the bearing is, the *more likely* it is to *fail.*

Several models and methods are used in reliability modelling. *Statistical-based reliability models*, which can be based on Exponential, Weibull, and Log-normal distributions, utilise historical failure data to estimate the probability of failure. Nelson, 2005 provides a comprehensive guide on how these models are used to understand different failure rates and patterns.



(a) Bike system

(b) Bike components

(c) (Partial) Fault Tree of a bike

Figure 1.4: Example of a Fault Tree model of a bike.

The *Physics of Failure (PoF)* approach examines the root causes of component failures by assessing how materials, defects, and stresses affect reliability. It identifies and models individual failure mechanisms in components based on environmental and usage stresses (H. Wang, Liserre, Blaabjerg, et al., 2014). For example, Zhu, Huang, W. Peng, et al., 2016 proposes a PoF-based framework for fatigue reliability analysis of an aircraft turbine engine disc.

System reliability models, on the other hand, evaluate the reliability of the entire system by considering the interaction and configuration of system components. Graph-based techniques, such as *Reliability Block Diagrams* (Signoret and Leroy, 2021), and systematic approaches, such as *Failure Modes and Effects Analysis* (Stamatis, 2003), are employed to identify potential failures and their consequences within a system.

One of the most prominent system reliability methods we encounter in the literature is Fault Tree Analysis, which we discuss further in the next section.

Fault Tree Analysis

Fault Tree Analysis (FTA) (Ruijters and Stoelinga, 2015) is a key technique in reliability engineering and risk analysis, used since the 1960s across various sectors such as automotive, aerospace, and nuclear industries (Kabir, 2017). FTA helps in modelling complex systems by illustrating logical relationships, which are crucial for understanding potential system failures, tracing root causes, pinpointing critical components, and computing probabilities of failure at both system and sub-system levels.

Fault Trees (FTs) are graphical models composed of *logic gates* and *basic events*. See Section I.4.1 for formal definitions. As an example, Figure 1.4(a) illustrates a *bike* system. Figure 1.4(b) presents the bike system components, such as wheels, handle, chain, disc brake, and cassette. Figure 1.4(c) models the bike's inability to ride safely using FTs. The system is divided into sub-systems and components until the desired resolution is reached. The *top event: bike cannot ride safely* represents the event of interest.

The *logic gates* in Figure 1.4(c) determines how a failure propagates. For example, if the *chain breaks*, the bike cannot ride, triggering the top event through the "OR" gate. The "AND" gate models failure when the *wheels fail*, indicating that all basic events under the gate must activate for the top event to occur. If only the front wheel fails, the bike can still ride with effort, but if the disc brake rotor also fails, it becomes unsafe.

This simple example demonstrates the usefulness of FT models in showing relationships between components and failure propagation, aiding strategic actions to prevent system-level failures. When basic events in the FT are modelled with *probability density functions*, the FT can provide quantitative data to support decision-making for managing the system.

One of the main challenges associated with FTs is *building* the model itself. To address this, in this dissertation, we used Multi-Objective Evolutionary Algorithms, which we discuss in the next section.

Multi-Objective Evolutionary Algorithms

Multi-Objective Evolutionary Algorithms (MOEAs) are population-based search strategies with conflicting objectives to be simultaneously optimised in a multi-dimensional space (Deb, 2011).

To explain their concept, let us consider the following example: assume that older computers have lower performance due to outdated technology and lower market costs, while modern computers have higher performance and higher market costs. Plotting market cost versus performance for a set of computers creates a visualisation like Figure 1.5.

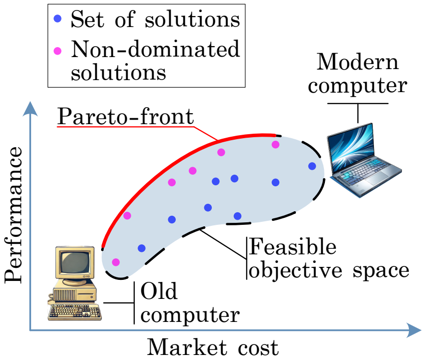


Figure 1.5: Computer market cost vs performance (example).

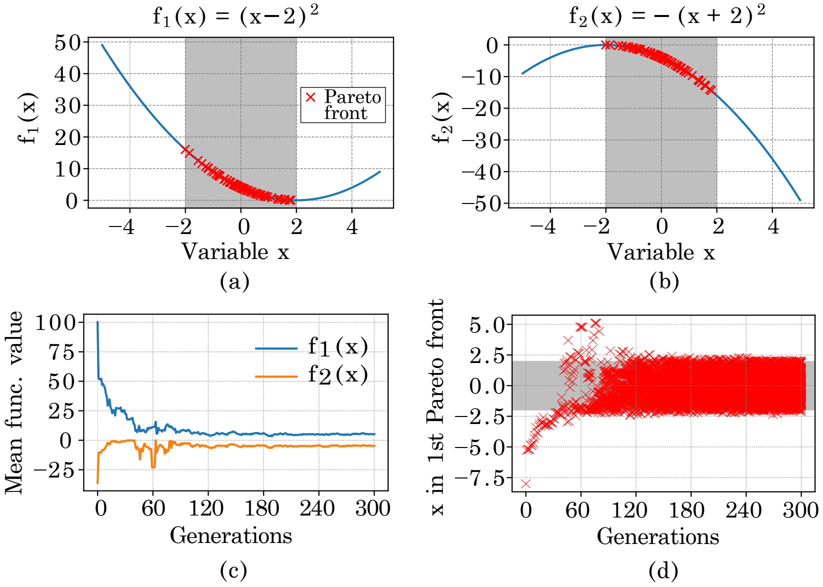


Figure 1.6: Solving $f_1(x) = (x-2)^2$ and $f_2(x) = -(x+2)^2$ using MOEAs (example). In (a) Pareto front for $f_1(x)$; (b) Pareto front for $f_2(x)$; (c) convergence; and (d) convergence of solutions in the 1st Pareto front.

This plot shows the *feasible objective space*, representing trade-offs between market cost and performance for the *set of solutions*. By setting objectives to minimise market cost *and* maximise performance, we identify *non-dominated solutions*, forming a *Pareto front*. Solutions on a Pareto front are optimal as they cannot improve in one objective without sacrificing the other (Deb, 2005).

MOEAs employ natural selection, where the best individuals are more likely to reproduce and pass on to the next generations (Ojha, Singh, Chakraborty, et al., 2019). The following example illustrates how they work. For functions $f_1(x) = (x-2)^2$ and $f_2(x) = -(x+2)^2$, the minimum and maximum values are at $x = 2$ and $x = -2$, respectively. However, when optimising both objectives *simultaneously*, we will obtain a set of non-dominated solutions instead. For this, we use a MOEA called the `NSGA-II` (Deb, 2005), and the results are shown in Figure 1.6(a)-(b). We observe that the first Pareto front ranges from -2 to $+2$ for both functions. This means that all solutions in this set are optimal for minimising $f_1(x)$ while maximising $f_2(x)$. Figure 1.6(c) displays convergence of the algorithm over generations, and Figure 1.6(d) shows x values in the first Pareto front across generations, showing convergence of x between -2 and $+2$.

1.2.3 Markov Process-based Prognostics

Prognostics: An overview

Prognostics—from the Greek *prognostikos*—is the cornerstone concept enabling predictability in PHM (Stage 4 in Figure 1.2). Within an engineering context, the science of prognostics attempts to answer: *how long will it be until a particular future event or state is reached?* (Goebel, Celaya, Sankararaman, et al., 2017). Therefore, the main aim of prognostics is to predict an event or state before its occurrence, making *time* a critical variable (Lee, F. Wu, Zhao, et al., 2014).

Among the most popular outcomes of prognostics is the estimation of the Remaining Useful Life (RUL), which measures the time until *failure*. However, estimating the time to reach alternative states to failure may be relevant for some applications, as discussed later in this section. This capability makes prognostics key within the PHM paradigm, enabling taking actions before failures occur, thus allowing better planning while minimising reactive costs and downtime (Elattar, Elminir, and Riad, 2016).

Engineering applications of prognostics are vast, including rotating machinery (Heng, S. Zhang, Tan, et al., 2009); Li-ion batteries (J. Zhang and Lee, 2011); gas turbines (Tahan, Tsoutsanis, Muhammad, et al., 2017); manufacturing (T. Xia, Dong, Xiao, et al., 2018); aircraft (Che, H. Wang, Fu, et al., 2019); and wind turbines (Rezamand, Kordestani, Carriveau, et al., 2020).

Prognostic models operate at both system and component levels (S. Kim, Choi, and N. H. Kim, 2021) and come in various types; however, the literature lacks consensus regarding their classifications (Mrugalska, 2019). Therefore, in this dissertation, we discuss the following categories: physics-based, data-driven, and hybrid. For completeness, the literature also mentions *knowledge-based* prognostics, though they are significantly less prevalent. Thus, we do not discuss them here. For more information, see Sikorska, Hodkiewicz, and Ma, 2011; J. Peng, G. Xia, Y. Li, et al., 2022; Xue, J. Yang, M. Yang, et al., 2023.

Physics-based prognostics—also referred to as *model-based* (Zio, 2022; Xue, J. Yang, M. Yang, et al., 2023)—use explicit mathematical representations to formalise physical failure modes and degradation phenomena. This requires a deep understanding of the system's physics, operating conditions, and life cycle loads (Elattar, Elminir, and Riad, 2016; Javed, Gouriveau, and Zerhouni, 2017; T. Xia, Dong, Xiao, et al., 2018). The process generally involves model identification, simulations under loads, tracking degradation measures, and predicting RUL (Cubillo, Perinpanayagam, and Esperon-Miguez, 2016).

These models are tailored to specific applications, such as crack growth, spall progression, and wear, relying on accurate parameterisation using laboratory or real-time data (Rezamand, Kordestani, Carriveau, et al., 2020; D. An, N. H. Kim, and Choi, 2015). Challenges in physics-based prognostics stem from the complexity of the physical failure modes and degradation phenomena, which may not be fully understood, dynamic, and highly non-linear, making their modelling difficult (Zio, 2022).

Data-driven prognostics have gained prominence in the PHM community with the rise of big data and industrial Internet of Things technologies (Xue, J. Yang, M. Yang, et al., 2023), owing to their low cost and ease of deployment, particularly when understanding of physics is limited (Elattar, Elminir, and Riad, 2016; Javed, Gouriveau, and Zerhouni, 2017; Guo, Z. Li, and M. Li, 2020; Heng, S. Zhang, Tan, et al., 2009). These methods extract degradation patterns from historical and condition monitoring data to predict future degradation or state behaviour (Tsui, N. Chen, Q. Zhou, et al., 2015; T. Xia, Dong, Xiao, et al., 2018; Zio, 2022).

The literature distinguish mainly two sub-classes of data-driven prognostics: Statistical models and Machine Learning models (Heng, S. Zhang, Tan, et al., 2009; D. An, N. H. Kim, and Choi, 2015; Elattar, Elminir, and Riad, 2016; Tahan, Tsoutsanis, Muhammad, et al., 2017; T. Xia, Dong, Xiao, et al., 2018; Guo, Z. Li, and M. Li, 2020). *Statistical*—also referred to as *Stochastic* (Prakash, Yuan, Hazra, et al., 2021)—models include Markov chains, Gaussian process regression, Gamma process, and Bayesian networks. *Machine Learning (ML)* models—also referred to as Artificial Intelligence (AI) techniques (Javed, Gouriveau, and Zerhouni, 2017)—include neural networks, support vector machines, and fuzzy logic, among others. As an observation, the distinction between statistical and ML models may be ambiguous, and a more thorough discussion around this classification is needed.

The shortcomings of data-driven prognostics include the dependency on the availability of extensive, multivariate data covering all operational phases and conditions of the system (Elattar, Elminir, and Riad, 2016; Zio, 2022), which is challenging in some cases, especially in safety-critical systems. Additionally, these types of models may be considered grey- or black-box models (e.g., neural networks) (Xue, J. Yang, M. Yang, et al., 2023), making interpretability difficult.

Hybrid-based prognostics—also known as *blended* (Xue, J. Yang, M. Yang, et al., 2023) or *fusion* (Elattar, Elminir, and Riad, 2016)—combine the strengths of physics-based and data-driven approaches to address their weaknesses. Javed, Gouriveau, and Zerhouni, 2017 identifies two types: *series* and *parallel*. In *series*, a physics-based model is updated with new data using an online parameter estimation technique. In *parallel*, a data-driven model predicts the residuals not explained by the first model. T. Xia, Dong, Xiao, et al., 2018 further classify hybrid prognostics into physics-based + data-driven and data-driven + data-driven. As an example, Rezamand, Kordestani, Carriveau, et al., 2020 provide a detailed analysis of the use of hybrid prognostics for wind turbine components. To date, hybrid-based prognostics receives increased attention as integrated solutions can lead to better problem-solving (Xue, J. Yang, M. Yang, et al., 2023).

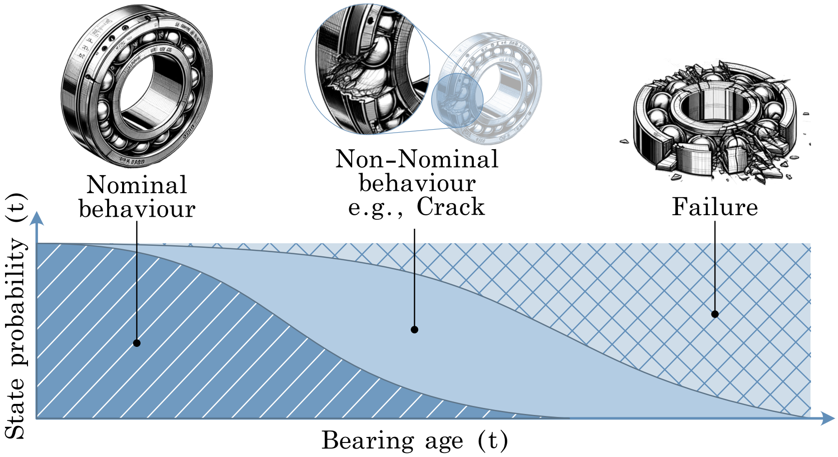


Figure 1.7: Multi-State Deterioration of a bearing with three states (example).

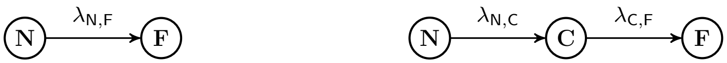
In this dissertation, we focus on *Markovian Process-based models* for prognostics—a sub-class of data-driven prognostics (Tsui, N. Chen, Q. Zhou, et al., 2015)—due to their ability to model alternative states to failure, crucial for safety-critical systems like bridges (Ranjith, Setunge, Gravina, et al., 2013) and components with slow degradation like sewer mains (Barraud, Bosco, Clemens-Meyer, et al., 2024). We use Markov chains for Multi-State Deterioration modelling, discussed next.

Markov chains for Multi-State Deterioration

Proposed by the Russian mathematician Andrey Markov, a *Markov process* models the behaviour of systems—physical or mathematical—based on *states* and their *transitions*. It assumes that the future evolution of the system depends only on its current state, not its past (Stewart, 2009).

This characteristic forms the basis of Markov *chains*, where state transitions create a *sequence* over time in which each future state depends only on the present state, disregarding prior events. Formal definitions of Markov chains are provided in Section II.4.1, and details of the Markov chain structures used in this dissertation are found in Section II.4.4. Here, we illustrate the concept behind Markov chains with the following example. Figure 1.3 depicts the reliability function of a bearing, where two states are distinguished: *nominal* (**N**) and *failure* (**F**). The underlying degradation process can be modelled using a two-state Markov chain, with a transition from **N** to **F**, as shown in Figure 1.8(a). In this example, the Markov chain does not allow repairs, i.e., once the state changes to failure, it cannot revert. However, repairs can still be modelled with Markov chains, though this would yield a different example.

By introducing an intermediate state, *non-nominal behaviour* (e.g., due to a crack), we enrich the degradation model, as depicted in Figure 1.7. This can be



(a) With 2 states

(b) With 3 states

Figure 1.8: Markov chains, where N: nominal behavior; C: non-nominal behavior e.g., due to a crack; F: failure; λ is the *hazard rate*.

modelled using a three-state Markov chain, as shown in Figure 1.8(b). Note that the y-axis in Figure 1.7 represents *state probability*, indicating the *probability* of being in a state given the bearing's age. For completeness, the Markov chain in Figure 1.7 operates in *continuous-time*, which differs from *discrete-time* Markov chains, which operate at specific equally spaced intervals. Further details are provided in Section II.4.1. Additionally, in Appendix A.1, this example is further elaborated with mathematical details.

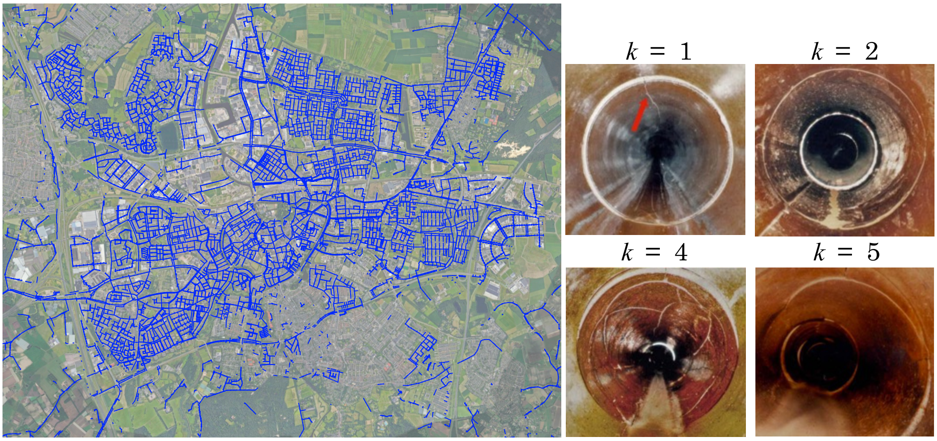
Regarding the application of Markov processes for prognostics applications, we encounter: X. Zhang, Xu, Kwan, et al., 2005; Tobon-Mejia, Medjaher, Zerhouni, et al., 2012 apply hidden Markov chains for degradation assessment and RUL estimation in bearings; Ranjith, Setunge, Gravina, et al., 2013 models four degradation condition states (good, condition, minor decay, decay, crushing) in timber bridge elements; D. Zhou, Yu, H. Zhang, et al., 2016 uses Markov chains for degradation assessment in gas turbines; J. Chiachío, Jalón, M. Chiachío, et al., 2020 proposes a Markov chains prognostics framework and uses a case study on fatigue crack propagation to perform stochastic damage modelling, estimating time-dependent reliability; Tanwar, H. Park, and Raghavan, 2021 uses hidden Markov chains to model the degradation in lubricating oil.

In this dissertation, we use Markov chains for Multi-State Deterioration (MSD) modelling, where the states in the model explicitly represent a well-defined state associated with deterioration. Our case study focuses on sewer networks, where the inspection data contains damage *severities*. More on this case study is presented in the next section.

Modelling deterioration in sewer mains

Sewer mains are vital components of urban infrastructure, necessary for maintaining sanitary conditions and public health (M.A. Cardoso and Silva, 2016). Figure 1.9(a) illustrates the sewer network system of Breda (in blue), The Netherlands. This network comprises approximately 25,000 pipes of various materials and functionalities, covering an area of 10 by 12 kilometres, highlighting the system's large scale. Figure 1.9(b) displays the data type collected via Closed Circuit Television camera inspections, with severities classified according to the EN 13508:1 norm. Here, a higher severity level (k) indicates larger and more pronounced cracks in the sewer main.

Degradation modelling in sewer mains is crucial for prioritising them in annual programming and renovation activities; however, developing accurate deterioration



(a) (b)

Figure 1.9: Sewer main system and damage severities. (a) Sewer network (in blue) of the city of Breda, The Netherlands (*View of Breda* 2024). (b) Cracks of different severity levels (indicated with k) taken from the (EN 13508:1).

models is particularly challenging. The complexity arises from several factors: the inability to conduct run-to-failure experiments due to the decades-long evolution of various deterioration modes; the vast scale and complexity of sewer systems combined with limited knowledge of the various failure mechanisms (Barraud, Bosco, Clemens-Meyer, et al., 2024); and the lack of reliable, high-quality data. This data scarcity is aggravated by the “data-loop problem”, where insufficient data makes it difficult to demonstrate its value, which in turn remains difficult to obtain without adequate data (Cherqui, Clemens-Meyer, Tscheikner-Gratl, et al., 2024). Furthermore, this lack of reliable data introduces various types of biases, a topic discussed in detail in Auger, Besnier, Bijnen, et al., 2024.

Current attempts to model the deterioration of sewer mains are classified into physics-based, Machine Learning-based, and probabilistic models (Hawari, Alkadour, Elmasry, et al., 2020; Saddiqi, Zhao, Cotterill, et al., 2023). However, the literature does not conclusively determine which type of model is best for sewer main deterioration modelling (Zeng, Z. Wang, H. Wang, et al., 2023).

1.2.4 Maintenance Optimisation

According to ISO 14224:2016 and CEN-EN:13306, *maintenance* is *“the mix of all technical, administrative, and managerial actions, aimed at retaining or restoring an item to a state in which it can perform as required”*.

Discussed since the 1960s, Maintenance Optimisation (MO) is the systematic improvement of maintenance activities, focusing on answering *when* and *what* to maintain (Arts, Boute, Loeys, et al., 2024). MO involves developing and analysing mathematical models to derive maintenance strategies (de Jonge and Scarf, 2020), where both costs and benefits are quantified, and an optimal balance between the two is obtained (Dekker, 1996). This may result in reduced maintenance costs, extended asset life, maximised availability, and ensured workplace safety (Ogunfowora and Najjaran, 2023).

The set of *rules* that provide guidance for effective maintenance management is the aim of a *maintenance policy*, and the framework to optimise such policy is called Maintenance Policy Optimisation (MPO). There are different types of maintenance policies, such as reactive, proactive, and aggressive (for more, see Tinga, 2013), which are tailored to address varying operational demands and asset conditions.

Applications of MO include manufacturing (Chin, Varbanov, Klemeš, et al., 2020; X. An, G. Si, T. Xia, et al., 2022), energy (Shafiee and Sørensen, 2019; Bermejo, Gomez Fernandez, Pino, et al., 2019), and civil infrastructure, such as wind farms (J. Xia and Zou, 2023), pavements (W. Chen and Zheng, 2021; Pourgholamali, Labi, and K. C. Sinha, 2023), bridges (Frangopol and Bocchini, 2012), railways (Guler, 2016), sewer mains (Obradović, Šperac, and Marenjak, 2019), and nuclear power plants (Lapa, Pereira, and A. Mol, 2000). MO also extends beyond engineering to applications in the healthcare domain (Mahfoud and Biyaali, 2018).

Dealing with MO is a complex optimisation problem due to factors such as uncertainty, scalability, dynamic environments, high-dimensional search spaces, multiple and conflicting objectives. Consequently, numerous approaches have been explored to address MO, and the literature extensively covers these methods. The most relevant reviews include: Deshmukh, Sharma, and Yadava, 2011, which distinguishes between qualitative and quantitative (also noted in Y. Sinha and Steel, 2015), as well as discrete- and continuous-time MO, providing 13 types of model classifications.

Ding and Kamaruddin, 2015 categorises models into three types: *certainty*, *risk*, and *uncertainty*, each with further sub-categories. The degree of certainty corresponds to the information available regarding the states of nature that influence the system under optimisation analysis. Goyal, Pabla, Dhami, et al., 2017 discusses *soft-computing* as an approach that mimics the human mind's ability to handle uncertainty and imprecision, aiming for adaptability, robustness, and cost-effectiveness in reaching near-optimal solutions.

Shafiee and Sørensen, 2019 enlists different solution techniques, including Markov models, operations research models (e.g., dynamic programming), Petri nets, simulations (e.g., using Monte Carlo sampling), Bayesian networks, fuzzy models, and Intelligent-based models (e.g., employing ML and Expert Systems). Also discusses group maintenance and opportunistic replacement. Syan and Ramsoobag, 2019 focuses on multi-criteria optimisation, often dealing with resource constraints and conflicting objectives, and discusses non-evolutionary and evolutionary algorithms.

Pourgholamali, Labi, and K. C. Sinha, 2023 identifies the *objective function*, *constraints*, and *decision variables* as key components of MO and distinguishes between single- and multi-objective MO, as well as exact solutions and heuristics/meta-heuristics. de Jonge and Scarf, 2020 explores MO for single- and multi-unit systems with continuous and discrete condition states, considering economic and stochastic dependencies. Ogunfowora and Najjaran, 2023 further discusses these aspects with a focus on the Reinforcement Learning paradigm.

J. Xia and Zou, 2023 proposes a framework for maintenance using digital twins. Dui, X. Wu, S. Wu, et al., 2024 explores how to perform MO based on *importance measures* (e.g., based on Minimal Cut Set in a Fault Tree), including ML- and Deep Learning-based approaches, highlighting prolonged training and computational times as major drawbacks.

Arts, Boute, Loeys, et al., 2024 identifies two main paradigms for MO: *renewal reward theory* and Markov Decision Processes (MDPs). The first identifies repeating cycles in a stochastic system for which a decision rule has been established, while the second accounts for possible system states and decision options. In this dissertation, we use MDPs, detailed in the next section.

Markov Decision Processes

A Markov Decision Process (MDP) is a stochastic process in which changes of state occur according to a Markov chain (Ding and Kamaruddin, 2015), and it can be used to prove that a certain type of decision rule is optimal (Arts, Boute, Loeys, et al., 2024). In an MDP, the available actions, rewards, and transition probabilities depend solely on the current state and action, not on past states or actions, making MDPs broad enough to model most realistic sequential *decision-making* problems (Puterman, 2014).

Section III.4.1 provides formal definitions of MDPs, and here we offer an example to illustrate this concept. Figure 1.10(a) depicts a 3 by 3 grid-world problem where a mouse navigates through the grids, potentially encountering traps or cheese.

To model this problem as an MDP, we must define the four main components in an MDP: *states*, *actions*, *transition probabilities*, and *rewards*. *States* represent a specific condition or status of the system being modelled; for example, Figure 1.10(b) shows the 9 possible positions in which the mouse can be at a given time. This can be represented by a vector including positions A1, A2, . . . , C3. *Actions* are choices made from a set of allowable options in a given state; for instance, Figure 1.10(c) shows the possible actions the mouse can take, such as moving *up*, *down*, *right*, or *left*.

Transition probabilities describe the likelihood of moving from one state to another given a particular action. For example, Figure 1.10(d) illustrates that the mouse has an 80% probability of moving from grid C3 to grid B3 by going

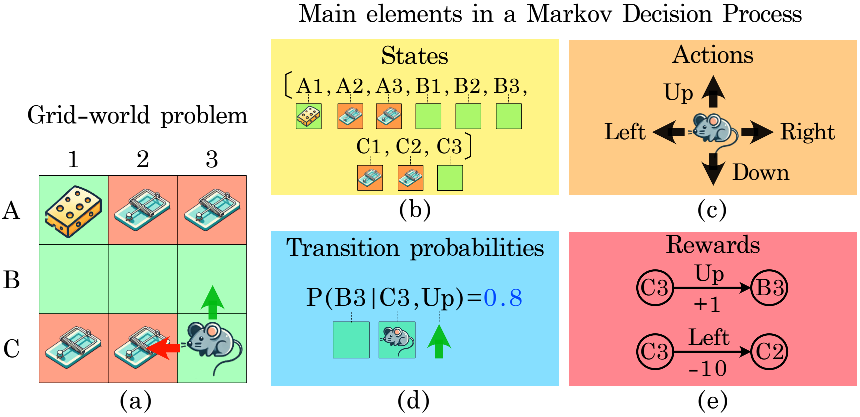


Figure 1.10: Modelling a grid-world problem using an MDP (example).

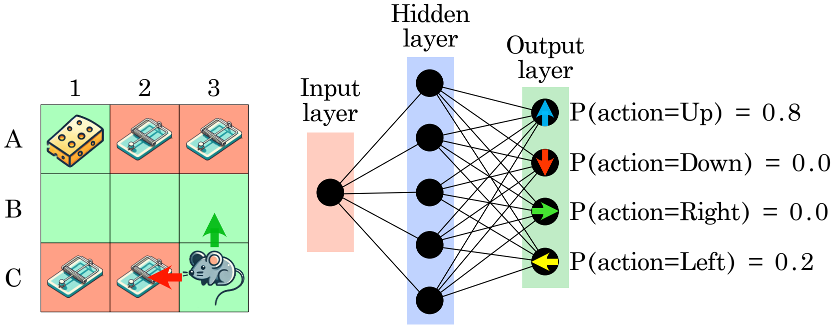
Up. *Rewards* are scalar feedback signals assigned to state transitions, quantifying the immediate outcome of an action in a given state. For instance, in Figure 1.10(e), if the mouse moves from C3 to B3, it receives a +1 reward, indicating a favourable action, while moving from C3 to C2 results in a −10 reward, indicating an undesired action leading to a trap.

As mentioned, the MDP enables the formulation of the optimisation problem, but 'solving' the MDP is the next step to obtain the desired *optimal policy*, which, for the example in Figure 1.10(a), is: [*Up*, *Left*, *Left*, *Up*]. If the mouse executes these actions, it will get the cheese! However, how can we mathematically obtain this? There are different algorithms for this, including *value* and *policy* iteration, *linear* programming (Puterman, 2014), *dynamic* programming (Bertsekas, 2012), and *Reinforcement Learning* (Sutton and Barto, 2018). In this dissertation, we use Deep Reinforcement Learning, which we discuss in the next section.

Deep Reinforcement Learning

Before delving into Deep Reinforcement Learning (DRL), it is important to first consider Reinforcement Learning (RL). RL is a type of ML paradigm, distinct from supervised and unsupervised learning, in that it involves an agent learning behaviour through trial and error by interacting with a virtual environment (Kaelbling, Littman, and A. W. Moore, 1996). Sutton and Barto, 2018 defines RL as "*learning what to do—how to map situations to actions—so as to maximise a numerical reward signal*". For further details on RL, we recommend the latter reference.

The concept of RL solidified in the 1980s and has gained significant momentum since 2017 with the advent of Deep Reinforcement Learning (DRL). DRL emerged with the aim of revolutionising the field of AI by equipping RL with the capabilities of Deep Neural Networks (DNNs), making RL more scalable and capable of handling complex problems. This has led to popular applications in robotics and



(a) Grid-world problem (b) Policy Network

Figure 1.11: Solving grid-world problem using an DRL (example).

gaming (Arulkumaran, Deisenroth, Brundage, et al., 2017), as well as applications in Maintenance Management (Fink, Q. Wang, Svensén, et al., 2020; Ogunfowora and Najjaran, 2023; Siraskar, Kumar, Patil, et al., 2023). X. Wang, S. Wang, Liang, et al., 2024 classify DRL algorithms into three categories: *value-based*, *policy-based*, and *maximum-entropy-based*, where the nature of the classification resides primarily in the learning objectives.

Section III.4.2 provides formal definitions of DRL; here, we present an example to illustrate this concept. Figure 1.11(a) presents the same grid-world problem discussed earlier. Figure 1.11(b) demonstrates how DRL could be applied to "solve" the MDP. Here, a Neural Network (NN) is shown, conceptually representing the mouse's behaviour in the grid world; essentially, the NN *is* the virtual agent.

The *state*—the mouse's current position on the grid—is encoded in the *input layer*, and this information is propagated through the *hidden layers*—a series of mathematical transformations—to the output layer, where four nodes correspond to the possible actions the agent can take. As mentioned, various DRL algorithms can train the agent. After training, the virtual mouse should ideally navigate the grid-world using the optimal policy; for the state in Figure 1.11(a), the NN suggests moving *Up* with 80% probability, bringing the mouse a step closer to the cheese!

The background and examples provided in this section should adequately convey the philosophy behind the key concepts in this dissertation. In the next section, we will discuss the research gaps addressed by this work.

1.3 Research gaps

The previous section provided an overview of the main concepts and relevant literature used in this dissertation, with PHM as the overarching concept. This dissertation is organised into three parts: **Part I**—with the main concepts discussed in Section 1.2.2—addresses *how to obtain efficient and compact Fault Tree models* *from failure datasets in a robust and scalable manner*; **Part II**—with the main concepts discussed in Section 1.2.3—focuses on *how and to what extent it is possible to accurately model Multi-State Deterioration with applications in sewer mains*; and **Part III**—with the main concepts discussed in Section 1.2.4—explores *how to devise optimal maintenance strategies for components with Multi-State Deterioration such as sewer mains using Deep Reinforcement Learning*. Each problem and its associated research gaps are further detailed in the subsequent sections.

Part I: Data-driven Inference of Fault Tree models

One challenge in Fault Tree Analysis (FTA) is constructing accurate, efficient, and reliable Fault Tree (FT) models. This process is referred to as *construction*, *synthesis*, or *induction* of FTs (Salem, Apostolakis, and Okrent, 1976; Hunt, Kelly, Mullhi, et al., 1993; Madden and Nolan, 1994). In this research, we refer to it as *inference* of FT models, which involves deducing the structure of a FT based on observed data, identifying relationships and dependencies among basic and intermediate events.

Research in this area is expanding, driven by the principles of Industry 4.0 (Oztemel and Gursev, 2020) and 5.0 (Maddikunta, Pham, B, et al., 2022), where modern societies are increasingly enhancing processes and automation, partly by leveraging large, structured datasets. The inference of FT models has been explored since the 1970s through three main approaches: knowledge-based, model-based, and data-driven. These methods are discussed in detail in Section I.3, with an overview provided below.

Knowledge-based approaches, relying on semi-automated techniques using a knowledge base and heuristics, were among the first explored, but they are limited by experts' knowledge, which may be biased and incomplete, potentially missing unseen relations in the FT models. *Model-based* approaches involve translating existing system models into FTs, utilising frameworks like Simulink (Karris, 2006) and SysML (Friedenthal, A. Moore, and Steiner, 2015), but they require pre-existing system models. *Data-driven* approaches have gained prominence with the growth in data collection, automatically generating FTs by analysing structured datasets.

In Part I of this dissertation, we focus on data-driven approaches for the inference of FT models, as these methods may require minimal domain expertise and reduce human intervention in identifying causal relationships within the data. However, we identify challenges in terms of scalability, robustness and completeness.

Scalability is the ability to maintain efficiency and performance as the dataset size increases. In our context, it denotes the algorithm's capacity to infer a FT structure for larger datasets (e.g., more basic events) without significant resource consumption increases. *Robustness* is the ability to perform consistently across diverse inputs. A robust algorithm should reliably produce correct and consistent FT models, even with noisy data or when re-evaluated on the same dataset. *Completeness* is the challenge of generating FT structures that include all failure modes present in the data.

Part II: Multi-state deterioration modelling

Approaches to model deterioration in sewer main systems can be categorised into three classes: physics-based, Machine Learning (ML), and probabilistic models (Hawari, Alkadour, Elmasry, et al., 2020; Saddiqi, Zhao, Cotterill, et al., 2023). In Part II.3, we provide an overview of the related literature. Conceptually, the most robust approach is *physics-based* models, as they utilise fundamental physics laws to describe degradation processes. However, due to epistemic uncertainty, applying these models in diverse contexts is challenging, as various factors influencing degradation may not be considered (Ana and Bauwens, 2010).

With the increasing availability of diverse datasets, ML models have potential applications in assessing sewer mains conditions by identifying useful relationships within the data (Zeng, Z. Wang, H. Wang, et al., 2023). However, challenges in the quality of data collected from sewer main systems (Auger, Besnier, Bijnen, et al., 2024) can negatively impact the performance of ML models. Moreover, these models are inadequate for long-term condition assessment (Kantidakis, Putter, Litière, et al., 2023) when they fail to account for properties such as *monotonicity*—the consistent change (increase or decrease) of degradation metrics over time due to factors like wear.

Even though *probabilistic* models can also be learned from data, their mathematical foundations make them more suitable for long-term risk assessment (El Morer, Wittek, and Rausch, 2024). Most probabilistic approaches to model degradation in sewer mains are based on *Markov chains* (Ana and Bauwens, 2010), where these models explicitly represent *damage severity levels* through well-defined discrete states, information collected through CCTV cameras and classified based on norms such as the EN 13508:1.

For using Markov chains in the stochastic degradation of sewer mains, we identified gaps in case studies, Markov chain structure, assumptions, and calibration. For *case studies*, the research community needs to share existing studies to enhance evidence on sewer main degradation models (Tscheikner-Gratl, Caradot, Cherqui, et al., 2019). Regarding Markov chains, assumptions about their *structure* when modelling transitions between severity levels and the *time homogeneity* assumption—indicating that stochastic transitions between states are time-independent—lack comparisons. Calibrating Markov chains for efficient convergence and addressing dataset issues, such as data censorship related to the imprecise dating of transitions between condition states (Cherqui, Clemens-Meyer, Tscheikner-Gratl, et al., 2024), remains an under-explored research direction.

Part III: Maintenance optimisation of multi-state components

A follow-up research direction from the modelling of components with multi-state deterioration is how to use this information to support maintenance decision-making. In our case study of sewer main systems, we find that asset management has been approached in various ways, as detailed in Section III.3. These approaches include risk-based methods (Lee, C. Y. Park, Baek, et al., 2021), multi-objective optimisation problems (Elmasry, Zayed, and Hawari, 2019), Markov Decision Processes (Wirahadikusumah and Abraham, 2003), considering the network structure (Qasem and Jamil, 2021), and ML-based methods (Marc Ribalta and Rubión, 2023).

However, techniques such as RL, widely used for MPO applications (Ogunfowora and Najjaran, 2023; Marugán, 2023), are still largely under-explored for sewer asset management. Current applications of RL to sewer asset management mainly correspond to control problems (Yin, Leon, Sharifi, et al., n.d.) and grouping of maintenance actions (Kerkkamp, Bukhsh, Y. Zhang, et al., 2022).

A significant challenge in developing maintenance strategies for sewer pipes arises from the complex state space, defined by various system parameters, which often experiences “dimensionality explosion”. This complexity renders traditional exhaustive search techniques and exact solvers, such as STORM (Hensel, Junges, Katoen, et al., 2022), inefficient. Consequently, there is a pressing need for methods capable of managing larger state spaces and delivering near-optimal policies by accounting for system characteristics. Our research addresses this gap by employing Deep Reinforcement Learning (DRL) for MPO in sewer systems. While it is well-known that DRL does not guarantee globally optimal policies, we explore the effectiveness of these techniques for the MPO of sewer mains. We consider the stochastic nature of the Multi-State Deterioration, evaluate the impact of different degradation models on policy effectiveness, and take into account the context of the components.

1.4 Research questions

Each gap covered in this thesis has a dedicated part with the following general research questions:

- **Part I**: *how to obtain efficient and compact Fault Tree models from failure datasets in a robust and scalable manner?*
- **Part II**: *how and to what extent is it possible to accurately model Multi-State Deterioration with applications in sewer mains?*
- **Part III**: *how can optimal maintenance strategies be devised for components with Multi-State Deterioration such as sewer mains using Deep Reinforcement Learning?*

It is worth noting that the answers from Part II on sewer mains degradation modelling are utilised in Part III on maintenance optimisation. In the following section, we discuss the research methodology employed in this dissertation.

1.5 Research methodology

The methodology to answer the research questions in Section 1.4 is divided into three main phases (Figure 1.12), with each phase associated with a part of the dissertation.

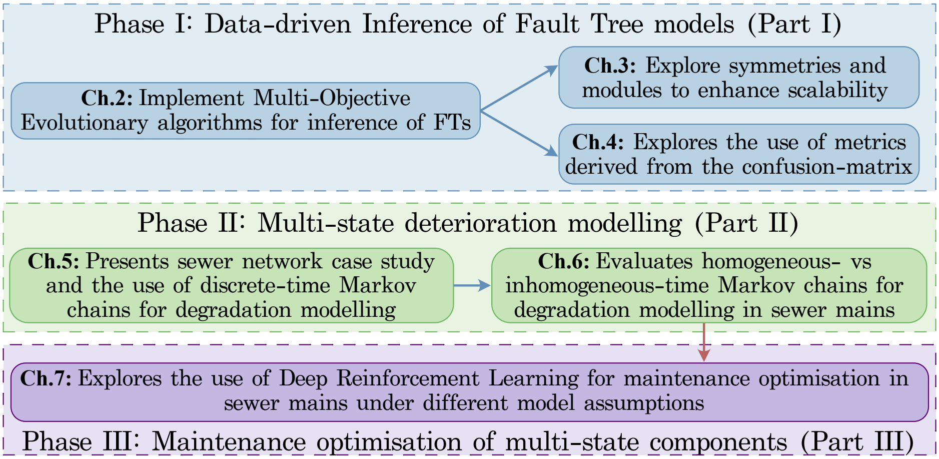

Phase I: Data-driven Inference of Fault Tree models (Part I)
Ch.2: Implement Multi-Objective Evolutionary algorithms for inference of FTs
Ch.3: Explore symmetries and modules to enhance scalability
Ch.4: Explores the use of metrics derived from the confusion-matrix
Phase II: Multi-state deterioration modelling (Part II)
Ch.5: Presents sewer network case study and the use of discrete-time Markov chains for degradation modelling
Ch.6: Evaluates homogeneous- vs inhomogeneous-time Markov chains for degradation modelling in sewer mains
Ch.7: Explores the use of Deep Reinforcement Learning for maintenance optimisation in sewer mains under different model assumptions
Phase III: Maintenance optimisation of multi-state components (Part III)

Figure 1.12: General methodology used in this dissertation.

Phase I focuses on the data-driven inference of Fault Tree (FT) models (Part I). For this, we employ multi-objective optimisation and evolutionary algorithms, leading to the development of the `FT-MOEA` algorithm (Jimenez-Roa, Heskes, Tinga, et al., 2023) (see Chapter 2). This phase produced two extensions: the investigation of symmetries and modules to enhance scalability in systems with symmetrical failure data, resulting in the `SymLearn` tool-chain (see Chapter 3); and the exploration of metrics derived from the confusion matrix to improve scalability and robustness (see Chapter 4).

Phase II centres on multi-state deterioration modelling (Part II). We initially provide a case study from a sewer network in the Netherlands, exploring discrete Markov chains to model deterioration in sewer mains across different cohorts (see Chapter 5). This work is further extended by comparing and evaluating homogeneous and inhomogeneous time Markov chains (see Chapter 6).

Phase III concentrates on policy optimisation of multi-state components (Part III). Building on the degradation models developed in Phase II, we explore the application of these models (see relations in Figure 1.12) within the context of maintenance policy optimisation. The problem is framed using Markov Decision Process and addressed using Deep Reinforcement Learning, with an evaluation of different assumptions in the degradation model (see Chapter 7).

1.6 Thesis outline

Figure 1.13 provides an overview of the sections and their respective chapters. Chapter 1 introduces and frames the research. Chapters 2 to 7 consist of peer-reviewed journal or conference papers. Chapter 8 presents the general discussion, while Chapter 9 offers the conclusion. The primary contributions of this dissertation are outlined in the following section.

1.7 Main contributions

Contributions on Reliability Modelling: Data-driven Inference of Fault Tree models

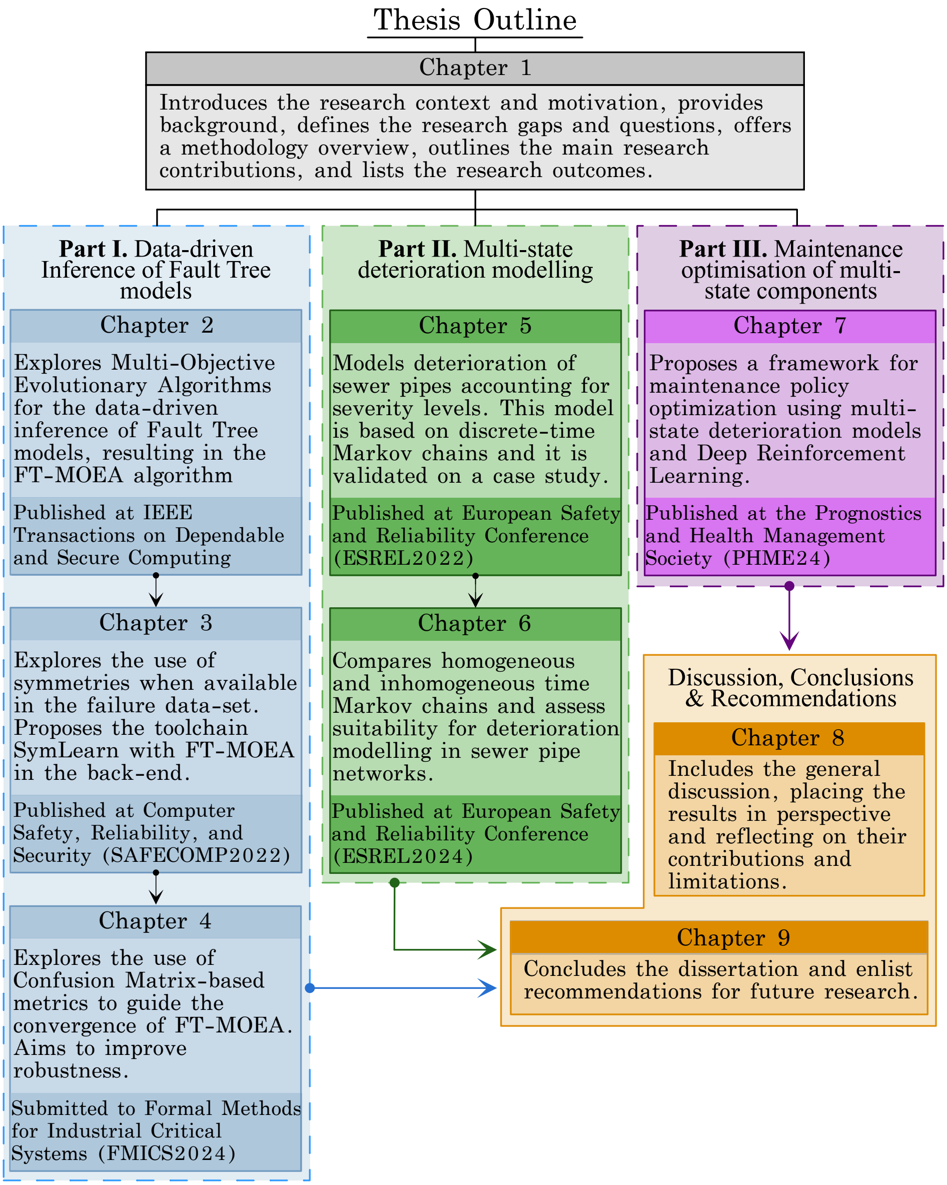
In Part I of this dissertation, we explored, for the first time, Multi-Objective Evolutionary Algorithms (MOEAs) to automatically infer FTs from failure datasets. In the domain of reliability modelling, our contributions are three-fold:

1. The `FT-MOEA` algorithm (Chapter 2), based on a MOEA, accounts for three optimisation metrics, including minimising FT size and accuracy-related error metrics. With `FT-MOEA`, we can consistently obtain compact FT structures. Data and implementations are available at `zenodo.org/record/5536431`.
2. The `SymLearn` toolchain (Chapter 3) employs a ‘divide and conquer’ strategy, exploiting symmetries and modules that may be present in the failure dataset. With `SymLearn`, we can handle larger problems and thus improve scalability. Data and implementations are available at `zenodo.org/record/5571811`.
3. The `FT-MOEA-CM` extension (Chapter 4) expands the multi-objective optimisation function by incorporating metrics computed from the confusion matrix. With `FT-MOEA-CM`, we improved robustness by consistently achieving global optima for larger problems. Data and implementations are available at `https://gitlab.utwente.nl/fmt/fault-trees/ft-moea`.

Our findings suggest that using MOEAs for the inference of FT models generally has a positive impact in terms of robustness, scalability, and convergence speed.

Contributions on Markov Process-based Prognostics: Multi-state deterioration modelling

In Part II, we used Markov chains to model Multi-State Deterioration (MSD) in sewer mains. Our contributions are three-fold:



Thesis Outline

Chapter 1

Introduces the research context and motivation, provides background, defines the research gaps and questions, offers a methodology overview, outlines the main research contributions, and lists the research outcomes.

Part I. Data-driven Inference of Fault Tree models

Chapter 2

Explores Multi-Objective Evolutionary Algorithms for the data-driven inference of Fault Tree models, resulting in the FT-MOEA algorithm

Published at IEEE Transactions on Dependable and Secure Computing

Chapter 3

Explores the use of symmetries when available in the failure data-set. Proposes the toolchain SymLearn with FT-MOEA in the back-end.

Published at Computer Safety, Reliability, and Security (SAFECOMP2022)

Chapter 4

Explores the use of Confusion Matrix-based metrics to guide the convergence of FT-MOEA. Aims to improve robustness.

Submitted to Formal Methods for Industrial Critical Systems (FMICS2024)

Part II. Multi-state deterioration modelling

Chapter 5

Models deterioration of sewer pipes accounting for severity levels. This model is based on discrete-time Markov chains and it is validated on a case study.

Published at European Safety and Reliability Conference (ESREL2022)

Chapter 6

Compares homogeneous and inhomogeneous time Markov chains and assess suitability for deterioration modelling in sewer pipe networks.

Published at European Safety and Reliability Conference (ESREL2024)

Part III. Maintenance optimisation of multi-state components

Chapter 7

Proposes a framework for maintenance policy optimization using multi-state deterioration models and Deep Reinforcement Learning.

Published at the Prognostics and Health Management Society (PHME24)

Discussion, Conclusions & Recommendations

Chapter 8

Includes the general discussion, placing the results in perspective and reflecting on their contributions and limitations.

Chapter 9

Concludes the dissertation and enlist recommendations for future research.

Figure 1.13: Thesis outline: Overview.

1. We present a real-world case study from the Netherlands (Section II.4.3). Part of the data is publicly available at `zenodo.org/record/6535853`.
2. We evaluate two types of Markov chain structures (Chapter 5) typically used for MSDM in sewer mains, discussing their benefits and drawbacks. Additionally, we extend and propose a Markov chain structure (Chapter 6) that accounts for functional failure states.
3. We compare the assumptions of homogeneous and inhomogeneous time Markov chains (Chapter 6), identifying inhomogeneous-time Markov chains as more suitable for long-lived assets like sewer mains. Data and implementations are available at `https://gitlab.utwente.nl/fmt/degradation-models/ihctmc`.

Our findings suggest that using Markov chains for MSDM in sewer mains has the potential to infer the severity level across populations of sewer mains.

Contributions on Maintenance Optimisation: Maintenance optimisation of multi-state components

In Part III, we used Multi-State Deterioration Model (MSDM) and Deep Reinforcement Learning to devise component-level strategic maintenance planning with applications in sewer mains. Our contributions are two-fold:

1. In Chapter 7, we propose a DRL framework for devising maintenance policies at the pipe level, considering MSDM. We detail model calibration and have made our models and dataset publicly available in the repository: `zenodo.org/records/11258904`.
2. We evaluate the influence of homogeneous and inhomogeneous MSDM on devising strategic maintenance, comparing agent behaviours against well-known maintenance policy heuristics.

Our findings suggest that DRL offers a flexible framework with untapped potential for maintenance strategies, and it is crucial to integrate degradation model assumptions, as they significantly influence policy behaviour.

1.8 List of publications

Eight scientific papers were generated during this doctoral project. This dissertation compiles six of these, all published in peer-reviewed journals and conferences.

Journal papers

8. **L. A. Jimenez-Roa**, T. Heskes, T. Tinga and M. Stoelinga, *"Automatic Inference of Fault Tree Models via Multi-Objective Evolutionary Algorithms",* in IEEE Transactions on Dependable and Secure Computing, vol. 20, no. 4, pp. 3317-3327, 1 July-Aug. 2023, doi: 10.1109/TDSC.2022.3203805.

Peer-reviewed conference papers

7. **L. A. Jimenez-Roa**, T. D. Simão, Z. Bukhsh, T. Tinga, H. Molegraaf, N. Jansen, M. Stoelinga, *"Maintenance Strategies for Sewer Pipes with Multi-State Degradation and Deep Reinforcement Learning",* in *PHM Society European Conference*, vol. 8, no. 1, pp. 14, 2024, doi:10.36001/phme.2024.v8i1.4091.

6. **L. A. Jimenez-Roa**, N. Rusnac, M. Volk, M. Stoelinga, *"Fault Tree inference using Multi-Objective Evolutionary Algorithms and Confusion Matrix-based metrics",* in Formal Methods for Industrial Critical Systems (FMICS), 2024. Springer's Lecture Notes in Computer Science (LNCS). doi: 10.1007/978-3-031-68150-9_5.

5. **L. A. Jimenez-Roa**, T. Heskes, T. Tinga, M. Stoelinga, *"Comparing Homogeneous and Inhomogeneous Time Markov Chains for Modelling Deterioration in Sewer Pipe Networks",* in 34th European Safety and Reliability Conference, ESREL 2024: Advances in Reliability, Safety and Security, 2024, arxiv.org/abs/2407.12557.

4. **L. A. Jimenez-Roa**, M. Volk, M. Stoelinga, *"Data-Driven Inference of Fault Tree Models Exploiting Symmetry and Modularisation",* in Trapp, M., Saglietti, F., Spisländer, M., Bitsch, F. (eds) Computer Safety, Reliability, and Security. SAFECOMP 2022. Lecture Notes in Computer Science, vol 13414. Springer, Cham. doi: 10.1007/978-3-031-14835-4_4.

3. **L. A. Jimenez-Roa**, T. Heskes, T. Tinga, H. J. A. Molegraaf and M. Stoelinga, *"Deterioration Modelling of Sewer Pipes via Discrete-Time Markov Chains: A Large-Scale Case Study in the Netherlands",* in 32nd European Safety and Reliability Conference, ESREL 2022: Understanding and Managing Risk and Reliability for a Sustainable Future, pp. 1299-1306. 2022, doi:10.3850/978-981-18-5183-4_R22-13-482-cd.

Not included in this thesis

2. R. Stribos, R. Bouman, **L. A. Jimenez-Roa**, M. Slot and M. Stoelinga. *"A Comparison of Anomaly Detection Algorithms with applications on Recoater Streaking in an Additive Manufacturing Process"* submitted to Rapid Prototyping Journal, August 2024, doi:10.1108/RPJ-03-2024-0125.

1. **L. A. Jimenez-Roa**, T. Heskes and M. Stoelinga *"Fault Trees, Decision Trees, And Binary Decision Diagrams: A Systematic Comparison,"* in *Proceedings of the 31st European Safety and Reliability Conference (ESREL 2021)*, pp. 673-680, doi:10.3850/978-981-18-2016-8_241-cd.

1.9 References

An, D., N. H. Kim, and J.-H. Choi (2015). “Practical options for selecting data-driven or physics-based prognostics algorithms with reviews”. In: *Reliability Engineering & System Safety* 133, pp. 223–236. ISSN: 0951-8320. DOI: 10.1016/j.ress.2014.09.014.

An, X., G. Si, T. Xia, Q. Liu, Y. Li, and R. Miao (2022). “Operation and Maintenance Optimization for Manufacturing Systems with Energy Management”. In: *Energies* 15.19. ISSN: 1996-1073. DOI: 10.3390/en15197338.

Ana, E. and W. Bauwens (2010). “Modeling the structural deterioration of urban drainage pipes: the state-of-the-art in statistical methods”. In: *Urban Water Journal* 7.1, pp. 47–59. DOI: 0.1080/15730620903447597.

Arts, J., R. N. Boute, S. Loeys, and H. E. van Staden (2024). “Fifty years of maintenance optimization: Reflections and perspectives”. In: *European Journal of Operational Research.* ISSN: 0377-2217. DOI: 10.1016/j.ejor.2024.07.002.

Arulkumaran, K., M. P. Deisenroth, M. Brundage, and A. A. Bharath (2017). “Deep Reinforcement Learning: A Brief Survey”. In: *IEEE Signal Processing Magazine* 34.6, pp. 26–38. DOI: 10.1109/MSP.2017.2743240.

Assaf, R. (2018). *Prognostics and Health Management for Multi-component Systems.* University of Salford (United Kingdom). URL: https://books.google.nl/books?id=1HBSwAEACAAJ.

Auger, S., J.-B. Besnier, M. van Bijnen, F. Cherqui, G. Chuzeville, F. Clemens-Meyer, M. G. Jaatun, J. Langeveld, Y. Le Gat, S. Moin, G. E. Oosterom, W. van Riel, B. Roghani, M. M. Rokstad, J. Røstum, F. Tscheikner-Gratl, and R. Ugarelli (June 2024). “Data management and quality control”. In: *Asset Management of Urban Drainage Systems: If anything exciting happens, we’ve done it wrong!* IWA Publishing. ISBN: 9781789063059. DOI: 10.2166/9781789063059_0299.

Barraud, S., E. Bosco, F. Clemens-Meyer, G. Fernandes, Y. Le Gat, C. de Haan, R. Luimes, K. Makris, F. Rooyackers, I. Scheperboer, J. Skjelde, and A. Suiker (June 2024). “Deterioration processes and modelling in urban drainage systems”. In: *Asset Management of Urban Drainage Systems: If anything exciting happens, we’ve done it wrong!* IWA Publishing. ISBN: 9781789063059. DOI: 10.2166/9781789063059_0131.

Bermejo, J., J. F. Gomez Fernandez, R. Pino, A. Crespo Marquez, and A. J. Guillén Lopez (Oct. 2019). “Review and Comparison of Intelligent Optimization Modelling Techniques for Energy Forecasting and Condition-Based Maintenance in PV Plants”. In: *Energies* 12, p. 4163. DOI: 10.3390/en12214163.

Bertsekas, D. (2012). *Dynamic Programming and Optimal Control: Volume I.* Athena Scientific optimization and computation series. Athena Scientific. ISBN: 9781886529434. URL: https://books.google.nl/books?id=qVBEEAAAQBAJ.

Che, C., H. Wang, Q. Fu, and X. Ni (2019). “Combining multiple deep learning algorithms for prognostic and health management of aircraft”. In: *Aerospace Science and Technology* 94, p. 105423. ISSN: 1270-9638. DOI: https://doi.org/10.1016/j.ast.2019.105423.

Chen, W. and M. Zheng (2021). “Multi-objective optimization for pavement maintenance and rehabilitation decision-making: A critical review and future directions”. In: *Automation in Construction* 130, p. 103840. ISSN: 0926-5805. DOI: https://doi.org/10.1016/j.autcon.2021.103840.

Cherqui, F., F. Clemens-Meyer, F. Tscheikner-Gratl, and B. van Duin (June 2024). *Asset Management of Urban Drainage Systems: If anything exciting happens, we’ve done it wrong!* IWA Publishing. ISBN: 9781789063059. DOI: 10.2166/9781789063059.

Chiachío, J., M. L. Jalón, M. Chiachío, and A. Kolios (2020). “A Markov chains prognostics framework for complex degradation processes”. In: *Reliability Engineering & System Safety* 195, p. 106621. ISSN: 0951-8320. DOI: 10.1016/j.ress.2019.106621.

Chin, H. H., P. S. Varbanov, J. J. Klemeš, M. F. D. Benjamin, and R. R. Tan (2020). "Asset maintenance optimisation approaches in the chemical and process industries – A review". In: *Chemical Engineering Research and Design* 164, pp. 162–194. ISSN: 0263-8762. DOI: 10.1016/j.cherd.2020.09.034.

Cubillo, A., S. Perinpanayagam, and M. Esperon-Miguez (2016). "A review of physics-based models in prognostics: Application to gears and bearings of rotating machinery". In: *Advances in Mechanical Engineering* 8.8, p. 1687814016664660. DOI: 10.1177/1687814016664660.

de Jonge, B. and P. A. Scarf (2020). "A review on maintenance optimization". In: *European Journal of Operational Research* 285.3, pp. 805–824. ISSN: 0377-2217. DOI: 10.1016/j.ejor.2019.09.047.

Deb, K. (2005). "Multi-Objective Optimization". In: *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*. Ed. by E. K. Burke and G. Kendall. Boston, MA: Springer US, pp. 273–316. ISBN: 978-0-387-28356-2. DOI: 10.1007/0-387-28356-0_10.

– (2011). "Multi-objective optimisation using evolutionary algorithms: an introduction". In: *Multi-objective evolutionary optimisation for product design and manufacturing*. Springer, pp. 3–34. DOI: 10.1007/978-0-85729-652-8_1.

Dekker, R. (1996). "Applications of maintenance optimization models: a review and analysis". In: *Reliability Engineering & System Safety* 51.3. Maintenance and reliability, pp. 229–240. ISSN: 0951-8320. DOI: 10.1016/0951-8320(95)00076-3.

Deshmukh, S. G., A. Sharma, and G. Yadava (Mar. 2011). "143. Sharma , A., Yadava G.S. and Deshmukh S.G., 2011. A literature review and future perspectives on maintenance optimization. Journal of Quality in Maintenance Engineering, 17(1), 5-25." In: *Journal of Quality in Maintenance Engineering* 17, pp. 5–25. DOI: 10.1108/13552511111116222.

Ding, S.-H. and S. Kamaruddin (2015). "Maintenance policy optimization—literature review and directions". In: *The International Journal of Advanced Manufacturing Technology* 76, pp. 1263–1283. DOI: 10.1007/s00170-014-6341-2.

Dui, H., X. Wu, S. Wu, and M. Xie (2024). "Importance measure-based maintenance strategy optimization: Fundamentals, applications and future directions in AI and IoT". In: *Frontiers of Engineering Management*, pp. 1–26. DOI: 10.1007/s42524-024-4003-0.

El Morer, F., S. Wittek, and A. Rausch (2024). "Assessment of the suitability of degradation models for the planning of CCTV inspections of sewer pipes". In: *Urban Water Journal* 21.2, pp. 190–203. DOI: 10.1080/1573062X.2023.2282126.

Elattar, H. M., H. K. Elminir, and A. M. Riad (2016). "Prognostics: a literature review". In: *Complex & Intelligent Systems* 2.2, pp. 125–154. DOI: 10.1007/s40747-016-0019-3.

Elmasry, M., T. Zayed, and A. Hawari (2019). "Multi-Objective Optimization Model for Inspection Scheduling of Sewer Pipelines". In: *Journal of Construction Engineering and Management* 145.2, p. 04018129. DOI: 10.1061/(ASCE)CO.1943-7862.0001599.

EN 13508:1 -Investigation and assessment of drain and sewer systems outside buildings - Part 1: General Requirements (Oct. 2012). Standard. Avenue Marnix 17, B-1000 Brussels: European Committee for Standardization (CEN).

Esteban, A., A. Zafra, and S. Ventura (2022). "Data mining in predictive maintenance systems: A taxonomy and systematic review". In: *WIREs Data Mining and Knowledge Discovery* 12.5, e1471. DOI: https://doi.org/10.1002/widm.1471.

Fink, O., Q. Wang, M. Svensén, P. Dersin, W.-J. Lee, and M. Ducoffe (2020). "Potential, challenges and future directions for deep learning in prognostics and health management applications". In: *Engineering Applications of Artificial Intelligence* 92, p. 103678. ISSN: 0952-1976. DOI: 10.1016/j.engappai.2020.103678.

Frangopol, D. M. and P. Bocchini (2012). "Bridge network performance, maintenance and optimisation under uncertainty: accomplishments and challenges". In: *Structure and Infrastructure Engineering* 8.4, pp. 341–356. DOI: 10.1080/15732479.2011.563089.

Friedenthal, S., A. Moore, and R. Steiner (2015). *A practical guide to SysML: the systems modeling language*. Morgan Kaufmann. DOI: 10.1016/C2013-0-14457-1.

Goebel, K., J. Celaya, S. Sankararaman, I. Roychoudhury, M. Daigle, and A. Saxena (Apr. 2017). *Prognostics: The Science of Making Predictions*. ISBN: 978-1539074830.

Goyal, D., B. Pabla, S. Dhami, and K. Lachhwani (2017). "Optimization of condition-based maintenance using soft computing". In: *Neural Computing and Applications* 28, pp. 829–844. DOI: 10.1007/s00521-016-2377-6.

Guler, H. (Dec. 2016). "Optimisation of railway track maintenance and renewal works by genetic algorithms". In: *Journal of the Croatian Association of Civil Engineers* 68, pp. 979–993. DOI: 10.14256/JCE.1458.2015.

Guo, J., Z. Li, and M. Li (2020). "A Review on Prognostics Methods for Engineering Systems". In: *IEEE Transactions on Reliability* 69.3, pp. 1110–1129. DOI: 10.1109/TR.2019.2957965.

Hawari, A., F. Alkadour, M. Elmasry, and T. Zayed (2020). "A state of the art review on condition assessment models developed for sewer pipelines". In: *Engineering Applications of Artificial Intelligence* 93, p. 103721. DOI: 10.1016/j.engappai.2020.103721.

Heng, A., S. Zhang, A. C. Tan, and J. Mathew (2009). "Rotating machinery prognostics: State of the art, challenges and opportunities". In: *Mechanical Systems and Signal Processing* 23.3, pp. 724–739. ISSN: 0888-3270. DOI: 10.1016/j.ymssp.2008.06.009.

Hensel, C., S. Junges, J. Katoen, T. Quatmann, and M. Volk (2022). "The probabilistic model checker Storm". In: *International Journal on Software Tools for Technology Transfer* 24.4, pp. 589–610. DOI: 10.1007/s10009-021-00633-z.

Hu, Y., X. Miao, Y. Si, E. Pan, and E. Zio (2022). "Prognostics and health management: A review from the perspectives of design, development and decision". In: *Reliability Engineering & System Safety* 217, p. 108063. ISSN: 0951-8320. DOI: 10.1016/j.ress.2021.108063.

Hunt, A., B. E. Kelly, J. S. Mullhi, F. P. Lees, and A. G. Rushton (1993). "The propagation of faults in process plants: 6, Overview of, and modelling for, fault tree synthesis". In: *Reliability Engineering & System Safety* 39 (2), pp. 173–194. DOI: 10.1016/0951-8320(93)90041-V.

IEC 60050-192:2015 - International Electrotechnical Vocabulary - Part 192: Dependability, Definition 192-01-24 (2015). Available at https://www.electropedia.org/iev/iev.nsf/display?openform&ievref=192-01-24. Accessed: 2024-06-10. International Electrotechnical Commission.

IEC 60050-192:2015 - International Electrotechnical Vocabulary - Part 192: Dependability, Definition 192-03-01 (2015). Available at https://www.electropedia.org/iev/iev.nsf/display?openform&ievref=192-03-01. Accessed: 2024-06-10. International Electrotechnical Commission.

International Organization for Standardization (Sept. 2016). *Petroleum, petrochemical and natural gas industries — Collection and exchange of reliability and maintenance data for equipment*. ISO/TC 67. International Standard confirmed [90.93]. Corrected version published October 2016. International Organization for Standardization. URL: https://www.iso.org/standard/64076.html.

Javed, K., R. Gouriveau, and N. Zerhouni (2017). "State of the art and taxonomy of prognostics approaches, trends of prognostics applications and open issues towards maturity at different technology readiness levels". In: *Mechanical Systems and Signal Processing* 94, pp. 214–236. ISSN: 0888-3270. DOI: 10.1016/j.ymssp.2017.01.050.

Jimenez-Roa, L. A., T. Heskes, T. Tinga, and M. Stoelinga (2023). "Automatic Inference of Fault Tree Models Via Multi-Objective Evolutionary Algorithms". In: *IEEE Transactions on Dependable and Secure Computing* 20.4, pp. 3317–3327. DOI: 10.1109/TDSC.2022.3203805.

Kabir, S. (2017). “An overview of fault tree analysis and its application in model based dependability analysis”. In: *Expert Systems with Applications* 77, pp. 114–135. DOI: 10.1016/j.eswa.2017.01.058.

Kaelbling, L. P., M. L. Littman, and A. W. Moore (1996). “Reinforcement learning: A survey”. In: *Journal of artificial intelligence research* 4, pp. 237–285. DOI: 10.48550/arXiv.cs/9605103.

Kantidakis, G., H. Putter, S. Litière, and M. Fiocco (2023). “Statistical models versus machine learning for competing risks: development and validation of prognostic models”. In: *BMC Medical Research Methodology* 23.1, p. 51. DOI: 10.1186/s12874-023-01866-z.

Karris, S. (2006). *Introduction to Simulink with Engineering Applications*. Orchard Publications. ISBN: 9780974423975. URL: https://books.google.nl/books?id=L2JYZYI_l_wC.

Kerkkamp, D., Z. Bukhsh, Y. Zhang, and N. Jansen (2022). “Grouping of Maintenance Actions with Deep Reinforcement Learning and Graph Convolutional Networks”. English. In: *Proceeding of the 14th International Conference on Agents and Artificial Intelligence*. Vol. 2, pp. 574–585. DOI: 10.5220/0000155600003116.

Kim, S., J.-H. Choi, and N. H. Kim (2021). “Challenges and Opportunities of System-Level Prognostics”. In: *Sensors* 21.22. ISSN: 1424-8220. DOI: 10.3390/s21227655.

Lapa, C. M., C. M. Pereira, and A. C. de A. Mol (2000). “Maximization of a nuclear system availability through maintenance scheduling optimization using a genetic algorithm”. In: *Nuclear Engineering and Design* 196.2, pp. 219–231. ISSN: 0029-5493. DOI: 10.1016/S0029-5493(99)00295-2.

Lee, J., F. Wu, W. Zhao, M. Ghaffari, L. Liao, and D. Siegel (2014). “Prognostics and health management design for rotary machinery systems—Reviews, methodology and applications”. In: *Mechanical Systems and Signal Processing* 42.1, pp. 314–334. ISSN: 0888-3270. DOI: 10.1016/j.ymssp.2013.06.004.

Lee, J., C. Y. Park, S. Baek, S. H. Han, and S. Yun (2021). “Risk-Based Prioritization of Sewer Pipe Inspection from Infrastructure Asset Management Perspective”. In: *Sustainability* 13.13. ISSN: 2071-1050. DOI: 10.3390/su13137213.

M.A. Cardoso, M. A. and M. S. Silva (2016). “Sewer asset management planning – implementation of a structured approach in wastewater utilities”. In: *Urban Water Journal* 13.1, pp. 15–27. DOI: 10.1080/1573062X.2015.1076859.

Madden, M. G. and P. J. Nolan (1994). “Generation of fault trees from simulated incipient fault case data”. In: *WIT Transactions on Information and Communication Technologies* 6. DOI: 10.2495/AI940611.

Maddikunta, P. K. R., Q.-V. Pham, P. B, N. Deepa, K. Dev, T. R. Gadekallu, R. Ruby, and M. Liyanage (2022). “Industry 5.0: A survey on enabling technologies and potential applications”. In: *Journal of Industrial Information Integration* 26, p. 100257. DOI: https://doi.org/10.1016/j.jii.2021.100257.

Mahfoud, H. and A. Biyaali (Mar. 2018). “Dependability based maintenance optimization in healthcare domain”. In: *Journal of Quality in Maintenance Engineering* 24, pp. 00–00. DOI: 10.1108/JQME-07-2016-0029.

Marc Ribalta Ramon Bejar, C. M. and E. Rubión (2023). “Machine learning solutions in sewer systems: a bibliometric analysis”. In: *Urban Water Journal* 20.1, pp. 1–14. DOI: 10.1080/1573062X.2022.2138460.

Marugán, A. P. (2023). “Applications of Reinforcement Learning for maintenance of engineering systems: A review”. In: *Advances in Engineering Software* 183, p. 103487. ISSN: 0965-9978. DOI: 10.1016/j.advengsoft.2023.103487.

McPherson, J. W. (2019). *Reliability Physics and Engineering: Time-To-Failure Modeling*. 3rd ed. Springer Cham, pp. XVII, 463. ISBN: 978-3-319-93682-6. DOI: 10.1007/978-3-319-93683-3.

Modarres, M., M. P. Kaminskiy, and V. Krivtsov (2016). *Reliability Engineering and Risk Analysis: A Practical Guide*. 3rd ed. CRC Press. DOI: 10.1201/9781315382425.

Mrugalska, B. (2019). "Remaining Useful Life as Prognostic Approach: A Review". In: *Human Systems Engineering and Design*. Ed. by T. Ahram, W. Karwowski, and R. Taiar. Cham: Springer International Publishing, pp. 689–695. ISBN: 978-3-030-02053-8. DOI: 10.1007/978-3-030-02053-8_105.

Nelson, W. B. (2005). *Applied Life Data Analysis*. John Wiley & Sons. ISBN: 978-0-471-72522-0.

NENEN13306: Maintenance - Maintenance terminology (2017). https://standards.globalspec.com/std/10272557/en-13306. Accessed: 2023-06-30.

O'Connor, P. and A. Kleyner (Jan. 2012). *Practical Reliability Engineering, Fifth Edition*. ISBN: 978-0-470-97981-5. DOI: 10.1002/9781119961260.

Obradović, D., M. Šperac, and S. Marenjak (2019). "Possibilities of using expert methods for sewer system maintenance optimisation". In: *Građevinar* 71.9, pp. 769–779. DOI: 10.14256/JCE.2589.2018.

Ogunfowora, O. and H. Najjaran (2023). "Reinforcement and deep reinforcement learning-based solutions for machine maintenance planning, scheduling policies, and optimization". In: *Journal of Manufacturing Systems* 70, pp. 244–263. ISSN: 0278-6125. DOI: 10.1016/j.jmsy.2023.07.014.

Ojha, M., K. P. Singh, P. Chakraborty, and S. Verma (2019). "A review of multi-objective optimisation and decision making using evolutionary algorithms". In: *International Journal of Bio-Inspired Computation* 14.2, pp. 69–84. DOI: 10.1504/ijbic.2019.101640.

Oztemel, E. and S. Gursev (2020). "Literature review of Industry 4.0 and related technologies". In: *Journal of intelligent manufacturing* 31.1, pp. 127–182. DOI: 10.1007/s10845-018-1433-8.

Peng, J., G. Xia, Y. Li, Y. Song, and M. Hao (2022). "Knowledge-based prognostics and health management of a pumping system under the linguistic decision-making context". In: *Expert Systems with Applications* 209, p. 118379. ISSN: 0957-4174. DOI: 10.1016/j.eswa.2022.118379.

Pourgholamali, M., S. Labi, and K. C. Sinha (2023). "Multi-objective optimization in highway pavement maintenance and rehabilitation project selection and scheduling: A state-of-the-art review". In: *Journal of Road Engineering* 3.3, pp. 239–251. ISSN: 2097-0498. DOI: 10.1016/j.jreng.2023.05.003.

Prakash, G., X.-X. Yuan, B. Hazra, and D. Mizutani (May 2021). "Toward a Big Data-Based Approach: A Review on Degradation Models for Prognosis of Critical Infrastructure". In: *Journal of Nondestructive Evaluation, Diagnostics and Prognostics of Engineering Systems* 4. DOI: 10.1115/1.4048787.

Puterman, M. (2014). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley Series in Probability and Statistics. John Wiley & Sons. ISBN: 9781118625873. URL: https://books.google.nl/books?id=VvBjBAAAQBAJ.

Qasem, A. and R. Jamil (2021). "GIS-Based Financial Analysis Model for Integrated Maintenance and Rehabilitation of Underground Pipe Networks". In: *Journal of Performance of Constructed Facilities* 35.5, p. 04021046. DOI: 10.1061/(ASCE)CF.1943-5509.0001623.

Ranjith, S., S. Setunge, R. Gravina, and S. Venkatesan (2013). "Deterioration Prediction of Timber Bridge Elements Using the Markov Chain". In: *Journal of Performance of Constructed Facilities* 27.3, pp. 319–325. DOI: 10.1061/(ASCE)CF.1943-5509.0000311.

Rezamand, M., M. Kordestani, R. Carriveau, D. S.-K. Ting, M. E. Orchard, and M. Saif (2020). "Critical Wind Turbine Components Prognostics: A Comprehensive Review". In: *IEEE Transactions on Instrumentation and Measurement* 69.12, pp. 9306–9328. DOI: 10.1109/TIM.2020.3030165.

Ruijters, E. and M. Stoelinga (2015). “Fault tree analysis: A survey of the state-of-the-art in modeling, analysis and tools”. In: *Computer Science Review* 15-16, pp. 29–62. ISSN: 1574-0137. DOI: https://doi.org/10.1016/j.cosrev.2015.03.001.

Saddiqi, M. M., W. Zhao, S. Cotterill, and R. K. Dereli (2023). “Smart management of combined sewer overflows: From an ancient technology to artificial intelligence”. In: *Wiley Interdisciplinary Reviews: Water* 10.3, e1635. DOI: 10.1002/wat2.1635.

Salem, S. L., G. E. Apostolakis, and D. Okrent (Nov. 1976). “Computer-oriented approach to fault-tree construction”. In: DOI: 10.2172/7132148.

Shafiee, M. and J. D. Sørensen (2019). “Maintenance optimization and inspection planning of wind energy assets: Models, methods and strategies”. In: *Reliability Engineering & System Safety* 192. Complex Systems RAMS Optimization: Methods and Applications, p. 105993. ISSN: 0951-8320. DOI: 10.1016/j.ress.2017.10.025.

Sheppard, J. W., M. A. Kaufman, and T. J. Wilmering (2008). “IEEE standards for prognostics and health management”. In: *2008 IEEE AUTOTESTCON*, pp. 97–103. DOI: 10.1109/AUTEST.2008.4662592.

Signoret, J.-P. and A. Leroy (Feb. 2021). “Reliability Block Diagrams (RBDs)”. In: *Reliability Assessment of Safety and Production Systems*. Springer Series in Reliability Engineering. Springer. Chap. 0, pp. 195–208. DOI: 10.1007/978-3-030-64708-7.

Sikorska, J., M. Hodkiewicz, and L. Ma (2011). “Prognostic modelling options for remaining useful life estimation by industry”. In: *Mechanical Systems and Signal Processing* 25.5, pp. 1803–1836. ISSN: 0888-3270. DOI: 10.1016/j.ymssp.2010.11.018.

Sinha, Y. and J. Steel (2015). “A progressive study into offshore wind farm maintenance optimisation using risk based failure analysis”. In: *Renewable and Sustainable Energy Reviews* 42, pp. 735–742. ISSN: 1364-0321. DOI: https://doi.org/10.1016/j.rser.2014.10.087.

Siraskar, R., S. Kumar, S. Patil, A. Bongale, and K. Kotecha (2023). “Reinforcement learning for predictive maintenance: A systematic technical review”. In: *Artificial Intelligence Review* 56.11, pp. 12885–12947. DOI: 10.1007/s10462-023-10468-6.

Stamatis, D. (2003). *Failure Mode and Effect Analysis: FMEA from Theory to Execution*. ASQ Quality Press. ISBN: 9780873895989. URL: https://books.google.nl/books?id=T9TxNHWJZmIC.

Stewart, W. J. (2009). *Probability, Markov chains, queues, and simulation: the mathematical basis of performance modeling*. Princeton University Press. Chap. 9. Discrete- and Continuous-Time Markov chains, pp. 193–284. ISBN: 9780691140629. DOI: 10.2307/j.ctvcm4gtc.12.

Sutton, R. S. and A. G. Barto (2018). *Reinforcement learning: An introduction*. MIT press, p. 552. ISBN: 978-0262039246.

Syan, C. S. and G. Ramsoobag (2019). “Maintenance applications of multi-criteria optimization: A review”. In: *Reliability Engineering & System Safety* 190, p. 106520. ISSN: 0951-8320. DOI: https://doi.org/10.1016/j.ress.2019.106520.

Tahan, M., E. Tsoutsanis, M. Muhammad, and Z. Abdul Karim (2017). “Performance-based health monitoring, diagnostics and prognostics for condition-based maintenance of gas turbines: A review”. In: *Applied Energy* 198, pp. 122–144. ISSN: 0306-2619. DOI: 10.1016/j.apenergy.2017.04.048.

Tanwar, M., H. Park, and N. Raghavan (2021). “Multistate Diagnosis and Prognosis of Lubricating Oil Degradation Using Sticky Hierarchical Dirichlet Process–Hidden Markov Model Framework”. In: *Applied Sciences* 11.14. ISSN: 2076-3417. DOI: 10.3390/app11146603.

Tinga, T. (Jan. 2013). *Principles of Loads and Failure Mechanisms. Applications in Maintenance, Reliability and Design*. Springer. Chap. 5, pp. 161–186. ISBN: 978-1-4471-4917-0. DOI: 10.1007/978-1-4471-4917-0.

Tobon-Mejia, D. A., K. Medjaher, N. Zerhouni, and G. Tripot (2012). “A Data-Driven Failure Prognostics Method Based on Mixture of Gaussians Hidden Markov Models”. In: *IEEE Transactions on Reliability* 61.2, pp. 491–503. DOI: 10.1109/TR.2012.2194177.

Tscheikner-Gratl, F., N. Caradot, F. Cherqui, J. P. Leitão, M. Ahmadi, J. G. Langeveld, Y. Le Gat, L. Scholten, B. Roghani, J. P. Rodríguez, et al. (2019). “Sewer asset management–state of the art and research needs”. In: *Urban Water Journal* 16.9, pp. 662–675. DOI: 10.1080/1573062X.2020.1713382.

Tsui, K. L., N. Chen, Q. Zhou, Y. Hai, and W. Wang (2015). “Prognostics and Health Management: A Review on Data Driven Approaches”. In: *Mathematical Problems in Engineering* 2015.1, p. 793161. DOI: 10.1155/2015/793161.

View of Breda (Aug. 2024). Google Earth. Available from: https://earth.google.com/web/search/breda/@51.55998129,4.76433911,6.42890411a,46248.16933708d,35y,0h,0t,0r, Accessed on 2024-08-08.

Wang, H., M. Liserre, F. Blaabjerg, P. de Place Rimmen, J. B. Jacobsen, T. Kvisgaard, and J. Landkildehus (2014). “Transitioning to Physics-of-Failure as a Reliability Driver in Power Electronics”. In: *IEEE Journal of Emerging and Selected Topics in Power Electronics* 2.1, pp. 97–114. DOI: 10.1109/JESTPE.2013.2290282.

Wang, X., S. Wang, X. Liang, D. Zhao, J. Huang, X. Xu, B. Dai, and Q. Miao (2024). “Deep Reinforcement Learning: A Survey”. In: *IEEE Transactions on Neural Networks and Learning Systems* 35.4, pp. 5064–5078. DOI: 10.1109/TNNLS.2022.3207346.

Wirahadikusumah, R. and D. M. Abraham (2003). “Application of dynamic programming and simulation for sewer management”. In: *Engineering, Construction and Architectural Management* 10.3, pp. 193–208. DOI: 10.1108/09699980310478449.

Xia, J. and G. Zou (2023). “Operation and maintenance optimization of offshore wind farms based on digital twin: A review”. In: *Ocean Engineering* 268, p. 113322. ISSN: 0029-8018. DOI: 10.1016/j.oceaneng.2022.113322.

Xia, T., Y. Dong, L. Xiao, S. Du, E. Pan, and L. Xi (2018). “Recent advances in prognostics and health management for advanced manufacturing paradigms”. In: *Reliability Engineering & System Safety* 178, pp. 255–268. ISSN: 0951-8320. DOI: https://doi.org/10.1016/j.ress.2018.06.021.

Xue, K., J. Yang, M. Yang, and D. Wang (2023). “An Improved Generic Hybrid Prognostic Method for RUL Prediction Based on PF-LSTM Learning”. In: *IEEE Transactions on Instrumentation and Measurement* 72, pp. 1–21. DOI: 10.1109/TIM.2023.3251391.

Yin, Z., A. S. Leon, A. Sharifi, and M. H. Amini (n.d.). “Optimal Control of Combined Sewer Systems to Minimize Sewer Overflows by Using Reinforcement Learning”. In: *World Environmental and Water Resources Congress 2023*, pp. 711–722. DOI: 10.1061/9780784484852.067.

Zeng, X., Z. Wang, H. Wang, S. Zhu, and S. Chen (2023). “Progress in Drainage Pipeline Condition Assessment and Deterioration Prediction Models”. In: *Sustainability* 15.4, p. 3849. DOI: 10.3390/su15043849.

Zhang, J. and J. Lee (2011). “A review on prognostics and health monitoring of Li-ion battery”. In: *Journal of Power Sources* 196.15, pp. 6007–6014. ISSN: 0378-7753. DOI: 10.1016/j.jpowsour.2011.03.101.

Zhang, L., J. Lin, B. Liu, Z. Zhang, X. Yan, and M. Wei (2019). “A Review on Deep Learning Applications in Prognostics and Health Management”. In: *IEEE Access* 7, pp. 162415–162438. DOI: 10.1109/ACCESS.2019.2950985.

Zhang, W., D. Yang, and H. Wang (2019). “Data-Driven Methods for Predictive Maintenance of Industrial Equipment: A Survey”. In: *IEEE Systems Journal* 13.3, pp. 2213–2227. DOI: 10.1109/JSYST.2019.2905565.

Zhang, X., R. Xu, C. Kwan, S. Liang, Q. Xie, and L. Haynes (2005). “An integrated approach to bearing fault diagnostics and prognostics”. In: *Proceedings of the 2005,* *American Control Conference, 2005.* 2750–2755 vol. 4. DOI: 10.1109/ACC.2005.1470385.

Zhou, D., Z. Yu, H. Zhang, and S. Weng (2016). "A novel grey prognostic model based on Markov process and grey incidence analysis for energy conversion equipment degradation". In: *Energy* 109, pp. 420–429. ISSN: 0360-5442. DOI: 10.1016/j.energy.2016.05.008.

Zhu, S.-P., H.-Z. Huang, W. Peng, H.-K. Wang, and S. Mahadevan (2016). "Probabilistic Physics of Failure-based framework for fatigue life prediction of aircraft gas turbine discs under uncertainty". In: *Reliability Engineering & System Safety* 146, pp. 1–12. ISSN: 0951-8320. DOI: 10.1016/j.ress.2015.10.002.

Zio, E. (2022). "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, p. 108119. ISSN: 0951-8320. DOI: 10.1016/j.ress.2021.108119.

Part I

Data-driven Inference of Fault Tree models

I.1 Introduction

Part I focuses on the data-driven inference of Fault Tree models. The general research question we address here is *how to obtain efficient and compact Fault Tree models from failure datasets in a robust and scalable manner?* This part is structured as follows: Section I.2 summarises the nomenclature used in Part I. Section I.3 reviews the related work common to all chapters. Section I.4 presents the formal definitions used in Part I. The chapters contained here are:

Chapter 2. *Automatic Inference of Fault Tree Models via Multi-Objective Evolutionary Algorithms* 41

Chapter 3. *Data-Driven Inference of Fault Tree Models Exploiting Symmetry and Modularisation* 61

Chapter 4. *Fault Tree inference using Multi-Objective Evolutionary Algorithms and Confusion Matrix-based metrics* 75

I.2 Nomenclature

Fault Trees:

$\mathcal{F}$	A Fault Tree model
BE	Basic Event (BE)
AND	Logic gate AND
OR	Logic gate OR
V	Set of nodes in a Fault Tree
G	Set of logic gates in a Fault Tree
TE	Top Event
$\boldsymbol{b}$	Status vector
$\mathbb{C}$	Minimal Cut Set (MCS)

Inference of Fault Trees:

$\mathbb{D}$	Failure dataset
$\mathcal{F}_{\mathbb{D}}$	Inferred FT from $\mathbb{D}$
$\mathsf{M}_{\mathbb{D}}$	MCS matrix computed from $\mathbb{D}$
$\mathsf{M}_{\mathcal{F}}$	MCS matrix computed from a given $\mathcal{F}$
ε_s	Size of the Fault Tree model
ε_d	Error computed from $\mathbb{D}$
ε_c	Error computed from $\mathsf{M}_{\mathcal{F}}$ and $\mathsf{M}_{\mathbb{D}}$
ρ	Number of superfluous BEs

Multi-Objective Evolutionary Algorithms:

ps	Population size
uc	Max. generations with unchanged best
ng	Max. number of generations

I.3 Related work

Different methods for inferring Fault Trees (FTs) have been discussed in the literature. We categorise these methods into three main groups: *knowledge-based*, *model-based*, and *data-driven*. *Knowledge-based* approaches primarily utilise various heuristics for knowledge representation and domain expertise (Latif-Shabgahi, 2002); *model-based* approaches translate existing system models and/or graphs into FTs; and *data-driven* approaches use structured databases as the main information source, aiming to *identify causal relationships in a failure dataset with minimal domain expertise and human intervention.*

Carpignano and Poucet, 1994 provides a comprehensive review of knowledge-based approaches. An example of a model-based approach is found in Mhenni, Nguyen, and Choley, 2014, where the authors employ *SysML* System Models to derive FT models. However, a significant limitation of model-based approaches is the need for a pre-existing model (Dickerson, Roslan, and Ji, 2018). Our focus is on *data-driven* approaches, which encompass the applications of machine learning techniques and data analytics. Appendix B.1, Table B.1 (divided into two parts), summarises and compares relevant literature on data-driven methods for the automatic inference of FT models.

Early techniques for *data-driven* FT inference include the `IFT` algorithm (Madden and Nolan, 1994), which employs Quinlan's ID3 algorithm to generate Decision Trees, and the approach in Mukherjee and Chakraborty, 2007, which uses *text mining* techniques with maintenance records as input data. Mukherjee and Chakraborty, 2007 addresses the challenge of inferring FTs through *linguistic analysis* and domain knowledge to extract failure characteristics from brief descriptions of equipment faults. Roth, Wolf, and Lindemann, 2015 propose a method using the *Structural Complexity Management* methodology to deduce dependencies, which are later used to infer the Boolean logic operators of FT models. Inspired by *Causal Decision Trees*, the `LIFT` algorithm (Nauta, Bucur, and Stoelinga, 2018) utilises the *Mantel-Haenszel* test to identify dependencies between events, requiring both basic event data and intermediate event failure information.

The `ILTA` (Waghen and M.-S. Ouali, 2019) and `MILTA` (Waghen and M.-S. Ouali, 2021) algorithms combine *Knowledge Discovery in Datasets* (KDD), *Interpretable Logic Tree Analysis*, and *Bayesian probability rules*. Another method, described in Linard, Bueno, Bucur, et al., 2020, constructs a *Bayesian Network* before converting it into an FT model, employing *blacklists* and *whitelists* to identify the presence or absence of arcs. The `DDFTA` algorithm (Lazarova-Molnar, Niloofar, and Barta, 2020) derives FTs from failure data time series through binarisation and Boolean equation simplification. The `DDFTAe` algorithm (Niloofar and Lazarova-Molnar, 2021), an extension of `DDFTA`, addresses missing information in time series fault occurrence data, while the `DDFTAnb` algorithm (Niloofar and Lazarova-Molnar, 2023b) extends `DDFTA` by focusing on FT models using *naïve Bayes classification* and time series data.

Evolutionary algorithm-based methods include `FT-EA` (Linard, Bucur, and Stoelinga, 2019), `FT-MOEA` (Jimenez-Roa, Heskes, Tinga, et al., 2023), and `FT-MOEA-CM` (Jimenez-Roa, Rusnac, Volk, et al., 2024), with `FT-MOEA` demonstrating improvements through a multi-objective cost function over the single-dimensional cost function used in `FT-EA`, and `FT-MOEA-CM` proved larger scalability by considering metrics computed from the Confusion Matrix. The `SymLearn` tool chain (Jimenez-Roa, Volk, and Stoelinga, 2022) enhances scalability by exploiting symmetries and modules within failure datasets. Additionally, Dorfhuber, Eisentraut, and Křetínský, 2023 employs genetic algorithms to learn attack trees from sets of traces.

The method in Verkuil, Budde, and Bucur, 2022 generates FT models from sensor time series data, exemplified by a domestic heater case study, which aims to identify thresholds that differentiate between *normal* and *error* conditions. The `ITCA` methodology (Waghen and M. Ouali, 2022) focuses on fault hierarchy causality through KDD and FTs, employing causality analysis and NASA's turbofan dataset as an example. Similarly, the methodology in Nadim, Ragab, and M. Ouali, 2023 uses interpretable machine learning and causal analysis to derive *Petri Nets* from event logs, with potential applications to FTs.

Further advancements in the field include the method proposed in Niloofar and Lazarova-Molnar, 2023c for learning *Dynamic* FTs from temporal *Truth Tables* and time series data, and the study in Niloofar and Lazarova-Molnar, 2023a that integrates *collaborative data analytics* into the derivation of FTs models, showing enhanced accuracy in data-driven FT inference.

I.4 Preliminaries

I.4.1 Fault Trees

A Fault Tree (FT) models how failures occur and propagate in a system, and lead to a system failure (NASA, 2002; Ruijters and Stoelinga, 2015). Formally, a FT is a directed acyclic graph where the leaves, called *basic events (BE)*, correspond to (atomic) system components. The intermediate nodes are equipped with logical *gates* modelling the failure propagation.

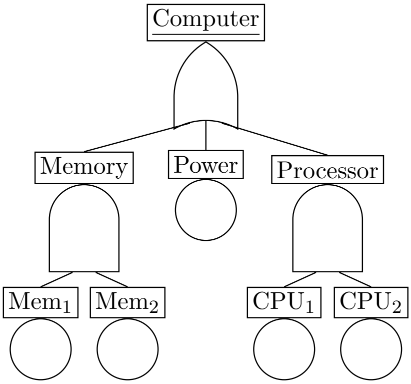


Figure I.1: Example FT.

A logical AND-gate fails, if all successor nodes fail, an OR-gate fails if at least one successor fails. An FT $\mathcal{F}$ fails if the dedicated root node fails. Figure I.1 depicts an example FT modelling a computer. The root *Computer* is an OR-gate, *Memory* and *Processor* are AND-gates, circles indicate BE.

Definition 1 (Fault tree). *A Fault Tree (FT) is a rooted directed acyclic graph* (V, E) *assigning the type to nodes via function* $Tp : V \to \{\mathit{BE}, \mathit{AND}, \mathit{OR}\}$ *s.t.* $Tp(v) = \mathit{BE}$ *iff* v *is a leaf. The* inputs *of a node* v*, denoted* $I(v)$*, are the successors of* v*. All nodes* $v \in V$ *must be reachable from the root* Top*.*

All nodes of type BE are denoted by $\mathsf{BEs} := \{v \in V \mid Tp(v) = \mathsf{BE}\}$.

The *semantics* of an FT $\mathcal{F}$ are given by its *structure function* $f^{\mathcal{F}}$. Let $\boldsymbol{b} = \langle \boldsymbol{b}_1, \ldots, \boldsymbol{b}_{|\mathsf{BEs}|} \rangle \in \{0,1\}^{|\mathsf{BEs}|}$ be a *status vector* where $\boldsymbol{b}_i = 1$ indicates that the i-th BE has failed, and $\boldsymbol{b}_i = 0$ that it functions properly, respectively.

Definition 2 (Semantics of FT). *Let* $\boldsymbol{b}$ *be a status vector and* $\mathcal{F}$ *an FT. The structure function* $f^{\mathcal{F}} : \{0,1\}^{|\mathit{BEs}|} \times V \to \{0,1\}$ *returns the status of node* v *and is defined as:*

$$f^{\mathcal{F}}(\boldsymbol{b}, v) := \begin{cases} \boldsymbol{b}_i & \textit{if } Tp(v) = \mathsf{BE} \textit{ and } v \textit{ is the } i\textit{-th } \mathsf{BE}, \\ \bigwedge_{v' \in I(v)} f(\boldsymbol{b}, v') & \textit{if } Tp(v) = \mathsf{AND}, \\ \bigvee_{v' \in I(v)} f(\boldsymbol{b}, v') & \textit{if } Tp(v) = \mathsf{OR}. \end{cases}$$

We say FT $\mathcal{F}$ fails for $\boldsymbol{b}$ if $f^{\mathcal{F}}(\boldsymbol{b}, \mathsf{Top}) = 1$. A status vector $\boldsymbol{b}$ can also be given as the set $\mathbb{C} = \{\boldsymbol{b}_i \in \boldsymbol{b} \mid \boldsymbol{b}_i = 1\}$ of failed BE, and we write $f^{\mathcal{F}}(\mathbb{C})$ instead of $f^{\mathcal{F}}(\boldsymbol{b})$. $\mathbb{C}$ is as *minimal cut set (MCS)* if $f^{\mathcal{F}}(\mathbb{C}) = 1$ and $\forall \mathbb{C}' \subset \mathbb{C} : f^{\mathcal{F}}(\mathbb{C}') = 0$.

For the FT in Figure I.1, the set $\mathbb{C} = \{\mathrm{Mem}_1, \mathrm{Mem}_2, \mathrm{CPU}_1\}$ of failed BE leads to a failure of the overall FT: $f^{F}(\mathbb{C}) = 1$. The FT has three MCS: $\{\mathrm{Mem}_1, \mathrm{Mem}_2\}$, $\{\mathrm{Power}\}$ and $\{\mathrm{CPU}_1, \mathrm{CPU}_2\}$.

I.4.2 Failure Dataset

We assume the failure data is given in a format such that a series of data points (vector) represents the possible state of each component as well as the overall system state.

Table I.1 gives an example dataset corresponding to the FT in Figure I.1. A row in the *failure dataset* $\mathbb{D}$ corresponds to a status vector $\boldsymbol{b}_k$—giving the status of each (atomic) component—together with the overall system status $f^{\mathbb{D}}(\boldsymbol{b}_k)$. For instance, the first row (status vector $\boldsymbol{b}_0$) represents that only component CPU_2 has failed, and the system is still operational. The second row (status vector $\boldsymbol{b}_1$) represents the failure of components CPU_1 and CPU_2, leading to a system failure.

Table I.1: Example failure dataset $\mathbb{D}$.

	Mem_1	Mem_2	Power	CPU_1	CPU_1	Sys.
$\boldsymbol{b}_0$	0	0	0	0	1	0
$\boldsymbol{b}_1$	0	0	0	1	1	1
$\boldsymbol{b}_2$	0	0	1	0	0	1
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

We assume the dataset is *coherent*, i.e., a failed system stays failed for further component failures, and *noise-free*, i.e., the same status of components always yields the same system state.

Failure Dataset. The failure dataset $\mathbb{D}$ is given as a labelled binary dataset indicating the failure status of each component, together the corresponding status of the overall system. Table I.1 gives an example corresponding to the FT in Figure I.1 where M_1 corresponds to Mem_1, etc.

We assume the data is *coherent*, i.e., once the system fails, it cannot become operational again through further component failures, and it is *noise-free*, i.e., observations with unchanged component states always yield the same system state.

We can also identify MCSs in the failure data $\mathbb{D}$. A (minimal) cut set $\mathbb{C}$ of $\mathbb{D}$ is a (minimal) set of BEs s.t. the corresponding status vector $\boldsymbol{b}$ yields a system failure in $\mathbb{D}$. The set of all MCSs in $\mathbb{D}$ is denoted by $\mathsf{M}_{\mathbb{D}}$.

I.4.3 Inference of Fault Tree models

We define the inference of FT models as the process of finding a compact FT $\mathcal{F}_{\mathbb{D}}$ that matches a given failure dataset $\mathbb{D}$. In a perfectly accurate FT, assigning the BEs in $\mathcal{F}_{\mathbb{D}}$ identical values to $\boldsymbol{b}_k$ in the dataset $\mathbb{D}$ results in the same overall system status, $f^{\mathcal{F}_{\mathbb{D}}}(\boldsymbol{b}_k) = f^{\mathbb{D}}(\boldsymbol{b}_k)$.

Problem statement. Given a failure dataset $\mathbb{D} = \big(\boldsymbol{b}_k \quad f(\boldsymbol{b}_k)\big)$, create a FT $\mathcal{F}_{\mathbb{D}}$ that is both

1. *small*, i.e., the number of nodes ε_s is minimal, and
2. *accurate*, i.e., the structure function $f^{\mathcal{F}_{\mathbb{D}}}$ of the FT coincides with the given failure dataset $f^{\mathbb{D}}(\boldsymbol{b}_k)$.

I.5 References

Carpignano, A. and A. Poucet (1994). “Computer assisted fault tree construction: a review of methods and concerns”. In: *Reliability Engineering & System Safety* 44.3, pp. 265–278. DOI: 10.1016/0951-8320(94)90018-3.

Dickerson, C. E., R. Roslan, and S. Ji (2018). “A formal transformation method for automated fault tree generation from a UML activity model”. In: *IEEE Transactions on Reliability* 67 (3), pp. 1219–1236. DOI: 10.1109/TR.2018.2849013.

Dorfhuber, F., J. Eisentraut, and J. Křetínský (2023). “Learning Attack Trees by Genetic Algorithms”. In: *Theoretical Aspects of Computing – ICTAC 2023*. Ed. by E. Ábrahám, C. Dubslaff, and S. L. T. Tarifa. Cham: Springer Nature Switzerland, pp. 55–73. ISBN: 978-3-031-47963-2. DOI: 10.1007/978-3-031-47963-2_5.

Jimenez-Roa, L. A., N. Rusnac, M. Volk, and M. Stoelinga (2024). “Fault Tree Inference Using Multi-objective Evolutionary Algorithms and Confusion Matrix-Based Metrics”. In: *Formal Methods for Industrial Critical Systems*. Ed. by A. E. Haxthausen and W. Serwe. Cham: Springer Nature Switzerland, pp. 80–96. ISBN: 978-3-031-68150-9. DOI: 10.1007/978-3-031-68150-9_5.

Jimenez-Roa, L. A., T. Heskes, T. Tinga, and M. Stoelinga (2023). “Automatic Inference of Fault Tree Models Via Multi-Objective Evolutionary Algorithms”. In: *IEEE Transactions on Dependable and Secure Computing* 20.4, pp. 3317–3327. DOI: 10.1109/TDSC.2022.3203805.

Jimenez-Roa, L. A., M. Volk, and M. Stoelinga (2022). “Data-Driven Inference of Fault Tree Models Exploiting Symmetry and Modularization”. In: *Computer Safety, Reliability, and Security*. Ed. by M. Trapp, F. Saglietti, M. Spisländer, and F. Bitsch. Cham: Springer International Publishing, pp. 46–61. DOI: 10.1007/978-3-031-14835-4_4.

Latif-Shabgahi, G. (2002). “Comparing selected knowledge-based fault tree construction tools”. In: *Proceedings of the IASTED International Conference on Intelligent Systems and Control*.

Lazarova-Molnar, S., P. Niloofar, and G. K. Barta (2020). “Data-Driven Fault Tree Modeling for Reliability Assessment of Cyber-Physical Systems”. In: *2020 Winter Simulation Conference (WSC)*, pp. 2719–2730. DOI: 10.1109/WSC48552.2020.9383882.

Linard, A., M. Bueno, D. Bucur, and M. Stoelinga (2020). “Induction of fault trees through Bayesian networks”. In: *Proceedings of the 29th European Safety and Reliability Conference, ESREL 2019*, pp. 910–917. DOI: 10.3850/978-981-11-2724-3_0596-cd.

Linard, A., D. Bucur, and M. Stoelinga (2019). "Fault Trees from Data: Efficient Learning with an Evolutionary Algorithm". In: *International Symposium on Dependable Software Engineering: Theories, Tools, and Applications*. Vol. 11951 LNCS, pp. 19–37. DOI: 10.1007/978-3-030-35540-1_2.

Madden, M. G. and P. J. Nolan (1994). "Generation of fault trees from simulated incipient fault case data". In: *WIT Transactions on Information and Communication Technologies* 6. DOI: 10.2495/AI940611.

Mhenni, F., N. Nguyen, and J.-Y. Choley (2014). "Automatic fault tree generation from SysML system models". In: *2014 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*. IEEE, pp. 715–720. DOI: 10.1109/AIM.2014.6878163.

Mukherjee, S. and A. Chakraborty (2007). "Automated fault tree generation: bridging reliability with text mining". In: *2007 Annual Reliability and Maintainability Symposium*. IEEE, pp. 83–88. DOI: 10.1109/RAMS.2007.328096.

Nadim, K., A. Ragab, and M. Ouali (2023). "Data-driven dynamic causality analysis of industrial systems using interpretable machine learning and process mining". In: *Journal of Intelligent Manufacturing* 34.1, pp. 57–83. DOI: 10.1007/s10845-021-01903-y.

NASA (2002). *Fault Tree Handbook with Aerospace Applications*. Handbook. U.S. National Aeronautics and Space Administration.

Nauta, M., D. Bucur, and M. Stoelinga (2018). "LIFT: Learning fault trees from observational data". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 11024 LNCS, pp. 306–322. DOI: 10.1007/978-3-319-99154-2_19.

Niloofar, P. and S. Lazarova-Molnar (2021). "Data-Driven Modelling Of Repairable Fault Trees From Time Series Data With Missing Information". In: *2021 Winter Simulation Conference (WSC)*, pp. 1–12. DOI: 10.1109/WSC52266.2021.9715375.

– (2023a). "Collaborative data-driven reliability analysis of multi-state fault trees". In: *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability* 237.5, pp. 886–896. DOI: 10.1177/1748006X221076290.

– (2023b). "Data-driven extraction and analysis of repairable fault trees from time series data". In: *Expert Systems with Applications* 215, p. 119345. ISSN: 0957-4174. DOI: 10.1016/j.eswa.2022.119345.

– (2023c). "Learning Temporal Truth Tables of Dynamic Fault Trees from Time Series Data on Faults". In: *2023 7th International Conference on System Reliability and Safety (ICSRS)*. IEEE, pp. 449–453. DOI: 10.1109/icsrs59833.2023.10381460.

Roth, M., M. Wolf, and U. Lindemann (2015). "Integrated Matrix-based Fault Tree Generation and Evaluation". In: *Procedia Computer Science* 44. 2015 Conference on Systems Engineering Research, pp. 599–608. ISSN: 1877-0509. DOI: 10.1016/j.procs.2015.03.027.

Ruijters, E. and M. Stoelinga (2015). "Fault tree analysis: A survey of the state-of-the-art in modeling, analysis and tools". In: *Computer Science Review* 15-16, pp. 29–62. ISSN: 1574-0137. DOI: https://doi.org/10.1016/j.cosrev.2015.03.001.

Verkuil, B., C. E. Budde, and D. Bucur (2022). "Automated fault tree learning from continuous-valued sensor data: a case study on domestic heaters". In: *International Journal of Prognostics and Health Management* 13.2. DOI: 10.36001/ijphm.2022.v13i2.3160.

Waghen, K. and M.-S. Ouali (2019). "Interpretable logic tree analysis: A data-driven fault tree methodology for causality analysis". In: *Expert Systems with Applications* 136, pp. 376–391. DOI: 10.1016/j.eswa.2019.06.042.

Waghen, K. and M.-S. Ouali (2021). "Multi-level interpretable logic tree analysis: A data-driven approach for hierarchical causality analysis". In: *Expert Systems with Applications* 178, p. 115035. ISSN: 0957-4174. DOI: 10.1016/j.eswa.2021.115035.

Waghen, K. and M. Ouali (2022). “A Data-Driven Fault Tree for a Time Causality Analysis in an Aging System”. In: *Algorithms* 15.6, p. 178. DOI: 10.3390/a15060178.

Chapter 2

Automatic Inference of Fault Tree Models via Multi-Objective Evolutionary Algorithms

Paper published at **L. A. Jimenez-Roa**, T. Heskes, T. Tinga and M. Stoelinga, *"Automatic Inference of Fault Tree Models via Multi-Objective Evolutionary Algorithms",* in IEEE Transactions on Dependable and Secure Computing, vol. 20, no. 4, pp. 3317-3327, 1 July-Aug. 2023, doi: 10.1109/TDSC.2022.3203805.

Abstract

Fault Tree Analysis is a well-known technique in reliability engineering and risk assessment, which supports decision-making processes and the management of complex systems. Traditionally, Fault Tree (FT) models are built manually together with domain experts, considered a time-consuming process prone to human errors. With Industry 4.0, there is an increasing availability of inspection and monitoring data, making techniques that enable knowledge extraction from large datasets relevant. Thus, our goal with this work is to propose a data-driven approach to infer efficient FT structures that achieve a complete representation of the failure mechanisms contained in the failure dataset without human intervention. Our algorithm, the `FT-MOEA`, based on multi-objective evolutionary algorithms, enables the simultaneous optimisation of different relevant metrics such as the FT size, the error computed based on the failure dataset and the Minimal Cut Sets. Our results show that, for six case studies from the literature, our approach successfully achieved automatic, efficient, and consistent inference of the associated FT models. We also present the results of a parametric analysis that tests our algorithm for different relevant conditions that influence its performance, as well as an overview of the data-driven methods used to automatically infer FT models.

2.1 Introduction

Fault Tree Analysis (FTA) is a widely used method in reliability engineering and risk analysis, mainly because it enables modelling complex systems by encoding and displaying logical relationships that can be used, among others, to understand how a system might fail, trace the root cause of the failure, identify critical components, and calculate the system and subsystem failure probabilities.

Fault Tree (FT) models exist since the 1960s and have been used in a wide range of domains, including the automotive, aerospace, and nuclear industries (Kabir, 2017). However, a major drawback of FTs is related to their construction, which is traditionally carried out in conjunction with domain expertise and in a hand-crafted manner, resulting in a tedious and time-consuming task. In the case of complex industrial systems, manual development of these models can lead to incompleteness, inconsistencies, and even errors (Signoret and Leroy, 2021).

The above challenge has been discussed since the 1970s, and it is referred to in the literature as *construction* (Salem, Apostolakis, and Okrent, 1976), *synthesis* (Hunt, Kelly, Mullhi, et al., 1993), or *induction* (Madden and Nolan, 1994) of FTs. In this work, we refer to this as *automatic inference of FT models*, which in general, is the process that automatically (with limited human intervention) produces an FT model given compatible input information.

This problem shares some similarities with *System Identification* (SI), where the objective is to identify the mathematical model of a given system (Johnson and Husbands, 1990), although one difference we observe between SI and FTs inference is that for SI it is necessary to pre-define a model structure (e.g., based on laws of physics), which is not possible in the case of FTs inference as this is a task of the inference process.

We identify FTs inference challenging because there are many possible FTs for a given failure dataset, and finding the best match is not trivial. Existing methods fail as (i) they need too much human intervention to add assumptions e.g., to deal with complex dependencies between components; (ii) they do not scale adequately in real-world applications, especially algorithms that perform exhaustive search have exponential time complexity; (iii) they result in complex FT structures, (iv) it is unknown how reliable they are under noisy data.

Table 2.1: Toy input failure dataset.

BE1	BE2	BE3	TE
1	1	1	1
1	0	1	1
0	1	1	1
1	1	0	1
1	0	0	0
0	1	0	0
0	0	1	1
0	0	0	0

We are interested in data-driven approaches, whose challenge is illustrated by the following example: Table 2.1 shows a toy input *failure dataset* (Section I.4.2). Suppose the associated system is composed of the components BE1, BE2 and BE3, where 0 and 1 are used as non-faulty and faulty states, respectively. TE corresponds to the system-level failure.

Thus, for this failure dataset, we want to find the FT structure $\mathcal{F}$ that best encodes the logic that describes the failure propagation in the system. Moreover, we are interested in the FT $\mathcal{F}$ composed of a minimal amount of elements.

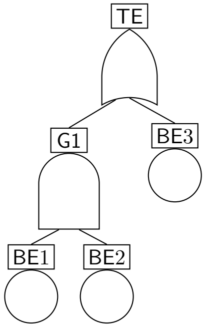


Figure 2.1: Inferred FT $\mathcal{F}$.

The desired output is presented in Figure 2.1 , where the inferred FT is composed of basic events BE1, BE2, connected to an AND-gate (G1), which together with BE3, is connected with an OR-gate to the top event TE.

Here the gates, which connect the basic events and the top event, result from the inference process following the logic described by the failure dataset. And although for this example the solution is rather simple, for larger failure datasets (i.e., with more basic events) the solution is not straightforward.

Contributions. This chapter offers:

(i) A demonstration that Multi-Objective Evolutionary Algorithms (MOEAs) can achieve more consistent and efficient FT structures by optimising multiple criteria simultaneously.

(ii) A new metric for comparing FT structures using Minimal Cut Sets and the RV-coefficient.

(iii) A parametric analysis elucidating the algorithm's performance under varying conditions.

(iv) Evidence that compact FTs improve convergence speed. Data and implementation are available at zenodo.org/record/5536431.

Outline. The remaining part of this chapter is organised as follows. Section 2.2 provides background on Fault Tree Analysis (FTA). Section 2.3 provides the technical background of Multi-Objective Evolutionary Algorithms (MOEAs). Section 2.4 explains our methodology. Section 2.5 presents how we apply the `NSGA-II` (an MOEA) to infer FTs. Section 2.6 presents the results of a thorough parametric analysis. Section 2.7 discusses our findings and presents our conclusions.

2.2 Fault Tree Analysis

Fault Tree Analysis (FTA) is a widely recognised method in reliability engineering that supports design and maintenance decisions for complex systems. FTA enables both *qualitative* and *quantitative* analyses. The qualitative analysis, based on the FT structure, identifies critical system components, with a focus on MCSs—minimal combinations of component failures leading to system failure. Smaller minimal cut sets highlight system vulnerabilities.

Quantitative analysis calculates dependability metrics, such as system *Reliability*, *Availability*, and *Mean-Time-to-Failure.* These calculations require the FT leaves to have assigned failure probabilities. For formal definitions and terminology related to *Fault Trees*, see Section I.4.1.

Figures 2.2(a) and 2.2(b) illustrate the *event* and *gate* symbols used in constructing the FT model. Figure 2.2(c) presents an example of an FT for a *Container Seal Design*, adapted from NASA, 2002. In this FT, the top event, *sealing function fails*, occurs either due to a *common cause seal failure* or independent seal failures. The former requires both *contamination tape failure* and a *basic cause seal failure*, while the latter necessitates failures in the *metal-to-metal seal*, *fused plug*, and at least two of the three *compression seals.*

2.3 Multi-Objective Evolutionary Algorithms

Evolutionary Algorithms (EAs) are population-based search strategies inspired by natural selection, where the most fit individuals are more likely to reproduce and pass on their traits to subsequent generations (Ojha, Singh, Chakraborty, et al., 2019). When EAs are used to optimise several conflicting objective functions simultaneously in a multi-dimensional space, they are termed Multi-Objective Evolutionary Algorithms (MOEAs) (Deb, 2011). MOEAs yield a set of solutions with trade-offs, known as *Pareto-optimal* solutions, from which users can select based on higher-level qualitative considerations (Deb, 2005).

To address the challenge of automatically inferring FTs from a failure dataset while optimising different metrics, we chose to employ the *Elitist Non-dominated Sorting Genetic Algorithm* (`NSGA-II`) (Section 2.3.1) and the *Crowding-Distance* approach (Section 2.3.2), both of which are widely used in multi-objective optimisation.

2.3.1 Elitist Non-dominated Sorting Genetic Algorithms

The *Elitist Non-dominated Sorting Genetic Algorithm* (`NSGA-II`) (Deb, Pratap, Agarwal, et al., 2002) is designed to find multiple Pareto-optimal solutions. `NSGA-II` employs the *elitist* principle, a diversity-preserving mechanism that focuses on non-dominated solutions (Deb, 2005). This principle ensures solution quality by allowing the best individual(s) of the current generation to advance to the next.

Non-dominated MOEAs rely on the concept of *dominance*, comparing two solutions to determine if one dominates the other. Non-dominated sorting is important for identifying elitist efficient solutions in MOEAs, but it is computationally intensive due to the numerous comparisons required (Long, X. Wu, and C. Wu, 2021). A set of solutions that do not dominate each other forms a *non-dominated front.* Appendix B.2 provides a detailed explanation of these concepts and demonstrates the application of `NSGA-II` in the automatic inference of FTs.

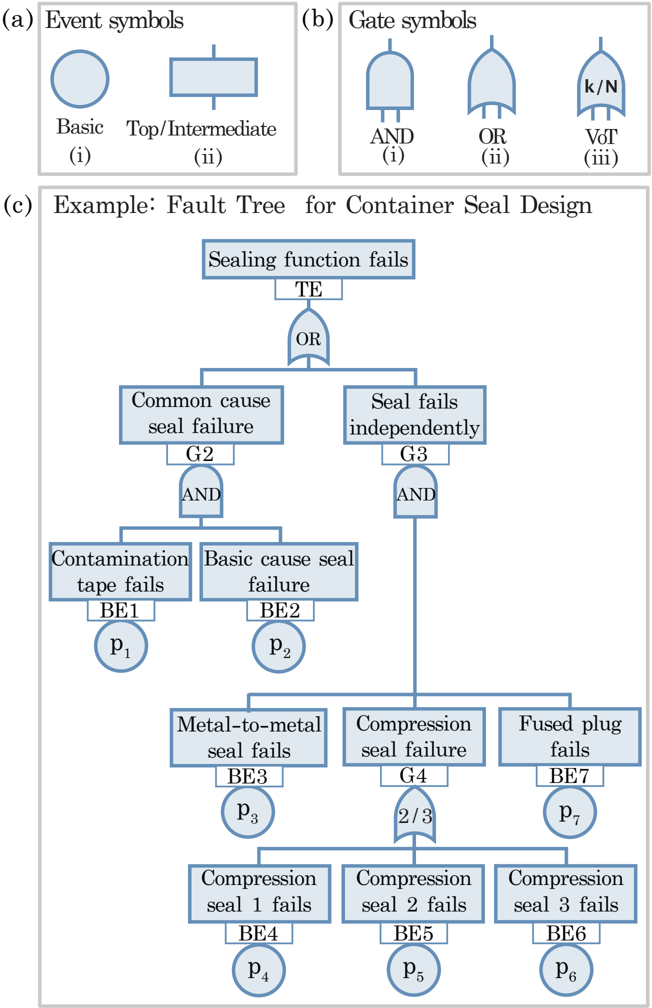


Figure 2.2: Elements in an FT: (a) event symbols, (b) gate symbols, and (c) example of an FT, adapted from NASA, 2002.

2.3.2 Crowding-Distance

If the last non-dominated front obtained through NSGA-II does not completely fill the available slots for the new population, the *Crowding-Distance* is utilised to determine which individuals from the last front should advance to the next generation. This mechanism promotes diversity (Martí, Segredo, Śnchez-Pi, et al., 2017). Appendix B.2.2 provides details on the application of Crowding-Distance in the automatic inference of FTs.

2.4 Methodology

Figure 2.3 depicts the general methodology we followed in this chapter. First, we selected some case studies of existing FTs (Section 2.6.2), these FTs act as ground truth. Then, we selected some parameters of interest (Section 2.6.4) to be evaluated in the parametric analysis. We used the Monte Carlo method (Section 2.6.1) to generate failure datasets (Section 2.4.1) based on selected case studies (Section 2.6.2). Then we used our `FT-MOEA` algorithm (Section 2.5) to infer the FT based on the provided failure dataset. Finally, we compared the ground truth with the inferred FTs and evaluate the experiment (Section 2.6).

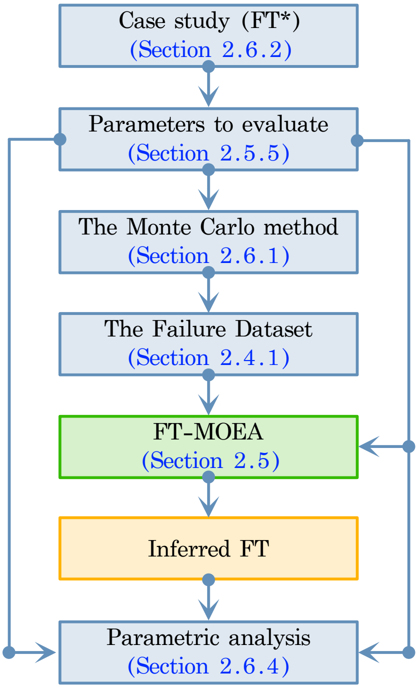


Figure 2.3: General methodology followed in this chapter.

2.4.1 The Failure Dataset

We formally defined the failure dataset in Section I.4.2. For this chapter, we also assume the following:

- *Labelled*: The dataset contains combinations of BE and their corresponding TE.
- *Binary*: Both BE and TE are binary, allowing for the use of Boolean operations, which enhances algorithm efficiency. Here, 0 and 1 represent non-faulty and faulty states, respectively.
- *Monotonic/consistent*: For any set of BE, if a BE changes from 0 to 1, TE may change from 0 to 1 but will never change from 1 to 0.
- *Complete*: The number of unique BE combinations in the failure dataset matches the space complexity $\mathcal{O}(2^w)$, where w is the number of unique BE for a given FT.
- *Noise-free*: The failure dataset contains no corrupted information; the relationship BE → TE is always accurate for a given FT.

Table 2.2 illustrates an example dataset corresponding to the FT depicted in Figure 2.2. This dataset was generated using the Monte Carlo method outlined in Section 2.6.1, with $\mathrm{N} = 250,000$ data points and a failure rate of $p_i = 0.5$ for each BE. *Ob.* denotes the *observation* associated with a unique combination of BE values and the corresponding TE. The columns BE1, BE2, ..., BE7 represent the *states* of the BE set, while TE indicates the *top event*. The final column shows the *count* of each observation in the failure dataset.

Table 2.2: Example of failure dataset associated to the example in Figure 2.2.

Ob.	BE1	BE2	BE3	BE4	BE5	BE6	BE7	TE	*Count*
1	0	0	0	0	0	0	0	0	1,968
2	0	0	0	0	0	0	1	0	2,039
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
24	0	0	1	0	1	1	1	1	1,976
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
128	1	1	1	1	1	1	1	1	1,947
$p \approx$	0.50	0.50	0.50	0.50	0.50	0.50	0.50	$N =$	250,000

2.5 Inferring fault trees via multi-objective evolutionary algorithms (FT-MOEA)

The algorithm takes as input the failure dataset (Section 2.4.1) and the initial MOEA parameters (Section 2.5.1). It outputs a string describing the inferred FT structure (AND- and OR-gates) and relevant error metrics between the inferred FT and the failure dataset, as detailed in Section 2.5.5. Figure 2.4 outlines our approach, comprising five steps:

1. Initialise the algorithm by loading the failure dataset (Section 2.4.1) and initial parameters (Section 2.5.1). Optionally, extract MCSs from the failure dataset (Section 2.5.2) if the objective function uses MCSs-based metrics.
2. Initialise the population with parent fault tree(s) (Section 2.5.3) and apply genetic operators to reach the desired population size.
3. Apply genetic operators (Section 2.5.4) to modify the FTs structure, repeating until the desired population size is achieved.
4. Calculate the optimisation metrics for each FT in the offspring population (Section 2.5.5). Determine the next generation's FTs using NSGA-II and *Crowding-Distance* (Section 2.3).
5. Evaluate convergence criteria (Section 2.5.6). If they are not met, repeat Steps 3 to 5 until a criterion is satisfied, resulting in a Pareto set of inferred FTs. Select the best individual with the smallest size and error(s) from the first Pareto set.

2.5.1 Step 1 - Initialisation

We have the following initial parameters:

- *Population size* (*ps*): Corresponds to the number of FTs within a generation. Only the best *ps* FTs can pass to the next generation.
- *Selection strategy*: For the NSGA-II algorithm we only use the *elitist* selection strategy.

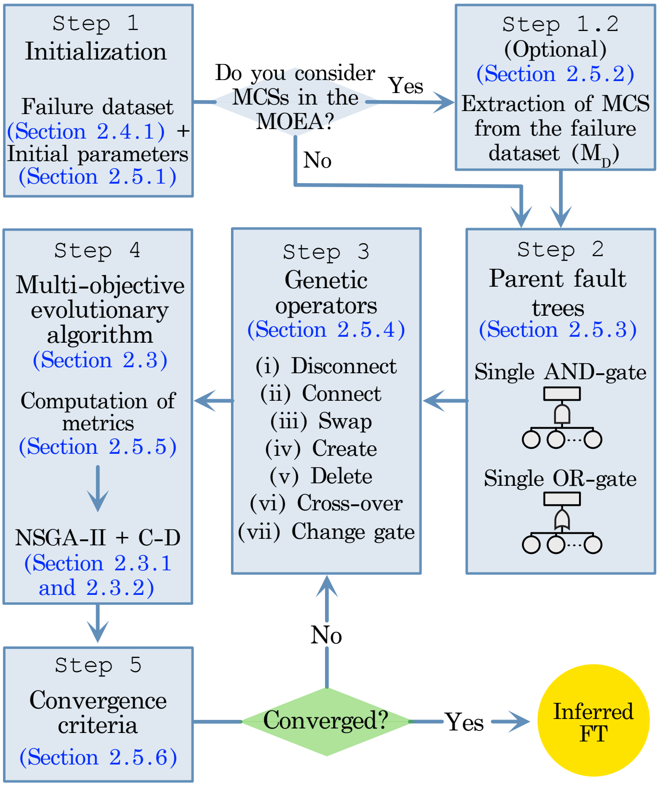


Figure 2.4: General process of the FT-MOEA algorithm to infer FTs from a failure dataset.

- *Max. generations with unchanged best candidate* (*uc*): if after *uc* number of generations the best individual (i.e., the FT with the smallest size, and smallest error(s) within the best Pareto set) remains unchanged, then we assume the process has converged and is therefore terminated.
- *Max. number of generations* (*ng*): Terminates the optimisation process if the number of generations exceeds *ng* and none of the other convergence criteria is met.

2.5.2 Step 1.2 - Extraction of MCSs from the failure dataset (optional step)

Minimal Cut Sets (MCSs) are minimal combinations of component failures leading to system failure, encoding the system's failure modes. Including this information in optimisation can enhance algorithm efficiency. However, using MCSs in the optimisation process is optional.

MCSs should be considered only if the failure dataset is *noise-free* and the expected FT is not overly complex (see Section 2.6.4 for more on FTs complexity). Otherwise, inaccurate MCSs or high computational costs could extend convergence time (discussed in Section 2.5.5).

The extraction of MCSs from a failure dataset follows Lazarova-Molnar, Niloofar, and Barta, 2020: *Step 1:* Identify all BE combinations resulting in class 1 (TE = 1). *Step 2:* Select the combination with the minimal order (fewest *true* BE) and add it to the MCSs matrix ($\mathsf{M}_{\mathbb{D}}$). *Step 3:* Remove any remaining combinations in the sub-set containing the identified MCS. *Step 4:* Repeat Steps 2 and 3 until the sub-set is empty.

The resulting $\mathsf{M}_{\mathbb{D}}$ can be used to compute accuracy based on MCSs (ε_c) (see Eq. 2.3). Table 2.3 shows an example of $\mathsf{M}_{\mathbb{D}}$ for the failure dataset in Table 2.2.

Table 2.3: Example of MCS matrix ($\mathsf{M}_{\mathbb{D}}$) computed from the failure dataset described in Table 2.2 associated with the example in Figure 2.2.

MCS	BE1	BE2	BE3	BE4	BE5	BE6	BE7
1	1	1	0	0	0	0	0
2	0	0	1	1	1	0	1
3	0	0	1	0	1	1	1
4	0	0	1	1	0	1	1

2.5.3 Step 2 - Parent fault tree(s)

The *parent fault tree(s)* serve as the basis from which the offspring population is generated using *genetic operators* (Section 2.5.4). The choice of parent FT(s) is crucial as it influences the initial distance from the global optimum in the optimisation process. In Linard, Bucur, and Stoelinga, 2019, two parent FTs are used: one connects the set of BE to a single OR gate, and the other to a single AND gate.

2.5.4 Step 3 - Genetic operators

The genetic operators are mathematical functions designed to alter the structure of an FT. We employ the seven operators proposed by Linard, Bucur, and Stoelinga, 2019, which also provides their formal definitions A brief overview is given next. Let $G = V \setminus \mathsf{BE}$ be the set of *gates* in the FT:

(i) G-create: Randomly creates an AND or OR gate under an existing gate in the set G for a given FT.

(ii) G-mutate: Randomly selects a gate in the set G and changes its type (i.e., OR → AND or AND → OR).

(iii) G-delete: Selects a gate in the set G of a given FT and deletes it along with its children.

(iv) BE-disconnect: Selects a basic event in the set BE of a given FT and disconnects it.

(v) BE-connect: Takes a disconnected basic event from a given FT and randomly places it under a gate in the set G.

(vi) BE-swap: Moves a basic event in the set BE of a given FT to a different parent gate in the set G.

(vii) Crossover: Randomly selects two FTs in the offspring population, then chooses an element in the set V of each FT to exchange.

2.5.5 Step 4 - Multi-objective Function

Section 2.5.5 defines the metrics to be minimised, while Section 2.5.5 outlines the various setups for our multi-objective optimisation function.

Metrics Calculation

Our multi-objective function considers three metrics: *fault tree size* (ε_s), *error based on failure data* (ε_d), and *error based on MCSs* (ε_c).

- *Fault Tree Size* (ε_s): Number of elements V in a FT (Eq. 2.1):

$$\varepsilon_s = |\mathrm{V}| = |\mathsf{BE}| + |\mathrm{G}|, \tag{2.1}$$

  here, $\varepsilon_s \geq 2$ since every FT has at least one BE and one G (i.e., the TE).
- *Error Based on Failure Data* (ε_d): Calculated using a vector P with N values, where N is the number of data points. P contains the TE values for a given set BE. The corresponding ground truth top event (TE*) is provided in the failure dataset (Section 2.4.1). ε_d is computed as:

$$\varepsilon_d = 1 - \frac{\sum_{i=1}^{\mathrm{N}} x_i}{\mathrm{N}} \begin{cases} x_i = 1, \textit{ if } \mathrm{P_i} = \mathsf{TE}^*_{\mathrm{i}}. \\ x_i = 0, \text{ Otherwise.} \end{cases} \tag{2.2}$$

  ε_d ranges from 0 to 1, where 0 indicates perfect mapping of BE to the corresponding TE in the failure dataset.
- *Error Based on MCSs* (ε_c): Computed using the *RV-coefficient* (Robert and Escoufier, 1976), which generalises the *squared Pearson correlation coefficient* to measure similarity between the MCS matrix from the failure dataset ($\mathsf{M}_{\mathbb{D}}$) and the MCS matrix of a given FT ($\mathsf{M}_{\mathcal{F}}$).

  $\mathsf{M}_{\mathcal{F}}$ is derived from the *disjunctive normal form* (DNF), which requires transforming an FT into its DNF to identify MCSs and construct $\mathsf{M}_{\mathcal{F}}$. This transformation is computationally intensive for large FT sizes.

  $\mathsf{M}_{\mathbb{D}}$ and $\mathsf{M}_{\mathcal{F}}$ are matrices of sizes $p \times w$ and $q \times w$ respectively, where w is the number of unique BEs, and p and q are the numbers of MCSs in the failure dataset and FT, respectively. The computation of ε_c is given by:

$$\varepsilon_c = 1 - \frac{\mathrm{tr}(\mathsf{M}_{\mathbb{D}}\mathsf{M}_{\mathcal{F}}^T\mathsf{M}_{\mathcal{F}}\mathsf{M}_{\mathbb{D}}^T)}{\sqrt{\mathrm{tr}(\mathsf{M}_{\mathbb{D}}\mathsf{M}_{\mathbb{D}}^T)^2\,\mathrm{tr}(\mathsf{M}_{\mathcal{F}}\mathsf{M}_{\mathcal{F}}^T)^2}} \tag{2.3}$$

Here, tr(.) represents the *trace*, and ε_c ranges from 0 to 1, where 0 denotes perfect correlation between $\mathsf{M}_{\mathbb{D}}$ and $\mathsf{M}_{\mathcal{F}}$. The RV-coefficient is utilised due to the consistent number of unique BEs across all problems, despite the differing numbers of MCSs in FTs relative to those in the failure dataset ($p \neq q$).

Setups of the multi-objective functions

Given that our multi-objective function (m.o.f.) has three arguments, we can explore various configurations to evaluate their impact (see Section 2.6.4 for parametric analysis results). For instance, to minimise only the error based on MCSs (ε_c), we can deactivate ε_s and ε_d by setting them to constants (e.g., $\varepsilon_s = \varepsilon_d = 1$). To distinguish between different m.o.f. configurations, we adopt the nomenclature in Table 2.4, where '✓' indicates whether a metric is considered (or active).

Table 2.4: Different setups of the m.o.f.

m.o.f.	ε_s	ε_d	ε_c
sdc	✓	✓	✓
dc		✓	✓
sc	✓		✓
sd	✓	✓	
c			✓
d		✓	

2.5.6 Step 5 - Convergence criterion

Our convergence criterion is defined by two initial parameters: the *max. number of generations* (ng) and the *max. generations with unchanged best candidate* (see Section 2.5.1). The convergence process is also terminated if $\varepsilon_c = 0$ or $\varepsilon_d = 0$ when the minimisation of the FT size is deactivated, specifically for the m.o.f.'s cd, c, or d.

2.6 Experimental evaluation

For our experimental evaluation, we selected six case studies from the literature (Section 2.6.2) and implemented our FT-MOEA algorithm in Python; the source code and data are available at zenodo.org/record/5536431. We evaluated the algorithm using synthetic failure datasets (Section 2.6.1). Section 2.6.3 compares FT-MOEA with FT-EA and provides details on convergence, while Section 2.6.4 presents our parametric analysis.

2.6.1 The Monte Carlo method

We use the Monte Carlo method to generate synthetic failure datasets based on the case studies in Section 2.6.2, maintaining the properties of the input dataset described in Section 2.4.1. To generate the synthetic dataset, we: (i) randomly generate (N) data points by drawing the BE *independently* from a *binomial distribution* with a success probability of p_i (where i represents a basic event), and (ii) compute the corresponding TE using the logical rules of the given FT (e.g., case studies in Section 2.6.2). To ensure the failure dataset is *complete* (see Section 2.4.1), we draw sufficient data points from the Monte Carlo simulation

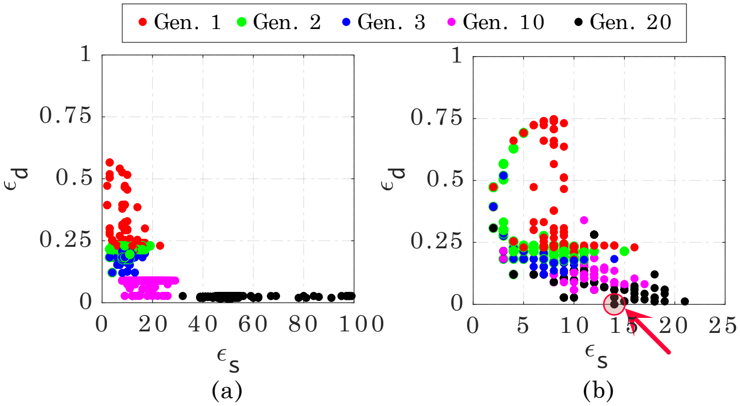


Figure 2.5: Evolution of metrics over generations: (a) using the m.o.f. `d`, and (b) using the m.o.f. `sdc`, for the MPPS case study ($ps = 400$, $ng = 100$, $uc = 20$). The red circle with an arrow at the bottom of (b) marks the global optimum.

so that the number of unique observations of BE matches the space complexity $\mathcal{O}(2^w)$, where w is the number of unique BE for a given FT.

2.6.2 Case studies

To establish a reliable ground truth, we utilised existing FTs from the literature across different applications. Our selection criteria included the number of elements in the FT, the number of MCSs, and their orders. Table 2.5 lists the selected case studies, providing details on the number of BE, the total number of AND, OR, and VoT gates, the number of MCSs, their orders, and the space complexity, measured as $\mathcal{O}(2^w)$. Since we only work with complete datasets, the space complexity in Table 2.5 also reflects the size of the failure dataset. We distinguish between the number of unique BE (w) and the total number of BE (W) because some FTs contain shared BEs.

2.6.3 Key findings of the `FT-MOEA` algorithm

To illustrate our findings and main contributions, we use the Mono-propellant propulsion system (MPPS) case study from Table 2.5. We first generate the failure dataset as described in Section 2.6.1, containing $N = 250,000$ data points. This dataset is then used as input for the `FT-MOEA` algorithm, with the following initial parameters: $ps = 400$, $ng = 100$, and $uc = 20$.

We compare the evolutionary process across generations for two m.o.f.s, `d` and `sdc`, to contrast the approach by Linard, Bucur, and Stoelinga, 2019 (`FT-EA`, minimising only ε_d) with our multi-objective optimisation process (`FT-MOEA`, minimising ε_s, ε_d, and ε_c).

Table 2.5: Case studies and associated relevant information. The number of: unique BEs (w), total BEs (W), AND gates (#AND), OR gates (#OR), VoT gates (#VoT), Minimal Cut Sets (#MCS), order of MCSs (O-MCSs), and space complexity ($\mathcal{O}(2^w)$).

Case study	*Ref.*	w	W	#AND	#OR	#VoT	FT_{size}	#MCSs	O-MCSs	$\mathcal{O}(2^w)$
CSD[a]	NASA, 2002	6	6	2	2	0	10	3	{2,3,3}	64
PT[b]	NASA, 2002	6	6	1	4	0	11	5	{1,1,2,2,2}	64
COVID-19[c]	Bakeli, Hafidi, et al., 2020	9	21	9	3	0	33	6	{3,4,4,4,4,4}	512
ddFT[d]	Lazarova-Molnar, Niloofar, and Barta, 2020	8	8	3	1	1	13	6	{3,4,4,5,6,6}	256
MPPS[e]	NASA, 2002	8	12	3	8	0	23	7	{2,2,2,2,2,2,2}	256
SMS[f]	Mentes and Helvacioglu, 2011	13	17	0	8	0	25	13	{1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1}	8192

[a] CSD: Container Seal Design; [b] PT: Pressure Tank; [c] COVID-19: COVID-19 infection risk; [d] ddFT: Data-driven Fault Tree; [e] MPPS: Mono-propellant propulsion system; [f] SMS: Spread Monitoring System.

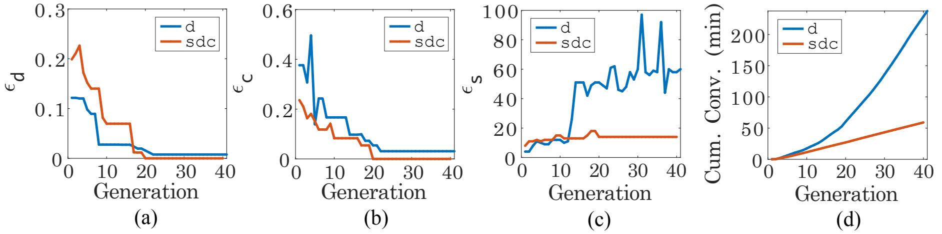


Figure 2.6: Metrics over generations for the best FTs using the m.o.f.s `d` and `sdc`. Convergence of (a) ε_d, (b) ε_c, (c) ε_s, (d) cumulative convergence time. Using the MPPS case study ($ps = 400$, $ng = 100$, $uc = 20$).

Figure 2.5(a) shows the results across generations when minimising solely ε_d. A rapid decrease in ε_d is observed in the initial generations, but from the 10th generation onwards, there is a sharp increase in FTs size (ε_s) (up to $\varepsilon_s = 100$) without further reduction in ε_d, while the ground truth FT has $\varepsilon_s = 23$. Conversely, using `FT-MOEA` (Figure 2.5(b)) results in a more balanced decrease in all directions. Moreover, `FT-MOEA` reaches the global optimum (i.e., $\varepsilon_d = \varepsilon_c = 0.0$) by the 20th generation with $\varepsilon_s = 14$ (red dashed circle with an arrow in 2.5(b)), an equivalent compressed version of the ground truth FT.

In Figure 2.6, we compare both m.o.f.s across generations, focusing on the metrics of the best FT per generation (i.e., the one on the first Pareto front with the smallest errors ε_d and ε_c). Figure 2.6(a) shows ε_d across generations for both objective functions, indicating that the m.o.f. `d` more quickly minimises ε_d than m.o.f. `sdc`. However, m.o.f. `sdc` achieves the global optimum by the 20th generation, whereas m.o.f. `d` does not.

Figure 2.6(b) compares ε_c, showing similar trends. Figure 2.6(c) depicts ε_s variation over generations, illustrating that our m.o.f. maintains smaller FT structures. Although the ground truth FT size is 23, `FT-MOEA` finds one with $\varepsilon_s = 14$, an equivalent and compressed version of the original (see Appendix B.3, Figure B.2 for details).

Figure 2.6(d) shows the cumulative convergence time (t) for both m.o.f.s. Our algorithm finds the optimal solution in about 20 minutes, whereas minimising only ε_d takes about 4 hours without reaching the global optimum. Details on the convergence of metrics across generations for the entire population are provided in Appendix B.4.

2.6.4 Parametric analysis

In our parametric analysis, we consider the population size, the multi-objective functions, the FT complexity, and the effect of superfluous BEs. Additionally, we assess the impact of varying the parent FT in Appendix B.5. These parameters were explored to understand their influence on computational time and convergence.

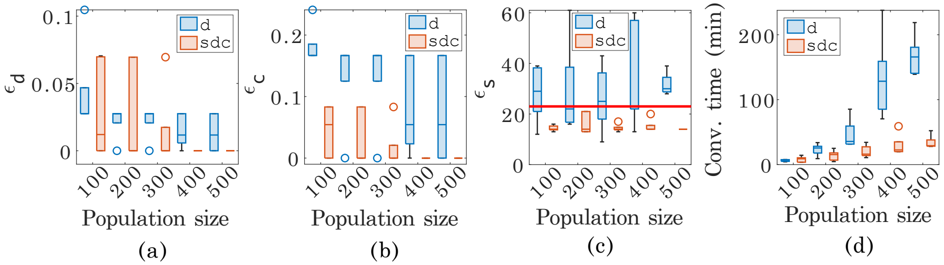


Figure 2.7: Influence of population size (ps) on (a) ε_d, (b) ε_c, (c) ε_s, and (d) convergence time for the m.o.f.s `sdc` and `d` in the MPPS case study ($ps = 400$, $ng = 100$, $uc = 20$).

We generate the failure dataset as described in Section 2.6.1. Given that the evolutionary algorithm is a stochastic process, we run our algorithm *five* times for each parameter combination until convergence. Using *box charts* in Matlab (e.g., Figure 2.7), we represent the groups of numerical data through their quartiles.

Population size

Figure 2.7 presents the results of the parametric analysis when varying the population size for the m.o.f.s `d` and `sdc`, using the MPPS case study (Table 2.5).

Figures 2.7(a) and 2.7(b) show that the m.o.f. `sdc` is more consistent with larger population sizes, with both errors (ε_c and ε_d) generally decreasing as the population size increases. Conversely, with m.o.f. `d`, errors also decrease with larger population sizes, but with less consistency. Figure 2.7(c) indicates that the m.o.f. `sdc` consistently produces smaller FTs than m.o.f. `d`, often even smaller than the ground truth (i.e., $\varepsilon_s \leq 23$), denoted by the horizontal red line. Figure 2.7(d) illustrates that larger population sizes exponentially increase computational time for both m.o.f.s. However, the m.o.f. `sdc` remains consistently faster.

Multi-objective functions

We evaluate all m.o.f. setups (Table 2.4) using the case studies from Table 2.5 and fixed input parameters ($ps = 400$, $ng = 100$, and $uc = 20$). Figure 2.8 presents results for the case studies COVID-19, MPPS, and ddFT. Results for the case studies CSD, PT, and SMS are shown in Figure B.6 (Appendix B.6).

Figure 2.8(a) shows the error based on the failure dataset (ε_d). Different objective functions exhibit varied behaviours. The m.o.f. `dc` achieves the exact solution for all cases, whereas the other m.o.f.s fail to find the global optimum in at least one case.

Similarly, Figure 2.8(b) shows the error based on MCSs (ε_c). As expected, the m.o.f. `dc` achieves $\varepsilon_c = 0.0$ for all cases. However, note that a low ε_d does not necessarily imply an optimal FT; for instance, compare ε_d and ε_c for the MPPS case using m.o.f.s `d` and `sd`.

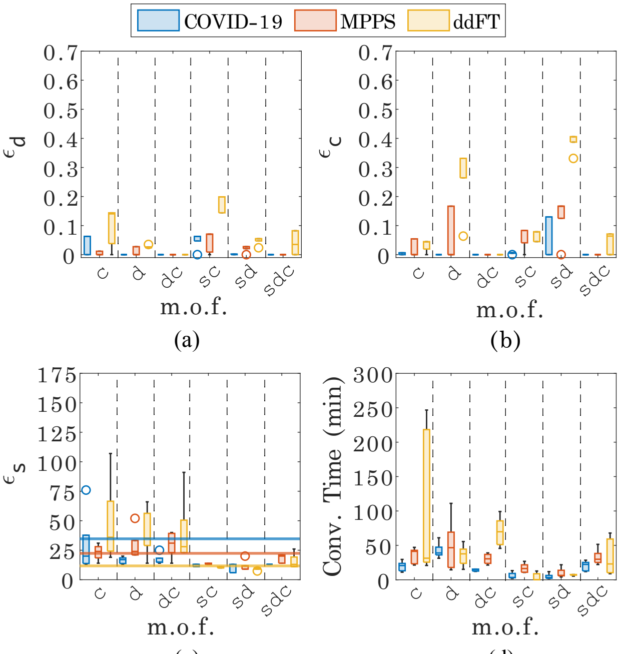


Figure 2.8: Comparison of m.o.f. performance for all case studies in Table 2.5, with $ps = 400$, $ng = 100$, and $uc = 20$: (a) ε_d, (b) ε_c, (c) ε_s, and (d) convergence time.

Errors ε_d and ε_c, particularly for the ddFT case study, tend to increase when ε_s is minimised (i.e., m.o.f.s sc, sd, and sdc), suggesting that FT-MOEA may converge to a FT with a slightly larger error but smaller size.

Figure 2.8(c) shows the sizes of the inferred FTs (ε_s). The influence of minimising ε_s is evident; when considered, ε_s in most cases is equal to or smaller than the ground truth, as indicated by the horizontal lines for different case studies (see Appendix C for examples and details). When not considered, ε_s may exceed the ground truth in some cases.

Fault tree complexity

Figure 2.8(d) depicts convergence time. Generally, for all m.o.f.s, case studies ordered by convergence time from longest to shortest are *ddFT*, *MPPS*, *COVID-19*, *CSD*, *PT*, and *SMS*. This suggests a relationship between the *complexity* of the underlying FT model and the time required for the algorithm to find it.

We hypothesise that this complexity is influenced by the number of MCSs and their orders (see *O-MCSs* in Table 2.5). For example, the *ddFT* case study, with six MCSs and orders between 3 and 6, typically took the longest to converge. In contrast, the *SMS* case study, with 13 MCSs all of order 1, converged almost immediately.

Thus, a higher number of MCSs and their orders may increase the time needed to reach the global optimum. Further research is needed to better quantify this type of complexity.

Influence of superfluous Basic Events

Real-world datasets may include varying numbers of BEs, not all contributing to system failure. Superfluous variables, which do not affect the TE regardless of their state, are termed *superfluous BEs*. We evaluate number of superfluous BEs (ρ) ranging from 0 to 6 using the MPPS case study with $ps = 400$, $ng = 100$, $uc = 20$, and m.o.f.s sdc and d. Figure 2.9(a) illustrates ε_s across different ρ values. When using m.o.f. sdc, ε_s is smaller than the ground truth (dashed horizontal red line). In contrast, with m.o.f. d, ε_s increases with higher ρ values. Figure 2.9(b) shows the additional or missing number of BEs (± BEs) for different ρ values. The MPPS case study has 7 unique BEs; we subtract this from the unique BEs of each inferred FT. The m.o.f. sdc consistently yields an FT with 7 BEs (± BEs $= 0$), indicating that superfluous BEs are removed during optimisation. Conversely, m.o.f. d shows variability and performs less effectively as ρ increases.

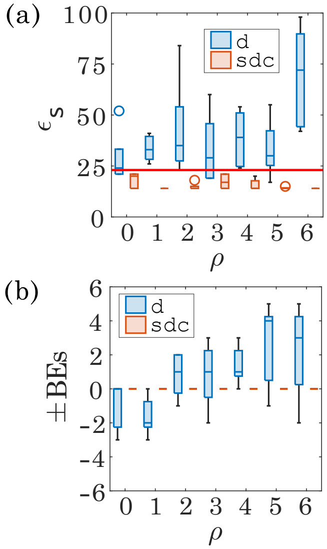


Figure 2.9: Influence of *number of superfluous Basic Events* (ρ) on (a) FT size (ε_s), (b) additional or missing number of BEs (± BEs). Using the m.o.f.s sdc and d, and the MPPS case study ($ps = 400$, $ng = 100$, $uc = 20$).

2.7 Discussion and Conclusions

We demonstrate that efficient and interpretable Fault Tree (FT) structures can be inferred using multi-objective evolutionary algorithms. However, several challenges must be addressed before real-world applications can be considered. Although our algorithm outperforms alternative approaches, its *scalability* presents a challenge for FTs with numerous BEs. A potential solution is to employ "*guided*" *multi-objective evolutionary algorithms*, which enhance the optimisation process by incorporating additional knowledge. One approach involves identifying *patterns* in the failure dataset to assemble parts of the FT (e.g., Waghen and Ouali, 2021). Another involves *guiding* the application of *genetic operators* by targeting FT components that likely require modification, potentially achieved through Bayesian optimisation. Alternatively, deep learning-based approaches, such as the one proposed by Cranmer, Sanchez-Gonzalez, Battaglia, et al., 2020, could be explored to derive symbolic rules from Graph Neural Networks.

Incorporating MCSs into the multi-objective optimisation function significantly improves the process. However, the `FT-MOEA` algorithm computes MCSs using disjunctive normal form, which is computationally expensive for large FTs. Additionally, MCSs cannot be computed in noisy data (see Appendix B.7 for preliminary results on noise effects). Therefore, exploring alternatives to address these issues is crucial. Finally, several challenges deserve further investigation:

- Obtaining noise-free, balanced, and complete failure datasets for complex engineering systems is nearly impossible. Thus, further evaluation of our algorithm's performance with incomplete, noisy, and unbalanced datasets is necessary.
- Real-world problems often contain symmetries, such as fully exchangeable basic events. Leveraging these symmetries could reduce the solution space and accelerate convergence, requiring research in this direction.
- Exploring methods to infer more sophisticated gates (e.g., VoT gates) is key for obtaining more compact and efficient FT structures.
- The methodology used in this chapter could be extended to infer other reliability models, such as reliability block diagrams and Boolean circuits.
- System identification techniques may be applicable to FT inference, warranting further research.
- Further research is needed to understand and quantify the complexity in FT model inference. Developing guidelines and metrics to assess the practical capabilities of FT inference algorithms remains an open challenge.

Overall, our novel algorithm, `FT-MOEA`, outperforms its predecessor, `FT-EA`, by converging faster, inferring more compact FT structures, achieving lower error levels, better removing superfluous variables, and maintaining consistency.

2.8 References

Bakeli, T., A. A. Hafidi, et al. (2020). “COVID-19 infection risk management during construction activities: An approach based on Fault Tree Analysis (FTA)”. In: *Journal of Emergency Management* 18.7, pp. 161–176. DOI: 10.5055/jem.0539.

Cranmer, M., A. Sanchez-Gonzalez, P. Battaglia, R. Xu, K. Cranmer, D. Spergel, and S. Ho (2020). *Discovering Symbolic Models from Deep Learning with Inductive Biases.* DOI: 10.48550/arXiv.2006.11287. arXiv: 2006.11287 [cs.LG].

Deb, K., A. Pratap, S. Agarwal, and T. Meyarivan (2002). “A fast and elitist multiobjective genetic algorithm: NSGA-II”. In: *IEEE Transactions on Evolutionary Computation* 6 (2), pp. 182–197. DOI: 10.1109/4235.996017.

Deb, K. (2005). “Multi-Objective Optimization”. In: *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques.* Ed. by E. K. Burke and G. Kendall. Boston, MA: Springer US, pp. 273–316. ISBN: 978-0-387-28356-2. DOI: 10.1007/0-387-28356-0_10.

– (2011). “Multi-objective optimisation using evolutionary algorithms: an introduction”. In: *Multi-objective evolutionary optimisation for product design and manufacturing.* Springer, pp. 3–34. DOI: 10.1007/978-0-85729-652-8_1.

Hunt, A., B. E. Kelly, J. S. Mullhi, F. P. Lees, and A. G. Rushton (1993). "The propagation of faults in process plants: 6, Overview of, and modelling for, fault tree synthesis". In: *Reliability Engineering & System Safety* 39 (2), pp. 173–194. DOI: 10.1016/0951-8320(93)90041-V.

Johnson, T. and P. Husbands (1990). "System identification using genetic algorithms". In: *International Conference on Parallel Problem Solving from Nature*. Springer, pp. 85–89.

Kabir, S. (2017). "An overview of fault tree analysis and its application in model based dependability analysis". In: *Expert Systems with Applications* 77, pp. 114–135. DOI: 10.1016/j.eswa.2017.01.058.

Lazarova-Molnar, S., P. Niloofar, and G. K. Barta (2020). "Data-Driven Fault Tree Modeling for Reliability Assessment of Cyber-Physical Systems". In: *2020 Winter Simulation Conference (WSC)*, pp. 2719–2730. DOI: 10.1109/WSC48552.2020.9383882.

Linard, A., D. Bucur, and M. Stoelinga (2019). "Fault Trees from Data: Efficient Learning with an Evolutionary Algorithm". In: *International Symposium on Dependable Software Engineering: Theories, Tools, and Applications*. Vol. 11951 LNCS, pp. 19–37. DOI: 10.1007/978-3-030-35540-1_2.

Long, Q., X. Wu, and C. Wu (2021). "Non-dominated sorting methods for multi-objective optimization: Review and numerical comparison". In: *Journal of Industrial & Management Optimization* 17.2, p. 1001. DOI: 10.3934/jimo.2020009.

Madden, M. G. and P. J. Nolan (1994). "Generation of fault trees from simulated incipient fault case data". In: *WIT Transactions on Information and Communication Technologies* 6. DOI: 10.2495/AI940611.

Martí, L., E. Segredo, N. Śnchez-Pi, and E. Hart (2017). "Impact of selection methods on the diversity of many-objective Pareto set approximations". In: *Procedia Computer Science* 112. Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 21st International Conference, KES-20176-8 September 2017, Marseille, France, pp. 844–853. ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2017.08.077.

Mentes, A. and I. H. Helvacioglu (2011). "An application of fuzzy fault tree analysis for spread mooring systems". In: *Ocean Engineering* 38 (2-3), pp. 285–294. DOI: 10.1016/j.oceaneng.2010.11.003.

NASA (2002). *Fault Tree Handbook with Aerospace Applications*. Handbook. U.S. National Aeronautics and Space Administration.

Ojha, M., K. P. Singh, P. Chakraborty, and S. Verma (2019). "A review of multi-objective optimisation and decision making using evolutionary algorithms". In: *International Journal of Bio-Inspired Computation* 14.2, pp. 69–84. DOI: 10.1504/ijbic.2019.101640.

Robert, P. and Y. Escoufier (1976). "A unifying tool for linear multivariate statistical methods: The RV-coefficient". In: *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 25 (3), pp. 257–265. DOI: 10.2307/2347233.

Salem, S. L., G. E. Apostolakis, and D. Okrent (Nov. 1976). "Computer-oriented approach to fault-tree construction". In: DOI: 10.2172/7132148.

Signoret, J.-P. and A. Leroy (2021). "Automated Fault Tree Building". In: *Reliability Assessment of Safety and Production Systems: Analysis, Modelling, Calculations and Case Studies*. Cham: Springer International Publishing, pp. 423–426. ISBN: 978-3-030-64708-7. DOI: 10.1007/978-3-030-64708-7_28.

Waghen, K. and M.-S. Ouali (2021). "Multi-level interpretable logic tree analysis: A data-driven approach for hierarchical causality analysis". In: *Expert Systems with Applications* 178, p. 115035. ISSN: 0957-4174. DOI: 10.1016/j.eswa.2021.115035.

Chapter 3

Data-Driven Inference of Fault Tree Models Exploiting Symmetry and Modularisation

Paper published at **L. A. Jimenez-Roa**, M. Volk, M. Stoelinga, *"Data-Driven Inference of Fault Tree Models Exploiting Symmetry and Modularisation",* in Trapp, M., Saglietti, F., Spisländer, M., Bitsch, F. (eds) Computer Safety, Reliability, and Security. SAFECOMP 2022. Lecture Notes in Computer Science, vol 13414. Springer, Cham. doi: 10.1007/978-3-031-14835-4_4.

Abstract

We present *SymLearn*, a method to automatically infer fault tree (FT) models from data. *SymLearn* takes as input failure data of the system components and exploits evolutionary algorithms to learn a compact FT matching the input data. *SymLearn* achieves scalability by leveraging two common phenomena in FTs: (i) We automatically identify symmetries in the failure dataset, learning symmetric FT parts only once. (ii) We partition the input data into independent modules, subdividing the inference problem into smaller parts. We validate our approach via case studies, including several truss systems, which are symmetric structures commonly found in infrastructures, such as bridges. Our experiments show that, in most cases, the exploitation of modules and symmetries accelerates the FT inference from hours to under three minutes.

3.1 Introduction

Fault Tree Analysis (FTA) (NASA, 2002; Ruijters and Stoelinga, 2015) is one of the most prominent methods in reliability engineering, used on a daily basis by thousands of engineers. *Fault Trees (FTs)* are a graphical model describing how failures occurring in (atomic) system components propagate through a system and eventually lead to an overall system failure. The quantitative and qualitative analysis of FTs is important for the risk management of complex engineering systems.

An important challenge in FTA is the creation of faithful FT models. Therefore, inference of FTs, also known as *construction* (Salem, Apostolakis, and Okrent, 1976), *synthesis* Hunt, Kelly, Mullhi, et al., 1993, or *induction* Madden and Nolan, 1994, has been investigated since the 1970s. Three categories of approaches exist: (i) *Knowledge-based* methods were investigated first, and are semi-automated approaches that derives an FT from a knowledge-based representation using heuristics (Carpignano and Poucet, 1994). These deploy techniques such as decision tables (Salem, Apostolakis, and Okrent, 1976; Wang and Liu, 1993), mini FTs Powers and Tompkins, 1974; Taylor, 1982, and Piping and Instrumentation Diagrams Taylor, 1982; Xie, Xue, and Xi, 1993. (ii) *Model-based* techniques derive an FT by translating a system model (e.g., using AADL Joshi, Vestal, and Binns, 2007; Mahmud and Mian, 2014, Digraphs De Vries, 1990; Lapp and Powers, 1977, Simulink Xiang, Yanoo, Maeno, et al., 2011, or SysML Mhenni, Nguyen, and Choley, 2014; Xiang, Yanoo, Maeno, et al., 2011) into a FT.

(iii) Due to the increasing availability of inspection and monitoring data, *data-driven* inference methods have emerged. These automatically infer an FT closely matching a given structured dataset, exploiting techniques like Bayesian networks (Linard, Bueno, Bucur, et al., 2020) and genetic algorithms (Linard, Bucur, and Stoelinga, 2019; Jimenez-Roa, Heskes, Tinga, et al., 2023). The resulting FTs closely match the given dataset but only contain events also present in the data—and therefore may lack rare events. Nevertheless, data-driven inference can provide a good basis for fault tree creation. A key drawback of data-driven inference methods is that they still lack sufficient *scalability* for larger systems.

In this work, we tackle the scalability challenge of FT inference by exploiting two concepts commonly used in FTs: symmetries and modules. *Symmetries* between components are commonly present in real-world systems, e.g., due to structural properties or redundancies in safety-critical systems. *Modules* correspond to subsystems and allow to subdivide the inference problem into smaller, possibly independent, problems. Our approach, called ***SymLearn***, automatically identifies symmetries and modules, and exploits them to reduce the solution space.

We implemented the SymLearn method in Python and numerically evaluated it in five case studies, including three truss system models, which are structural systems typically found in civil infrastructures such as roofs, transmission towers, and bridges. We compare SymLearn to the previous FT-MOEA implementation (Jimenez-Roa, Heskes, Tinga, et al., 2023), which was shown to be faster than its predecessor FT-EA (Linard, Bucur, and Stoelinga, 2019). Our experiments show that: (1) SymLearn is orders of magnitude faster than FT-MOEA if modules and symmetries can be exploited; (2) SymLearn is in some cases slower than inference based on Boolean formulas, it yields, however, more compact FTs than Boolean methods.

Contributions. Our main contributions are:

(i) We define modules and symmetries based on the MCSs.
(ii) We present algorithms to automatically identify modules and symmetries from the MCSs.
(iii) We introduce `SymLearn`, an approach to automatically infer FTs from failure datasets by exploiting modules and symmetries.
(iv) We implemented `SymLearn` in Python and numerically evaluated it in several case studies.

The implementation and data are available at zenodo.org/record/5571811.

Outline. Section 3.2 defines modules and symmetries. Section 3.3 details the `SymLearn` approach. In Section 3.4, we evaluate `SymLearn` on truss system models and discuss the results. We conclude in Section 3.5 and present future work.

3.2 Modules and Symmetries

3.2.1 Modules

Instead of directly inferring an FT $\mathcal{F}_{\mathbb{C}D}$ from the MCSs $\mathbb{C}D$, we aim to first partition $\mathbb{C}D$ into multiple parts, infer individual FTs for each of them, and then combine the FTs into the overall FT $\mathcal{F}_{\mathbb{C}D}$.

Definition 3 (MCS partitioning)**.** *Let $M_1, \dots, M_n \subseteq \mathbb{C}$ be a* partitioning of the set $\mathbb{C}$ of MCSs, *i.e., $M_i \cap M_j = \emptyset$ for all $i \neq j$ and $M_1 \cup \dots \cup M_n = \mathbb{C}$. For a partition M_i, we let $\mathsf{BEs}^{M_i} := \bigcup_{C \in M_i} C$ denote the set of* BE *occurring in M_i.* BE *occurring in multiple partitions are called the* shared BE*.*

In the case of a large number of shared BEs, the inferred FTs—which each might be optimal individually—can yield an overall FT which is sub-optimal. For example, gates with (some of the) shared BEs as input might occur in multiple FTs. Thus, the goal is to find a partitioning such that the number of shared BEs is as small as possible. If no BE are shared, the resulting partitioning of BEs forms *independent modules.* In FTs, (independent) modules are independent sub-trees, where only the root node is connected to other parts of the FT (Dutuit and Rauzy, 1996). Modules can therefore be thought of as coherent entities in the context of the overall system, e.g., components. Modularisation is used to simplify the FT analysis.

Definition 4 (Modules)**.** *A partitioning $M_1, \dots M_n$ of the set $\mathbb{C}$ of MCSs is called a* module partitioning *if the corresponding $\mathsf{BEs}^{M_1}, \dots, \mathsf{BEs}^{M_n}$ form a partitioning of* BEs*. A subset $\mathfrak{M}$ of* BEs *is called an* independent module *if it is part of a module partitioning, i.e., all* BE *of $\mathfrak{M}$ are included in MCSs of a single M_i.*

An independent module $\mathfrak{M}$ does not share BE. Thus, the BE in $\mathfrak{M}$ are not connected to other parts of the FT and they belong to an independent sub-tree.

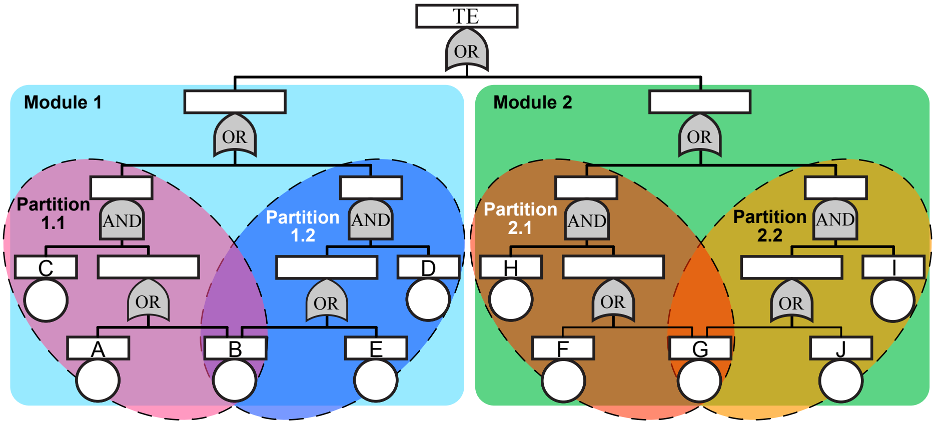


Figure 3.1: FT with independent modules and further partitioning.

Example 1 (Modules)**.** *The partitioning for the FT in Figure 3.1 is given by coloured boxes. The* BEs $\{A, B, C, D, E\}$ *and* $\{F, G, H, I, K\}$ *form independent modules. The corresponding MCSs can be further subdivided. For instance, Partition 1.1 with* $\{\{A, C\}\{B, C\}\}$ *and Partition 1.2 with* $\{\{B, D\}, \{D, E\}\}$ *share* BE B.

3.2.2 Symmetries

Symmetries in an FT describe components, e.g., BE or complete sub-trees, that can be swapped without changing the failure behaviour of the FT. In our setting, symmetries reduce the computational effort for inferring FTs as only one of the sub-trees must be constructed; other sub-tree(s) can be copied from the (original) sub-tree because of the symmetry. We define symmetries on the MCSs. Applying a symmetry on the MCSs yields the same MCSs, i.e., swapping symmetric BE does not change the structure function of the FT.

Definition 5 (Symmetry on MCSs)**.** *A* symmetry *on the set* $\mathbb{C}$ *of all MCSs is a permutation* ψ : BEs $\rightarrow$ BEs *which preserves* $\mathbb{C}$*, i.e.,* $\psi(\mathbb{C}) = \mathbb{C}$ *where* $\psi(\mathbb{C}) := \{\psi(C) \mid C \in \mathbb{C}\}$ *and* $\psi(C) := \{\psi(b) \mid b \in C\}$.

We denote all possible symmetries on $\mathbb{C}$ by $\mathcal{Z}_{\mathbb{C}}$. A *symmetry between sets* $A, B \subseteq$ BEs is a symmetry $\psi \in \mathcal{Z}_{\mathbb{C}}$ with $\psi(A) \subseteq B$ and $\psi(B) \subseteq A$. Note that we define symmetries only on BEs and not on gates. The definition is thus more general and allows symmetries even in cases where sub-trees are not isomorphic.

Lemma 1 (Necessary condition for symmetry)**.** *If* $\psi \in \mathcal{Z}_{\mathbb{C}}$ *is a symmetry on the MCSs* $\mathbb{C}$*, then* $count(b) = count(\psi(b))$ *for all* $b \in$ BEs*, where* $count(b) := |\{C \in \mathbb{C} \mid b \in C\}|$ *denotes the number of occurrences of* b *in* $\mathbb{C}$.

Example 2 (Symmetry)**.** *Consider again the FT* $\mathcal{F}$ *in Figure 3.1. The permutation* $\psi_1 = (AF)(BG)(CH)(DI)(EJ)$ *is a symmetry in* $\mathcal{F}$ *(between the independent*

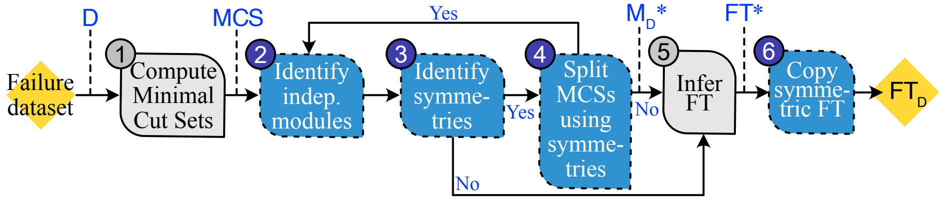


Figure 3.2: `SymLearn` tool chain overview. Blue boxes indicate novel steps.

modules). For example, $\psi_1(\{A, C\}) = \{F, H\} \in \mathbb{C}\mathcal{F}$*. Symmetries within the modules are given by* $\psi_2 = (AE)(CD) \in \mathcal{Z}_{\mathbb{C}\mathcal{F}}$ *and* $\psi_3 = (FJ)(HI) \in \mathcal{Z}_{\mathbb{C}\mathcal{F}}$*.*

3.3 Exploiting Modules and Symmetries in Fault Tree Inference

Our `SymLearn` approach is outlined in Figure 3.2 and consists of 6 steps:

1. computes the set of all *MCSs* $\mathbb{C}\mathbb{D}$ associated with input dataset $\mathbb{D}$.
2. finds a partitioning $M_1, \ldots, M_n$ of $\mathbb{C}\mathbb{D}$ s.t. the corresponding BEs form *independent modules* $\mathfrak{M}_1, \ldots, \mathfrak{M}_n$. In the worst case, no proper partitioning is possible and the independent module consists of all BEs.
3. identifies the *symmetries* $\mathcal{Z}_{\mathbb{C}\mathbb{D}}$ on $\mathbb{C}\mathbb{D}$. If symmetries exist between independent modules, then only one of these modules needs to be considered in the following. Otherwise, SymLearn directly goes to Step 5.
4. tries to further *split* the MCSs M_i of each module $\mathfrak{M}_i$ via a symmetry $\psi \in \mathcal{Z}_{\mathbb{C}\mathbb{D}}$. The split into M_i^1 and M_i^2 should satisfy $\psi(M_i^1) = M_i^2$ and preferably have a small number of shared BE. If a split is found, SymLearn recursively starts again with Step 2 for M_i^1; otherwise it proceeds with Step 5.
5. infers an FT $\mathcal{F}_M$ for each partition M of the MCSs. Several approaches can be used, e.g., `FT-MOEA` (Jimenez-Roa, Heskes, Tinga, et al., 2023) or simplification of Boolean formulas (Lazarova-Molnar, Niloofar, and Barta, 2020).
6. creates for each set of symmetric MCSs M_i^2 a corresponding *symmetric FT* $\mathcal{F}_{M_i^2}$ by copying the "original" FT $\mathcal{F}_{M_i^1}$ and renaming the BEs according to the symmetry ψ. Last, all inferred FTs are joined under an OR-gate.

We provide details on all steps of SymLearn in the following.

Step 1: Compute Minimal Cut Sets.

SymLearn starts by extracting all the MCSs $\mathbb{C}\mathbb{D}$ from the data $\mathbb{D}$. We use the algorithm from (Lazarova-Molnar, Niloofar, and Barta, 2020), but employ an improved computation of the MCSs from the cut sets. Here, we iteratively select a cut set C with minimal cardinality and remove all cut sets that include C. The runtime complexity of the algorithm is quadratic in $\mathbb{D}$, i.e., $\mathcal{O}(\mathbb{D}^2) = \mathcal{O}(2^{2 \cdot |\text{BEs}|})$.

Algorithm 1 Identifying independent modules $\mathfrak{M}_1, \ldots, \mathfrak{M}_n$ from the MCSs $\mathbb{CD}$.

```
Input: MCSs $\mathbb{CD}$.
Output: Partitioning $M_1, \ldots, M_n$ of $\mathbb{CD}$, corresponds to independent modules
    $\mathfrak{M}_1, \ldots, \mathfrak{M}_n$.
  $\mathit{Partitioning} \leftarrow \{\{C\} \mid C \in \mathbb{CD}\}$
  while $\exists M, M' \in \mathit{Partitioning}$ with $M$ and $M'$ sharing BE do
      $\mathit{Partitioning} \leftarrow (\mathit{Partitioning} \setminus \{M, M'\}) \cup \{M \cup M'\}$
  return $\mathit{Partitioning} = \{M_1, \ldots, M_n\}$, modules $\{\mathfrak{M}_1 = \mathsf{BEs}^{M_1}, \ldots, \mathfrak{M}_n = \mathsf{BEs}^{M_n}\}$
```

Step 2: Identify Independent Modules. Our aim is to partition the MCSs $\mathbb{CD}$ s.t. an FT for each partition can be learned individually. This allows for a more efficient inference which could even be performed in parallel.

We start by trying to find independent modules from $\mathbb{CD}$ as described in Algorithm 1. The initial partitioning uses each cut set of $\mathbb{CD}$ as its own partition. If two partitions share BE, they must be merged to satisfy the constraint for independent modules in Def. 4. We iteratively merge partitions until their BEs are disjoint. The BEs then form the independent modules. The following Steps 3-5 are performed for each independent module and corresponding MCSs individually. The FTs created for the modules are combined by an OR-gate in the end.

Example 3 (Identify independent modules)**.** *We use the MCSs* $\mathbb{CD} = \{\{A, C\}, \{B, C\}, \{B, D\}, \{D, E\}, \{F, H\}, \{G, H\}, \{G, I\}, \{I, K\}\}$ *corresponding to Figure 3.1. Applying the algorithm, cut sets* $\{A, C\}$ *and* $\{B, C\}$*, for instance, are merged as they share* BE C*. In the end, the independent modules and partitioning are:*

$$\begin{aligned} \mathfrak{M}_1 &= \{A, B, C, D, E\} & M_1 &: \{\{A, C\}\{B, C\}, \{B, D\}, \{D, E\}\} \\ \mathfrak{M}_2 &= \{F, G, H, I, K\} & M_2 &: \{\{F, H\}\{G, H\}, \{G, I\}, \{I, K\}\} \end{aligned}$$

Extraction of BE. As an additional optimisation, we automatically derive BE which occur in all minimal cut sets of a partition. In order for the partition to cause a system failure, all these BE must fail. Hence, they are excluded from all MCSs and the approach continues on the reduced MCS. In the end, the excluded BE are joined under an AND-gate with the FT resulting from the reduced MCSs.

Step 3: Identify Symmetries. Next, we identify the symmetries $\mathcal{Z}_{\mathbb{CD}}$ from $\mathbb{CD}$ in a fully automated manner. The simplest way is a brute-force approach trying out all possible permutations and checking whether they are valid symmetries according to Def. 5. While this approach is factorial in $|\mathsf{BEs}|$, we obtain good performance in practice by exploiting two optimisations.

Algorithm 2 Splitting of MCS M_i into two symmetric parts M_i^1 and M_i^2.

```
Input: MCS $M_i$, symmetry $\psi \in \mathcal{Z}_{\mathbb{CD}}$
Output: Symmetric MCSs $M_i^1, M_i^2$ with corresponding contained BE
  $\mathsf{BEs}^{M_i^1}, \mathsf{BEs}^{M_i^2}$
  $M_i^1 \leftarrow \emptyset$, $M_i^2 \leftarrow \emptyset$, $\mathsf{BEs}_1 \leftarrow \emptyset$, $\mathsf{BEs}_2 \leftarrow \emptyset$
  $Q \leftarrow \mathbb{CD}$
  while $C \in Q$ do
      if $C = \psi(C)$ then return $M_i, \emptyset, \mathsf{BEs}^{M_i}, \emptyset$
      $Q \leftarrow Q \setminus \{C, \psi(C)\}$
      if $|C \cap \mathsf{BEs}_1| \geq |C \cap \mathsf{BEs}_2|$ then
          $M_i^1 \leftarrow M_i^1 \cup \{C\}$, $M_i^2 \leftarrow M_i^2 \cup \{\psi(C)\}$, $\mathsf{BEs}_1 \leftarrow \mathsf{BEs}_1 \cup C$, $\mathsf{BEs}_2 \leftarrow \mathsf{BEs}_2 \cup \psi(C)$
      else
          $M_i^1 \leftarrow M_i^1 \cup \{\psi(C)\}$, $M_i^2 \leftarrow M_i^2 \cup \{C\}$, $\mathsf{BEs}_1 \leftarrow \mathsf{BEs}_1 \cup \psi(C)$, $\mathsf{BEs}_2 \leftarrow \mathsf{BEs}_2 \cup C$
  return $M_i^1, M_i^2, \mathsf{BEs}_1, \mathsf{BEs}_2$
```

Symmetries between independent modules. The most efficient approach is to exploit the independent modules from the previous step. Symmetries between two independent modules $\mathfrak{M}, \mathfrak{M}'$ can be quickly found by restricting the permutations to only the ones matching each BE in $\mathfrak{M}$ to one in $\mathfrak{M}'$.

Fast exclusion of non-symmetric BEs. If only one independent module was found in Step 2, then the symmetries must be computed by an exhaustive search. However, we can exclude infeasible permutation candidates early on by using Lemma 1. Two BE with different numbers of occurrences in $\mathbb{CD}$ cannot be symmetric and thus, all permutations containing such mappings are excluded.

Example 4 (Identify symmetries)**.** *Continuing Example 3, we find the symmetry $\psi_1 = (AF)(BG)(CH)(DI)(EK)$ between independent modules $\mathfrak{M}_1$ and $\mathfrak{M}_2$. As a result, the symmetric set of MCSs M_2 will not be considered in the remainder. We continue by searching for symmetries within $\mathfrak{M}_1$ according to M_1. Candidate permutations such as (AC) are quickly excluded, because $count(A) = 1 \neq 2 = count(C)$. In the end, symmetry $\psi_2 = (AE)(CD)$ is found.*

Step 4: Split MCSs using Symmetries. A symmetry ψ found in the previous step can be used to split the MCSs M_i. We restrict ourselves to splits into two parts here, but more parts work in the same manner. A successful split creates two symmetric subsets M_i^1 and M_i^2 of M_i with $\psi(M_i^1) = M_i^2$.

Algorithm 2 describes the split of the MCSs M_i according to a symmetry $\psi \in \mathcal{Z}_{\mathbb{CD}}$. Initially, the queue Q contains all MCSs from $\mathbb{CD}$. For each MCS C we compute the symmetric MCS $\psi(C)$. If C is symmetric to itself ($C = \psi(C)$), a split would add the same MCS to both parts. As this would only increase the size of the resulting FTs, we do not proceed further. If both MCSs are distinct, we add C to the set of MCSs with which it shares the most BE. For example, we add C to M_i^1 if $|C \cap \mathsf{BEs}_1| \geq |C \cap \mathsf{BEs}_2|$. By this choice, we ensure that adding C to M_i^1 does not add too many new BE to BEs_1 and we keep the number of shared BE between BEs_1 and BEs_2 small.

Note that the split can still yield two parts which share a significant amount of BE. Composing the two resulting FTs can therefore yield an FT which is larger than the single FT inferred without the split. However, the composed FT will capture the symmetric structure present in the given MCSs.

Example 5 (Split the Minimal Cut Sets)**.** *We continue with symmetry* $\psi_2 = (AE)(CD)$ *and MCSs* $M_1 = \{\{A, C\}, \{B, C\}, \{B, D\}, \{D, E\}\}$ *from Example 4. We start the algorithm with MCS* $\{A, C\}$*. The symmetric MCS is* $\psi(\{A, C\}) = \{D, E\}$*. The first split yields* $M_1^1 = \{\{A, C\}\}$ *and* $M_1^2 = \{\{D, E\}\}$*. The next MCS* $\{B, C\}$ *is added to* M_1^1 *because they both share BE* C*. The final split is:*

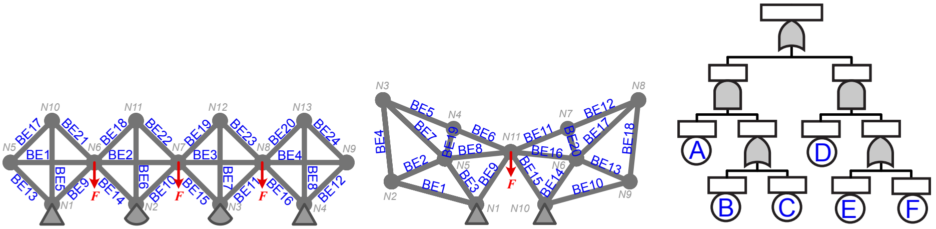
$$M_1^1 = \{\{A, C\}, \{B, C\}\} \qquad \mathit{BEs}_1 = \{A, B, C\},$$
$$M_1^2 = \{\{D, E\}, \{B, D\}\} \qquad \mathit{BEs}_2 = \{B, D, E\}.$$

The split corresponds to the purple and dark blue sub-trees in Figure 3.1.

Step 5: Infer Fault Tree. If no further partitioning of the MCSs M_i w.r.t. Steps 2-4 is possible, we use existing techniques to infer an FT from the (reduced) MCSs. SymLearn is modular and supports the use of any learning approach in this step, for example, based on genetic algorithms (Linard, Bucur, and Stoelinga, 2019) or Boolean logic (Lazarova-Molnar, Niloofar, and Barta, 2020). In our setting, we use the multi-objective evolutionary algorithm `FT-MOEA` (Jimenez-Roa, Heskes, Tinga, et al., 2023).

`FT-MOEA` starts in the first generation by default with two *parent FTs*: one FT consists of an AND-gate connected to all BEs, and the other one uses an OR-gate. In each generation, several *genetic operators* are applied which randomly modify the FT structure. Each FT is evaluated according to three metrics given in Sect 3.2: size of the FT $|\mathcal{F}|$, error based on the failure dataset (ε_d), and error based on the set of MCSs (ε_c). The aim is to minimise the multi-objective function $(|\mathcal{F}|, \varepsilon_d, \varepsilon_c)$ by applying the *Elitist Non-dominated Sorting Genetic Algorithm* (NSGA-II) (Deb, Pratap, Agarwal, et al., 2002) and obtain the Pareto sets. Only the best candidates according to the metrics are then passed to the next generation. The algorithm stops if no improvement was made in a given number of generations and returns the FTs ordered according to the multi-objective function.

Example 6 (`FT-MOEA`)**.** *Given the MCS* $\{\{A, C\}, \{B, C\}\}$*, we use* `FT-MOEA` *to infer a FT. The resulting FT is the sub-tree indicated by purple colour in Figure 3.1.*



(a) Case TS2.

(b) Case TS3.

(c) Case SC.

Figure 3.3: Visualisation of case studies TS2, TS3 and SC.

Step 6: Copy Symmetric FTs. After obtaining an FT $\mathcal{F}_M$ for MCSs M, we obtain the symmetric FT $\mathcal{F}_{M'}$ for the symmetric MCSs $M' = \psi(M)$ by copying $\mathcal{F}_M$ and replacing each BE b with its symmetric BE $\psi(b)$. The original and the symmetric FT are then joined under an OR-gate.

Example 7 (Copy symmetric FT)**.** *We continue with Example 6. Copying the purple sub-tree in Figure 3.1 and applying symmetry $\psi_2 = (AE)(CD)$ yields the symmetric (dark blue) FT. Joining both FTs with an* OR*-gate yields Module 1.*

3.4 Experimental Evaluation

We implemented the SymLearn methodology in a Python toolchain, available at zenodo.org/record/5571811, and evaluate our approach on five case studies, see Table 3.1: *Cases SC* and *SS* are two small systems, depicted in Figure 3.3(c) and running example of Figure 3.1, respectively. Further, we consider three *truss system models*.

Truss System Cases. Truss systems, commonly used in civil infrastructure like transmission towers, and bridges (see Figure 3.4(a)), are composed of elements connected by nodes, forming rigid bodies under tensile stress.

Table 3.1: Overview of case studies.

Case	#BEs	$\lvert\mathbb{D}\rvert$	$\lvert\mathbb{CD}\rvert$
SC	6	64	4
SS	10	1,024	8
TS1	10	1,024	16
TS2	24	16,777,216	26
TS3	20	1,048,576	18

These systems exhibit a high degree of symmetry and a modular structure. Additionally, they allow for failure datasets to be generated through structural analysis, similar to Byun and Song, 2020. Thus, truss systems provide a highly suitable model for evaluating SymLearn in realistic scenarios.

We use three truss system variants: Cases TS1 (Figure 3.4(a)) and TS2 (Figure 3.3(a)) are typical configurations in bridges, while Case TS3 (Figure 3.3(b)) is found in roofs. Note that Case TS1

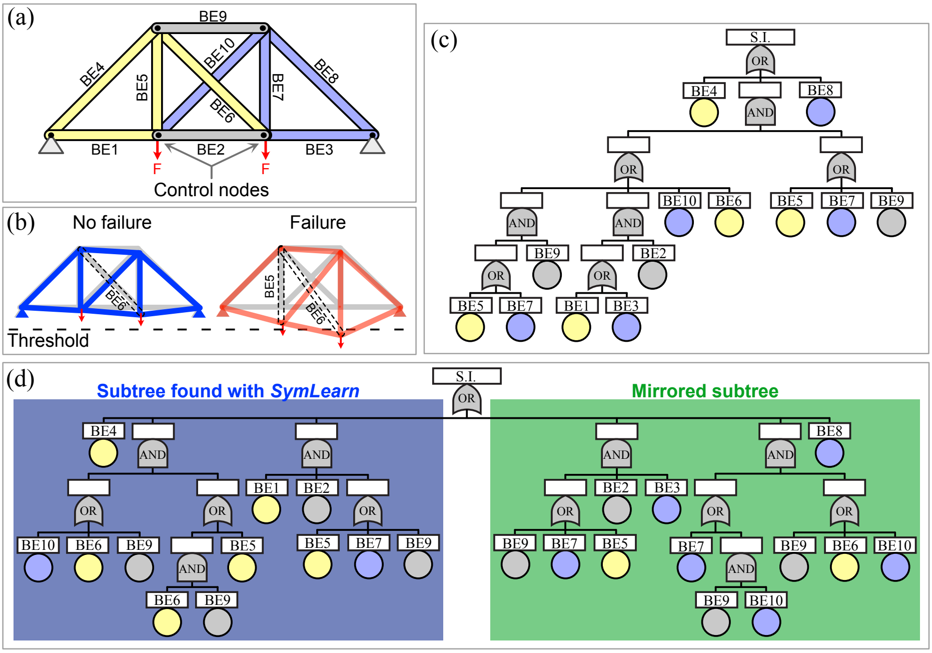


Figure 3.4: Example case TS1 modelling a symmetric truss bridge system. (a) Model. (b) Depiction of failure/no-failure states. (c) FT inferred by FT-MOEA. (d) FT inferred by SymLearn. Top corresponds to the truss system instability.

contains no independent modules, whereas TS2 and TS3 contain four and two modules, respectively.

Generation of Failure Dataset. Based on case TS1 (Figure 3.4) we explain how we use numerical truss system models to generate complete failure datasets. TS1 consists of 10 elements (interpreted as BEs), and two symmetric loads applied on the control nodes. We model damage by reducing close to zero the cross-sectional area of at least one element in the truss system model, and by determining the displacements and stresses in the components due to the applied loads at the nodes of the numerical model. We generate a synthetic failure dataset $\mathbb{D}$ by randomly drawing 10^6 data points for the status of elements in the truss model via Monte Carlo simulation, and evaluating *structural instability* (S.I.) based on the displacement of control nodes.

Experimental Setup.

We compare the SymLearn tool with 3 different back-ends in Step 5, to infer the FT from data.

- FT-MOEA is used in 4 different settings: (1) *All* is the default setting using both modules and symmetries; (2) *No Sym* is *All* but without symmetries; (3) *No*

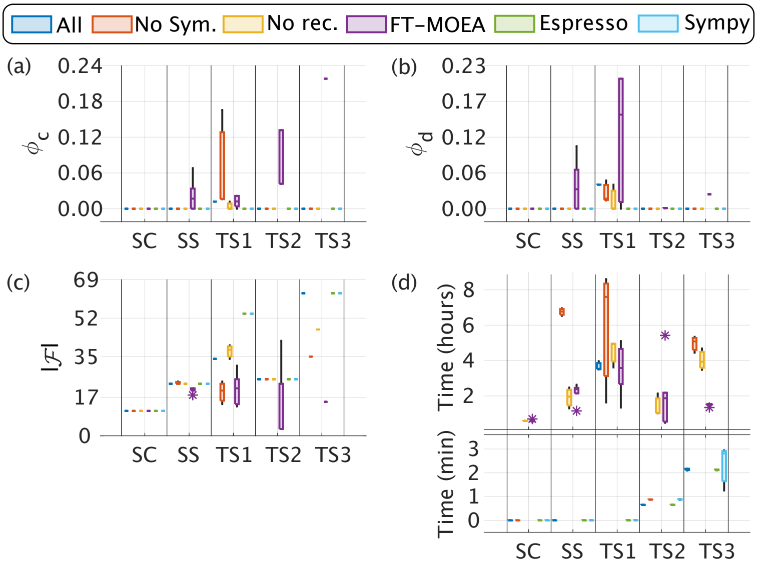


Figure 3.5: Results for the case studies and different metrics: (a) error ϕ_c based on the MCSs, (b) error ϕ_d based on dataset, (c) FT size $|\mathcal{F}|$, and (d) runtime.

rec. is *All* but without recursive calls for further sub-division; (4) `FT-MOEA` is the original implementation (Jimenez-Roa, Heskes, Tinga, et al., 2023) without modules and symmetries.

- *Espresso* translates a set of MCSs $\mathbb{CD}$ into a Boolean formula $\bigvee_{C\in\mathbb{CD}} \bigwedge_{b\in C} b$ and simplifies it via the *ESPRESSO* algorithm (Brayton, Hachtel, McMullen, et al., 1984) available in PYEDA[*]. The resulting formula is then translated into an FT.
- *Sympy* is similar to *Espresso* but uses the SYMPY library[†] for simplification.

We ran all case studies three times on a CPU with 2.3 GHz and 8 GB of RAM.

Results. We compare the FTs for case TS1 inferred via `FT-MOEA` (Figure 3.4(c)) and via SymLearn in configuration *All* (Figure 3.4(d)). Colours depict the connections of the BEs to the components in Figure 3.4(a). SymLearn identified the symmetry (between yellow and blue BE) and was able to infer the left sub-tree using `FT-MOEA` while the right sub-tree was obtained by simple mirroring.

The box charts in Figure 3.5 compare the different configurations in all five cases w.r.t. the three metrics in Section 3.2: the size $|\mathcal{F}|$ of the FT, the error ε_d based on the failure dataset, and the error ε_c based on the MCSs. From Figure 3.5(a) and 3.5(b), we see that the SymLearn configurations based on Boolean functions as a back-end (i.e., *Espresso* and *Sympy*) always yield an FT that exactly matches the

[*] `https://pyeda.readthedocs.io/en/latest/2llm.html`

[†] `https://docs.sympy.org/latest/modules/logic.html`

input, i.e., $\varepsilon_c = \varepsilon_d = 0$. This is expected since the Boolean logic formula perfectly encodes all the MCSs. In contrast, the other configurations using FT-MOEA did not always yield a completely accurate FT (i.e., $\varepsilon_c, \varepsilon_d > 0.0$), for example, case TS1. The error stems from the multi-objective optimisation which also aims to provide a small FT and the evolutionary algorithm which can fall into local optima. However, for the cases TS2 and TS3 (with independent modules), all configurations of SymLearn (*All*, *No Sym*, *No rec.*) outperformed FT-MOEA by returning an FT that accurately reflects the input ($\varepsilon_c = \varepsilon_d = 0.0$). This shows the clear benefit of subdividing the problem using independent modules.

Figure 3.5(c) shows the advantage of using FT-MOEA as a back-end compared to Boolean logic, since the sizes of the returned FTs can be considerably smaller. The FTs inferred using Espresso or Sympy can be twice as large as the ones resulting from FT-MOEA. The reason is that for the Boolean logic formulas, no simplifications were performed by the libraries and the resulting FTs are therefore exactly encoding all the MCSs. Notice that the original FT-MOEA yields smaller or equal FT sizes than any of the configurations of SymLearn. This smaller size can however also come at the cost of losing accuracy, as demonstrated by case TS2. The larger FTs in SymLearn mostly stem from the composition of partitions where shared BE occur in both sub-trees, see for example Figure 3.4(c) and 3.4(d). While explicitly capturing the symmetries can therefore increase the size of the resulting FT, it also provides more insights into the system.

Figure 3.5(d) shows that SymLearn (*All*) runs significantly faster than FT-MOEA alone. If independent modules are present (cases TS2, TS3, SC and SS), SymLearn yields an FT within at most 2 min while FT-MOEA requires at least 1 h. The benefit of exploiting symmetries and modules can also be seen when comparing configuration *All* to *No Sym* and *No. rec.* which both run longer. Note that for SymLearn nearly all computation time is spent in the FT-MOEA back-end (Step 5). Computing the modules and symmetries (Steps 2-4) took 50 ms at most whereas the computation of the MCSs (Step 1) took 43 s at most (for case TS2). Configurations based on Boolean functions always yield a result within minutes, but yield significantly larger FTs.

3.5 Conclusions

We presented SymLearn, a data-driven algorithm that infers a Fault Tree (FT) model from given failure data in a fully automatic way by identifying and exploiting modules and symmetries. Our evaluation based on truss system models shows that SymLearn is significantly faster than only using evolutionary algorithms when modules and symmetries can be exploited.

In the future, we aim to further improve the scalability by *optimising the inference process*. First, the current partitioning of the Minimal Cut Sets requires the top gate to be an OR-gate. We aim to support the AND-gate as well. In addition, the inference back-end can be improved by either optimising FT-MOEA or developing new inference approaches.

We also plan to *relax restrictions on the input data.* In the current approach, the resulting FTs are only as good as the given input data, which may be incomplete, e.g., due to rare events not present in the data. Moreover, the input may not completely represent the reality due to noise in the data. Hence, we aim to extend our approach to account for missing information and noise.

3.6 References

Brayton, R. K., G. D. Hachtel, C. T. McMullen, and A. L. Sangiovanni-Vincentelli (1984). *Logic Minimization Algorithms for VLSI Synthesis.* Vol. 2. The Kluwer International Series in Engineering and Computer Science. Springer. DOI: 10.1007/978-1-4613-2821-6.

Byun, J.-E. and J. Song (2020). "Efficient probabilistic multi-objective optimization of complex systems using matrix-based Bayesian network". In: *Reliability Engineering & System Safety* 200, p. 106899. DOI: 10.1016/j.ress.2020.106899.

Carpignano, A. and A. Poucet (1994). "Computer assisted fault tree construction: a review of methods and concerns". In: *Reliability Engineering & System Safety* 44.3, pp. 265–278. DOI: 10.1016/0951-8320(94)90018-3.

De Vries, R. C. (1990). "An automated methodology for generating a fault tree". In: *IEEE Transactions on Reliability* 39.1, pp. 76–86. DOI: 10.1109/24.52615.

Deb, K., A. Pratap, S. Agarwal, and T. Meyarivan (2002). "A fast and elitist multiobjective genetic algorithm: NSGA-II". In: *IEEE Transactions on Evolutionary Computation* 6 (2), pp. 182–197. DOI: 10.1109/4235.996017.

Dutuit, Y. and A. Rauzy (1996). "A linear-time algorithm to find modules of fault trees". In: *IEEE Transactions on Reliability* 45.3, pp. 422–425. DOI: 10.1109/24.537011.

Hunt, A., B. E. Kelly, J. S. Mullhi, F. P. Lees, and A. G. Rushton (1993). "The propagation of faults in process plants: 6, Overview of, and modelling for, fault tree synthesis". In: *Reliability Engineering & System Safety* 39 (2), pp. 173–194. DOI: 10.1016/0951-8320(93)90041-V.

Jimenez-Roa, L. A., T. Heskes, T. Tinga, and M. Stoelinga (2023). "Automatic Inference of Fault Tree Models Via Multi-Objective Evolutionary Algorithms". In: *IEEE Transactions on Dependable and Secure Computing* 20.4, pp. 3317–3327. DOI: 10.1109/TDSC.2022.3203805.

Joshi, A., S. Vestal, and P. Binns (2007). "Automatic generation of static fault trees from AADL models". In: URL: https://hdl.handle.net/11299/217313.

Lapp, S. A. and G. J. Powers (1977). "Computer-aided synthesis of fault-trees". In: *IEEE Transactions on Reliability* 26 (1). DOI: 10.1109/TR.1977.5215060.

Lazarova-Molnar, S., P. Niloofar, and G. K. Barta (2020). "Data-Driven Fault Tree Modeling for Reliability Assessment of Cyber-Physical Systems". In: *2020 Winter Simulation Conference (WSC)*, pp. 2719–2730. DOI: 10.1109/WSC48552.2020.9383882.

Linard, A., M. Bueno, D. Bucur, and M. Stoelinga (2020). "Induction of fault trees through Bayesian networks". In: *Proceedings of the 29th European Safety and Reliability Conference, ESREL 2019*, pp. 910–917. DOI: 10.3850/978-981-11-2724-3_0596-cd.

Linard, A., D. Bucur, and M. Stoelinga (2019). "Fault Trees from Data: Efficient Learning with an Evolutionary Algorithm". In: *International Symposium on Dependable Software Engineering: Theories, Tools, and Applications.* Vol. 11951 LNCS, pp. 19–37. DOI: 10.1007/978-3-030-35540-1_2.

Madden, M. G. and P. J. Nolan (1994). “Generation of fault trees from simulated incipient fault case data”. In: *WIT Transactions on Information and Communication Technologies* 6. DOI: 10.2495/AI940611.

Mahmud, N. and Z. Mian (2014). “Automatic generation of Temporal Fault Trees from AADL models”. In: *Safety, Reliability and Risk Analysis: Beyond the Horizon - Proceedings of the European Safety and Reliability Conference, ESREL 2013*, pp. 2741–2749.

Mhenni, F., N. Nguyen, and J.-Y. Choley (2014). “Automatic fault tree generation from SysML system models”. In: *2014 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*. IEEE, pp. 715–720. DOI: 10.1109/AIM.2014.6878163.

NASA (2002). *Fault Tree Handbook with Aerospace Applications*. Handbook. U.S. National Aeronautics and Space Administration.

Powers, G. J. and F. C. Tompkins (1974). “Fault tree synthesis for chemical processes”. In: *AIChE Journal* 20.2, pp. 376–387. DOI: 10.1002/aic.690200226.

Ruijters, E. and M. Stoelinga (2015). “Fault tree analysis: A survey of the state-of-the-art in modeling, analysis and tools”. In: *Computer Science Review* 15-16, pp. 29–62. ISSN: 1574-0137. DOI: https://doi.org/10.1016/j.cosrev.2015.03.001.

Salem, S. L., G. E. Apostolakis, and D. Okrent (Nov. 1976). “Computer-oriented approach to fault-tree construction”. In: DOI: 10.2172/7132148.

Taylor, J. R. (1982). “An algorithm for fault-tree construction”. In: *IEEE Transactions on Reliability* 31 (2), pp. 137–146. DOI: 10.1109/TR.1982.5221276.

Wang, J. and T. Liu (1993). “A component behavioural model for automatic fault tree construction”. In: *Reliability Engineering & System Safety* 42.1, pp. 87–100. DOI: 10.1016/0951-8320(93)90058-7.

Xiang, J., K. Yanoo, Y. Maeno, and K. Tadano (2011). “Automatic Synthesis of Static Fault Trees from System Models”. In: *2011 Fifth International Conference on Secure Software Integration and Reliability Improvement*, pp. 127–136. DOI: 10.1109/SSIRI.2011.32.

Xie, G., D. Xue, and S. Xi (1993). “TREE-EXPERT: a tree-based expert system for fault tree construction”. In: *Reliability Engineering & System Safety* 40.3, pp. 295–309. DOI: 10.1016/0951-8320(93)90066-8.

Chapter 4

Fault Tree inference using Multi-Objective Evolutionary Algorithms and Confusion Matrix-based metrics

Paper published at **L. A. Jimenez-Roa**, N. Rusnac, M. Volk, M. Stoelinga, *"Fault Tree inference using Multi-Objective Evolutionary Algorithms and Confusion Matrix-based metrics",* in Formal Methods for Industrial Critical Systems (FMICS), 2024. Springer's Lecture Notes in Computer Science (LNCS). doi: 10.1007/978-3-031-68150-9_5.

Abstract

In the domain of reliability engineering and risk assessment, the development of Fault Tree (FT) models is pivotal for decision-making in complex systems. Traditional FT model development, relying on manual efforts and expert collaboration, is both time-consuming and error-prone. The era of Industry 4.0 introduces capabilities for automatically deriving FTs from inspection and monitoring data.

This chapter presents FT-MOEA-CM, an extension of the FT-MOEA algorithm for inferring FT models from failure data using multi-objective optimisation. FT-MOEA-CM enhances its predecessor by integrating confusion matrix-derived metrics and incorporating parallelisation and caching mechanisms. Our evaluation on six FTs from diverse application areas showcases that FT-MOEA-CM exhibits (1) enhanced robustness, (2) faster convergence and (3) better scalability than FT-MOEA, suggesting its potential in efficiently inferring larger FT models.

4.1 Introduction

Fault Tree Analysis (FTA) (NASA, 2002; Ruijters and Stoelinga, 2015) is a critical tool in reliability engineering and risk analysis, utilised extensively in industry for its ability to model complex systems and assess failure probabilities. Despite its creation in the 1960s and widespread application across a large range of industrial domains, the construction of fault trees remains a significant effort. Traditional methods involve manual, expert-driven development, a process that is not only laborious but also prone to errors and inconsistencies, especially in complex systems.

A promising approach is the automatic inference of Fault Tree (FT) models from failure data. Given a set of data points representing the status of components (operational/failed) and the corresponding overall system status, the aim is to automatically infer a compact FT model capturing the failure behaviour present in the dataset. While first inference approaches date back to the 1970s (Madden and Nolan, 1994), the recent surge of data collection allows new approaches for FT inference (Nauta, Bucur, and Stoelinga, 2018; Waghen and Ouali, 2019; Lazarova-Molnar, Niloofar, and Barta, 2020; Jimenez-Roa, Volk, and Stoelinga, 2022). A recent algorithm for creating FT models from failure datasets is `FT-MOEA` (Jimenez-Roa, Heskes, Tinga, et al., 2023), which employs both the *Elitist Non-Dominated Sorting Genetic Algorithm* (`NSGA-II`) (Deb, Pratap, Agarwal, et al., 2002) and the *Crowding-Distance* (Martí, Segredo, Śnchez-Pi, et al., 2017). The former leverages multi-objective optimisation and Pareto front concepts to infer FT models, while the latter serves as a diversity criterion, prioritising diverse solutions over overcrowded ones.

However, `FT-MOEA` encounters challenges related to *robustness*, *scalability*, and *convergence speed*. Robustness pertains to its ability to consistently yield the same results. Scalability refers to the capacity to handle larger FTs, characterised by a greater number of *basic events* and Minimal Cut Sets (MCSs). Convergence speed concerns efficiency in completing the task.

These problems may stem from the limited features considered in `FT-MOEA`'s optimisation process. Specifically, `FT-MOEA` incorporates only three features: error metrics based on accuracy and MCS, and the FT size, with the first two being correlated. Furthermore, computing MCSs for larger FTs is notably computationally expensive, which aggravates scalability and convergence speed concerns.

To address these challenges, this chapter explores alternative features to guide the multi-objective optimisation's convergence process, resulting in the development of the `FT-MOEA-CM` algorithm. This algorithm leverages 16 metrics derived from the well-established Confusion Matrix (CM) and eliminates the need for the computationally expensive MCS calculations.

Our methodology is structured into two phases: In a first phase, we perform *feature assessment* and identify the most informative and effective features for guiding the

Table 4.1: List of 17 metrics evaluated to guide the inference process of FTs.

Metric name	Range	Comment
Fault Tree Size	$[2, \infty)$	Number of Fault Tree nodes $\lvert\mathcal{F}_{\mathbb{D}}\rvert := \lvert V\rvert$.
Precision, Specificity, Sensitivity, Negative predictive value, Accuracy, Threat score, Balanced accuracy, Negative likelihood ratio, Positive likelihood ratio, Diagnostic odds ratio, F1 Score, Fowlkes-Mallows Index	$[0, 1]$	Metrics that range between 0 and 1 (or 0 to infinity) are normalised to $[0, 1]$ with 0 being the optimum value.
Matthews correlation coefficient, Informedness, Markedness, Kappa statistic	$[0, 2]$	Metrics that range between -1 and 1 are normalised to $[0, 2]$ with 0 being the optimum value.

FT inference process. To achieve this, we conduct a Principal Component Analysis (PCA), a technique utilised for dimensionality reduction and feature selection. In a second phase, we perform an extensive *evaluation* of the new approach on six FTs from diverse application areas, and compare with `FT-MOEA`. In particular, we investigate how the inclusion of additional information in `FT-MOEA-CM` influences the robustness, scalability, and convergence speed of the FT inference process.

Contributions. The primary contributions of this work are as follows:

(i) Introduction of the `FT-MOEA-CM` algorithm, employing confusion matrix-based metrics for the automatic inference of FTs, which enhances robustness, scalability, and convergence speed over its predecessor `FT-MOEA`.

(ii) Improved performance through the integration of features like caching and parallelisation, particularly beneficial for larger FT structures.

(iii) `FT-MOEA-CM` is available at `https://gitlab.utwente.nl/fmt/fault-trees/ft-moea`

Outline. Section 4.2 introduces FTs and formally defines their inference. Section 4.3 details the `FT-MOEA-CM` methodology. Section 4.4 describes our experimental setup, and Section 4.5 presents our results from evaluating `FT-MOEA-CM` on six case studies. We conclude in Section 4.7 and present future work.

4.2 Confusion Matrix-based metrics

Our inference approach is guided by metrics based on the Confusion Matrix. The *Confusion Matrix (CM)* is a performance evaluation tool commonly used in machine learning classification tasks (Sokolova and Lapalme, 2009). In binary classification, a 2×2 CM categorises predictions into four outcomes: True Positives (`TP`), True Negatives (`TN`), False Positives (`FP`), and False Negatives (`FN`). In our setting `TP` and `TN` correspond to both the FT and the data giving the same result, i.e.,

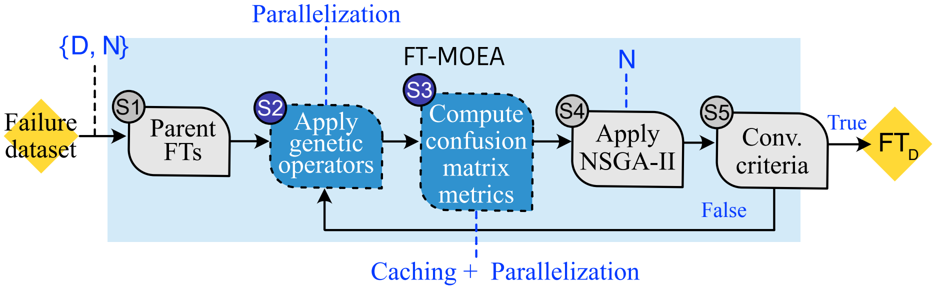


Figure 4.1: FT-MOEA-CM methodology. Blue boxes indicate novel steps.

$f^{\mathcal{F}_{\mathbb{D}}}(\boldsymbol{b}_k) = f^{\mathbb{D}}(\boldsymbol{b}_k)$. FP and FN indicate that the outcome of the FT differs from the data, i.e., $f^{\mathcal{F}_{\mathbb{D}}}(\boldsymbol{b}_k) \neq f^{\mathbb{D}}(\boldsymbol{b}_k)$.

To assess the FT's performance relative to input data $\mathbb{D}$, we utilise 17 metrics outlined in Table 4.1. The first metric evaluates the FT's size via the number of nodes $F_{\mathbb{D}}$, the remaining 16 metrics are derived from the CM.

We normalise all CM-based metrics to the interval $[0, 1]$ such that 0 represents optimal values. Metrics ranging from -1 to 1, such as the *Matthews Correlation Coefficient* are scaled to the interval $[0, 2]$ to enhance the interpretability of simulation outcomes. Further details on the CM and associated metrics can be found in (Božić, Runje, Lisjak, et al., 2023).

4.3 FT-MOEA-CM's methodology

Figure 4.1 illustrates FT-MOEA-CM's FT inference process, which utilises multi-objective evolutionary algorithms and metrics derived from the confusion matrix.

The approach is based on the standard steps of genetic algorithms: each generation of FTs is mutated based on operators such as adding or removing gates. The resulting FTs are evaluated based on metrics and the best FTs are then used in the next generation. As FT-MOEA-CM is based on FT-MOEA, we direct the reader to Jimenez-Roa, Heskes, Tinga, et al., 2023 for detailed methodology information. Below, we outline each main step of the process.

(1) - **Input:** The input includes the failure dataset $\mathbb{D}$ (Section 4.2) as well as FT-MOEA-CM's parameters, such as the maximum population size N.
- **Process:** The initial population consists of two *parent* FTs, one a single AND-gate and the other an OR-gate, each connecting to all BEs in $\mathbb{D}$.
- **Output:** The parent FTs constitute the FT population.

(2) - **Input:** The existing FT population.
- **Process:** Seven genetic operators (e.g., adding or removing gates, crossover of sub-trees) are applied to alter the structure for each FT in the population (see Jimenez-Roa, Heskes, Tinga, et al., 2023 for details). We improve

upon Jimenez-Roa, Heskes, Tinga, et al., 2023 by introducing *parallelisation*, enabling the use of multiple system cores for FT generation.

- **Output:** An expanded FT population featuring new FTs.

(3) - **Input:** The expanded FT population.

- **Process:** Each FT in the population is processed to calculate the 17 metrics listed in Table 4.1. *Caching* is used to avoid recalculating metrics for previously evaluated FTs, thus enhancing efficiency. *Parallelisation* is implemented to further improve this process.
- **Output:** The enlarged FT population with corresponding metric values.

(4) - **Input:** The enlarged FT population with corresponding metric values.

- **Process:** The `NSGA-II` algorithm (Deb, Pratap, Agarwal, et al., 2002) and `Crowding-Distance` (Martí, Segredo, Śnchez-Pi, et al., 2017) are utilised for multi-objective optimisation to construct Pareto fronts of non-dominated FTs based on the metrics.
- **Output:** The top N FTs—where N is a user-defined parameter—are selected for the next generation.

(5) - **Input:** The top N FTs.

- **Process:** Evaluate convergence criteria: (i) the maximum number of generations is reached, or (ii) the best FT candidate remains unchanged for a specified number of generations. If neither condition is fulfilled, the top N FTs are used as input for Step 2, and Steps 2 to 5 are repeated until one of the convergence criteria is met.
- **Output:** The inferred FT, $\mathcal{F}_{\mathbb{D}}$, identified as the best FT candidate in the first Pareto front.

4.4 Experimental Evaluation

Case studies. We evaluate our approach on six FTsstemming from various application areas. The *Data-driven Fault Tree (ddFT)* (Lazarova-Molnar, Niloofar, and Barta, 2020) was obtained from time series data. The *Mono-propellant Propulsion System (MPPS)* (NASA, 2002) is used for a small space flight vehicle. The *COVID-19* FT (Bakeli, Hafidi, et al., 2020) is used in infection risk management. The *Truss System (TS1)* (Jimenez-Roa, Volk, and Stoelinga, 2022) models a symmetric truss bridge system. The two FTs*GPT12BE* and *GPT15BE* were generated with GPT-4 (OpenAI, Achiam, Adler, et al., 2024), representing larger FTsdesigned to test scalability. The prompts used for generation included examples of existing FTs, the number of nodes, and the number of gates. Table 4.2 outlines for each case study the number of Basic Events ($\lvert$BEs$\rvert$),

Table 4.2: Overview of case studies.

Case	$\lvert$BEs$\rvert$	$\lvert\mathcal{F}\rvert$	$\lvert D\rvert$	$\lvert\mathbb{C}D\rvert$	All FTs
ddFT	8	19	256	6	83,600
MPPS	8	14	256	7	73,200
COVID-19	9	13	512	6	60,400
TS1	10	21	1,024	16	127,200
GPT12BE	12	25	4,096	13	139,200
GPT15BE	15	27	32,768	10	108,000

FT Size ($|\mathcal{F}|$), failure dataset size ($|\mathbb{D}|$), total number of MCSs ($|\mathbb{CD}|$), and the number of all FTsacross generations. The latter is further discussed in Section 4.5.1.

Implementation. The implementation of `FT-MOEA-CM`, available online[*], is complemented by a dedicated database server designed for storing and processing data produced by `FT-MOEA-CM`. This server, developed in GO, employs a MySQL database and can be accessed online[†].

Generation of Failure Dataset. Access to real-life failure data is typically very limited. Instead, we evaluate our approach on synthetic failure datasets which are generated from realistic reliability models. We consider existing FTs from the literature as ground truth, see Table 4.2. For each FT, we generate a synthetic failure dataset $\mathbb{D}$ by evaluating all the unique combinations of BEs in the respective FT, ensuring the completeness of the failure dataset. Our dataset allows us to compare the FT inferred from the dataset with the ground-truth FT, and thereby evaluate the quality of the inferred FT.

Experimental Setup. Our case studies were executed five times on an E5-2683V4 CPU at 2.10 GHz, with 16 cores supporting 2 threads each on the EEMCS-HPC Cluster of the University of Twente. The evaluation comprises two primary sections: the first, elaborated in Section 4.5.1, concentrates on *feature assessment* through Principal Component Analysis to discern the most informative features from the CM for inferring FTs. Section 4.5.2 compares the efficacy of the CM-based metrics with the original `FT-MOEA` implementation, involving two configurations: `FT-MOEA-CM-All` includes all 17 features from Table 4.1, and `FT-MOEA-CM-Best` employs only the top 7 features identified in Section 4.5.1. The evaluation addresses robustness, convergence speed, and scalability. Lastly, we also evaluate `FT-MOEA-CM`'s features of parallelisation and caching in Section 4.5.3.

4.5 Results

4.5.1 Feature Assessment

For Step 3 in the FT inference (cf. Figure 4.1), we need to identify the most informative metrics listed in Table 4.1. We conduct this by performing *feature assessment* by evaluating the importance of different variables in a dataset. Here, the *features* are the *metrics* computed from the CM.

We use Principal Component Analysis (PCA), a multivariate statistical technique that extracts information in the form of *principal components* (Abdi and Williams, 2010), which are *orthogonal* vectors that maximise variance, capturing the most significant features. This technique is commonly used for dimensionality reduction

[*] https://gitlab.utwente.nl/fmt/fault-trees/ft-moea

[†] https://github.com/killB0x/ft-moea-cm-server

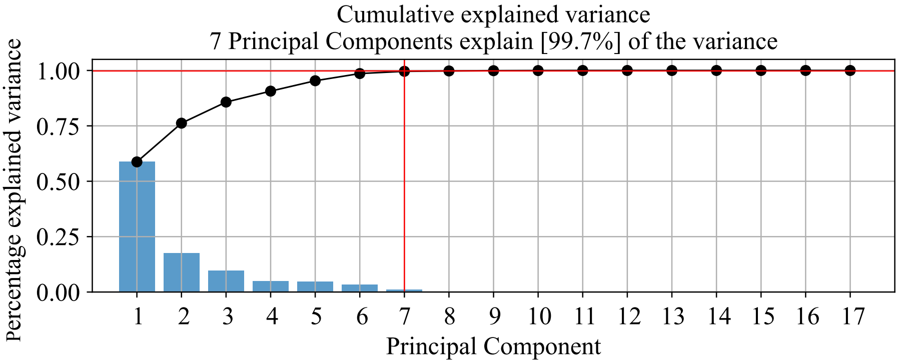


Figure 4.2: Scree plot: cumulative explained variance per principal component.

and data analysis. We apply PCA using the Python `PCA` package (Taskesen, 2020). This process was applied to all case studies listed in Table 4.2, compiling a *feature dataset* for each case study with the 17 metrics in Table 4.1.

Metrics were calculated for *every FT in each generation* to identify the most uncorrelated metrics. This is crucial as uncorrelated metrics enhance convergence in multi-objective evolutionary algorithms by improving diversity and preventing biased searches. The column titled “All FTs” in Table 4.2 specifies the total number of data samples available for PCA analysis in each case study.

Data pre-processing. To mitigate the potential dominance of any case study due to data volume discrepancies, random sampling is employed across all cases, ensuring uniformity by aligning dataset sizes with that of the smallest one (i.e., case COVID-19). Similarly, to avoid dominance of one feature over others due to magnitude disparities, we normalise each feature. This normalisation involves subtracting the mean and scaling to unit variance, a process executed using the `StandardScaler` function from the `preprocessing` module in `scikit-learn` (Pedregosa, Varoquaux, Gramfort, et al., 2011).

Principal Component Analysis. We use PCA to identify the most informative metrics. We examine the explained variance percentage of each Principal Component (PC), depicted in the scree plot in Figure 4.2. This plot reveals that the first 7 out of 17 PCs account for 99.78% of the variance in the features dataset. This suggests that only 7 of the 17 PCs are informative.

The analysis of *loadings* reflects each feature’s contribution magnitude to a particular PC and is crucial for identifying the most informative metrics for FT inference. Table 4.3 presents the loadings for each metric across the seven main PCs. A higher absolute loading value indicates a stronger contribution to the respective PC, and the loading’s sign shows the correlation nature. This analysis reveals that the most informative metrics are: *Matthews correlation coefficient*, *Specificity*, *Negative predictive value*, *Precision*, *Diagnostic odds ratio*, *FT size*, and *Accuracy*.

Table 4.3: Loading analysis per metric.

	PC	**Feature**	**Loading**	**Type**
1	PC1	Matthews correlation coef.	0.296	best
2	PC2	Specificity	0.538	best
3	PC3	Negative predictive value	0.656	best
4	PC4	Precision	0.525	best
5	PC5	Diagnostic odds ratio	0.702	best
6	PC6	FT Size	0.791	best
7	PC7	Accuracy	0.873	best
8	PC2	Sensitivity	−0.370	weak
9	PC1	Threat Score	0.283	weak
10	PC4	Balanced accuracy	0.348	weak
11	PC1	F1 Score	0.283	weak
12	PC1	Fowlkew-Mallows Index	0.284	weak
13	PC4	Informedness	−0.393	weak
14	PC4	Markedness	0.348	weak
15	PC1	Kappa statistic	0.293	weak
16	PC2	Negative likelihood ratio	−0.315	weak
17	PC2	Positive likelihood ratio	0.487	weak

The *Matthews correlation coefficient* (0.296) has the highest loading on PC1. However, other features have similar loading on PC1, such as the *Threat Score* (0.283). This similarity may indicate a correlation between these metrics, so only the one with the highest loading is considered. For other PCs, such as PC6 and PC7, *FT Size* and *Accuracy* are respectively the highest contributors, indicating that these metrics are uncorrelated with the others. Thus, the seven *best* metrics consistently contribute the most uniquely to their respective PC and are minimally correlated across different case studies, whereas the *weak* features show higher correlations to one or more of the best features, and therefore left out of the analysis.

4.5.2 Comparing `FT-MOEA` and `FT-MOEA-CM`

The comparison between `FT-MOEA` and `FT-MOEA-CM` focuses on three key aspects: robustness, scalability, and convergence speed. Robustness is assessed by examining the variability in the output FT. Convergence speed is evaluated by analysing the rate of convergence. Finally, scalability analysis involves studying case studies of various sizes.

Results interpretation. Part of the comparative analysis includes box plots constructed from the outcomes of each experimental setup. In these setups, the algorithm is executed *five* times to generate distinct instances of the experiment, yielding five separate results for the same configuration. This repeated execution is crucial for accurately assessing the outcomes due to the stochastic nature of the optimisation process, where genetic operators are randomly applied. By running the algorithm multiple times, we can effectively evaluate the impact of this randomness on the results.

Robustness. Robustness is assessed by analysing the variability in the output FT upon convergence. An algorithm is considered robust if it consistently yields the same FT structure, though this criterion may not be universally applicable due to the potential existence of multiple optimal FT structures for the same failure dataset.

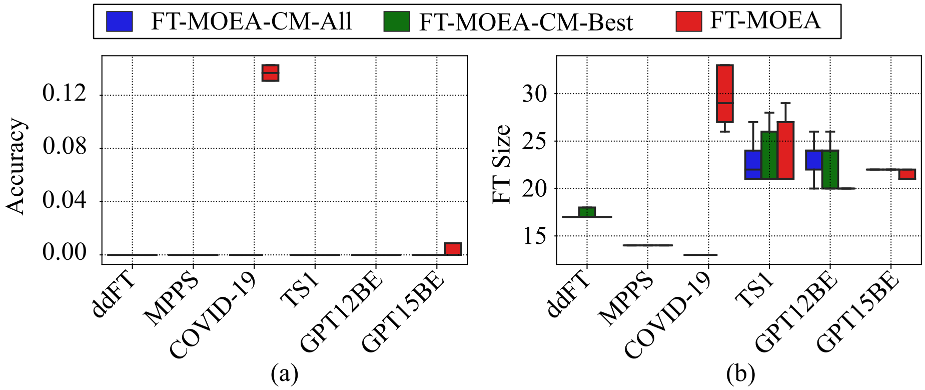


Figure 4.3: Evaluating robustness for all case studies and algorithms. In (a), based on accuracy, FT-MOEA-CM reached global optima for all case studies, while FT-MOEA failed for *COVID-19* and *GPT15BE*. In (b) based on FT size.

Discussion. Robustness is evaluated by examining the variance in box plots for accuracy and FT size, where a variance close to zero suggests higher robustness. In Figure 4.3(a), *accuracy* results for all case studies are presented using box plots to compare FT-MOEA-CM-All, FT-MOEA-CM-Best, and FT-MOEA. An accuracy of 0 is optimal. It is observed that FT-MOEA-CM's configurations consistently achieved an Accuracy of 0.0 with zero variance, indicating they consistently reach the global optima across all case studies. In contrast, FT-MOEA failed to achieve the global optima in the COVID-19 and GPT15BE case studies.

Regarding *FT size*, Figure 4.3(b) shows that results vary significantly among algorithms. For instance, in the TS1 case study, all algorithms yield varying FT sizes. For GPT12BE, FT-MOEA showed greater robustness compared to FT-MOEA-CM-All and FT-MOEA-CM-Best, whereas the opposite holds for COVID-19.

Conclusion. The results indicate that in terms of Accuracy, FT-MOEA-CM-All and FT-MOEA-CM-Best are more consistent compared to FT-MOEA, achieving the global optima for all case studies. However, regarding FT Size and the related FT structure, the consistency of the results varies.

Scalability. The scalability analysis assesses the algorithms' efficiency in managing more complex case studies. The complexity of a FT is linked to its number of elements (i.e., BEs and logic gates) and MCSs, indicating that a more complex FT represents a more challenging inference process.

Discussion. Table 4.2 presents the FT Size and the number of MCSs per case study, arranged from the one with the fewest BEs to that with the most. The underlying concept is that case studies with fewer BEs and MCSs should be simpler to manage. As indicated by the results in Figure 4.3, FT-MOEA-CM-All and FT-MOEA-CM-Best consistently reached the global optima across all cases,

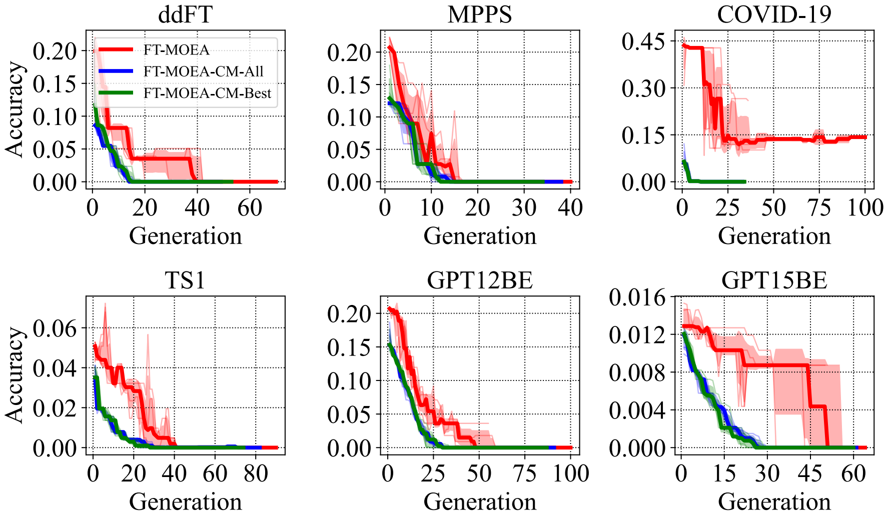


Figure 4.4: Accuracy over generations for all case studies and algorithms.

while FT-MOEA did not in the COVID-19 and GPT15BE case studies. However, in the GPT12BE case, all three algorithms exhibited strong performance, especially FT-MOEA, which surpassed FT-MOEA-CM-All and FT-MOEA-CM-Best. Nevertheless, given that GPT12BE contains more BEs and MCSs than COVID-19, the reasons for FT-MOEA's difficulties with the latter are not clear.

Figure 4.4 depicts the convergence over generations for each case study in terms of accuracy. It is observed that FT-MOEA-CM's configurations converge more rapidly to the optimal accuracy compared to FT-MOEA. These findings suggest that FT-MOEA-CM may scale better than FT-MOEA due to its superior convergence profile, indicating enhanced capabilities to manage larger problems. Further research is necessary to examine this hypothesis more comprehensively.

Conclusion. Regarding scalability, our findings suggest that FT-MOEA-CM may be more scalable than FT-MOEA.

Convergence Speed. Convergence speed refers to the time required by an algorithm to automatically infer an FT from a failure dataset.

Discussion. Figure 4.5 shows the convergence speed, measured in minutes. In specific case studies, such as ddFT, MPPS, and COVID-19, FT-MOEA-CM outpaced FT-MOEA. Notably, FT-MOEA-CM-Best demonstrated superior performance in the MPPS case study. For other cases, like TS1 and GPT12BE, the algorithms' convergence speeds were comparable. Nonetheless, in the GPT15BE case study, FT-MOEA converged faster but to a local optima.

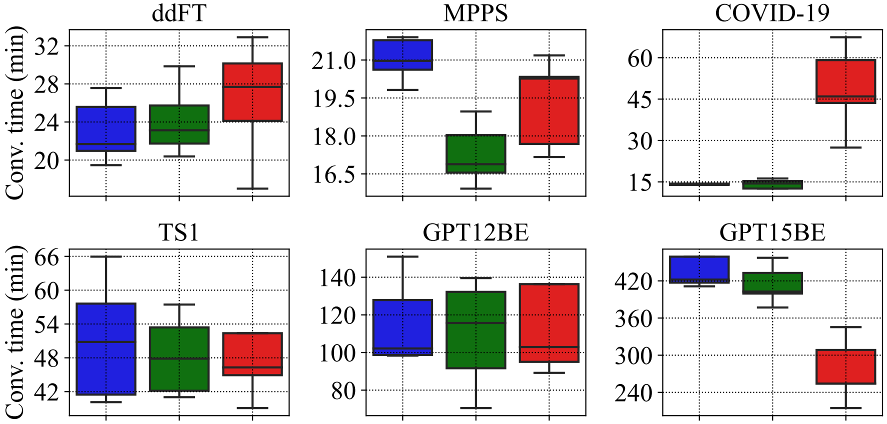


Figure 4.5: Convergence time per case study and algorithm: `FT-MOEA-CM-All` (Blue box); `FT-MOEA-CM-Best` (Green box); `FT-MOEA` (Red box).

A different perspective on the convergence process is illustrated in Figure 4.6, focusing on FT Size across generations and Figure 4.7, showing the computational time per generation. `FT-MOEA-CM` approaches larger FT sizes more rapidly (to attain global optima), then transitions to optimising the FT structure by reducing FT Size, a pattern also identified in Jimenez-Roa, Heskes, Tinga, et al., 2023.

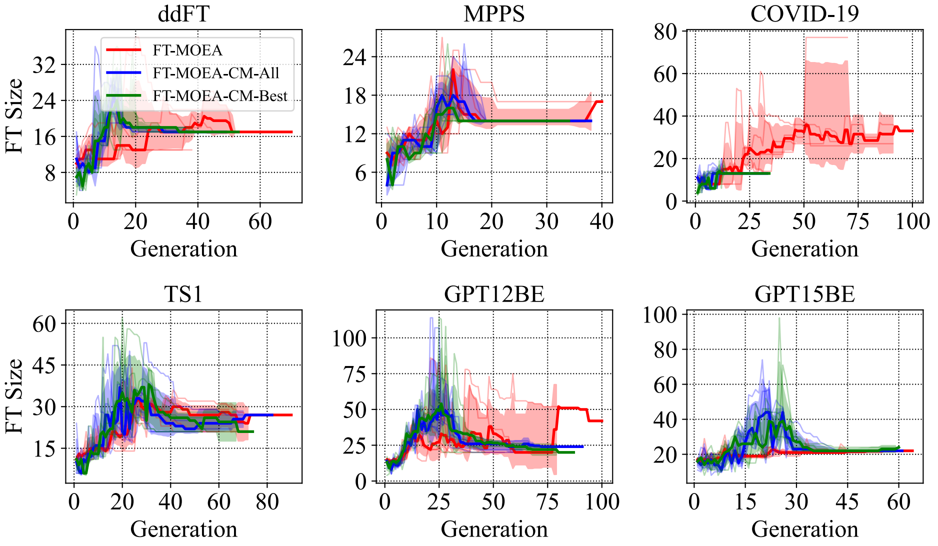


Figure 4.6: FT size across generations: Convergence for all case studies and algorithms based on FT size.

Conclusion. The results in terms of convergence speed are less conclusive; in some cases, `FT-MOEA-CM` outperforms `FT-MOEA`, but in others, it is rather similar or even slower. However, notice `FT-MOEA-CM` always achieved global optima.

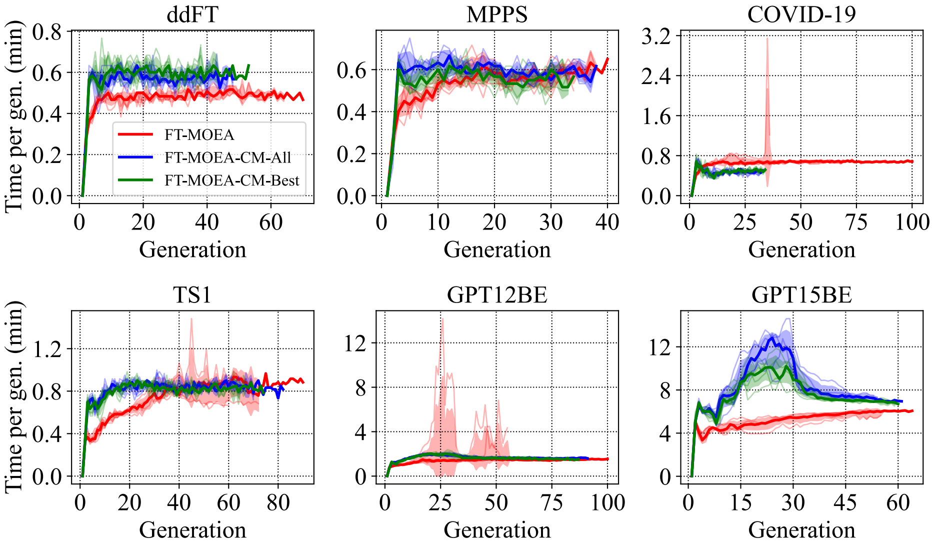


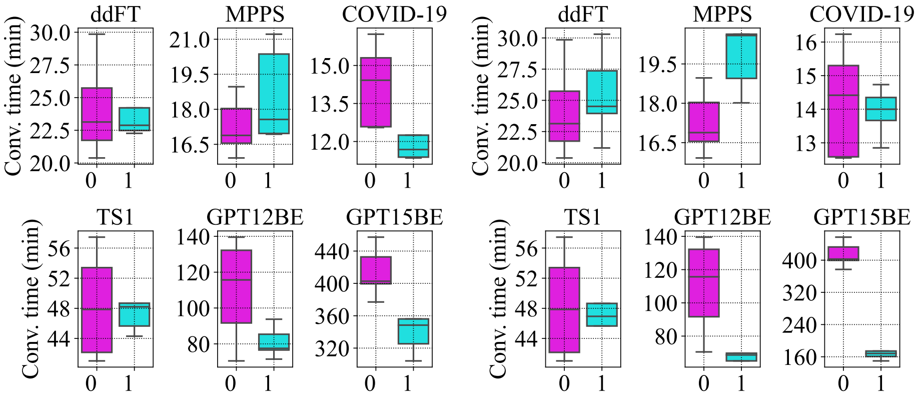
Figure 4.7: Computational time per generation: Convergence for all case studies and algorithms based on computational time per generation.

Comparing `FT-MOEA-CM-All` and `FT-MOEA-CM-Best`. Both configurations yield the global optima in terms of robustness, yet exhibited less consistency in the TS1 and GPT12BE case studies. In terms of scalability, the two setups are on par. Their convergence speed is also similar, except in the MPPS case study, where `FT-MOEA-CM-Best` consistently outperformed. Importantly, `FT-MOEA-CM-Best` used only the top 7 features (identified through PCA in Section 4.5.1), compared to `FT-MOEA-CM-All`, which utilised all 17 features. This suggests that `FT-MOEA-CM-Best` provides a more efficient setup for `FT-MOEA-CM`.

4.5.3 `FT-MOEA-CM`'s Features: Parallelisation and Caching

We evaluate the effects of parallelisation and caching on `FT-MOEA-CM-Best`. Caching stores intermediate results to avoid redundant computations, thereby enhancing efficiency. Parallelisation employs multiple processors to perform tasks concurrently, thus decreasing execution time by distributing the workload.

According to Figure 4.8(a), caching benefits all case studies, excluding MPPS, by enabling quicker convergence. From Figure 4.8(b), parallelisation is shown to improve convergence speed by approximately 45% for larger case studies, namely GPT12BE and GPT15BE. While parallelisation ensures enhanced consistency for TS1 and COVID-19, it negatively affects convergence time in ddFT and MPPS.



(a) Caching

(b) Parallelisation

Figure 4.8: Evaluation of (a) caching and (b) parallelisation, where 0 (magenta box) and 1 (cyan box) indicate the feature "off" or "on", respectively.

4.6 General discussion

Comparing state-of-the-art algorithms is difficult due to varying input data types, such as time series (Niloofar and Lazarova-Molnar, 2023) and the lack of publicly available implementations. Since FT-MOEA uses failure datasets, we focus on publicly available algorithms with the same input data type. Among these, LIFT (Nauta, Bucur, and Stoelinga, 2018) requires intermediate event data, and FT-BN (Linard, Bucur, and Stoelinga, 2019) needs white- and black-listing information, neither of which FT-MOEA requires. Therefore, we primarily compare FT-MOEA with its predecessor FT-EA (Linard, Bucur, and Stoelinga, 2019), and its extensions SymLearn and FT-MOEA-CM.

For comparison, we identify robustness, scalability, and convergence speed as relevant criteria. Additionally, to substantiate this comparison, the most important results on this topic addressed in this thesis are comprehensively presented in Table 4.4 in terms of inferred FT size; Table 4.5 in terms of correctly encoded Minimal Cut Sets (MCSs); and Table 4.6 in terms of convergence speed.

Robustness refers to consistently yielding correct FTs with similar structures. From Table 4.4, we observe that for case studies *CSD*, *PT*, *COVID-19*, *ddFT*, *MPPS*, and *SMS*, FT-MOEA was more robust than FT-EA, indicated by a smaller difference between the first (*Q1*) and third (*Q3*) quantiles for most case studies. This suggests that casting the optimisation problem as a multi-objective function enhances robustness. Similarly, FT-MOEA-CM is more robust than FT-MOEA (and thus FT-EA), as shown in Table 4.5, where FT-MOEA-CM consistently achieved global optima by encoding all MCSs in the failure dataset, a task not always achieved by FT-MOEA.

FT inference algorithms that achieve global optima, handle large datasets, and

Table 4.4: Sizes of the inferred Fault Trees (FTs), per algorithm across all the case studies (evaluated 5 times) in Part I of this thesis. |BEs| is the number of Basic Events; $|\mathcal{F}|$ is the FT size; $|\mathbb{CD}|$ is the number of MCSs in the ground truth problem. *Q1*, *Q2*, and *Q3* are respectively the 25%, 50%, and 75% quantiles.

Case	|BEs|	$\|\mathcal{F}\|$	$\|\mathbb{CD}\|$	FT-EA			FT-MOEA			SymLearn			FT-MOEA-CM		
				Q1	*Q2*	*Q3*	*Q1*	*Q2*	*Q3*	*Q1*	*Q2*	*Q3*	*Q1*	*Q2*	*Q3*
CSD$^{(a)}$	6	10	3	10	10	11	10	10	10	-	-	-	10	10	10
PT$^{(b)}$	6	11	5	9	10	11	9	9	9	-	-	-	9	9	9
COVID-19$^{(c)}$	9	13	6	14	17	18	13	13	13	-	-	-	13	13	13
ddFT$^{(d)}$	8	13	6	34	35	53	11	13	17	-	-	-	17	17	18
MPPS$^{(e)}$	8	23	7	21	24	27	14	20	21	-	-	-	14	14	14
SMS$^{(f)}$	13	25	13	14	14	14	14	14	14	-	-	-	14	14	14
gpt12$^{(g1)}$	12	25	13	-	-	-	20	20	20	-	-	-	20	24	24
gpt15$^{(g2)}$	15	27	10	-	-	-	21	21	22	-	-	-	22	22	22
SS$^{(h1)}$	10	23	8	-	-	-	20	21	21	23	23	23	-	-	-
SC$^{(h2)}$	6	11	4	-	-	-	11	11	11	11	11	11	-	-	-
TS1$^{(i1)}$	10	34†	16	-	-	-	15	21	25	34	34	34	21	21	26
TS2$^{(i2)}$	24	25†	26	-	-	-	8	23	23	25	25	25	-	-	-
TS3$^{(i3)}$	20	63†	18	-	-	-	15	15	15	63	63	63	-	-	-

† Fault Trees associated to truss systems (Jimenez-Roa, Volk, and Stoelinga, 2022).
$^{(a)}$**CSD**: Container Seal Design (NASA, 2002); $^{(b)}$**PT**: Pressure Tank (NASA, 2002); $^{(c)}$**COVID-19**: COVID-19 FT (Jimenez-Roa, Heskes, Tinga, et al., 2023); $^{(d)}$**ddFT**: Data-driven FT (Lazarova-Molnar, Niloofar, and Barta, 2020); $^{(e)}$**MPPS**: Mono-propellant propulsion system (NASA, 2002); $^{(f)}$**SMS**: Spread Monitoring System (Mentes and Helvacioglu, 2011); $^{(g1)}$**gpt12**: GPT generated FT with 12 BEs (Jimenez-Roa, Rusnac, Volk, et al., 2024); $^{(g2)}$**gpt15**: GPT generated FT with 15 BEs (Jimenez-Roa, Rusnac, Volk, et al., 2024); $^{(h1)}$**SS**: symmetric toy-example (Jimenez-Roa, Volk, and Stoelinga, 2022); $^{(h2)}$**SC**: symmetric toy-example (Jimenez-Roa, Volk, and Stoelinga, 2022); $^{(h3)}$**SC**: Truss system case TS1 (Jimenez-Roa, Volk, and Stoelinga, 2022); $^{(h4)}$**SC**: Truss system case TS2 (Jimenez-Roa, Volk, and Stoelinga, 2022); $^{(h5)}$**SC**: Truss system case TS3 (Jimenez-Roa, Volk, and Stoelinga, 2022)

adapt efficiently from smaller to larger systems are crucial for *scalability*. Table 4.5 indicates that, for failure datasets with symmetries, SymLearn handles larger problems (up to 24 BEs, FTs with up to 63 elements, and 26 MCSs) compared to FT-MOEA. This highlights the benefits of using information like symmetries, especially in the case study *TS3*, where FT-MOEA struggles with local optima. Additionally, FT-MOEA-CM shows greater scalability than FT-MOEA, even without harnessing symmetries, emphasising the advantages of incorporating more information into the multi-objective function.

Finally, we measure the time taken by the algorithms to complete the task, as reported in Table 4.6. Generally, FT-MOEA converges faster than FT-EA, suggesting that multi-objective optimisation enables more efficient convergence. Between FT-MOEA and FT-MOEA-CM, the results are less conclusive; in some cases, FT-MOEA was faster than FT-MOEA-CM, and vice versa. However, FT-MOEA-CM consistently achieves global optima. SymLearn often outperforms FT-MOEA when symmetries are present in the failure dataset. Notably, FT-MOEA-CM converges to the global optima four times faster than both FT-MOEA and SymLearn for the case *TS1*.

Table 4.5: Percentage of Minimal Cut Sets (MCSs) that were correctly encoded by the inferred Fault Tree (FT) per algorithm across all the case studies (evaluated 5 times) in Part I of this thesis. |BEs| is the number of Basic Events; $|\mathcal{F}|$ is the FT size; $|\mathbb{CD}|$ is the number of MCSs in the ground truth problem. *Q1*, *Q2*, and *Q3* are respectively the 25%, 50%, and 75% quantiles.

Case	|BEs|	$\|\mathcal{F}\|$	$\|\mathbb{CD}\|$	FT-EA			FT-MOEA			SymLearn			FT-MOEA-CM		
				Q1	*Q2*	*Q3*	*Q1*	*Q2*	*Q3*	*Q1*	*Q2*	*Q3*	*Q1*	*Q2*	*Q3*
CSD[a]	6	10	3	100%	100%	100%	100%	100%	100%	-	-	-	100%	100%	100%
PT[b]	6	11	5	40%	100%	100%	100%	100%	100%	-	-	-	100%	100%	100%
COVID-19[c]	9	13	6	100%	100%	100%	100%	100%	100%	-	-	-	100%	100%	100%
ddFT[d]	8	13	6	50%	50%	50%	17%	67%	100%	-	-	-	100%	100%	100%
MPPS[e]	8	23	7	57%	100%	100%	100%	100%	100%	-	-	-	100%	100%	100%
SMS[f]	13	25	13	100%	100%	100%	100%	100%	100%	-	-	-	100%	100%	100%
gpt12[g1]	12	25	13	-	-	-	100%	100%	100%	-	-	-	100%	100%	100%
gpt15[g2]	15	27	10	-	-	-	90%	90%	100%	-	-	-	100%	100%	100%
SS[h1]	10	23	8	-	-	-	88%	94%	100%	100%	100%	100%	-	-	-
SC[h2]	6	11	4	-	-	-	100%	100%	100%	100%	100%	100%	-	-	-
TS1[h3]	10	34†	16	-	-	-	77%	84%	88%	62%	62%	62%	100%	100%	100%
TS2[h4]	24	25†	26	-	-	-	35%	100%	100%	100%	100%	100%	-	-	-
TS3[h5]	20	63†	18	-	-	-	0%	0%	0%	100%	100%	100%	-	-	-

† Fault Trees associated to truss systems (Jimenez-Roa, Volk, and Stoelinga, 2022).
[a] **CSD**: Container Seal Design (NASA, 2002); [b] **PT**: Pressure Tank (NASA, 2002); [c] **COVID-19**: COVID-19 FT (Jimenez-Roa, Heskes, Tinga, et al., 2023); [d] **ddFT**: Data-driven FT (Lazarova-Molnar, Niloofar, and Barta, 2020); [e] **MPPS**: Mono-propellant propulsion system (NASA, 2002); [f] **SMS**: Spread Monitoring System (Mentes and Helvacioglu, 2011); [g1] **gpt12**: GPT generated FT with 12 BEs (Jimenez-Roa, Rusnac, Volk, et al., 2024); [g2] **gpt15**: GPT generated FT with 15 BEs (Jimenez-Roa, Rusnac, Volk, et al., 2024); [h1] **SS**: symmetric toy-example (Jimenez-Roa, Volk, and Stoelinga, 2022); [h2] **SC**: symmetric toy-example (Jimenez-Roa, Volk, and Stoelinga, 2022); [h3] **SC**: Truss system case TS1 (Jimenez-Roa, Volk, and Stoelinga, 2022); [h4] **SC**: Truss system case TS2 (Jimenez-Roa, Volk, and Stoelinga, 2022); [h5] **SC**: Truss system case TS3 (Jimenez-Roa, Volk, and Stoelinga, 2022)

4.7 Conclusions

In this chapter, we introduced `FT-MOEA-CM`, an extension of the `FT-MOEA` algorithm, specifically designed for inferring Fault Tree models from failure datasets using the `NSGA-II` and Crowding Sorting algorithms for multi-objective optimisation.

An important distinction of `FT-MOEA-CM` from its predecessor is the incorporation of features derived from the *confusion matrix*. We conducted a Principal Component Analysis on 17 available features, identifying 7 as the most important: **Matthews correlation coefficient**, **Specificity**, **Negative predictive value**, **Precision**, **Diagnostic odds ratio**, **FT size**, and **Accuracy**.

We compared `FT-MOEA-CM` (this chapter), `FT-MOEA` (Chapter 2), `SymLearn` (Chapter 3), and `FT-EA` (Linard, Bucur, and Stoelinga, 2019) across 9 case studies, and the results suggest that `FT-MOEA-CM` is more robust, consistently achieving the global optima across all case studies, unlike the other implementations, and producing similar FT structures. In terms of scalability, `FT-MOEA-CM` appears

Table 4.6: Convergence time in minutes taken per algorithm across all the case studies (evaluated 5 times) in Part I of this thesis. |BEs| is the number of Basic Events; $|\mathcal{F}|$ is the FT size; $|\mathbb{CD}|$ is the number of MCSs in the ground truth problem. *Q1*, *Q2*, and *Q3* are respectively the 25%, 50%, and 75% quantiles.

Case	|BEs|	$\|\mathcal{F}\|$	$\|\mathbb{CD}\|$	FT-EA			FT-MOEA			SymLearn			FT-MOEA-CM		
				Q1	*Q2*	*Q3*	*Q1*	*Q2*	*Q3*	*Q1*	*Q2*	*Q3*	*Q1*	*Q2*	*Q3*
CSD$^{(a)}$	6	10	3	6.5	27.5	42.1	4.6	5.0	5.0	-	-	-	9.7	10.8	12.1
PT$^{(b)}$	6	11	5	0.1	0.1	4.0	2.0	2.8	3.0	-	-	-	6.5	6.8	7.2
COVID-19$^{(c)}$	9	13	6	37.8	39.0	42.9	13.2	22.3	24.1	-	-	-	12.6	14.4	15.3
ddFT$^{(d)}$	8	13	6	26.8	37.7	40.9	10.9	23.0	56.3	-	-	-	21.7	23.1	25.7
MPPS$^{(e)}$	8	23	7	19.0	46.6	55.3	26.2	29.9	33.9	-	-	-	16.6	16.9	18.0
SMS$^{(f)}$	13	25	13	0.1‡	0.1‡	0.1‡	214.0	218.8	219.0	-	-	-	215.3	216.4	216.5
gpt12$^{(g1)}$	12	25	13	-	-	-	95.0	102.9	136.3	-	-	-	91.7	115.7	132.2
gpt15$^{(g2)}$	15	27	10	-	-	-	254.2	254.2	308.5	-	-	-	399.5	402.8	432.8
SS$^{(h1)}$	10	23	8	-	-	-	131.5	142.9	146.9	0.0	0.0	0.0	-	-	-
SC$^{(h2)}$	6	11	4	-	-	-	33.8	34.6	35.8	0.0	0.0	0.0	-	-	-
TS1$^{(i1)}$	10	34†	16	-	-	-	163.8	214.7	273.3	211.0	211.9	224.9	42.2	47.9	53.4
TS2$^{(i2)}$	24	25†	26	-	-	-	51.7	112.2	127.1	0.7	0.7	0.7	-	-	-
TS3$^{(i3)}$	20	63†	18	-	-	-	89.4	91.4	92.8	2.1	2.2	2.2	-	-	-

† Fault Trees associated to truss systems (Jimenez-Roa, Volk, and Stoelinga, 2022).
$^{(a)}$**CSD**: Container Seal Design (NASA, 2002); $^{(b)}$**PT**: Pressure Tank (NASA, 2002); $^{(c)}$**COVID-19**: COVID-19 FT (Jimenez-Roa, Heskes, Tinga, et al., 2023); $^{(d)}$**ddFT**: Data-driven FT (Lazarova-Molnar, Niloofar, and Barta, 2020); $^{(e)}$**MPPS**: Mono-propellant propulsion system (NASA, 2002); $^{(f)}$**SMS**: Spread Monitoring System (Mentes and Helvacioglu, 2011); $^{(g1)}$**gpt12**: GPT generated FT with 12 BEs (Jimenez-Roa, Rusnac, Volk, et al., 2024); $^{(g2)}$**gpt15**: GPT generated FT with 15 BEs (Jimenez-Roa, Rusnac, Volk, et al., 2024); $^{(h1)}$**SS**: symmetric toy-example (Jimenez-Roa, Volk, and Stoelinga, 2022); $^{(h2)}$**SC**: symmetric toy-example (Jimenez-Roa, Volk, and Stoelinga, 2022); $^{(h3)}$**SC**: Truss system case TS1 (Jimenez-Roa, Volk, and Stoelinga, 2022); $^{(h4)}$**SC**: Truss system case TS2 (Jimenez-Roa, Volk, and Stoelinga, 2022); $^{(h5)}$**SC**: Truss system case TS3 (Jimenez-Roa, Volk, and Stoelinga, 2022)

superior to the alternatives, being capable of handling larger problems. Additionally, FT-MOEA-CM demonstrated a higher convergence speed, evaluated by both convergence time and convergence profile.

FT-MOEA-CM's features, caching and parallelisation, proved to improve convergence speed, with potential benefits to infer larger FTs.

Future research. Possible directions for future work include:

- Addressing the scalability issue in computing the algorithm's confusion matrix metrics for exponentially growing datasets by using approximate evaluations with subsets of the failure dataset during initial algorithm generations.
- Exploring methods to facilitate convergence to Directed Acyclic Graphs (DAGs) instead of trees, using tree decomposition and Tree Width metrics to measure a graph's resemblance to a tree, applicable to specific DAG instances.
- Developing a benchmark for fair comparison of algorithms for automatic Fault Tree model inference to understand their comparative advantages and drawbacks, ensuring uniform and thorough evaluation.
- Extending our approach to handle missing information and noise, is crucial for realistic scenarios where data may be incomplete or inaccurate.

4.8 References

Abdi, H. and L. J. Williams (2010). “Principal component analysis”. In: *WIREs Computational Statistics* 2.4, pp. 433–459. DOI: 10.1002/wics.101.

Bakeli, T., A. A. Hafidi, et al. (2020). “COVID-19 infection risk management during construction activities: An approach based on Fault Tree Analysis (FTA)”. In: *Journal of Emergency Management* 18.7, pp. 161–176. DOI: 10.5055/jem.0539.

Božić, D., B. Runje, D. Lisjak, and D. Kolar (2023). “Metrics Related to Confusion Matrix as Tools for Conformity Assessment Decisions”. In: *Applied Sciences* 13.14. ISSN: 2076-3417. DOI: 10.3390/app13148187.

Deb, K., A. Pratap, S. Agarwal, and T. Meyarivan (2002). “A fast and elitist multiobjective genetic algorithm: NSGA-II”. In: *IEEE Transactions on Evolutionary Computation* 6 (2), pp. 182–197. DOI: 10.1109/4235.996017.

Jimenez-Roa, L. A., N. Rusnac, M. Volk, and M. Stoelinga (2024). “Fault Tree Inference Using Multi-objective Evolutionary Algorithms and Confusion Matrix-Based Metrics”. In: *Formal Methods for Industrial Critical Systems*. Ed. by A. E. Haxthausen and W. Serwe. Cham: Springer Nature Switzerland, pp. 80–96. ISBN: 978-3-031-68150-9. DOI: 10.1007/978-3-031-68150-9_5.

Jimenez-Roa, L. A., T. Heskes, T. Tinga, and M. Stoelinga (2023). “Automatic Inference of Fault Tree Models Via Multi-Objective Evolutionary Algorithms”. In: *IEEE Transactions on Dependable and Secure Computing* 20.4, pp. 3317–3327. DOI: 10.1109/TDSC.2022.3203805.

Jimenez-Roa, L. A., M. Volk, and M. Stoelinga (2022). “Data-Driven Inference of Fault Tree Models Exploiting Symmetry and Modularization”. In: *Computer Safety, Reliability, and Security*. Ed. by M. Trapp, F. Saglietti, M. Spisländer, and F. Bitsch. Cham: Springer International Publishing, pp. 46–61. DOI: 10.1007/978-3-031-14835-4_4.

Lazarova-Molnar, S., P. Niloofar, and G. K. Barta (2020). “Data-Driven Fault Tree Modeling for Reliability Assessment of Cyber-Physical Systems”. In: *2020 Winter Simulation Conference (WSC)*, pp. 2719–2730. DOI: 10.1109/WSC48552.2020.9383882.

Linard, A., D. Bucur, and M. Stoelinga (2019). “Fault Trees from Data: Efficient Learning with an Evolutionary Algorithm”. In: *International Symposium on Dependable Software Engineering: Theories, Tools, and Applications*. Vol. 11951 LNCS, pp. 19–37. DOI: 10.1007/978-3-030-35540-1_2.

Madden, M. G. and P. J. Nolan (1994). “Generation of fault trees from simulated incipient fault case data”. In: *WIT Transactions on Information and Communication Technologies* 6. DOI: 10.2495/AI940611.

Martí, L., E. Segredo, N. Śnchez-Pi, and E. Hart (2017). “Impact of selection methods on the diversity of many-objective Pareto set approximations”. In: *Procedia Computer Science* 112. Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 21st International Conference, KES-20176-8 September 2017, Marseille, France, pp. 844–853. ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2017.08.077.

Mentes, A. and I. H. Helvacioglu (2011). “An application of fuzzy fault tree analysis for spread mooring systems”. In: *Ocean Engineering* 38 (2-3), pp. 285–294. DOI: 10.1016/j.oceaneng.2010.11.003.

NASA (2002). *Fault Tree Handbook with Aerospace Applications*. Handbook. U.S. National Aeronautics and Space Administration.

Nauta, M., D. Bucur, and M. Stoelinga (2018). “LIFT: Learning fault trees from observational data”. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 11024 LNCS, pp. 306–322. DOI: 10.1007/978-3-319-99154-2_19.

Niloofar, P. and S. Lazarova-Molnar (2023). "Data-driven extraction and analysis of repairable fault trees from time series data". In: *Expert Systems with Applications* 215, p. 119345. ISSN: 0957-4174. DOI: 10.1016/j.eswa.2022.119345.

OpenAI et al. (2024). *GPT-4 Technical Report*. DOI: 10.48550/ARXIV.2303.08774.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay (2011). "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12.85, pp. 2825–2830. URL: http://jmlr.org/papers/v12/pedregosa11a.html.

Ruijters, E. and M. Stoelinga (2015). "Fault tree analysis: A survey of the state-of-the-art in modeling, analysis and tools". In: *Computer Science Review* 15-16, pp. 29–62. ISSN: 1574-0137. DOI: https://doi.org/10.1016/j.cosrev.2015.03.001.

Sokolova, M. and G. Lapalme (2009). "A systematic analysis of performance measures for classification tasks". In: *Information Processing & Management* 45.4, pp. 427–437. ISSN: 0306-4573. DOI: 10.1016/j.ipm.2009.03.002.

Taskesen, E. (Oct. 2020). *PCA: A Python Package for Principal Component Analysis*. Version 1.8.4. URL: https://erdogant.github.io/pca.

Waghen, K. and M.-S. Ouali (2019). "Interpretable logic tree analysis: A data-driven fault tree methodology for causality analysis". In: *Expert Systems with Applications* 136, pp. 376–391. DOI: 10.1016/j.eswa.2019.06.042.

Part II

Multi-state deterioration modelling

II.1 Introduction

Part II focuses on Multi-State Deterioration modelling with application to sewer mains. The general research question we address here is *how and to what extent is it possible to accurately model Multi-State Deterioration with applications in sewer mains?* This part is structured as follows: Section II.2 summarises the nomenclature used in Part II. Section II.3 reviews the related work common to all chapters. Section II.4 presents the formal definitions used in Part II. The chapters contained here are:

Chapter 5. *Deterioration Modelling of Sewer Pipes via Discrete-Time Markov Chains: A Large-Scale Case Study in the Netherlands* 111

Chapter 6. *Comparing Homogeneous and Inhomogeneous Time Markov Chains for Modelling Deterioration in Sewer Pipe Networks* 123

II.2 Nomenclature

Markov chains for Multi-State Deterioration Modelling:

T	Parameter space.
t	Time (e.g., component *age*), with $t \in T$
X_t	Stochastic process over t.
$\mathbb{S}$	State space of X_t.
k	Severity index $k \in \mathbb{S}$.
$\boldsymbol{p}_k^{(0)}$	Initial state distribution over $\mathbb{S}$.
$\boldsymbol{p}_k^{(t)}$	State probability distribution over $\mathbb{S}$ at t.
$\mathcal{M}$	Markov chain hypothesis.
i	Sojourn state, where $i \in \mathbb{S}$.
j	Arrival state, where $j \in \mathbb{S}$.
$\mathbf{P}_{ij}(\cdot)$	Transition probability matrix (function).
$\mathbf{Q}_{ij}(\cdot)$	Transition rate matrix (function).

Survival Analysis:

$f(\cdot)$	Probability density function.
$\mathbf{S}(\cdot)$	Survival function.
$\lambda(\cdot)$	Hazard rate function.

II.3 Related work

Sewer networks, crucial for social and economic welfare, present management challenges due to limited budgets, environmental changes, and complex, hard-to-model deterioration processes. As these systems approach the end of their design life, predictive tools for deterioration become vital for efficient maintenance and logistics (Marc Ribalta and Rubión, 2023). Robust models for sewer main deterioration help in identifying high-risk pipes, thus facilitating proactive maintenance, decision-making, and strategic planning (Scheidegger, Hug, Rieckermann, et al., 2011; Egger, Scheidegger, Reichert, et al., 2013; Caradot, Sonnenberg, Kropp, et al., 2017). Deterioration models for sewer mains are typically developed using inspection data adhering to standards such as the EN 13508:1 and EN 13508:2. These standards guide the classification of damages observed via Closed Circuit Television (CCTV) inspections into *severity levels*.

Comprehensive reviews by Ana and Bauwens, 2010; Malek Mohammadi, Najafi, Kaushal, et al., 2019; Hawari, Alkadour, Elmasry, et al., 2020; Saddiqi, Zhao, Cotterill, et al., 2023; Zeng, Z. Wang, H. Wang, et al., 2023 categorise sewer main deterioration models into three main types: physics-based, Machine Learning (ML)-based, and probabilistic models. *Physics-based* models utilise mathematical relations grounded in physical principles. *ML-based* models are increasingly recognised for their ability to identify complex patterns in large datasets and use these insights for predictive and decision-making applications. Comparisons of different

ML models for sewer network condition assessment are made by Nguyen and Seidu, 2022; El Morer, Wittek, and Rausch, 2024. *Probabilistic* models, grounded in probability theory, treat factors related to sewer network deterioration as random variables. Barraud, Bosco, Clemens-Meyer, et al., 2024 provides a comprehensive and updated overview of different approaches used for deterioration modelling in sewer mains.

These approaches have inherent limitations. Physics-based approaches become too extensive when capturing the complex deterioration behaviours present in large-scale systems with varying contexts such as sewer networks. ML-based and probabilistic models are only as effective as the quality and completeness of the data they use, a known challenge in sewer network systems (Noshahri, olde Scholtenhuis, Doree, et al., 2021). Additionally, despite the widespread application of ML-based techniques for diagnostic purposes, such as anomaly detection and condition or defect classification, they may be unsuitable for generating reliable, monotonous deterioration curves. This limitation, highlighted by Rokstad and Ugarelli, 2015; Caradot, Rouault, Clemens, et al., 2018; Kantidakis, Putter, Litière, et al., 2023, constrains their effectiveness in long-term maintenance planning.

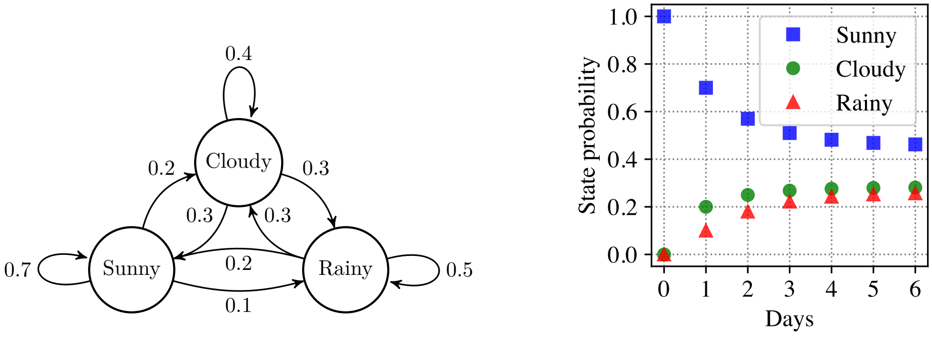
This research focuses on *Markov chains*, a probabilistic model used to predict the future distribution of the deterioration states. Markov chains are crucial in Multi-State Modelling (MSM) due to their ability to model state transitions. Tran, Setunge, and Shi, 2021 highlight the suitability of state-based Markov chains for sewer main deterioration modelling, especially when only a single inspection record is available. The key advantages of Markov chains are: (i) converting condition data into ordinal numbers, such as severity levels, commonly used in infrastructure asset rating (Tran, Lokuge, Setunge, et al., 2022); (ii) capturing the stochastic behaviour of sewer main deterioration; and (iii) providing outputs that indicate pipe condition proportions, essential for optimising maintenance planning.

II.4 Preliminaries

II.4.1 Markov Chains

A *Markov chain* models *states* and the *transitions* between them. As an example, Figure II.1(a) illustrates a Markov chain with three states: *Sunny*, *Cloudy*, and *Rainy*. The arrows represent transitions, with the arrow's start showing the origin state and the end showing the destination state. The numbers on the arrows indicate the probability of moving from one state to another.

Using the Markov chain in Figure II.1(a), we can calculate metrics such as the *state probability*, which is the probability of being in a particular state over time, given the current state. For instance, if the current state is *Sunny*, the Markov chain can determine the probabilities of being in each of the three states over time (i.e., days), as shown in Figure II.1(b). Here, the probability of staying



(a) Discrete-time Markov chain.

(b) State probability over time.

Figure II.1: Markov chain modelling the weather condition (example).

in *Sunny* decreases while it increases for the other states. As noted later in this section, Markov chains model random processes and are therefore useful for sampling; for example, from Figure II.1(a), the following sequence could be obtained: Sunny → Rainy → Rainy → Cloudy → $\cdots$. The definitions below were adopted from Stewart, 2009; Brémaud, 2020; Colombo, Abreu, and Martins, 2021.

Definition 6 (Markov chain)**.** *Let a stochastic process be defined by a family of random variables $X_t \in T$, where t represents* time *and X denotes the value of the random variable at time t; $T \subseteq \mathbb{R}$ is the* parameter space*. If T is discrete, the process is discrete-time; if T is continuous, the process is continuous-time. The values assumed by X_t are termed* states*, which may be either continuous or discrete; in this dissertation, we use the latter. A* Markov chain *is defined as a tuple $\mathcal{M} = \langle \mathbb{S}, \mathbf{P}_{ij}, \boldsymbol{p}^{(0)} \rangle$ where:*

- *$\mathbb{S}$ is a countable set called the* state space *which contains all possible states of the chain.*
- *$\mathbf{P}_{ij} : \mathbb{S} \times \mathbb{S} \to [0, 1]$ is a family of transition probabilities $\mathbb{P}(t, \tau; i, j)$ such that for any states $i, j \in \mathbb{S}$, time $t \in T$, and future time $\tau \in T$ where $\tau \geq t$, $\mathbb{P}(t, \tau; i, j) = \mathbb{P}(X_\tau = j \mid X_t = i)$ gives the probability of transitioning from state i at time t to state j at time τ. Additionally, $\mathbf{P}_{ij} \geq 0$ for all $i, j \in \mathbb{S}$, and $\sum_{j \in S} \mathbf{P}_{ij} = 1$ for all $i \in \mathbb{S}$.*
- *$\boldsymbol{p}^{(0)} : \mathbb{S} \to [0, 1]$ is the initial probability distribution over $\mathbb{S}$. This distribution satisfies $\boldsymbol{p}^{(0)}_k \geq 0$ for all $k \in \mathbb{S}$ and $\sum_{k \in \mathbb{S}} \boldsymbol{p}^{(0)}_k = 1$.*

If $\mathbf{P}_{ij}$ depends on the time the process was initiated, it is termed an *inhomogeneous-time* stochastic process; on the other hand, if $\mathbf{P}_{ij}$ do not depend on the time the process was initiated, it is termed a *homogeneous-time* stochastic process.

If X_t is invariant under any shift of the time origin, it is termed a *stationary* process; if it varies with the time of initiation, it is termed *non-stationary*. X_t has the *Markov property* if it satisfies the following definition.

Definition 7 (Markov property)**.** *The* Markov property *states that the conditional probability of transitioning to any future state depends only on the present state and not on the sequence of events that preceded it. Formally, for any n-length sequence of times $\langle t_0 \leq t_1 \leq \cdots \leq t_n \rangle$ in T, and states $\langle x_0, x_1, \ldots, x_n \rangle$ in $\mathbb{S}$, the property is given by:*

$$\mathbb{P}(X_t = x \mid X_{t_1} = x_1, X_{t_2} = x_2, \ldots, X_{t_n} = x_n) \\ = \mathbb{P}(X_t = x \mid X_{t_n} = x_n), \quad (4.1)$$

this equation holds for all states and time points in the index set T.

In this dissertation, we use *inhomogeneous*, *homogeneous*, *continuous*, and *discrete*-time Markov chains. Before providing their formal definitions and for completeness, below we define *probability density function*, *survival function*, and *hazard rate* (more in Appendix C.1):

Definition 8 (Stochastic Process Functions)**.** *For a stochastic process characterised by a set of functions parametrised by θ—scale, shape, or other distribution-specific parameters—the key functions are:*

- Probability Density Function (PDF), $f(t;\theta)$*: describes the probability density of a continuous random variable at a specific value t, where $f(t;\theta) \geq 0$ for all t and $\int f(t;\theta)\,dt = 1$ across the domain of t.*
- Survival Function (SF), $\mathbf{S}(t;\theta)$*: represents the probability that the lifetime of a system or component exceeds a specific time t, formally defined as $\mathbf{S}(t;\theta) = 1 - \int_0^t f(x;\theta)\,dx$. Note that $\mathbf{S}(\cdot)$ is also referred to as the* reliability function *in other fields.*
- Hazard Rate Function (HR), $\lambda(t;\theta)$*: rate at which failures occur given survival until time t. It is defined by the ratio $\lambda(t;\theta) = \frac{f(t;\theta)}{S(t;\theta)}$, assuming $\mathbf{S}(t;\theta) > 0$.*

Inhomogeneous Time Markov Chain

In an Inhomogeneous Time Markov Chain (IHTMC) the transitions between states are time-dependent and it is formally defined as follows.

Definition 9 (Inhomogeneous-Time Markov Chain)**.** *An* Inhomogeneous Time Markov Chain (IHTMC) *is defined by the tuple $\mathcal{M} = \langle \mathbb{S}, \boldsymbol{p}^{(0)}, \mathbf{Q}_{ij}(t,\theta) \rangle$, see $\mathbb{S}$ and $\boldsymbol{p}^{(0)}$ in Definition 6. Here:*

- $\mathbf{Q}_{ij}(t,\theta) : \mathbb{S} \times \mathbb{S} \to \mathbb{R}$ *is a time-dependent* transition rate matrix*. This matrix function, parametrised by time t and parameters θ, is structured as follows:*
  - *The non-diagonal entries $q_{ij}(t,\theta)$, for $i, j \in \mathbb{S}$ and $i \neq j$, represent the transition rate from state i to state j at time t.*
  - *The diagonal entries $q_{ii}(t,\theta)$ are chosen to ensure that each row sum of $\mathbf{Q}_{ij}$ is zero, signifying that the rate out of any state is equal to the sum of the rates into other states.*

- $\mathbf{Q}_{ij}(t,\theta)$ *can be parametrised by hazard rates* $\lambda(t;\theta)$*, derived from the probability density function* $f(t;\theta)$ *and the survival function* $\mathbf{S}(t;\theta)$*, where* θ *represents the hyper-parameters.*

The IHTMC's temporal evolution is described by the *Forward Kolmogorov* equation:

$$\frac{\partial \mathbf{P}_{ij}(t,\tau)}{\partial t} = \sum_{k\in\mathbb{S}} \mathbf{P}_{ik}(t,\tau)\mathbf{Q}_{kj}(t) \tag{4.2}$$

Here $\mathbf{P}_{ij}(t,\tau)$ is the *transition probability matrix* (see Definition 6). Using Eq. 4.2, the *master equation* of Markov chains is derived, expressing the probability flow between states by incorporating *inflow* and *outflow* terms:

$$\frac{\partial \boldsymbol{p}_k(t)}{\partial t} = \sum_{i\in\mathbb{S},i\neq k} \boldsymbol{p}_i(t)\mathbf{Q}_{ik}(t) - \boldsymbol{p}_k(t)\Big(\sum_{j\in\mathbb{S},j\neq k}\mathbf{Q}_{kj}(t)\Big) \tag{4.3}$$

Here, $\boldsymbol{p}_k(t)$ is the probability of *being* in state $k \in \mathbb{S}$ at time t. The term $\sum_{i\in\mathbb{S},i\neq k}\mathbf{Q}_{kj}(t)$ captures the rates of transition from state k to all other states j, excluding self-transitions.

Continuous-Time Markov Chain

A Continuous-Time Markov Chain (CTMC) is deemed homogeneous because its hazard rates $\lambda(t;\theta)$ are constant over time, exhibiting the *memoryless property.* Formally, a CTMC is defined as follows.

Definition 10 (Continuous-Time Markov Chain)**.** *A* Continuous-Time Markov Chain (CTMC) *is defined by the tuple* $\mathcal{M} = \langle \mathbb{S}, \boldsymbol{p}^{(0)}, \mathbf{Q}_{ij}(\theta)\rangle$*, see* $\mathbb{S}$ *and* $\boldsymbol{p}^{(0)}$ *in Definition* 6*. Here:*

- $\mathbf{Q}_{ij}(\theta) : \mathbb{S}\times\mathbb{S} \to \mathbb{R}$ *represents the* time-independent transition rate matrix. *Unlike the inhomogeneous case, this matrix function, parametrised only by* θ*, is constant over time.*
- *Since the transition rates* $\lambda(t;\theta)$ *are constant,* $\mathbf{Q}_{ij}$ *does not depend on time but may still be parametrised by* θ *which affect the transition rates.*

When T consists of discrete, equally spaced intervals, we transition to Discrete-Time Markov Chains, which are discussed below.

Discrete-Time Markov Chain

When discretising t into discrete, equally spaced intervals of length Δt, denoted as n, we transition from continuous to Discrete-Time Markov Chains (DTMCs). A connection to CTMCs can be established through the *matrix of exponents*:

$$\mathbf{P}_{ij}(t) = \exp(\mathbf{Q}_{ij}t) \tag{4.4}$$

Definition 11 (Discrete-Time Markov Chain)**.** *A Discrete-Time Markov Chain (DTMC) is defined by the tuple $\mathcal{M} = \langle \mathbb{S}, \boldsymbol{p}^{(0)}, \mathbf{P}_{ij} \rangle$; see $\mathbb{S}$, $\boldsymbol{p}^{(0)}$, and $\mathbf{P}_{ij}$ in Definition 6. Here:*

- $\mathbf{P}_{ij} : \mathbb{S} \times \mathbb{S} \to [0, 1]$ *represents the* transition probability matrix. *Each entry p_{ij} of the matrix $\mathbf{P}_{ij}$ specifies the probability of transitioning from state i to state j over the time step Δt, where $i, j \in \mathbb{S}$.*
- *The matrix $\mathbf{P}_{ij}$ is time-invariant, characteristic of a time-homogeneous Markov process. The uniform time step Δt ensures it applies to transitions at regular intervals, simplifying temporal modelling.*

This form is the simplest and most common type of Markov chains, where the state probabilities can be calculated with the *Chapman-Kolmogorov* equation:

$$\boldsymbol{p}^{(n)} = \boldsymbol{p}^{(0)} \mathbf{P}_{ij}^{n} \tag{4.5}$$

Here, $\boldsymbol{p}^{(n)}$ represents the state probability distribution at the nth step, and $\mathbf{P}_{ij}^{n}$ is the n-th power of the transition probability matrix.

II.4.2 Case studies in sewer networks: overview

Validating degradation models through real-world case studies is a common practice. Depending on the model, different types of data are collected from sewer networks. For instance, Machine Learning models—used e.g., for anomaly detection or damage classification—often utilise datasets containing *images*[‡] or *videos*[§]. Examples include Haurum and Moeslund, 2021.

We are interested in condition data collected through inspections using Closed Circuit Television (CCTV), which include reports of various damage types and their classification with a *severity* index, following guidelines such as EN 13508:1. Table II.1 presents a (non-exhaustive) overview of research utilising similar data to develop models for condition assessment and degradation modelling. For details on the datasets, such as construction year and population age, we recommend checking the sources.

Table II.1 presents the reference, city, and country of each case study; the total sewer network length and the portion analysed (e.g., after data cleaning); the number of pipes studied (considered as the *population* of pipes); and the availability of the data. N.A. indicates that the reference does not explicitly mention data availability, while U.R. denotes data available upon request. From Table II.1, we conclude that, to our knowledge, the only publicly available dataset for degradation assessment is that provided by Jimenez-Roa, Heskes, Tinga, et al., 2022, which is the result of this dissertation.

[‡] https://paperswithcode.com/dataset/sewer-ml

[§] https://videopipe.github.io/cctvpipe/index.html

Table II.1: Case studies on sewer main degradation modelling, detailing reference, city, country, network length (study/total km), number of pipes for model calibration, and dataset availability: Not Available (N.A.), Upon Request (U.P.).

Reference	City	Country	Len. (Km)	Pipes	Avail.
Micevski, Kuczera, and Coombes, 2002	Newcastle	Australia	17/380	497	N.A.
Baik, Jeong, and Abraham, 2006	San Diego	U.S.	90/4,800	-	N.A.
Le Gat, 2008	Dresden	Germany	287/870	21,966	N.A.
Dirksen and Clemens, 2008	-	Netherlands	95/-	-	N.A.
Lubini and Fuamba, 2011	St-Hyacinthe & Verdun	Canada	-	459	N.A.
Scheidegger, Hug, Rieckermann, et al., 2011††	-	-	-	-	N.A.
Duchesne, Beardsell, Villeneuve, et al., 2013	Quebec	Canada	936/5,333	15,122	N.A.
Egger, Scheidegger, Reichert, et al., 2013††	-	-	-	2,000	N.A.
Rokstad and Ugarelli, 2015	Oslo	Norway	499/1,848	12,003	N.A.
Caradot, Sonnenberg, Kropp, et al., 2017	Braunschweig	Germany	1,027/1,300	35,826	N.A.
Caradot, Riechel, Fesneau, et al., 2018	Berlin	Germany	4,825/9,710	102,258	N.A.
Hernández, Caradot, Sonnenberg, et al., 2021	Medellín	Colombia	537/5,213*	17,298	N.A.
Hernández, Caradot, Sonnenberg, et al., 2021	Bogotá	Colombia	432/9,391*	8,553	N.A.
Lin, Yuan, and Tovilla, 2019	-	Canada	-	-	N.A.
Tran, Lokuge, Karunasena, et al., 2022	-	Australia	-	-	N.A.
Khaleghian and Shan, 2023	†	U.S.	-	-	U.R.
Jimenez-Roa, Heskes, Tinga, et al., 2022	Breda	Netherlands	1,045/2,169	25,594	Yes

† Four states from the U.S.: Texas, Indiana, California, and Pennsylvania.
†† Synthetic data generated with NetCoS tool.
* Not explicitly mentioned in the paper, but inferred from the information.

For completeness, other available datasets exist[e][f][g], but no related publications were found at the time of this thesis's publication. In the next section we provide details on the dataset, which is used in Chapters 5 and 6 of this dissertation.

II.4.3 Breda's case study

The raw dataset for the case study includes 59,183 sewer mains (approximately 2,169 km) and 30,661 inspections. After cleaning, which involved removing uninspected pipes, those built before 1900, and those with missing or erroneous data, the dataset was reduced to 25,594 sewer mains (approximately 1,045 km) and 29,926 inspections.

Histograms for all categorical and numerical variables in the dataset are displayed in

[e] https://hub.arcgis.com/maps/09bb004af3cf4c65b7ea73f81f1fe2bc/about

[f] https://hub.arcgis.com/datasets/lahub::sewer-pipes/about

[g] https://sitepx.freensb.cloudns.org/https/catalogue.data.govt.nz/dataset/wastewater-network-wastewater-points2

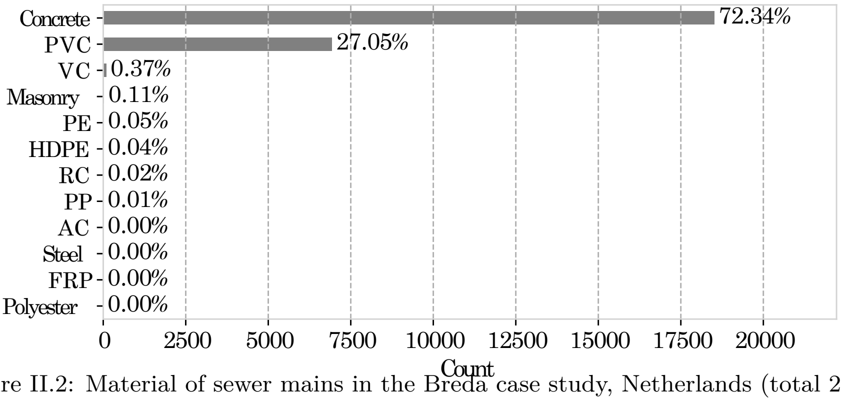


Figure II.2: Material of sewer mains in the Breda case study, Netherlands (total 25,594 pipes). High-Density Polyethylene (HDPE), Polyethylene (PE), Polypropylene (PP), Reinforced Concrete (RC), Fiberglass Reinforced Plastic (FRP), and Asbestos Cement (AC).

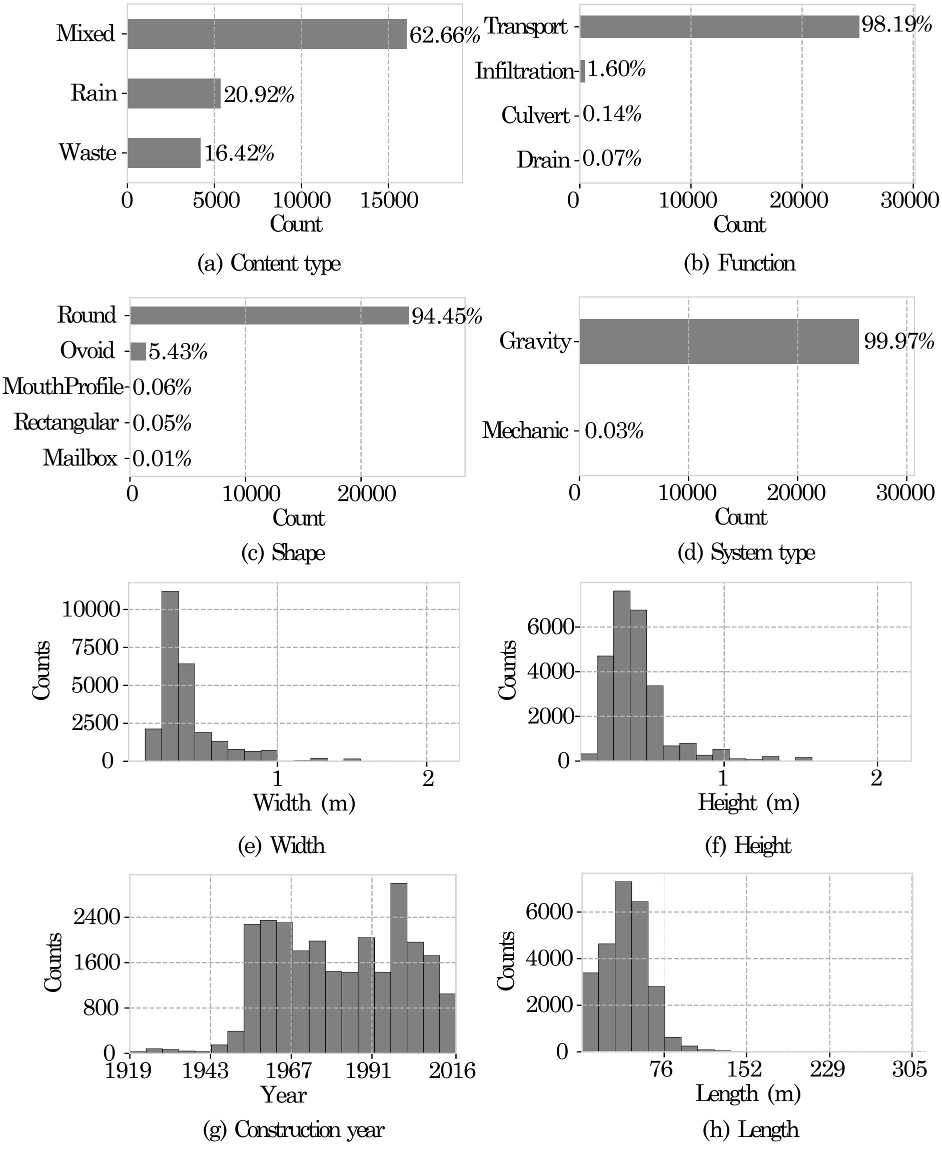
Figures II.2 and II.3. Figure II.2 shows the sewer main materials. Figures II.3.(a)-(d) present the categorical variables: sewer main content, function, shape, and system type, while Figures II.3.(e)-(h) depict the numerical variables: pipe width, height, construction year, and length.

Regarding the *categorical* variables, Figure II.2 shows that most pipes are made of *concrete* (72%) and *PVC* (27%). Figure II.3(a) indicates content types: Mixed (63%), Rain (21%), and Waste (16%). Figure II.3(b) reveals that 98% are used for *transport*. Figure II.3(c) shows that most shapes are *rounded* (94%) with a small percentage *ovoid* (5%). Finally, Figure II.3(d) indicates almost all sewer mains operate by *gravity*.

For the *numerical* variables, Figure II.3(e) shows that most pipes are less than 2 meters in width and height, as indicated in Figure II.3(f). Figure II.3(g) indicates that the pipes were built between 1919 and 2016, with the majority constructed around the 1950s. Figure II.3(h) shows that most pipes are up to 75 meters long, with the majority around 40 meters. Figure II.4 presents a visualisation of the sewer network in Breda, The Netherlands.

Through visual inspection conducted via CCTV at various sections along the sewer main length, and based on European standards EN 13508:1; EN 13508:2, the damage codes and severities—ranging from 1 to 5—present in the sewer mains (if any) are determined. As mentioned, a total of 29,926 inspections were registered for 25,594 sewer mains, with several conditions identified and recorded along the sewer main length per inspection. Table II.2 reports the counts per damage code and severity level in the inspection dataset from 1993 to 2016, where the total corresponds to the cumulative instances per damage code across all inspections.

Figure II.5(a) presents a histogram of inspections over the years, showing an increasing trend. Figure II.5(b) illustrates the maximum number of inspections



(a) Content type

(b) Function

(c) Shape

(d) System type

(e) Width

(f) Height

(g) Construction year

(h) Length

Figure II.3: Categorical and numerical variables in the Breda case study, Netherlands (total 25,594 pipes). (a)-(d) categorical variables; (e)-(h) numerical variables.

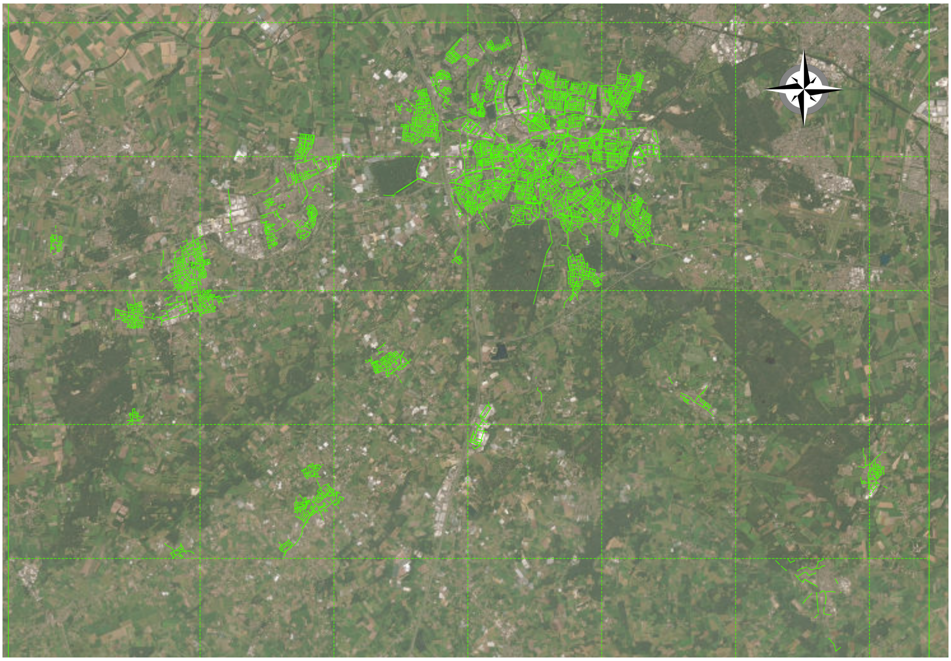


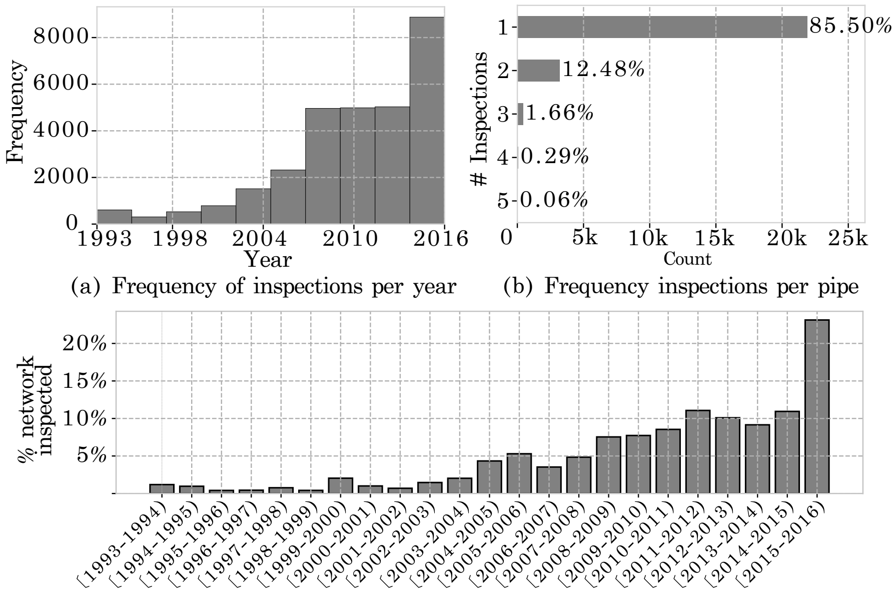
Figure II.4: Breda's sewer network in the Netherlands (*View of Breda* 2024) comprises 25,594 pipes, totalling approximately 1,045 km in length.

per pipe, with approximately 85% inspected only once, 12% twice, and less than 3% three or more times. Figure II.5(c) illustrates the percentage of the network inspected over the years. From approximately 2010 onwards, about 10% of the network is inspected annually. Details on the design and execution of the inspection campaigns are unavailable and beyond the scope of this case study.

II.4.4 Deterioration modelling in sewer mains using Markov Chains

Cherqui, Clemens-Meyer, Tscheikner-Gratl, et al., 2024 indicates that "*for long-term assessments on relatively large cohorts of elements (i.e., at the scale of an entire catchment or urban area), statistical/data-driven models based on condition class data seem to present a usable tool*". As mentioned in Section 1.2.3, various Markov chains have been utilised to model the stochastic deterioration of sewer mains. This section provides a summary and overview of some models discussed in the literature.

Inspection data are typically collected via CCTV along various sections of sewer mains. According to European standards EN 13508:1; EN 13508:2, damage codes and severities—ranging from 1 to 5—are then assigned. The need to explicitly

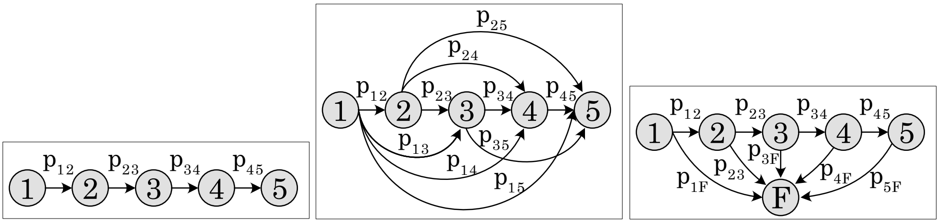


(a) Frequency of inspections per year (b) Frequency inspections per pipe

(c) Percentage of the sewer network inspected annually

Figure II.5: Histograms on inspection dataset (1,045 km of sewer network length).

model these severity levels as states in a Markov chain has led to the most common architectures found in the literature, as depicted in Figure II.6.



(a) Type "Single" (b) Type "Multi" (c) Type "Single+Failure"

Figure II.6: Types of Markov chain structures are discussed in this dissertation. For Discrete-Time Markov Chains, self-loops must be added to the states.

Since severities are ranked from 1 to 5, the state space can be defined as $\mathbb{S} = \langle 1, 2, 3, 4, 5, F \rangle$ with an index $k \in \mathbb{S}$. The models in Figure II.6 represent sequential states from *pristine* condition ($k = 1$) to *severe* deterioration ($k = 5$) or *functional failure* ($k = F$).

All structures shown in Figure II.6 correspond to *acyclic* Markov chains, meaning transitions can only occur from the current state to worse-case states ($p_{ij} = 0$ for $i > j$ with $i, j \in \mathbb{S}$), without the possibility of improving its condition without intervention (e.g., via repairs). Only the final state is absorbing, meaning $p_{|\mathbb{S}||\mathbb{S}|} = 1$.

Table II.2: Damage codes and severity levels available in the inspection dataset from 1993 to 2016 ($\star$ indicates damages codes used in this dissertation).

Description	Code	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$	Total
Infiltration*	BBF	16	61,694	8,078	2,818	324	72,930
Displaced joint	BAJ	20	1,545	379	239	36,867	39,050
Surface damage*	BAF	153	30,161	2,880	522	449	34,165
Water level	BDD	1,517	16,818	1,301	161	53	19,850
Defective connection	BAH	4	3,118	1,233	3,791	4,173	12,319
Protruding inlet	BAG	7,405	0	2,326	0	377	10,108
Crack*	BAB	47	606	0	3,581	5,848	10,082
Roots	BBA	0	6,219	1,517	506	697	8,939
Soil penetration	BBD	0	3,586	336	66	51	4,039
Attached deposits	BBB	1,877	1,536	291	36	43	3,783
Settled deposits	BBC	661	1,926	503	98	102	3,290
Intruding sealant	BAI	928	514	1,035	62	84	2,623
Porous pipe	BAN	0	0	0	0	1,638	1,638
Curvature in sewer	BCC	0	0	0	0	1,260	1,260
Soil exposed	BAO	0	0	0	0	810	810
Other obstacles	BBE	96	365	129	30	60	680
Visible hollow space	BAP	0	0	0	0	480	480
Deformation	BAA	6	219	52	19	16	312
Fracture/collapse*	BAC	1	131	0	57	27	216
Defective repair	BAL	3	33	141	0	10	187
Defective lining	BAK	0	1	14	125	25	165
Missing masonry mortar	BAE	0	9	1	2	1	13
Defective masonry	BAD	2	5	0	0	0	7
Exfiltration	BBG	0	0	0	0	4	4

The Chain "Single" (Figure II.6(a)) permits transitions only between consecutive states from better to worse-case (i.e., $0 \leq p_{ij} \leq 1$ for all i and $j = i + 1$, with $p_{ij} = 0$ for $j > i + 1$). The Chain "Multi" (Figure II.6(b)) allows transitions between both consecutive and non-consecutive states from better to worse-case (i.e., $0 \leq p_{ij} \leq 1$ for all $i \leq j$).

As severity level $k = 5$ is not a functional failure, the Chain "Single+Failure" (Figure II.6(c)) extends the Chain "Single" by adding an additional state, resulting in $\mathbb{S} = \langle 1, 2, 3, 4, 5, F \rangle$. This chain permits transitions between consecutive states, as in the Chain "Single", and from any state to the *functional failure* state (F). Information related to this state, such as fracture/collapse (code BAC), is detailed in Table II.2. This Markov chain structure was proposed in this thesis and is further discussed in Chapter 6.

Table II.3 lists references specifying the type of Markov structure ("Single" or "Multi") and whether the Markov chain is inhomogeneous (Inhom.) or homogeneous (Hom.).

Table II.3: Markov chain types used to model deterioration in sewer mains. "Single" and "Multi" are depicted in Figure II.6; Inhom. and Hom. indicate inhomogeneous or homogeneous-time Markov chains.

Reference	Single	Multi	Inhom.	Hom.
Kleiner, 2001	✓		✓	
Micevski, Kuczera, and Coombes, 2002		✓		✓
Baik, Jeong, and Abraham, 2006		✓		✓
Le Gat, 2008	✓		✓	
Dirksen and Clemens, 2008	✓			✓
Ana and Bauwens, 2010	✓	✓	✓	
Lubini and Fuamba, 2011		✓	✓	
Scheidegger, Hug, Rieckermann, et al., 2011	✓	✓	✓	✓
Egger, Scheidegger, Reichert, et al., 2013	✓		✓	
Duchesne, Beardsell, Villeneuve, et al., 2013	✓		✓	
Timashev and Bushinskaya, 2015	✓		✓	
Rokstad and Ugarelli, 2015	✓		✓	
Caradot, Sonnenberg, Kropp, et al., 2017	✓		✓	
Caradot, Riechel, Fesneau, et al., 2018	✓		✓	
Lin, Yuan, and Tovilla, 2019	✓			✓
Hawari, Alkadour, Elmasry, et al., 2020	✓	✓	✓	✓
Tran, Lokuge, Karunasena, et al., 2022		✓		✓
Jimenez-Roa, Heskes, Tinga, et al., 2022	✓	✓		✓
Jimenez-Roa, Tinga, Heskes, et al., 2024†			✓	✓

† Reference uses the Markov chain type "Single+Failure" in Figure II.6(c).

II.5 References

Ana, E. and W. Bauwens (2010). "Modeling the structural deterioration of urban drainage pipes: the state-of-the-art in statistical methods". In: *Urban Water Journal* 7.1, pp. 47–59. DOI: 0.1080/15730620903447597.

Baik, H.-S., H. S. Jeong, and D. M. Abraham (2006). "Estimating Transition Probabilities in Markov Chain-Based Deterioration Models for Management of Wastewater Systems". In: *Journal of Water Resources Planning and Management* 132.1, pp. 15–24. DOI: 10.1061/(ASCE)0733-9496(2006)132:1(15).

Barraud, S., E. Bosco, F. Clemens-Meyer, G. Fernandes, Y. Le Gat, C. de Haan, R. Luimes, K. Makris, F. Rooyackers, I. Scheperboer, J. Skjelde, and A. Suiker (June 2024). "Deterioration processes and modelling in urban drainage systems". In: *Asset Management of Urban Drainage Systems: If anything exciting happens, we've done it wrong!* IWA Publishing. ISBN: 9781789063059. DOI: 10.2166/9781789063059_0131.

Brémaud, P. (2020). "Non-homogeneous Markov Chains". In: *Markov Chains: Gibbs Fields, Monte Carlo Simulation and Queues*. Cham: Springer International Publishing, pp. 399–422. ISBN: 978-3-030-45982-6. DOI: 10.1007/978-3-030-45982-6_12.

Caradot, N., M. Riechel, M. Fesneau, N. Hernandez, A. Torres, H. Sonnenberg, E. Eckert, N. Lengemann, J. Waschnewski, and P. Rouault (2018). "Practical benchmarking of statistical and machine learning models for predicting the condition of sewer pipes in Berlin, Germany". In: *Journal of Hydroinformatics* 20.5, pp. 1131–1147. DOI: 10.2166/hydro.2018.217.

Caradot, N., P. Rouault, F. Clemens, and F. Cherqui (2018). “Evaluation of uncertainties in sewer condition assessment”. In: *Structure and Infrastructure Engineering* 14.2, pp. 264–273. DOI: 10.1080/15732479.2017.1356858.

Caradot, N., H. Sonnenberg, I. Kropp, A. Ringe, S. Denhez, A. Hartmann, and P. Rouault (2017). “The relevance of sewer deterioration modelling to support asset management strategies”. In: *Urban Water Journal* 14.10, pp. 1007–1015. DOI: 10.1080/1573062X.2017.1325497.

Cherqui, F., F. Clemens-Meyer, F. Tscheikner-Gratl, and B. van Duin (June 2024). *Asset Management of Urban Drainage Systems: If anything exciting happens, we’ve done it wrong!* IWA Publishing. ISBN: 9781789063059. DOI: 10.2166/9781789063059.

Colombo, D., D. T. Abreu, and M. R. Martins (2021). “Application of Markovian models in reliability and availability analysis: advanced topics”. In: *Safety and Reliability Modeling and its Applications*. Ed. by H. Pham and M. Ram. Advances in Reliability Science. Elsevier, pp. 91–160. ISBN: 978-0-12-823323-8. DOI: 10.1016/B978-0-12-823323-8.00015-5.

Dirksen, J. and F. Clemens (2008). “Probabilistic modeling of sewer deterioration using inspection data”. In: *Water Science and Technology* 57.10, pp. 1635–1641. DOI: 10.2166/wst.2008.308.

Duchesne, S., G. Beardsell, J.-P. Villeneuve, B. Toumbou, and K. Bouchard (2013). “A survival analysis model for sewer pipe structural deterioration”. In: *Computer-Aided Civil and Infrastructure Engineering* 28.2, pp. 146–160. DOI: 10.1111/j.1467-8667.2012.00773.x.

Egger, C., A. Scheidegger, P. Reichert, and M. Maurer (2013). “Sewer deterioration modeling with condition data lacking historical records”. In: *Water Research* 47.17, pp. 6762–6779. ISSN: 0043-1354. DOI: 10.1016/j.watres.2013.09.010.

El Morer, F., S. Wittek, and A. Rausch (2024). “Assessment of the suitability of degradation models for the planning of CCTV inspections of sewer pipes”. In: *Urban Water Journal* 21.2, pp. 190–203. DOI: 10.1080/1573062X.2023.2282126.

EN 13508:1 -Investigation and assessment of drain and sewer systems outside buildings - Part 1: General Requirements (Oct. 2012). Standard. Avenue Marnix 17, B-1000 Brussels: European Committee for Standardization (CEN).

EN 13508:2 -Investigation and assessment of drain and sewer systems outside buildings - Part 2: Visual inspection coding system (May 2011). Standard. Avenue Marnix 17, B-1000 Brussels: European Committee for Standardization (CEN).

Haurum, J. B. and T. B. Moeslund (2021). “Sewer-ML: A Multi-Label Sewer Defect Classification Dataset and Benchmark”. In: *CoRR* abs/2103.10895, pp. 13456–13467. URL: https://arxiv.org/abs/2103.10895.

Hawari, A., F. Alkadour, M. Elmasry, and T. Zayed (2020). “A state of the art review on condition assessment models developed for sewer pipelines”. In: *Engineering Applications of Artificial Intelligence* 93, p. 103721. DOI: 10.1016/j.engappai.2020.103721.

Hernández, N., N. Caradot, H. Sonnenberg, P. Rouault, and A. Torres (2021). “Optimizing SVM models as predicting tools for sewer pipes conditions in the two main cities in Colombia for different sewer asset management purposes”. In: *Structure and Infrastructure Engineering* 17.2, pp. 156–169. DOI: 10.1080/15732479.2020.1733029.

Jimenez-Roa, L. A., T. Heskes, T. Tinga, H. J. A. Molegraaf, and M. Stoelinga (2022). “Deterioration modeling of sewer pipes via discrete-time Markov chains: A large-scale case study in the Netherlands”. In: *Proceedings of the 32nd European Safety and Reliability Conference, ESREL 2022 - Understanding and Managing Risk and Reliability for a Sustainable Future*, pp. 1299–1306. DOI: 10.3850/978-981-18-5183-4_R22-13-482-cd.

Jimenez-Roa, L. A., T. Tinga, T. Heskes, and M. Stoelinga (2024). “Comparing Homogeneous and Inhomogeneous Time Markov Chains for Modelling Degradation in Sewer

Pipe Networks". In: *Proceedings of the European Safety and Reliability Conference (ESREL 2024)*. DOI: 10.48550/arXiv.2407.12557.

Kantidakis, G., H. Putter, S. Litière, and M. Fiocco (2023). "Statistical models versus machine learning for competing risks: development and validation of prognostic models". In: *BMC Medical Research Methodology* 23.1, p. 51. DOI: 10.1186/s12874-023-01866-z.

Khaleghian, H. and Y. Shan (2023). "Developing a Data Quality Evaluation Framework for Sewer Inspection Data". In: *Water* 15.11. ISSN: 2073-4441. DOI: 10.3390/w15112043.

Kleiner, Y. (2001). "Optimal scheduling of rehabilitation and inspection/condition assessment in large buried pipes". In: *NRCC-44487, 4th International Conference on Water Pipeline Systems—Managing Pipeline Assets in an Evolving Market*, pp. 181–197.

Le Gat, Y. (2008). "Modelling the deterioration process of drainage pipelines". In: *Urban Water Journal* 5.2, pp. 97–106. DOI: 10.1080/15730620801939398.

Lin, P., X.-X. Yuan, and E. Tovilla (2019). "Integrative modeling of performance deterioration and maintenance effectiveness for infrastructure assets with missing condition data". In: *Computer-Aided Civil and Infrastructure Engineering* 34.8, pp. 677–695. DOI: 10.1111/mice.12452.

Lubini, A. T. and M. Fuamba (2011). "Modeling of the deterioration timeline of sewer systems". In: *Canadian Journal of Civil Engineering* 38.12, pp. 1381–1390. DOI: 10.1139/l11-103.

Malek Mohammadi, M., M. Najafi, V. Kaushal, R. Serajiantehrani, N. Salehabadi, and T. Ashoori (2019). "Sewer pipes condition prediction models: A state-of-the-art review". In: *Infrastructures* 4.4, p. 64. DOI: 10.3390/infrastructures4040064.

Marc Ribalta Ramon Bejar, C. M. and E. Rubión (2023). "Machine learning solutions in sewer systems: a bibliometric analysis". In: *Urban Water Journal* 20.1, pp. 1–14. DOI: 10.1080/1573062X.2022.2138460.

Micevski, T., G. Kuczera, and P. Coombes (2002). "Markov Model for Storm Water Pipe Deterioration". In: *Journal of Infrastructure Systems* 8.2, pp. 49–56. DOI: 10.1061/(ASCE)1076-0342(2002)8:2(49).

Nguyen, L. V. and R. Seidu (2022). "Application of Regression-Based Machine Learning Algorithms in Sewer Condition Assessment for Ålesund City, Norway". In: *Water* 14.24, p. 3993. DOI: 10.3390/w14243993.

Noshahri, H., L. L. olde Scholtenhuis, A. G. Doree, and E. C. Dertien (2021). "Linking sewer condition assessment methods to asset managers' data-needs". In: *Automation in Construction* 131, p. 103878. ISSN: 0926-5805. DOI: 10.1016/j.autcon.2021.103878.

Rokstad, M. M. and R. M. Ugarelli (Apr. 2015). "Evaluating the role of deterioration models for condition assessment of sewers". In: *Journal of Hydroinformatics* 17.5, pp. 789–804. ISSN: 1464-7141. DOI: 10.2166/hydro.2015.122.

Saddiqi, M. M., W. Zhao, S. Cotterill, and R. K. Dereli (2023). "Smart management of combined sewer overflows: From an ancient technology to artificial intelligence". In: *Wiley Interdisciplinary Reviews: Water* 10.3, e1635. DOI: 10.1002/wat2.1635.

Scheidegger, A., T. Hug, J. Rieckermann, and M. Maurer (2011). "Network condition simulator for benchmarking sewer deterioration models". In: *Water Research* 45.16, pp. 4983–4994. ISSN: 0043-1354. DOI: 10.1016/j.watres.2011.07.008.

Stewart, W. J. (2009). *Probability, Markov chains, queues, and simulation: the mathematical basis of performance modeling.* Princeton University Press. Chap. 9. Discrete- and Continuous-Time Markov chains, pp. 193–284. ISBN: 9780691140629. DOI: 10.2307/j.ctvcm4gtc.12.

Timashev, S. and A. Bushinskaya (2015). "Markov approach to early diagnostics, reliability assessment, residual life and optimal maintenance of pipeline systems". In: *Structural Safety* 56, pp. 68–79. ISSN: 0167-4730. DOI: 10.1016/j.strusafe.2015.05.006.

Tran, H., W. Lokuge, W. Karunasena, and S. Setunge (2022). “Markov-based deterioration prediction and asset management of floodway structures”. In: *Sustainable and Resilient Infrastructure* 7.6, pp. 789–802. DOI: 10.1080/23789689.2022.2067950.

Tran, H., W. Lokuge, S. Setunge, and W. Karunasena (2022). “Network deterioration prediction for reinforced concrete pipe and box culverts using Markov model: Case study”. In: *Journal of Performance of Constructed Facilities* 36.6, p. 04022047. DOI: 10.1061/(ASCE)CF.1943-5509.000176.

Tran, H., S. Setunge, and L. Shi (2021). “Markov Chain–Based Inspection and Maintenance Model for Stormwater Pipes”. In: *Journal of Water Resources Planning and Management* 147.11, p. 04021077. DOI: 10.1061/(ASCE)WR.1943-5452.0001469.

View of Breda (Aug. 2024). Google Earth. Available from: https://earth.google.com/web/search/breda/@51.52709667,4.79307765,14.36430885a,46125.41637033d,35y,0h,0t,0r, Accessed on 2024-08-22.

Zeng, X., Z. Wang, H. Wang, S. Zhu, and S. Chen (2023). “Progress in Drainage Pipeline Condition Assessment and Deterioration Prediction Models”. In: *Sustainability* 15.4, p. 3849. DOI: 10.3390/su15043849.

Chapter 5

Deterioration Modelling of Sewer Pipes via Discrete-Time Markov Chains: A Large-Scale Case Study in the Netherlands

Paper published at **L. A. Jimenez-Roa**, T. Heskes, T. Tinga, H. J. A. Molegraaf and M. Stoelinga, *"Deterioration Modelling of Sewer Pipes via Discrete-Time Markov Chains: A Large-Scale Case Study in the Netherlands",* in 32nd European Safety and Reliability Conference, ESREL 2022: Understanding and Managing Risk and Reliability for a Sustainable Future, pp. 1299-1306. 2022, doi:10.3850/978-981-18-5183-4_R22-13-482-cd.

Abstract

Sewer network systems are a crucial component of civil infrastructure. To achieve an optimal balance between maintenance costs and system performance, reliable sewer main deterioration models are crucial. In this chapter, we present a large-scale case study in the city of Breda, Netherlands.

Our dataset includes information on sewer mains constructed since the 1920s and contains various covariates. We focus on three types of damage: infiltrations, surface damage, and cracks, each with an associated severity index ranging from 1 to 5. To account for the characteristics of sewer mains, we defined six cohorts of interest.

Two types of Discrete-Time Markov Chains (DTMCs), which we have termed Chain 'Multi' and 'Single' (with Chain 'Multi' containing additional transitions compared to Chain 'Single'), are commonly employed to model sewer main deterioration at the pipeline level. We aim to evaluate which model is more suitable for our case study. To calibrate the DTMCs, we define an optimisation process using Sequential

Least-Squares Programming to identify the DTMC parameter that best minimises the root mean weighted square error.

Our results indicate that, for our case study, there is no significant difference between Chain 'Multi' and 'Single'. However, the latter has fewer parameters and can be trained more easily. Our DTMCs are useful for comparing the cohorts via expected values. For instance, concrete pipes carrying mixed and waste content reach severe levels of surface damage more rapidly compared to concrete pipes carrying rainwater, which is a phenomenon typically observed in practice.

5.1 Introduction

Sewer network systems are an important part of the civil infrastructure required to achieve an adequate level of social and economic welfare. The management of these systems has become increasingly challenging due to the need to cope with limited budgets, environmental changes, uncertainty about network deterioration, and a lack of rigorous deterioration analysis. This often leads to conservative approaches that result in the early replacement of sewer mains.

Thus, aiming at finding a good trade-off between maintenance costs and system performance, robust and reliable *sewer main deterioration models* are needed to prioritise pipes at high risk of failure for proactive maintenance, support decision making, and strategic rehabilitation planning (Scheidegger, Hug, Rieckermann, et al., 2011; Egger, Scheidegger, Reichert, et al., 2013).

Moreover, there is a need in the research community for sharing existing case studies aiming at increasing the evidence on sewer main deterioration models (Tscheikner-Gratl, Caradot, Cherqui, et al., 2019).

Concerning sewer main deterioration models, three types can be identified: those based on physics, artificial intelligence, and statistics. A detailed review of the different types of models used to predict the deterioration of sewer networks is presented in Hawari, Alkadour, Elmasry, et al., 2020.

Physics-based models may be too complex to capture the complete deterioration behaviour, and artificial intelligence models require high computational costs and demands of data (Ana and Bauwens, 2010). Thus, given the limitations of these types of models, we centre our attention on statistical methods.

In particular, we are interested in Markov chain models, since they proved to be among the most reliable and widely used approaches to simulate sewer main deterioration (Kobayashi, Kaito, and Lethanh, 2012; Tscheikner-Gratl, Caradot, Cherqui, et al., 2019), and enable the modelling of sequential events, such as sewer mains deterioration (Ana and Bauwens, 2010).

Several types of Markov chains (MC) have been implemented for the modelling of sewer networks, examples are discrete-time MC (Micevski, Kuczera, and Coombes, 2002; Baik, Jeong, and Abraham, 2006), continuous-time MC (Lin, Yuan, and Tovilla, 2019), non-Homogeneous MC (Le Gat, 2008), hidden-MC (Kobayashi, Kaito, and Lethanh, 2012), semi-MC (Scheidegger, Hug, Rieckermann, et al., 2011).

As a first step, we decided to use discrete-time Markov chains (DTMCs) because these proved to be a straightforward approach to model deterioration patterns associated with sewer networks. Moreover, we are interested in two typical types of DTMCs (see Figure 5.1) that we call Chains *'Multi'* and *'Single'*, where the former contains additional transitions compared to the latter, and we are interested in evaluating which of them best suits our case study.

Similar to Caradot, Riechel, Fesneau, et al., 2018, our goal with these DTMCs is to predict the probability for a pipe to be in a severity class for a certain type of damage, based on the pipe age and a set of numerical or categorical variables (called covariates) organised in 6 cohorts (i.e., group of pipes with the same characteristics) of interest.

Our research questions are both application-oriented (RQ1) and methodological (RQ2): **RQ1** how do the predefined cohorts compare in terms of deterioration rate? **RQ2** how can DTMCs assist in getting this insight, and how do Chains 'Multi' and 'Single' compare in terms of performance?

The experimental evaluation is based on a large-scale case study in the city of Breda in the Netherlands, where we have information on sewer mains built since the 1920s which contains information on different covariates. We focus on three typical types of sewer mains damages namely *infiltration*, *surface damage*, and *cracks*. Each damage has an associated severity index ranging from 1 to 5.

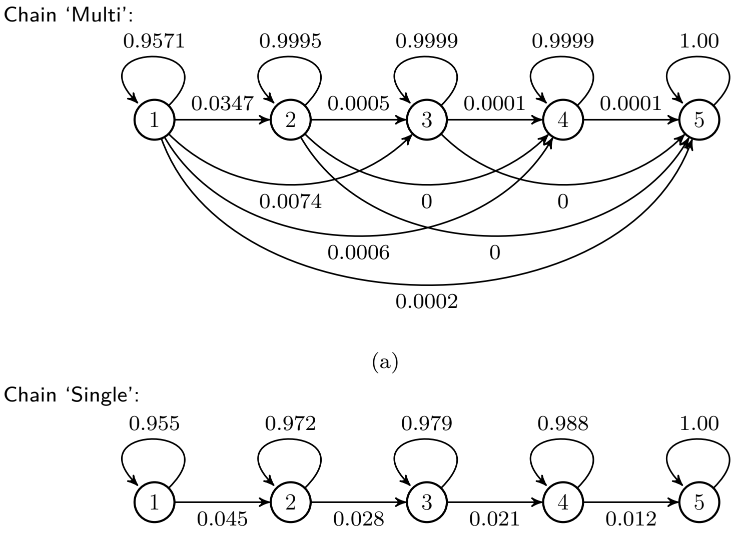
Our main contribution is to demonstrate the application of existing deterioration models in a large-scale case study. The present work is a valuable step toward the development of an evidence-based asset management framework. The scripts and comparative figures can be found at zenodo.org/record/6535853.

The structure of this chapter is as follows. Section 5.2 provides the theoretical background on DTMCs. Section 5.3 presents our methodology. In Section 5.4 we preset the case study, the experimental evaluation and the main results. We discuss and conclude in Section 5.5.

5.2 Homogeneous discrete-time Markov chain

Refer to Definition 11 on page 99 for the formal definition of Discrete-Time Markov Chains (DTMCs). Our DTMCs are defined by five states, $\mathbb{S} = \langle 1, 2, 3, 4, 5 \rangle$, with the index of the state denoted as $k \in \mathbb{S}$.

The values p_{ij} with $i, j \in \mathbb{S}$ in Figure 5.1 are discussed in Section 5.4.2. The *initial probability distribution* $\boldsymbol{p}^{(0)}$ indicates the probability that the sewer main is in the



(a)

(b)

Figure 5.1: DTMCs modelling the deterioration of sewer mains considering five deterioration states and no repairs. (a) Chain 'Multi'; (b) Chain 'Single'.

state k at the step $n = 0$,

$$\boldsymbol{p}^{(0)} = \langle \boldsymbol{p}_1^{(0)}, \boldsymbol{p}_2^{(0)}, \boldsymbol{p}_3^{(0)}, \boldsymbol{p}_4^{(0)}, \boldsymbol{p}_5^{(0)} \rangle^T \tag{5.1}$$

where $\sum_{k \in \mathbb{S}} \boldsymbol{p}_k^{(0)} = 1$ at any step n. To calculate the state probabilities associated with the n-step $\boldsymbol{p}^{(n)}$, we apply the Chapman-Kolmogorov equation in Eq. 4.5 (page 99). From here, the n-step transition probability matrix $\mathbf{P}_{ij}^{(n)}$ is obtained by multiplying the matrix $\mathbf{P}_{ij}$ by itself n times. If n is a positive real number instead of a natural number, one can compute the *fractional power of the matrix* using, for example, the `scipy.linalg` function in Python (Virtanen, Gommers, Oliphant, et al., 2020).

We compute the expected severity $\mathbb{E}^{(n)}$, as shown in Baik, Jeong, and Abraham, 2006, by multiplying the state probability distribution $\boldsymbol{p}^{(n)}$ at step n by the severity class vector $\psi \in \mathbb{S}$. In this context, 1 indicates pristine condition, and 5 represents the maximal severity that can be assigned to a type of damage. Then $\mathbb{E}^{(n)}$ is computed as follows:

$$\mathbb{E}^{(n)} = \boldsymbol{p}^{(n)} \psi \tag{5.2}$$

We adopt the structures of Markov chains types "Multi" and "Single" as discussed in Section II.4.4, page 103. These are typical Markov chain structures used to model the deterioration of sewer mains via DTMCs (Ana and Bauwens, 2010).

Table 5.1: Cohorts of interest, fraction (%) of total inspected pipes (25,507).

Cohort name	Description	Fraction (%)
CMW	Material: Concrete & Content: Mixed and Waste	59.29
CR	Material: Concrete & Content: Rainwater	12.85
PMW	Material: PVC & Content: Mixed and Waste	18.88
PR	Material: PVC & Content: Rainwater	7.89
CdL	Material: Concrete & Width $<$ 500 mm	50.16
CdG	Material: Concrete & Width $\geq$ 500 mm	22.02

Figure 5.1 shows an example of these structures used in this chapter. The number on the arrows indicate the probability of moving from one state to another.

5.3 Methodology

Our goal is to describe the deterioration of sewer mains based on a historic set of inspection data. To achieve this we calibrate DTMCs that quantify the probability of a pipe (from the historic dataset) being in a condition class given the age of the pipe. These DTMCs represent cohorts of pipes, i.e., they are trained with data from pipes that share similar characteristics. An overview of the four steps we follow includes: **Step 1**, pre-processing the data (cleaning); **Step 2**, defining cohorts; **Step 3**, creating a discretised table per cohort, which serves as the input data to calibrate the DTMC; and **Step 4**, calibrating the DTMC. The details about each of the steps are provided in the following sections.

5.3.1 Data pre-processing

Our work is based on the case study detailed in Section 5.4.1. The dataset for each sewer main includes information on inspections, such as (i) unique inspection identifiers, (ii) inspection dates, (iii) damage sizes, (iv) damage codes, (v) damage classes (described by ψ), and (vi) the damage's relative position. Each damage code is associated with a damage class. For this chapter, we exclude data on pipes built before 1920 and those with missing or erroneous construction year information.

5.3.2 Definition of cohorts

To account for explanatory variables beyond pipe age in deterioration modelling, it is necessary to construct cohorts (groups of sewer mains with similar characteristics) and calibrate a DTMC for each cohort. Table 5.1 presents six cohorts of interest and the number of pipes with specific characteristics, expressed as a fraction of the total inspected pipes. However, a drawback of defining cohorts is the potential for small subsets (e.g., Cohort PR), which may not be statistically representative.

Table 5.2: Discretised table $\hat{\boldsymbol{p}}_k^{(\hat{n})}$ for cohort `CMW`, surface damage (`BAF`), with $\Delta t = 3$ years.

Count	PipeAge	Time	Step	$\hat{\boldsymbol{p}}_k^{(\hat{n})}$				
(c)	(years)	(t)	$(\hat{n})$	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$
832	[0,3)	1.5	0	0.95	0.03	0.01	0.01	0.00
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
2,339	[48,51)	49.5	16	0.35	0.50	0.12	0.02	0.01
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
64	[75,78)	76.5	25	0.44	0.20	0.28	0.05	0.03
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

5.3.3 Discretised table

We construct a *discretised table* for each cohort (Table 5.2) and damage code to serve as input for calibrating the DTMCs. The *state of a sewer pipe* is identified as the *maximum damage class found during an inspection* for the relevant damage codes. This conservative approach helps determine which pipes require near-term repair. To create Table 5.2, we define a time interval Δt, group pipes by age at the time of inspection, and count the pipes in each damage class per group, normalising by the total number of pipes in that group.

For instance, in Table 5.2, for Cohort `CMW`, damage code `BAF` (surface damage), and $\Delta t = 3$ years, there were $2'339$ pipes with $48 \leq \text{PipeAge} < 51$ years at the time of inspection. The *count* vector (c) shows the total number of pipes within a PipeAge interval, t is the mean value of the PipeAge interval, and $\hat{n}$ represents the discretisation step. For $\hat{n} = 16$, corresponding to the interval $48 \leq \text{PipeAge} < 51$ at $t = 49.5$ years, 35% of pipes were in State 1 $\left(\text{i.e., } \hat{\boldsymbol{p}}_{k=1}^{(\hat{n}=16)} = 0.35\right)$, and 50% in State 2 $\left(\text{i.e., } \hat{\boldsymbol{p}}_{k=2}^{(\hat{n}=16)} = 0.50\right)$. Thus, $\hat{\boldsymbol{p}}_k^{(\hat{n})}$ forms a $|\hat{n}| \times |\mathbb{S}|$ matrix representing the *ground truth for calibrating the DTMCs.* The sum of counts (c) gives the total number of pipes in the network, which varies across PipeAge intervals. We incorporate this variation by defining a *weight* vector in the calibration of the DTMCs.

We disregard right-*censoring* in our dataset, assuming that the sewer pipe has just reached the condition observed during the inspection, though it may have reached this state earlier.

5.3.4 Calibration of the Discrete-Time Markov Chain

To calibrate a DTMC, we optimise the parameters $\boldsymbol{p}^{(0)}$ (Eq. 5.1) and $\mathbf{P}_{ij}$ (Definition 11, p. 99) to minimise the *Root Mean Weighted Square Error* (Err) (Eq. 5.4). First, we normalise the counts (c) (Table 5.2) to compute a weight vector $\bar{w}$ as shown in Eq. 5.3:

$$\bar{w} = \frac{c}{\max(c)}, \tag{5.3}$$

Err is then calculated as the difference between the discretised table $\left(\hat{\boldsymbol{p}}_k^{(\hat{n})}\right)$ and the DTMC predictions using the Chapman-Kolmogorov equation (Eq. 4.5, p. 99) $\left(\bar{\boldsymbol{p}}_k^{(\hat{n})}\right)$,

$$Err = \sqrt{\frac{\sum_{\hat{n},k}\left[\left(\bar{\boldsymbol{p}}_k^{(\hat{n})} - \hat{\boldsymbol{p}}_k^{(\hat{n})}\right)^2 * \bar{w}_{\hat{n}}\right]}{|\hat{n}| \times |\mathbb{S}|}} \tag{5.4}$$

Minimisation of *Err* is performed using the *Sequential Least-Squares Programming* (SLSQP) algorithm from `Scipy` (Virtanen, Gommers, Oliphant, et al., 2020) with default parameters. All optimisation parameters are bounded in $[0, 1]$ and initialised as follows: (i) in $\bar{\boldsymbol{p}}^{(0)}$, $\bar{\boldsymbol{p}}_{k=1}^{(0)} = 1$ and $\bar{\boldsymbol{p}}_{k\neq 1}^{(0)} = 0$; (ii) $\mathbf{P}_{ij}$ is the identity matrix. We apply the constraints outlined in Definition 6 (page 96) for both Markov chains.

We calibrate the DTMCs by randomly selecting 50% of the available data per cohort using repeated half-sample bootstrap (Saigo, Shao, and Sitter, 2001). The calibration process outputs the $\boldsymbol{p}^{(0)}$ and $\mathbf{P}_{ij}$ with the smallest *Err* for a given Δt.

5.4 Experimental Evaluation

5.4.1 Case study

The detailed description of the case study is provided in Section II.4.3 on page 100. Based on these data, as recommended by domain experts, we focus on the damage codes (d.c.): infiltration (`BBF`), surface damage (`BAF`), and cracks (`BAB`), which were observed in 44%, 35%, and 18% of the inspections, respectively.

5.4.2 Results

We visually compare the cohorts and chains using Figure 5.2. To construct these figures, we train a thousand DTMCs using repeated half-sampled bootstrap (Section 5.3.4) to account for uncertainty. Figure 5.2 presents selected results, with the full analysis available on Zenodo[†].

Figures 5.2(a)-(d) show the probability of being in state k given PipeAge, cohort, chain, and damage code. Markers represent the discretised table $\hat{\boldsymbol{p}}_k^{(\hat{n})}$ (with $\Delta t = 3$ years), where size visualises counts. Dashed lines indicate a 95% confidence interval based on the projections of a thousand calibrated DTMCs, while the solid line shows the median value. Figures for comparing cohorts and chains are in Zenodo[†] under */comparing_cohorts* and */comparing_chains*.

[†] All comparative figures and scripts are available at zenodo.org/record/6535853

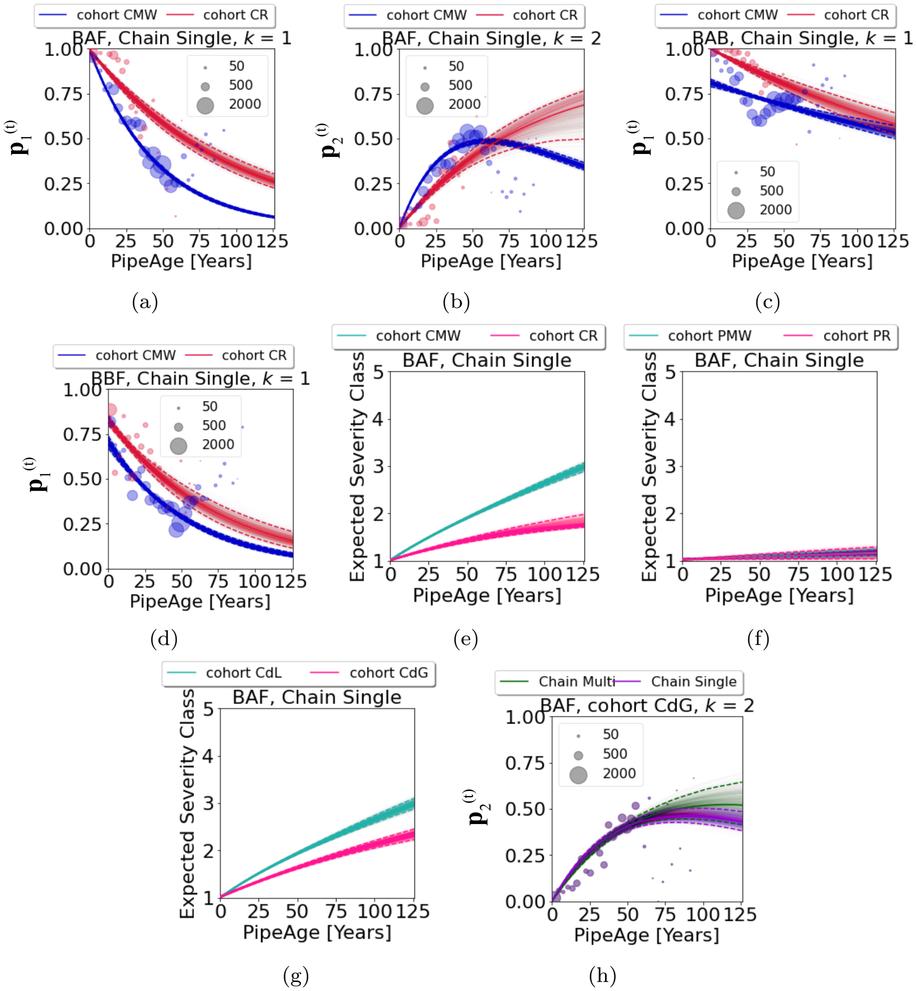


Figure 5.2: Comparisons between different cohorts and chains, for various degradation states and damage types. Find all the figures at zenodo.org/record/6535854.

Figures 5.2(e)-(g) display expectations computed using Eq. 5.2, corresponding to the expected severity class for a given damage class at a specific PipeAge. The dashed and solid lines represent the confidence interval and median value, respectively. These results facilitate cohort comparisons, available on Zenodo[†]: */comparing_expectations*.

Comparing Cohorts `CMW` and `CR`

Figures 5.2(a)-(b) indicate that older concrete pipes (25-75 years) carrying Mixed and Waste content (Cohort `CMW`) have a higher probability of reaching severe *surface damage* levels (`BAF`) compared to those carrying Rainwater (Cohort `CR`). Similarly, Figure 5.2(e) shows that the expected severity class for Cohort CMW is higher across all PipeAges, with a maximum expected class of $k = 3$ over 125 years.

For *cracks* (BAB), complex degradation patterns are observed, with DTMCs not fully capturing these changes (e.g., Cohort CMW in Figure 5.2(c)). It is also unlikely to find cracks in states $k = 2, 3, 4$[†].

For *infiltration* (BBF), Figure 5.2(d) shows an initial probability of roughly 25% for pipes in Cohorts CMW and CR to experience at least mild infiltrations (i.e., $k > 1$).

Comparing Cohorts PMW and PR

Figure 5.2(f) shows that Cohorts PMW (PVC pipes carrying Mixed and Waste content) and PR (PVC pipes carrying Rainwater) exhibit similar patterns under *surface damage* (BAF), with a maximum expected severity of $k < 2$ over 125 years. No significant differences were found between these cohorts for cracks and infiltration[†].

Comparing Cohorts CdL and CdG

Figure 5.2(g) compares Cohorts CdL (Concrete pipes, Width<500 mm) and CdG (Concrete pipes, Width≥500 mm). Narrow pipes show more severe *surface damage* (BAF) than wider ones. The same pattern is observed under *cracks* (BAB)[†]. For *infiltration* (BBF), wider pipes tend to reach severe states faster, though with considerable uncertainty[†].

Comparing Chains "Multi" and "Single"

When comparing Chains "Multi" and "Single", we generally observed no significant differences. However, for certain cases, such as Cohort CdG and surface damage (BAF) in Figure 5.2(h), Chain 'Single' transitions to more severe states faster than Chain "Multi" when PipeAge > 60 years (see $\mathbf{P}_{ij}$ in Figure 5.1, $\Delta t = 3$ years).

We hypothesise this occurs because Chain "Multi" may converge towards diagonal values in $\mathbf{P}_{ij}$ close to 1 (e.g., Figure 5.1(a), $p_{3,3} = p_{4,4} = 0.9999$), making these states nearly absorbing, a behaviour not seen in Chain "Single" (Figure 5.1(b)).

5.5 Discussion and Conclusions

We model sewer main deterioration in a large-scale case study in Breda using Discrete-Time Markov Chains (DTMCs). We describe a methodology to calibrate DTMCs and visually[†] compare deterioration patterns across cohorts (groups of sewer mains with similar characteristics) for three damage types: infiltration, surface damage, and cracks.

[†] All comparative figures and scripts are available at zenodo.org/record/6535853

Our DTMCs effectively project and estimate future deterioration states of sewer mains, enabling comparisons across cohorts, such as expected severity classes. For instance, we conclude that concrete pipes carrying Mixed and Waste content degrade faster than those carrying Rainwater, a phenomenon commonly observed in practice.

When comparing DTMC types, “Multi” and “Single” chains show similar performance. The “Single” chain is easier to calibrate due to fewer parameters, making it suitable for this study. However, the “Multi” chain requires a better implementation to avoid forming absorbing intermediate states.

Regarding limitations, we assume sewer main conditions are observed at inspection, which may not be accurate due to right-*censoring* (damage occurring before inspection). This assumption biases our DTMCs, predicting failures later than they actually occur. To address this, we plan to explore *multi-state survival models* for *interval-censored data* (Hout, 2016), which better account for censored data.

Additionally, we will improve DTMCs parameter inference by incorporating covariates via *Maximum Likelihood Estimation*, eliminating the need to discretise data based on time intervals.

5.6 References

Ana, E. and W. Bauwens (2010). “Modeling the structural deterioration of urban drainage pipes: the state-of-the-art in statistical methods”. In: *Urban Water Journal* 7.1, pp. 47–59. DOI: 0.1080/15730620903447597.

Baik, H.-S., H. S. Jeong, and D. M. Abraham (2006). “Estimating Transition Probabilities in Markov Chain-Based Deterioration Models for Management of Wastewater Systems”. In: *Journal of Water Resources Planning and Management* 132.1, pp. 15–24. DOI: 10.1061/(ASCE)0733-9496(2006)132:1(15).

Caradot, N., M. Riechel, M. Fesneau, N. Hernandez, A. Torres, H. Sonnenberg, E. Eckert, N. Lengemann, J. Waschnewski, and P. Rouault (2018). “Practical benchmarking of statistical and machine learning models for predicting the condition of sewer pipes in Berlin, Germany”. In: *Journal of Hydroinformatics* 20.5, pp. 1131–1147. DOI: 10.2166/hydro.2018.217.

Egger, C., A. Scheidegger, P. Reichert, and M. Maurer (2013). “Sewer deterioration modeling with condition data lacking historical records”. In: *Water Research* 47.17, pp. 6762–6779. ISSN: 0043-1354. DOI: 10.1016/j.watres.2013.09.010.

Hawari, A., F. Alkadour, M. Elmasry, and T. Zayed (2020). “A state of the art review on condition assessment models developed for sewer pipelines”. In: *Engineering Applications of Artificial Intelligence* 93, p. 103721. DOI: 10.1016/j.engappai.2020.103721.

Hout, A. van den (2016). *Multi-State Survival Models for Interval-Censored Data*. CRC Press, pp. 1–238. ISBN: 978-146656841-9. DOI: 10.1201/9781315374321.

Kobayashi, K., K. Kaito, and N. Lethanh (2012). “A statistical deterioration forecasting method using hidden Markov model for infrastructure management”. In: *Transportation Research Part B: Methodological* 46.4, pp. 544–561. DOI: 10.1016/j.trb.2011.11.008.

Le Gat, Y. (2008). “Modelling the deterioration process of drainage pipelines”. In: *Urban Water Journal* 5.2, pp. 97–106. DOI: 10.1080/15730620801939398.

Lin, P., X.-X. Yuan, and E. Tovilla (2019). “Integrative modeling of performance deterioration and maintenance effectiveness for infrastructure assets with missing condition data”. In: *Computer-Aided Civil and Infrastructure Engineering* 34.8, pp. 677–695. DOI: 10.1111/mice.12452.

Micevski, T., G. Kuczera, and P. Coombes (2002). “Markov Model for Storm Water Pipe Deterioration”. In: *Journal of Infrastructure Systems* 8.2, pp. 49–56. DOI: 10.1061/(ASCE)1076-0342(2002)8:2(49).

Saigo, H., J. Shao, and R. R. Sitter (2001). “A repeated half-sample bootstrap and balanced repeated replications for randomly imputed data”. In: *Survey Methodology* 27.2, pp. 189–196.

Scheidegger, A., T. Hug, J. Rieckermann, and M. Maurer (2011). “Network condition simulator for benchmarking sewer deterioration models”. In: *Water Research* 45.16, pp. 4983–4994. ISSN: 0043-1354. DOI: 10.1016/j.watres.2011.07.008.

Tscheikner-Gratl, F., N. Caradot, F. Cherqui, J. P. Leitão, M. Ahmadi, J. G. Langeveld, Y. Le Gat, L. Scholten, B. Roghani, J. P. Rodríguez, et al. (2019). “Sewer asset management–state of the art and research needs”. In: *Urban Water Journal* 16.9, pp. 662–675. DOI: 10.1080/1573062X.2020.1713382.

Virtanen, P., R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors (2020). “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python”. In: *Nature Methods* 17, pp. 261–272. DOI: 10.1038/s41592-019-0686-2.

Chapter 6

Comparing Homogeneous and Inhomogeneous Time Markov Chains for Modelling Deterioration in Sewer Pipe Networks

Paper published at **L. A. Jimenez-Roa**, T. Heskes, T. Tinga, M. Stoelinga, *"Comparing Homogeneous and Inhomogeneous Time Markov Chains for Modelling Deterioration in Sewer Pipe Networks"*, in 34th European Safety and Reliability Conference, ESREL 2024: Advances in Reliability, Safety and Security, 2024, arxiv.org/abs/2407.12557.

Abstract

Sewer systems are essential for social and economic welfare. Managing these systems requires robust predictive models for deterioration behaviour. This study focuses on probability-based approaches, particularly Markov chains, for their ability to associate random variables with deterioration. Literature predominantly uses homogeneous and inhomogeneous Markov chains for this purpose. However, their effectiveness in sewer main deterioration modelling is still debatable. Some studies support homogeneous Markov chains, while others challenge their utility. We examine this issue using a large-scale sewer network in the Netherlands, incorporating historical inspection data. We model deterioration with homogeneous discrete and continuous time Markov chains, and inhomogeneous-time Markov chains using Gompertz, Weibull, Log-Logistic and Log-Normal density functions. Our analysis suggests that, despite their higher computational requirements, inhomogeneous-time Markov chains are more appropriate for modelling the non-linear stochastic characteristics related to sewer main deterioration, particularly the Gompertz distribution. However, they pose a risk of over-fitting, necessitating significant improvements in parameter inference processes to effectively address this issue.

6.1 Introduction

Sewer networks are essential for societal and economic welfare but face management challenges such as budget constraints, environmental changes, and complex deterioration processes. Predictive tools for deterioration are becoming crucial as these systems reach the end of their design life, aiding in efficient maintenance and logistics (Marc Ribalta and Rubión, 2023). Robust models for sewer main deterioration are critical for balancing maintenance costs and system performance, enabling proactive maintenance, informed decision-making, and strategic planning (Caradot, Sonnenberg, Kropp, et al., 2017).

In this work, we focus on *Markov chains*, which are probabilistic models with the ability to predict future distributions associated with deterioration processes. Markov chains have several advantages: (i) they convert condition data into ordinal numbers such as severity levels, commonly used in industry to assess the condition of infrastructure assets (Tran, Lokuge, Setunge, et al., 2022); (ii) capture the stochastic nature of deterioration processes in sewer mains; (iii) their outputs can indicate the proportions of pipes in specific conditions, crucial for optimising maintenance planning.

Two primary types of Markov chains, homogeneous and inhomogeneous-time, are prevalent in the literature for modelling deterioration in sewer networks. However, the optimal Markov chain type remains debated. Proponents of homogeneous-time Markov chains, such as Micevski, Kuczera, and Coombes, 2002, argue for their sufficiency, while proponents of inhomogeneous-time Markov chains, such as Egger, Scheidegger, Reichert, et al., 2013, question homogeneous-time Markov chains efficacy. This gap is what we cover with this work since no studies have directly compared these models using the same dataset and discussed their suitability.

Understanding this is crucial for sewer asset managers implementing maintenance strategies, as different assumptions about the deterioration model can have distinct implications for maintenance policies.

For this, we employ homogeneous-time Markov chains, and for inhomogeneous chains, we use Gompertz, Weibull, Log-Logistic and Log-Normal functions, commonly used in reliability engineering. Our study, using a large-scale sewer network case study in the Netherlands, evaluates calibration complexity and model performance using cross-validation and various goodness-of-fit metrics. We employ the non-parametric Turnbull estimator for handling the interval-censored data in the inspection dataset, serving as a reference.

Contributions. Our key contributions with this work are:

- Presenting evidence that inhomogeneous-time Markov chains, despite their complexity, more effectively model non-linear stochastic behaviours in long-lived assets like sewer networks.

- Exploring alternative distributions, such as Log-Logistic and Log-Normal functions, in sewer network deterioration modelling.
- Provide comprehensive formal definitions of the deterioration models. Additionally, for calibration, we combine the Metropolis-Hastings (M-H) algorithm with the Sequential Least Squares Programming (SLSQP) algorithm for parameter inference in different Markov chains, a novel approach in this field.
- Our implementation is available at `https://gitlab.utwente.nl/fmt/degradation-models/ihctmc`.

Chapter outline. Section 6.2 describes the methods and materials. Section 6.3 details the experimental setup and results. Section 6.4 analyses the findings. Section 6.5 concludes the chapter and suggests future research directions.

Related work. The literature on sewer main deterioration modelling identifies two primary types of Markov chains: homogeneous and inhomogeneous (Table II.6 on page 104). Homogeneous-time Markov Chains (HTMCs) have *constant* transition probabilities, meaning the probabilities of transitioning between states do not change over time. In contrast, Inhomogeneous-time Markov Chains (IHTMCs) feature *time-variable* transition probabilities, indicating that the likelihood of state transitions can vary.

From the literature, we observe that HTMCs offer simplicity and computational efficiency, making them easier to analyse. However, they often cannot adequately model non-linear patterns found in stochastic degradation processes, where assuming constant transition probabilities may be overly simplistic. In contrast, IHTMCs can handle these complexities better by accommodating time-varying transition probabilities. Yet, these chains are computationally intensive and sometimes lack feasible closed-form solutions, complicating their application.

For completeness, other studies use different forms of Markov chains, such as *semi-Markov chains*, *fuzzy Markov chains*, and *ordered logistic models* (Kleiner, 2001; Kleiner, Sadiq, and Rajani, 2004; Lubini and Fuamba, 2011). These types of Markov chains are outside the scope of our analysis.

6.2 Methods and materials

Deterioration models for sewer mains are typically developed using inspection data conforming to standards like EN 13508:1 and EN 13508:2, which guide the classification of damages observed through CCTV inspections into *severity levels*.

This data situates these deterioration models within the domain of Multi-State Modelling (MSM), which captures deterioration through *finite states* with well-defined deterioration indicators, providing a granular view of the process (Compare, Baraldi, Bani, et al., 2017). Thus, stochastic deterioration modelling of sewer mains is conducted via Markov chains, with states corresponding to severity levels.

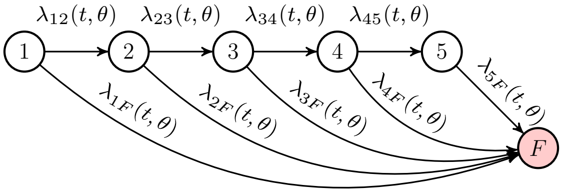


Figure 6.1: Markov chain structure modelling the deterioration of a sewer main considering five deterioration states and a functional failure state.

For formal definitions of IHTMCs, HTMCs, CTMCs, DTMCs, see Section II.4.1 on page 95.

6.2.1 Multi-state deterioration modelling for sewer networks using parametrised Markov chains

We first outline the structure of our Markov chain model (Figure 6.1) and then describe its parameterisation. The pipe element is defined with $K = 6$ sequential states $\boldsymbol{p} = [\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_K]$, where $\boldsymbol{p}_1$ represents the pristine state and $\boldsymbol{p}_K$ the most deteriorated state. The six states account for severity levels 1 to 5 and functional failure, as reported in sewer network inspections.

Further details on the type of Markov chain in Figure 6.1 are provided in Section II.4.4 (p. 103), applicable to both IHTMCs and HTMCs. We parametrise our Markov chains using probability density functions to model hazard rates. Specifically, we employ Exponential, Gompertz, Weibull, Log-Logistic, and Log-Normal density functions. For the Log-Normal function, lacking a closed-form hazard rate, we compute it as the ratio $f(t;\theta)/\mathbf{S}(t;\theta)$. Hazard rates and hyper-parameter ranges for the other functions are specified in Eq. 6.1.

$$\text{Exponential:} \quad \lambda(t;\alpha) = \alpha \tag{6.1a}$$

$$\text{Rate: } \alpha > 0$$

$$\text{Gompertz:} \quad \lambda(t;\alpha,\beta) = \alpha\beta e^{\beta t} \tag{6.1b}$$

$$\text{Shape: } \alpha > 0, \text{Scale: } \beta > 0$$

$$\text{Weibull:} \quad \lambda(t;\alpha,\beta) = \frac{\beta}{\alpha}\left(\frac{t}{\alpha}\right)^{\beta-1} \tag{6.1c}$$

$$\text{Scale: } \alpha > 0, \text{Shape: } \beta > 0$$

$$\text{Log-Logistic:} \quad \lambda(t;\alpha,\beta) = \frac{(\beta/\alpha)(t/\alpha)^{\beta-1}}{1+(t/\alpha)^{\beta}} \tag{6.1d}$$

$$\text{Scale: } \alpha > 0, \text{Shape: } \beta > 0$$

From Eq. 4.3 (page 98), we derive the system of differential equations related to the Markov chain in Figure 6.1 and present them in Eq. 6.2. To solve numerically

$$\frac{\partial \boldsymbol{p}_1(t)}{dt} = -\big(\lambda_{12}(t;\theta) + \lambda_{1F}(t;\theta)\big)\boldsymbol{p}_1(t) \tag{6.2a}$$

$$\frac{\partial \boldsymbol{p}_2(t)}{dt} = \lambda_{12}(t;\theta)\boldsymbol{p}_1(t) - \big(\lambda_{23}(t;\theta) + \lambda_{2F}(t;\theta)\big)\boldsymbol{p}_2(t) \tag{6.2b}$$

$$\frac{\partial \boldsymbol{p}_3(t)}{dt} = \lambda_{23}(t;\theta)\boldsymbol{p}_2(t) - \big(\lambda_{34}(t;\theta) + \lambda_{3F}(t;\theta)\big)\boldsymbol{p}_3(t) \tag{6.2c}$$

$$\frac{\partial \boldsymbol{p}_4(t)}{dt} = \lambda_{34}(t;\theta)\boldsymbol{p}_3(t) - \big(\lambda_{45}(t;\theta) + \lambda_{4F}(t;\theta)\big)\boldsymbol{p}_4(t) \tag{6.2d}$$

$$\frac{\partial \boldsymbol{p}_5(t)}{dt} = \lambda_{45}(t;\theta)\boldsymbol{p}_4(t) - \lambda_{5F}(t;\theta)\boldsymbol{p}_5(t) \tag{6.2e}$$

$$\frac{\partial \boldsymbol{p}_F(t)}{dt} = \lambda_{1F}(t;\theta)\boldsymbol{p}_1(t) + \lambda_{2F}(t;\theta)\boldsymbol{p}_2(t) + \lambda_{3F}(t;\theta)\boldsymbol{p}_3(t) + \lambda_{4F}(t;\theta)\boldsymbol{p}_4(t) + \lambda_{5F}(t;\theta)\boldsymbol{p}_5(t) \tag{6.2f}$$

the system of differential equations in Eq. 6.2, we use Python's `solve_ivp` function from the `scipy.integrate` module. This function, based on 'LSODA' from the FORTRAN `odepack` library, employs the `Adams/BDF` method with automatic stiffness detection (Virtanen, Gommers, Oliphant, et al., 2020).

6.2.2 Model calibration

To optimise the hyper-parameters of parametrised Markov chains, we employ a novel approach that combines the *Metropolis-Hastings (M-H)* algorithm (Hastings, 1970)—a Markov chain Monte Carlo method—with the *Sequential Least Squares Programming* (SLSQP) algorithm (Virtanen, Gommers, Oliphant, et al., 2020), specialised in solving constrained non-linear problems.

Although the M-H algorithm alone does not ensure optimal hyper-parameters, it provides a crucial initial guess that aids the SLSQP algorithm in avoiding premature convergence to local optima. To the best of our knowledge, this is the first time these two algorithms have been used together for this application.

Sewer inspections are considered *interval-censored*, where state transitions occur within certain intervals but are not exactly known (Duchesne, Beardsell, Villeneuve, et al., 2013). This complexity is omitted from our likelihood function, but its further exploration is suggested (Hout, 2016). We analyse the impact of interval-censored data using a non-parametric Turnbull estimator (see Section 6.2.3).

The initial part of our optimisation problem aligns with Micevski, Kuczera, and Coombes, 2002, starting with model calibration in a Bayesian optimisation context. We consider $y = [y_1, \ldots, y_n]$, representing the ages of pipes at inspection. Our likelihood function, $f(y|\gamma, \mathcal{M})$, where $\gamma = \langle \theta, \boldsymbol{p}^{(0)} \rangle$, evaluates the probability of observing the data y given the parameters γ and assuming the Markov model $\mathcal{M}$. Incorporating $\boldsymbol{p}^{(0)}$ in the optimisation introduces the constraint $\sum_{k \in \mathbb{S}} \boldsymbol{p}_k^{(0)} = 1$.

Initially, parameters γ are sampled from the prior $\mathbb{P}(\gamma|\mathcal{M})$. By applying Bayes' theorem, the posterior distribution $\mathbb{P}(\gamma|y,\mathcal{M})$ is expressed as:

$$\mathbb{P}(\gamma|y,\mathcal{M}) = \frac{f(y|\gamma,\mathcal{M})\mathbb{P}(\gamma|\mathcal{M})}{\mathbb{P}(y|\mathcal{M})},$$

where the posterior $\mathbb{P}(\gamma|y,\mathcal{M})$ updates beliefs about the parameters after observing data. The marginal likelihood $\mathbb{P}(y|\mathcal{M})$ is given by:

$$\mathbb{P}(y|\mathcal{M}) = \int f(y|\gamma,\mathcal{M})\mathbb{P}(\gamma|\mathcal{M})d\gamma,$$

reflecting how well model $\mathcal{M}$, across all parameter values, explains the observed data. Since the computation of $\mathbb{P}(y|\mathcal{M})$ is complex, it is assumed that the posterior is *proportional* to the product of the likelihood and the prior.

$$\mathbb{P}(\gamma|y,\mathcal{M}) \propto f(y|\gamma,\mathcal{M})\mathbb{P}(\gamma|\mathcal{M}) \tag{6.3}$$

For our optimisation problem, we first derive the following relations:

$$\mathbf{S}_k(t;\gamma,\mathcal{M}) = \sum_{m=1}^{k} \boldsymbol{p}_m(t;\gamma,\mathcal{M}),$$

$$f(y|\gamma,\mathcal{M}) = -\frac{d\mathbf{S}_k(t;\gamma,\mathcal{M})}{dt},$$

where $\mathbf{S}_k(t;\gamma,\mathcal{M})$ is the survival functions, notice that $\mathbf{S}_{k=F}(t;\gamma,\mathcal{M}) = 1$. Then the log-likelihood function (ℓ) is defined by:

$$\ell = \sum_{t\in y}\sum_{k\in\mathbb{S}} \mathbf{n}_{\underline{k},t} \log\Big(-\frac{d\mathbf{S}_k(t;\gamma,\mathcal{M})}{dt}\Big) \tag{6.4}$$

Here, $\ell \in (-\infty, 0]$, $\mathbf{n}_{\underline{k},t}$ denotes the number of pipes of age t found in states that transitioned from k, denoted as $\underline{k}$. E.g., if $k = 1$, then $\underline{k} = \langle 2, F\rangle$ (see Figure 6.1).

The acceptance distribution $\mathcal{A}$ of the M-H algorithm is given by:

$$\begin{aligned}\mathcal{A}(x_t, x_{t+1}) = \min\left(1, \frac{f(y|\gamma_{t+1},\mathcal{M})\mathbb{P}(\gamma_{t+1}|\mathcal{M})}{f(y|\gamma_t,\mathcal{M})\mathbb{P}(\gamma_t|\mathcal{M})}\right)& \\ > U(0,1)& \end{aligned} \tag{6.5}$$

Here, x_t and x_{t+1} are the current and proposed points in the parameter space, and γ_t and γ_{t+1} the corresponding sets of hyper-parameters. The prior $\mathbb{P}(\gamma_{t+1}|\mathcal{M})$ is a uniform distribution $U(\underline{\epsilon}, \bar{\epsilon})$, where $\underline{\epsilon}$ and $\bar{\epsilon}$ define the range for each hyper-parameter in γ.

$$\text{AIC} = 2|\gamma| - 2\ell \tag{6.6a}$$

$$\text{BIC} = \ln(|y|)|\gamma| - 2\ell \tag{6.6b}$$

$$\text{RMSE} = \sqrt{\frac{1}{|y| \times |\mathbb{S}|} \sum_{t \in y} \sum_{k \in \Omega} \big(\boldsymbol{p}_k(t) - \hat{\boldsymbol{p}}_k(t)\big)^2} \tag{6.6c}$$

The M-H algorithm executes 50,000 iterations, with the first 49,000 as the burn-in period, and the subsequent 1,000 samples used to compute mean values and the output γ_{M-H}.

Post convergence, γ_{M-H} serves as the initial guess for SLSQP, with γ parameters constrained between $\underline{\epsilon}$ and $\bar{\epsilon}$. SLSQP employs convergence tolerances of `eps = 1E - 5` and `ftol = 1E - 50`, and runs for up to `300` iterations. Upon SLSQP convergence, γ_{SLSQP} is derived and selected as the optimal set of hyper-parameters for further analysis.

6.2.3 Non-parametric modelling

Non-parametric survival curve estimators compute survival probabilities without assuming a specific distribution for survival times. This approach provides a reliable baseline crucial in our analysis to understand the effects of interval-censored data. Given the interval-censored nature of our data, we employ the *Turnbull* estimator (Turnbull, 1976), a non-parametric technique suitable for such data.

For each severity level k in our Markov chain, we calibrate a Turnbull estimator. Data binarization is achieved using $k_{\text{bin}} \in \Omega$ as a threshold. Observations with $k < k_{\text{bin}}$ are considered *non-events*, associated with the interval $[y_i, +\infty)$. Conversely, $k \geq k_{\text{bin}}$ are treated as *events*, defined by the interval $[0, y_i)$. These non-parametric Turnbull estimators are computed using the `lifelines` toolbox (Davidson-Pilon, 2019) in Python.

6.2.4 Goodness-of-fit metrics

Markov chains performance is evaluated via likelihood-based metrics: Akaike Information Criterion (AIC) (Akaike, 1998) (Eq. 6.6a) and Bayesian Information Criterion (BIC) (Schwarz, 1978) (Eq. 6.6b), which aid in model selection. Moreover, the Root Mean Squared Error (RMSE) (Eq. 6.6c) quantifies the Euclidean distance between the predictions of the Markov chains and the Turnbull estimator.

Both AIC and BIC include $|\gamma|$, the number of parameters in the model, with BIC additionally considering $\ln(|y|)$, the natural logarithm of the sample size. RMSE involves $\boldsymbol{p}_k(t)$ and $\hat{\boldsymbol{p}}_k(t)$, which denote the probabilities of being in state k at pipe age t, obtained from the Markov chains and the Turnbull estimator, respectively.

6.3 Experimental setup and evaluation

6.3.1 Case study

The case study is detailed in Section II.4.3 on page 100. Mohammadi, Najafi, Kermanshachi, et al., 2020; Salihu, Hussein, Mohandes, et al., 2022 identify age, material, and content as the primary factors affecting sewer pipe condition. Based on these, we categorise pipes into three cohorts and examine the `BAF` damage code, indicating infiltration.

Cohort `CMW`: Concrete pipes for mixed and waste content, Length: 469 km, Pipes: 11,942. Cohort `CS`: Concrete pipes for stormwater, Length: 172 km, Pipes: 4,701. Cohort `PMW`: PVC pipes for mixed and waste content, Length: 294 km, Pipes: 10,777.

6.3.2 Experimental setup

Our experiment aims to assess the efficacy of homogeneous and inhomogeneous Markov chains in predicting stochastic deterioration of sewer mains using the same dataset. Employing cross-validation, 70% of the sewer mains from the case study are randomly selected for model calibration, while the remaining 30% is used to compute goodness-of-fit metrics described in Section 6.2.4.

6.3.3 Results

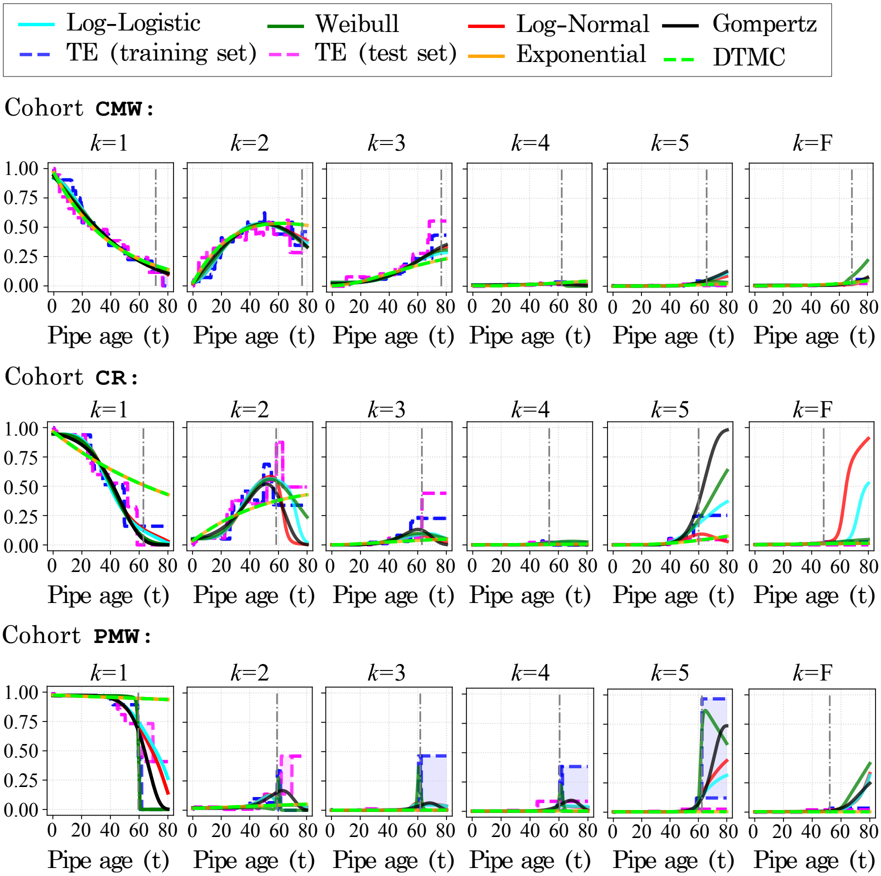
The different types of Markov chains are calibrated using data from cohorts `CMW`, `CS`, and `PMW` on infiltration, following the procedure described in Section 6.2.2 with the training set. Table 6.1 presents the goodness-of-fit metrics for both the training and test sets, while Figure 6.2 illustrates the state probabilities. The results from the Turnbull estimator for both sets are also displayed. The vertical grey dashed lines in the figures denote the last inspection used for model training. By solving Eq. 4.2 (page 98), we obtain the transition probability matrix over time $\boldsymbol{p}_{ij}(t, \tau)$. Figure 6.3 displays these probabilities for cohort `CS` and infiltration.

6.4 Findings

6.4.1 Comparison between cohorts

For all cohorts, the inhomogeneous time Markov chains (modelled with Gompertz, Weibull, Log-logistic, and Log-normal functions) outperform the homogeneous time Markov chains (modelled with the Exponential function and DTMCs) by achieving the lowest values in all goodness-of-fit metrics in Table 6.1.

Notice that a smaller RMSE in Table 6.1 suggests a closer alignment of the Markov chains with the Turnbull estimator. Also, the goodness-of-fit metrics for



(a) Cohort `CMW`:

(b) Cohort `CR`:

(c) Cohort `PMW`:

Figure 6.2: State probability $\boldsymbol{p}_k^{(t)}$ for different Markov chains. Dashed lines are the Turnbull estimators. For Cohort (a) `CMW`, (b) `CS`, (c) `PMW`.

both homogeneous time Markov chains are identical, which is consistent with the theoretical mapping of one into the other. This is visually corroborated in Figure 6.2.

6.4.2 Transition probabilities over time

For further clarification and illustrative purposes in understanding the behaviour within different types of Markov chains, Figure 6.3 displays the transition probability variations among Markov chains in the `CS` cohort. The homogeneous time Markov chain, employing the Exponential distribution, maintains constant transition probabilities over time, reflecting its homogeneous and memoryless properties. Conversely, the inhomogeneous time Markov chains reveal diverse behaviours in their transition probabilities, depicting distinct temporal variations. Notice that there are also differences in the transition probabilities between inhomogeneous Markov chains, due to the different assumptions on the density functions.

6.4.3 Overfitting

All inhomogeneous Markov chains map well where data is available (up to around 70-year-old pipes, see grey dashed vertical lines in Figure 6.2), however, beyond this point, these models tend to move faster to worse conditions. This is likely related to the additional degrees of freedom that inhomogeneous Markov provides.

This effect is less in homogeneous Markov chains because they have fewer degrees of freedom. Thus, future research should consider this aspect in the model calibration, to improve the predictive capabilities of inhomogeneous time Markov chains.

6.4.4 Comparing inhomogeneous Markov chains

Upon closer examination of inhomogeneous Markov chains modelled with Log-Normal, Log-Logistic, Weibull, and Gompertz density functions, Table 6.1 reveals that the Gompertz distribution consistently demonstrates good performance across all cohorts and goodness-of-fit metrics, followed by Weibull and Log-Logistic density functions. Notably, the Weibull distribution shows poor performance for cohort `PMW`, likely due to sub-optimal parameters resulting from convergence in local optima.

6.5 Conclusions and future research

We examine the effectiveness of homogeneous and inhomogeneous Markov chains in modelling stochastic deterioration in sewer mains. We introduce four inhomogeneous Markov chain models parametrised with Log-Normal, Log-Logistic, Weibull, and Gompertz density functions, and compare them against a homogeneous Markov chain with an Exponential distribution and discrete-time Markov chains using the same dataset.

These models are calibrated using Metropolis-Hastings and Sequential Least Squares Programming algorithms, utilising historical inspection data from a Dutch sewer network. Additionally, we employ the Turnbull estimator as a reference to account for the interval censoring in the dataset.

From the dataset, we establish three cohorts and assess the fit of the Markov chains using various goodness-of-fit metrics. Our findings suggest that, despite their complexity, inhomogeneous time Markov chains more effectively model the non-linear stochastic behaviours observed in sewer network inspection data. In particular, the inhomogeneous time Markov chain modelled with the Gompertz distribution consistently showed good performance.

This observation aligns with Mizutani and Yuan, 2023, which recommends inhomogeneous Markov chains to model time-varying transition probabilities in bridge

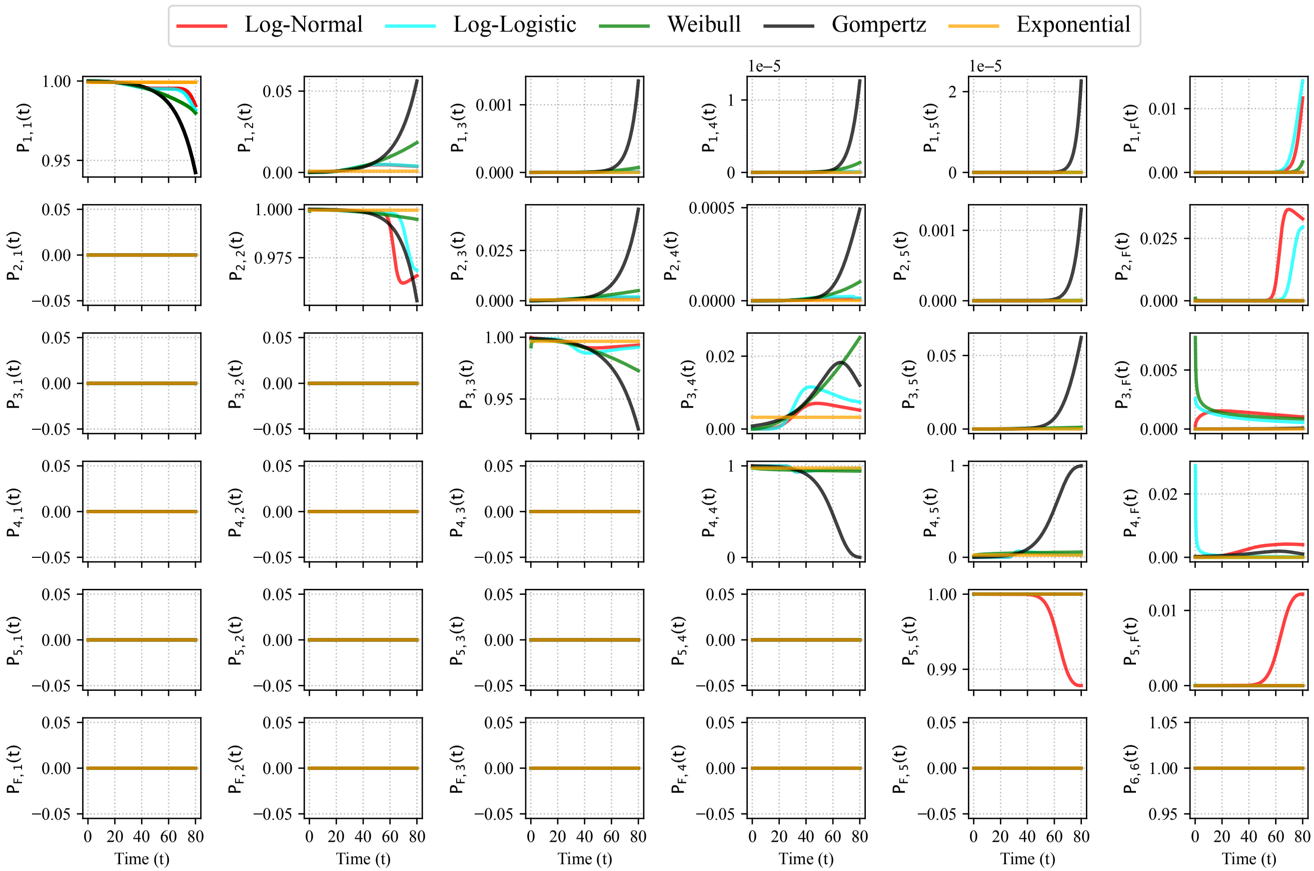


Figure 6.3: Transition probabilities $\mathbf{P}_{ij}(t, \tau)$ for cohort CS on infiltration.

Table 6.1: Goodness-of-fit metrics computed on the training and testing sets for different Markov chains. Blue and red colours indicate the best and worst scores, respectively.

				Training set			Test set		
Cohort	Type	Func.	$\lvert\gamma\rvert$	RMSE	AIC	BIC	RMSE	AIC	BIC
`CMW`	IHTMC	Gompertz	24	0.0250	57431.0	57601.8	**0.0337**	**20436.0**	**20583.2**
	IHTMC	Weibull	24	0.0233	**57414.8**	**57585.5**	0.0361	20478.0	20625.2
	IHTMC	Log-Logistic	24	**0.0219**	58544.4	58715.1	**0.0391**	20861.2	21008.4
	IHTMC	Log-Normal	24	0.0221	58553.6	58724.3	0.0385	20823.2	20970.3
	CTMC	Exponential	15	**0.0312**	**59574.6**	**59681.3**	0.0359	**21142.1**	**21234.0**
	DTMC	-	15	**0.0312**	**59574.6**	**59681.3**	0.0359	**21142.1**	**21234.0**
`CS`	IHTMC	Gompertz	24	0.0358	**3532.8**	**3665.8**	0.0468	**1179.4**	**1290.9**
	IHTMC	Weibull	24	0.0326	3850.0	3983.0	**0.0423**	1269.0	1380.5
	IHTMC	Log-Logistic	24	**0.0313**	**4111.3**	**4244.3**	0.0427	1345.9	**1457.4**
	IHTMC	Log-Normal	24	0.0330	4035.1	4168.1	0.0446	1324.4	1436.0
	CTMC	Exponential	15	**0.0593**	4006.7	4089.8	**0.0583**	**1359.1**	1428.8
	DTMC	-	15	**0.0593**	4006.7	4089.8	**0.0583**	**1359.1**	1428.8
`PWM`	IHTMC	Gompertz	24	0.0199	**2349.3**	**2495.3**	**0.0172**	**989.7**	1114.7
	IHTMC	Weibull	24	**0.0153**	**5522.0**	**5668.0**	**0.0403**	**1822.1**	**1947.0**
	IHTMC	Log-Logistic	24	0.0217	4699.7	4845.7	0.0178	1523.1	1648.0
	IHTMC	Log-Normal	24	0.0211	3588.2	3734.2	0.0179	1493.6	1618.6
	CTMC	Exponential	15	**0.0297**	2438.6	2529.9	0.0256	1002.8	**1080.9**
	DTMC	-	15	**0.0297**	2438.6	2529.9	0.0256	1002.8	**1080.9**

structures. This result is crucial for sewer asset managers, as deriving maintenance policies for sewer mains requires accounting for these non-linearities in deterioration models, since different assumptions may yield distinct maintenance policy implications.

To maintain the severity levels within the model and to address the non-linearities in the deterioration process of sewer mains, it is key to adequately evaluate the inhomogeneous behaviours. The use of homogeneous time Markov chains is advised only if the modeller can substantiate this assumption beforehand.

Future research. Future research directions include:

- Addressing the omission of interval censoring during the calibration of our inhomogeneous time Markov chains, which approximate the Turnbull estimator, requires further investigation to assess the validity of neglecting interval censoring.
- Expanding our models to consider pipe length and the distribution of deterioration along the sewer main, beyond focusing solely on the most severe pipe condition during inspections.
- Developing models that incorporate covariates without forming cohorts to minimise cohort selection biases.
- Integrating uncertainty quantification, vital for decision-making, requires studies on accurate uncertainty bound estimation.
- Despite our calibration process's efficacy, further exploration of alternative optimisation techniques for non-linear constrained problems is needed to improve parameter inference speed, aiming to reduce over-fitting.
- Future studies should also investigate the application of these models to optimise maintenance and inspection policies in sewer networks.

6.6 References

Akaike, H. (1998). "Information Theory and an Extension of the Maximum Likelihood Principle". In: *Selected Papers of Hirotugu Akaike*. Ed. by E. Parzen, K. Tanabe, and G. Kitagawa. New York, NY: Springer New York, pp. 199–213. ISBN: 978-1-4612-1694-0. DOI: 10.1007/978-1-4612-1694-0_15.

Caradot, N., H. Sonnenberg, I. Kropp, A. Ringe, S. Denhez, A. Hartmann, and P. Rouault (2017). "The relevance of sewer deterioration modelling to support asset management strategies". In: *Urban Water Journal* 14.10, pp. 1007–1015. DOI: 10.1080/1573062X.2017.1325497.

Compare, M., P. Baraldi, I. Bani, E. Zio, and D. Mc Donnell (2017). "Development of a Bayesian multi-state degradation model for up-to-date reliability estimations of working industrial components". In: *Reliability Engineering & System Safety* 166. Reliability and Performance of Multi-State Systems, pp. 25–40. ISSN: 0951-8320. DOI: 10.1016/j.ress.2016.11.020.

Davidson-Pilon, C. (2019). "lifelines: survival analysis in Python". In: *Journal of Open Source Software* 4.40, p. 1317. DOI: 10.21105/joss.01317.

Duchesne, S., G. Beardsell, J.-P. Villeneuve, B. Toumbou, and K. Bouchard (2013). "A survival analysis model for sewer pipe structural deterioration". In: *Computer-Aided Civil and Infrastructure Engineering* 28.2, pp. 146–160. DOI: 10.1111/j.1467-8667.2012.00773.x.

Egger, C., A. Scheidegger, P. Reichert, and M. Maurer (2013). "Sewer deterioration modeling with condition data lacking historical records". In: *Water Research* 47.17, pp. 6762–6779. ISSN: 0043-1354. DOI: 10.1016/j.watres.2013.09.010.

EN 13508:1 -Investigation and assessment of drain and sewer systems outside buildings - Part 1: General Requirements (Oct. 2012). Standard. Avenue Marnix 17, B-1000 Brussels: European Committee for Standardization (CEN).

EN 13508:2 -Investigation and assessment of drain and sewer systems outside buildings - Part 2: Visual inspection coding system (May 2011). Standard. Avenue Marnix 17, B-1000 Brussels: European Committee for Standardization (CEN).

Hastings, W. K. (Apr. 1970). "Monte Carlo sampling methods using Markov chains and their applications". In: *Biometrika* 57.1, pp. 97–109. ISSN: 0006-3444. DOI: 10.1093/biomet/57.1.97.

Hout, A. van den (2016). *Multi-State Survival Models for Interval-Censored Data.* CRC Press, pp. 1–238. ISBN: 978-146656841-9. DOI: 10.1201/9781315374321.

Kleiner, Y. (2001). "Optimal scheduling of rehabilitation and inspection/condition assessment in large buried pipes". In: *NRCC-44487, 4th International Conference on Water Pipeline Systems—Managing Pipeline Assets in an Evolving Market*, pp. 181–197.

Kleiner, Y., R. Sadiq, and B. Rajani (2004). "Modeling failure risk in buried pipes using fuzzy Markov deterioration process". In: *Pipeline Engineering and Construction: What's on the Horizon?*, pp. 1–12. DOI: 10.1061/40745(146)7.

Lubini, A. T. and M. Fuamba (2011). "Modeling of the deterioration timeline of sewer systems". In: *Canadian Journal of Civil Engineering* 38.12, pp. 1381–1390. DOI: 10.1139/l11-103.

Marc Ribalta Ramon Bejar, C. M. and E. Rubión (2023). "Machine learning solutions in sewer systems: a bibliometric analysis". In: *Urban Water Journal* 20.1, pp. 1–14. DOI: 10.1080/1573062X.2022.2138460.

Micevski, T., G. Kuczera, and P. Coombes (2002). "Markov Model for Storm Water Pipe Deterioration". In: *Journal of Infrastructure Systems* 8.2, pp. 49–56. DOI: 10.1061/(ASCE)1076-0342(2002)8:2(49).

Mizutani, D. and X.-X. Yuan (2023). "Infrastructure deterioration modeling with an inhomogeneous continuous time Markov chain: A latent state approach with analytic transition probabilities". In: *Computer-Aided Civil and Infrastructure Engineering* 38.13, pp. 1730–1748. DOI: 10.1111/mice.12976.

Mohammadi, M. M., M. Najafi, S. Kermanshachi, V. Kaushal, and R. Serajiantehrani (2020). "Factors Influencing the Condition of Sewer Pipes: State-of-the-Art Review". In: *Journal of Pipeline Systems Engineering and Practice* 11.4, p. 03120002. DOI: 10.1061/(ASCE)PS.1949-1204.0000483.

Salihu, C., M. Hussein, S. R. Mohandes, and T. Zayed (2022). "Towards a comprehensive review of the deterioration factors and modeling for sewer pipelines: A hybrid of bibliometric, scientometric, and meta-analysis approach". In: *Journal of Cleaner Production* 351, p. 131460. ISSN: 0959-6526. DOI: 10.1016/j.jclepro.2022.131460.

Schwarz, G. (1978). "Estimating the Dimension of a Model". In: *The Annals of Statistics* 6.2, pp. 461–464. ISSN: 00905364. URL: https://www.jstor.org/stable/2958889.

Tran, H., W. Lokuge, S. Setunge, and W. Karunasena (2022). "Network deterioration prediction for reinforced concrete pipe and box culverts using Markov model: Case study". In: *Journal of Performance of Constructed Facilities* 36.6, p. 04022047. DOI: 10.1061/(ASCE)CF.1943-5509.000176.

Turnbull, B. W. (1976). "The Empirical Distribution Function with Arbitrarily Grouped, Censored and Truncated Data". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 38.3, pp. 290–295. DOI: https://doi.org/10.1111/j.2517-6161.1976.tb01597.x.

Virtanen, P., R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors (2020). "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods* 17, pp. 261–272. DOI: 10.1038/s41592-019-0686-2.

Part III

Maintenance optimisation of multi-state components

III.1 Introduction

Part III focuses on Maintenance Policy Optimisation (MPO) of components with Multi-State Deterioration via Deep Reinforcement Learning, with applications to sewer mains. The general research question we address here is *how can optimal maintenance strategies be devised for components with Multi-State Deterioration such as sewer mains using Deep Reinforcement Learning?* This part is structured as follows: Section III.2 summarises the nomenclature used in Part III. Section III.3 reviews the related work common to all chapters. Section III.4 presents the formal definitions used in Part III. The chapters contained here are:

Chapter 7. *Maintenance Strategies for Sewer Pipes with Multi-State Deterioration and Deep Reinforcement Learning* 151

III.2 Nomenclature

Refer to the nomenclature on Markov chains for Multi-State Deterioration Model in Section II.2 on page 94.

(Contextual-) Markov Decision Processes:

T	Time horizon
$t \in T$	*Time* (e.g., component *age*)
$\mathcal{S}$	State space
$\mathcal{A}$	Action space
$\mathcal{C}$	Context space
$\mathcal{R}(\cdot)$	Reward function
$\mathbf{P}_{ij}(\cdot)$	Transition probability function
$s_t, s_{t+1} \in \mathcal{S}$	*Current* (t) and *Next* $(t+1)$ state instances
$a_t \in \mathcal{A}$	Action instance at time t
r_t	Reward at time t
γ	Discount factor
π_0	Initial policy distribution
$\mathbf{M}$	Markov Decision Process
$c \in \mathcal{C}$	Context instance
$\mathcal{K}(\cdot)$	Mapping function
$\mathbf{M}^c$	Contextual Markov Decision Process

Reinforcement Learnings:

$\mathbf{E}$	Environment
$\mathbf{A}$	Agent
π_t	Policy at t
$V(\cdot)$	Value function
$Q(s, a)$	State-action value function

Deep Neural Networks:

$\mathbf{N}$	Deep Neural Network
$\hat{L}$	Number of layers
n	Size of the input layer
m	Size of the output layer
$\hat{l}$	Layer in the network, with $\hat{l} = 1, \ldots, \hat{L}$
W	Weight matrix
b	Bias vector
$\sigma(\cdot)$	Activation function
$\mathcal{L}(\cdot)$	Loss function

Proximal Policy Optimisations:

$\mathbb{O}$	Surrogate objective function
$\hat{A}_t$	Advantage function at t
φ	Policy parameters

ρ	Probability ratio
δ	Temporal difference error
ϵ	Clipping parameter
ζ	Bias-variance balance parameter

Sewer mains maintenance optimisation:

$\mathbf{h}$	Health vector
L	Pipe length
ΔL	Pipe segment length
$\boldsymbol{\nu}$	Segments in a pipe
$\mathbf{d}$	Distribution of severities in the pipe

III.3 Related work

Over the past two decades, the need for integral sewer asset management has become evident (Abraham, Wirahadikusumah, Short, et al., 1998), highlighting the necessity of understanding deterioration mechanisms and developing predictive models for proactive and strategic sewer maintenance (R. Fenner, 2000).

Sewer asset management encompasses maintenance, rehabilitation, and inspection, and has been investigated through various methodologies, encompassing risk-based strategies, multi-objective optimisation, MDPs, analysis of network structure, Machine Learning applications, decision-support frameworks, and Reinforcement Learning. An overview of each approach is provided below.

Risk-based

A *risk-based* approach evaluates the potential for loss or negative outcomes due to uncertainty. Arthur and Crow, 2007 focus on *serviceability*, suggesting proactive maintenance of assets to prevent sewer overflow, flooding, and traffic disruptions. Fuchs-Hanusch, Günther, Möderl, et al., 2015 concentrate on cause-effect relationships to aid *inspection* planning, employing logistic regression to estimate the probability of sewer failure and its negative effects on the system's hydraulic performance. Baah, Dubey, Harvey, et al., 2015 utilises a *risk matrix* and a *weighted-sum multi-criteria decision matrix* for the assessment of consequences and risks associated with sewer main failure. Fontecha, Akhavan-Tabatabaei, Duque, et al., 2016 addresses sediment-related blockages, deriving *maintenance costs per time unit* through a convex non-linear function and a homogeneous Poisson process. Lee, Park, Baek, et al., 2021 concentrates on sewer inspection, employing a *risk matrix-based* method to assess economic, social, and environmental impacts.

Multi-objective optimisation

Multi-objective optimisation aims to optimise multiple, often conflicting, objectives simultaneously. João A. Zeferino and Cunha, 2010 addresses the minimisation of capital and operating maintenance costs, alongside the maximisation of dissolved oxygen, by employing a weighting method and a *simulated annealing algorithm*. Yang and Su, 2007 investigates sewer rehabilitation to achieve high effectiveness with minimal costs, utilising multi-objective *genetic algorithms* (GA). Similarly, Marzouk and Ibrahim, 2013 considers the condition of sewer networks and life-cycle maintenance costs as distinct objectives, integrating Monte Carlo simulations, discrete-time Markov chains for degradation modelling, and multi-objective genetic algorithms to identify optimal maintenance strategies. Furthermore, Elmasry, Zayed, and Hawari, 2019 concentrates on enhancing sewer inspection strategies by optimising inspection times, costs, and frequencies through *Mixed Integer Linear Programming* (MILP), demonstrating superior performance over GA.

Based on Markov decision processes

A *Markov decision process* is a widely used mathematical framework for modelling decision-making, also with applications in MPO. Abraham, Wirahadikusumah, Short, et al., 1998 employs *deterministic dynamic programming* to determine optimal strategies for sewer rehabilitation. Wirahadikusumah, Abraham, and Castello, 1999 models the rehabilitation of sewer networks as an MDP, incorporating life-cycle cost analysis and sewer network deterioration through discrete-time Markov chains, and addresses the problem using *dynamic programming* alongside the *policy improvement algorithm*. Further, Wirahadikusumah and Abraham, 2003 utilises *probabilistic dynamic programming* for sewer maintenance and rehabilitation, focusing on constraints to enhance the understanding of sewer life-cycle costs.

Considering sewer network structure

Moving towards *system-level* analysis, studies that explicitly model network structure and augment the optimisation problem with features pertinent to system-level analysis are distinguished. R. A. Fenner, Sweeting, and Marriott, 2000 proposes a method that combines *Geographical Information Systems* (GIS) tools, risk analysis and Bayesian statistics. Similarly, Inanloo, Tansel, Shams, et al., 2016 GIS-based risk assessment, integrating component failure probabilities, consequences, and potential *interactions* with other infrastructure networks for a comprehensive asset management analysis of transport, water, and sewer network systems. Hamid Zaman and Lorentz, 2017 tackles the schedule optimisation issue by framing it within the context of *combinatorial optimisation* and addressing it through genetic and heuristic algorithms. Qasem and Jamil, 2021 applies GIS-based financial analysis for integrated maintenance, rehabilitation, and replacement planning for water, sewer, and road networks. Kerkkamp, Bukhsh, Y. Zhang, et al., 2022 employs *Graphical Neural Networks* (GNN) and DRL to model the sewer network structure, focusing on grouping maintenance actions by leveraging sewer main proximity.

Machine learning

Machine learning (ML) enables computers to improve their performance on specific tasks by learning from data, with extensive applications in various domains, including sewer asset management. A review on this is provided by Marc Ribalta and Rubión, 2023. In the realm of *condition assessment*, techniques such as K-nearest neighbours, Support Vector Machines (SVM), Random Forest, Principal Component Analysis, and Gate Recurrent Unit are utilised for concrete defect detection and classification, as detailed by Gueye, Y. Wang, and Mushtaq, 2023, while Cheng and M. Wang, 2018 and Fang, Guo, Q. Li, et al., 2020 implement Deep Learning and methods like Isolate Forest, One-Class SVM, Local Outlier Factors, and Gaussian Distributed-based approaches for video sequences anomaly detection. For *degradation modelling* of sewer mains, Random Survival Forest, SVM, and Random Forest are applied to model sewer degradation and determine the distribution of time-to-failure, as explored by Caradot, Riechel, Fesneau, et al., 2018, Laakso, Kokkonen, Mellin, et al., 2019, and Hernández, Caradot, Sonnenberg, et al., 2021. Additionally, ML aids in maintenance *decision-making* by using data from sewer overflows and Decision Trees, as demonstrated by Montserrat, Bosch, Kiser, et al., 2015.

Decision-support frameworks

Decision-support tools, as high-level and generic management approaches, enhance decision-making by integrating various concepts, including those previously discussed. DeSilva, Burn, Tjandraatmadja, et al., 2005 explores sewer main leakage and introduces a *decision support tool* for rehabilitation prioritisation, utilising soil models, pipe properties, and operational conditions. Similarly, Ana and Bauwens, 2007 offers a decision support tool that employs multiple sewer management and rehabilitation models. Breysse, Vasconcelos, and Schoefs, 2007 combines social and technical cost indicators to provide a comprehensive tool for managers to assess alternatives. "Urban stormwater drainage management: The development of a multicriteria decision aid approach for best management practices" 2007 assists decision-makers in ranking options using *multi-criteria analysis.* Arthur, Crow, Pedezert, et al., 2009 suggests a holistic approach based on *Failure Mode and Effect Analysis* (FMECA), implementable with limited information without additional data collection. M.A. Cardoso and Silva, 2016 introduces the *AWARE-P* procedure, which combines various decision-support tools and methods considering performance, costs, and risk over an analysis horizon. Obradović, Šperac, and Marenjak, 2019 discusses the use of *expert systems* to support sewer maintenance optimisation. Lin, Yuan, and Tovilla, 2019 and Caradot, Sampaio, Guilbert, et al., 2021 focus on an integrated approach that considers modelling long-term sewer main degradation and maintenance strategies for sewer rehabilitation planning. F. Taillandier and Bennabi, 2020 proposes the *AGORA* method based on *multi-criteria decision analysis*, which incorporates uncertainties and enables comparing different management strategies. Khurelbaatar, Al Marzuqi, Van Afferden, et al., 2021 employs the *ALLOWS* method, enabling comparison of different management scenarios and stakeholder selection of the most cost-effective. Ramos-Salgado, Muñuzuri, Aparicio-Ruiz, et al., 2022 proposes a five-step framework for long-term infrastructure asset management and planning. Assaf and Assaad, 2023 adopts an agent-based approach combined with Monte Carlo analysis to determine optimal preventive maintenance, repair, and replacement policies.

Reinforcement Learning

The integration of Reinforcement Learning (RL) into sewer asset management is largely unexplored, with existing research mainly concentrating on *real-time control* for smart infrastructure, adapting to environmental changes such as storms. Mullapudi, Lewis, Gruden, et al., 2020 utilises DRL for controlling stormwater system valves through simulation of varied storm scenarios. Z. Yin, Leon, Sharifi, et al., n.d. employ RL for *near real-time* control to minimise sewer overflows. Meanwhile, Z. Zhang, Tian, and Liao, 2023 and Tian, Liao, Zhi, et al., 2022 both delve into enhancing urban drainage systems' robustness, the former through decentralised *multi-agent RL* and the latter via *Multi-RL*, with Tian, Fu, Xin, et al., 2024 further improving model *interpretability* using DRL. Additionally, Kerkkamp, Bukhsh, Y. Zhang, et al., 2022 investigates sewer network MPO by combining DRL with GNN to optimise maintenance actions grouping. Jeung, Jang, Yoon, et al., 2023 proposes a DRL-based *data assimilation* methodology to enhance stormwater and water quality simulation accuracy by integrating observational data with simulation outcomes.

Based on recent review papers, we offer an overview of the advantages of employing DRL for MPO tasks:

- The RL paradigm offers a *unified framework* for formulating problems that integrates both condition and predictive-based maintenance objectives with maintenance optimisation goals (Ogunfowora and Najjaran, 2023).
- DRL excels in complex, dynamic environments and is adaptable to uncertain conditions through its trial-and-error learning approach, which does not require pre-collected data or prior knowledge. This makes it ideal for addressing MPO challenges (Real Torres, Andreiana, Ojeda Roldán, et al., 2022).
- The proven effectiveness of DRL in MPO tasks within infrastructure systems, such as pavement (Yao, Dong, Jiang, et al., 2020), highlights its potential to improve maintenance strategies across other infrastructure systems (Marugán, 2023).
- DRL methods offer time-efficient and cost-effective solutions compared to traditional approaches by minimising maintenance costs and risks, and balancing performance with total maintenance costs over infrastructure life-cycles (Marugán, 2023).

- DRL's growth for maintenance planning is driven by increased access to IoT data and higher computing power, enabling seamless integration of predictive and optimisation models in maintenance (Ogunfowora and Najjaran, 2023).
- DRL holds significant potential for the future of smart manufacturing, promoting a cognitive, personalised approach (C. Li, Zheng, Y. Yin, et al., 2023).

III.4 Preliminaries

III.4.1 Markov Decision Process

A Markov Decision Process (MDP) is a well-known mathematical framework for formulating sequential decision-making problems (Puterman, 1990). Below, we provide a formal definition.

Definition 12 (Markov Decision Process)**.** *Let $T \subset \mathbb{N}_0$ represent the set of all non-negative integers. Let t be a discrete-time index such that $t \in T$, indexing the time steps in a stochastic sequential decision-making process. A* Markov Decision Process (MDP) *is formally defined by the tuple $\mathbf{M} = \langle \mathcal{S}, \mathcal{A}, \mathbf{P}, \mathcal{R}, \pi_0, \gamma \rangle$, where:*

- $\mathcal{S}$ *is a set of* states*.*
- $\mathcal{A}$ *is a set of* actions*.*
- $\mathbf{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ *is the transition probability function, $\mathbf{P}(s_{t+1}|s_t, a_t)$, giving the probability of transitioning from state s_t to state s_{t+1} under action a_t.*
- $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$ *is the reward function, $\mathcal{R}(s_t, a_t, s_{t+1})$, specifying the reward received after the transition.*
- $\pi_0 : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$ *is the initial policy distribution at $t = 0$.*
- $\gamma \in [0, 1]$ *is the discount factor, quantifying the importance of future rewards relative to immediate rewards.*

III.4.2 Deep Reinforcement Learning

Reinforcement Learning (RL) seeks to develop agents that learn optimal behaviours in virtual environments through trial and error guided by a reward signal (Arulkumaran, Deisenroth, Brundage, et al., 2017). Combining RL with Deep Neural Networks (DNNs) results in Deep Reinforcement Learning (DRL), offering greater scalability and the ability to tackle complex problems. Below we provide formal definitions on RL and DNNs.

Definition 13 (Reinforcement Learning)**.** Reinforcement Learning (RL) *is a learning paradigm where an agent interacts with an environment to maximise cumulative reward. Formally, it can be modelled as an* MDP *(see Definition 12). The key components of* RL *are the* Environment*,* Agent*, and* Objective*, detailed below:*

- Environment ($\mathbf{E}$)*: This comprises the state space $\mathcal{S}$, action space $\mathcal{A}$, state transition probabilities $\mathbf{P}(s_{t+1} \mid s_t, a_t)$, and reward function $\mathcal{R}(s_t, a_t, s_{t+1})$. Here,*

$s_t, s_{t+1} \in \mathcal{S}$ are the current *and* next *states, respectively, and $a_t \in \mathcal{A}$ is the action taken in s_t to reach s_{t+1}. The environment defines how actions affect the next state and rewards.*

- Agent (**A**)*: The agent is the* decision-maker *in* RL *that interacts with* **E** *by following a policy. It is characterised by the following:*
  - Policy (π)*: A function $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$ that maps* states *to a probability distribution over* actions. *The policy governs the agent's behaviour, determining the action to be taken in each state at time t.*
  - Value Function ($V(s)$)*: A function that estimates the expected return (cumulative discounted rewards) from each state $s \in \mathcal{S}$ under policy π, defined as:*

$$V(s) = \mathbb{E}_{\pi}\left[\sum_{k=0}^{\infty} \gamma^k \mathcal{R}_{t+k+1} \mid s_t = s\right], \textit{ for all } s \in \mathcal{S}, \tag{6.7}$$

where $\mathbb{E}_{\pi}[\cdot]$ denotes the expected value of a random variable when the **A** *follows policy π, and t represents any time step. The recursive form of $V(s)$ calculates the return over trajectories τ and is expressed as follows:*

$$V(s) = \mathbb{E}_{\pi}[\mathcal{R}_{t+1} + \gamma V(s_{t+1}) \mid s_t = s] \tag{6.8}$$

  - State-Action Value ($Q(s, a)$)*: The expected return of performing action $a \in \mathcal{A}$ for state $s \in \mathcal{S}$ can be defined as the pair* state-action *value function:*

$$Q(s, a) = \mathbb{E}_{\pi}[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) \mid s_t = s, a_t = a] \tag{6.9}$$

- Agent's Goal*: The agent* **A** *in* RL *aims to optimise its policy π by maximising the optimal Q function:*

$$\pi^*(s) = \arg\max_{a \in \mathcal{A}} Q^*(s, a) \tag{6.10}$$

where $\pi^*(\cdot)$ denotes the optimal policy *and $Q^*(\cdot)$ is the* optimal state-action pair. *This is achieved through iterative policy evaluation and improvement.*

Definition 14 (Deep Neural Network)**.** *A* Deep Neural Network (DNN) *is defined as a function $f : \mathbb{R}^n \to \mathbb{R}^m$ with $n, m \in \mathbb{N}$, and can be formally represented as a tuple $\mathbf{N} = \langle \hat{L}, \{d_{\hat{l}}\}_{\hat{l}=0}^{\hat{L}}, \sigma(\cdot), \{W_{\hat{l}}, b_{\hat{l}}\}_{\hat{l}=1}^{\hat{L}}, \mathcal{L}(\cdot)\rangle$, where:*

- *$\hat{L} \in \mathbb{N}$ denotes the number of layers in the network, when $\hat{L} \geq 2$ indicates a multi-layered structure.*
- *$\{d_{\hat{l}}\}_{\hat{l}=0}^{\hat{L}}$ specifies the dimensions of each layer $\hat{l}$, with $d_0 = n$ for the* input *layer and $d_{\hat{L}} = m$ for the* output *layer.*
- *$\sigma(\cdot) : \mathbb{R} \to \mathbb{R}$ is an* activation function, *which may be non-linear.*
- *For each $\hat{l} = 1, \ldots, \hat{L}$:*
  - *$W_{\hat{l}} \in \mathbb{R}^{d_{\hat{l}} \times d_{\hat{l}-1}}$ is the* weight matrix *of the $\hat{l}$-th layer, mapping inputs from dimension $d_{\hat{l}-1}$ to $d_{\hat{l}}$.*

  - $b_{\hat{l}} \in \mathbb{R}^{d_{\hat{l}}}$ *is the* bias vector *for the* $\hat{l}$*-th layer.*
  - *The function* $f_{\hat{l}} : \mathbb{R}^{d_{\hat{l}-1}} \to \mathbb{R}^{d_{\hat{l}}}$*, defined by* $f_{\hat{l}}(x) = \sigma(W_{\hat{l}}x + b_{\hat{l}})$*, describes the operation of the* $\hat{l}$*-th layer, where* $x \in \mathbb{R}^{d_{\hat{l}-1}}$ *is the input to the layer.*
- *The function* $f : \mathbb{R}^n \to \mathbb{R}^m$*, defined as* $f = f_{\hat{L}} \circ f_{\hat{L}-1} \circ \cdots \circ f_1$*, represents the overall* network operation *from input to output.*
- $\mathcal{L}(\cdot) : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$ *is the* loss function *that quantifies the error between the network's output* $\hat{y} = f(x)$ *and the target output* y*, which is crucial for training the network by adjusting the weights and biases.*

From Definitions 13 and 14, DRL involves modelling an agent **A** with a DNN-based function $f : \mathbb{R}^{d_s} \to \mathbb{R}^{d_a}$, where $d_s = \dim(\mathcal{S})$ and $d_a = \dim(\mathcal{A})$ denote the state and action space dimensions, and f aims to approximate π^* during training.

X. Wang, S. Wang, Liang, et al., 2024 discusses various families of methods employed to address DRL problems, including value-based, policy-based, and maximum entropy-based methods. In this dissertation, we utilise Proximal Policy Optimisation, a method belonging to the policy-based family, which is examined in greater detail in Section III.4.4.

III.4.3 Contextual Markov Decision Process

A Contextual Markov Decision Process (CMDP) extends MDPs (see Definition 12, page 143) by incorporating *context*. The goal of a CMDP is to learn a policy that optimises cumulative reward, accounting for varying hidden *static* parameters known as the *context* (Hallak, Di Castro, and Mannor, 2015). Below, we provide a formal definition.

Definition 15 (Contextual Markov Decision Process)**.** *A Contextual Markov Decision Process (CMDP) is formally described by the tuple* $\mathbf{M}^c = \langle \mathcal{C}, \mathcal{S}, \mathcal{A}, \mathcal{K}(c) \rangle$*, where:*

- $\mathcal{C}$ *is the* context space*, representing a set of all possible* static *parameters that influence the decision process.*
- $\mathcal{S}$ *and* $\mathcal{A}$ *denote the* state *and* action *spaces, respectively.*
- $\mathcal{K}(c)$ *is the function that maps any* context $c \in \mathcal{C}$ *to its corresponding* MDP *(see Definition 12, page 143). Each mapped* MDP *is characterised by the tuple* $\mathcal{K}(c) = \langle \mathcal{S}, \mathcal{A}, \mathbf{P}^{(c)}, \mathcal{R}^{(c)}, \pi_0^{(c)}, \gamma \rangle$*, where:*
  - $\pi_0^{(c)}$ *is the* initial probability distribution *influenced by* c*.*
  - $\mathbf{P}^{(c)}(s_{t+1}|s_t, a_t)$ *is the* transition probability function *influenced by* c*.*
  - $\mathcal{R}^{(c)}(s_t, a_t, s_{t+1})$ *is the* reward function *influenced by* c*.*

III.4.4 Proximal Policy Optimisation

The Proximal Policy Optimisation (PPO) algorithm (Schulman, Wolski, Dhariwal, et al., 2017) optimises an agent's policy to maximise expected returns while maintaining training stability and computational efficiency. PPO achieves this stability through an objective function that penalises large deviations from the previous policy, thereby keeping new and old policies closely aligned, reducing the risk of performance collapse. The introduction of a “clipped” surrogate objective function is a key innovation of PPO, enhancing stability and making it widely adopted in various DRL applications.

Clipped surrogate objective function $\mathbb{O}^{CLIP}(\varphi)$:

Let $T \subset \mathbb{R}$ represent a *time horizon* of fixed length, and let $t \in T$ denote a *time* indicating a decision point. The clipped surrogate objective function in PPO is given by:

$$\mathbb{O}^{CLIP}(\varphi) = \hat{\mathbb{E}}_t \left[\min(\rho_t(\varphi)\hat{A}_t, \text{clip}(\rho_t(\varphi), 1-\epsilon, 1+\epsilon)\hat{A}_t)\right], \qquad (6.11)$$

where φ denotes the *policy parameters.* The term $\rho_t(\varphi) = \frac{\pi_\varphi(a_t|s_t)}{\pi_{\varphi_{old}}(a_t|s_t)}$ represents the *probability ratio* of action $a_t \in \mathcal{A}$ in state $s_t \in \mathcal{S}$ under the *new* policy π_φ relative to the *old* policy $\pi_{\varphi_{old}}$. $\hat{A}_t$ is an estimator of the advantage function, indicating the relative benefit of an action. The clipping function clip$(\cdot)$ modifies $\rho_t(\varphi)$ so that it remains within the interval $[1-\epsilon, 1+\epsilon]$, here ϵ typically lies between 0.1 and 0.2. The expectation $\hat{\mathbb{E}}_t[\cdot]$ signifies an empirical average over a sample batch.

Advantage function $\hat{A}_t$:

The advantage function $\hat{A}_t$ is usually estimated using *generalised advantage estimation*, calculated as:

$$\hat{A}_t = \delta_t + (\gamma\zeta)\delta_{t+1} + (\gamma\zeta)^2\delta_{t+2} + \cdots + (\gamma\zeta)^{T-t-1}\delta_{T-1}, \qquad (6.12)$$

here, $\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$ is the temporal difference error for state $s_t \in \mathcal{S}$ and reward $r_t \in \mathcal{R}$; γ is the discount factor valuing current over future rewards, and $\zeta \in [0, 1]$ helps balance the bias-variance trade-off in $\hat{A}_t$. $V(s_{t+1})$ and $V(s_t)$ are the *value functions* (see Eq.6.7, page 144) for the states s_{t+1} and s_t, respectively.

Training with PPO:

Training involves sampling data through the agent’s (**A**) interaction with the environment (**E**) while executing the policy $\pi_{\varphi_{old}}$. Subsequently, the advantage function $\hat{A}_t$ (Eq. 6.12) is estimated, and the clipped surrogate objective function $\mathbb{O}^{CLIP}(\varphi)$ (Eq. 6.11) is optimised using *stochastic gradient descent* on the policy parameters φ. This process is repeated with the updated policy until meeting convergence. For details on PPO’s implementation see the documentation in (PPO:SB3).

References

Abraham, D. M., R. Wirahadikusumah, T. J. Short, and S. Shahbahrami (1998). "Optimization Modeling for Sewer Network Management". In: *Journal of Construction Engineering and Management* 124.5, pp. 402–410. DOI: 10.1061/(ASCE)0733-9364(1998)124:5(402).

Ana, E. and W. Bauwens (2007). "Sewer network asset management decision-support tools: a review". In: *International Symposium on New Directions in Urban Water Management*. Vol. 12. 14, pp. 1–8.

Arthur, S. and H. Crow (2007). "Prioritising sewerage maintenance using serviceability criteria". In: *Proceedings of the Institution of Civil Engineers - Water Management* 160.3, pp. 189–194. DOI: 10.1680/wama.2007.160.3.189.

Arthur, S., H. Crow, L. Pedezert, and N. Karikas (Apr. 2009). "The holistic prioritisation of proactive sewer maintenance". In: *Water Science and Technology* 59.7, pp. 1385–1396. ISSN: 0273-1223. DOI: 10.2166/wst.2009.134.

Arulkumaran, K., M. P. Deisenroth, M. Brundage, and A. A. Bharath (2017). "Deep Reinforcement Learning: A Brief Survey". In: *IEEE Signal Processing Magazine* 34.6, pp. 26–38. DOI: 10.1109/MSP.2017.2743240.

Assaf, G. and R. H. Assaad (2023). "Optimal Preventive Maintenance, Repair, and Replacement Program for Catch Basins to Reduce Urban Flooding: Integrating Agent-Based Modeling and Monte Carlo Simulation". In: *Sustainability* 15.11. ISSN: 2071-1050. DOI: 10.3390/su15118527.

Baah, K., B. Dubey, R. Harvey, and E. McBean (2015). "A risk-based approach to sanitary sewer pipe asset management". In: *Science of The Total Environment* 505, pp. 1011–1017. ISSN: 0048-9697. DOI: https://doi.org/10.1016/j.scitotenv.2014.10.040.

Breysse, D., E. Vasconcelos, and F. Schoefs (2007). "Management Strategies and Improvement of Performance of Sewer Networks". In: *Computer-Aided Civil and Infrastructure Engineering* 22.7, pp. 462–477. DOI: 10.1111/j.1467-8667.2007.00503.x.

Caradot, N., M. Riechel, M. Fesneau, N. Hernandez, A. Torres, H. Sonnenberg, E. Eckert, N. Lengemann, J. Waschnewski, and P. Rouault (2018). "Practical benchmarking of statistical and machine learning models for predicting the condition of sewer pipes in Berlin, Germany". In: *Journal of Hydroinformatics* 20.5, pp. 1131–1147. DOI: 10.2166/hydro.2018.217.

Caradot, N., P. R. Sampaio, A. Guilbert, H. Sonnenberg, V. Parez, and V. Dimova (2021). "Using deterioration modelling to simulate sewer rehabilitation strategy with low data availability". In: *Water Science and Technology* 83.3, pp. 631–640. DOI: 10.2166/wst.2020.604.

Cheng, J. C. and M. Wang (2018). "Automated detection of sewer pipe defects in closed-circuit television images using deep learning techniques". In: *Automation in Construction* 95, pp. 155–171. ISSN: 0926-5805. DOI: 10.1016/j.autcon.2018.08.006.

DeSilva, D., S. Burn, G. Tjandraatmadja, M. Moglia, P. Davis, L. Wolf, I. Held, J. Vollertsen, W. Williams, and L. Hafskjold (Feb. 2005). "Sustainable management of leakage from wastewater pipelines". In: *Water science and technology : a journal of the International Association on Water Pollution Research* 52, pp. 189–98. DOI: 10.2166/wst.2005.0459.

Elmasry, M., T. Zayed, and A. Hawari (2019). "Multi-Objective Optimization Model for Inspection Scheduling of Sewer Pipelines". In: *Journal of Construction Engineering and Management* 145.2, p. 04018129. DOI: 10.1061/(ASCE)CO.1943-7862.0001599.

F. Taillandier, S. M. E. and A. Bennabi (2020). "A decision-support framework to manage a sewer system considering uncertainties". In: *Urban Water Journal* 17.4, pp. 344–355. DOI: 10.1080/1573062X.2020.1781908.

Fang, X., W. Guo, Q. Li, J. Zhu, Z. Chen, J. Yu, B. Zhou, and H. Yang (2020). "Sewer Pipeline Fault Identification Using Anomaly Detection Algorithms on Video Sequences". In: *IEEE Access* 8, pp. 39574–39586. DOI: 10.1109/ACCESS.2020.2975887.

Fenner, R. (2000). "Approaches to sewer maintenance: a review". In: *Urban Water* 2.4. Sewer Systems and Processes, pp. 343–356. ISSN: 1462-0758. DOI: https://doi.org/10.1016/S1462-0758(00)00065-0.

Fenner, R. A., L. Sweeting, and M. J. Marriott (2000). "A new approach for directing proactive sewer maintenance". In: *Proceedings of the Institution of Civil Engineers-Water and Maritime Engineering*. Vol. 142. 2. Thomas Telford Ltd, pp. 67–77. DOI: 10.1680/wame.2000.142.2.67.

Fontecha, J. E., R. Akhavan-Tabatabaei, D. Duque, A. L. Medaglia, M. N. Torres, and J. P. Rodríguez (Mar. 2016). "On the preventive management of sediment-related sewer blockages: a combined maintenance and routing optimization approach". In: *Water Science and Technology* 74.2, pp. 302–308. ISSN: 0273-1223. DOI: 10.2166/wst.2016.160.

Fuchs-Hanusch, D., M. Günther, M. Möderl, and D. Muschalla (June 2015). "Cause and effect oriented sewer degradation evaluation to support scheduled inspection planning". In: *Water Science and Technology* 72.7, pp. 1176–1183. ISSN: 0273-1223. DOI: 10.2166/wst.2015.320.

Gueye, T., Y. Wang, and R. T. Mushtaq (2023). "Concrete deterioration detection in sewers using machine learning algorithms: an experiment-based study". In: *International Journal of Information Technology* 15.4, pp. 1949–1959. DOI: 10.1007/s41870-023-01231-9.

Hallak, A., D. Di Castro, and S. Mannor (2015). "Contextual Markov Decision Processes". In: *arXiv preprint arXiv:1502.02259*. DOI: 10.48550/arXiv.1502.02259.

Hamid Zaman Ahmed Bouferguene, M. A.-H. and C. Lorentz (2017). "Improving the productivity of drainage operations activities through schedule optimisation". In: *Urban Water Journal* 14.3, pp. 298–306. DOI: 10.1080/1573062X.2015.1112409.

Hernández, N., N. Caradot, H. Sonnenberg, P. Rouault, and A. Torres (2021). "Optimizing SVM models as predicting tools for sewer pipes conditions in the two main cities in Colombia for different sewer asset management purposes". In: *Structure and Infrastructure Engineering* 17.2, pp. 156–169. DOI: 10.1080/15732479.2020.1733029.

Inanloo, B., B. Tansel, K. Shams, X. Jin, and A. Gan (2016). "A decision aid GIS-based risk assessment and vulnerability analysis approach for transportation and pipeline networks". In: *Safety Science* 84, pp. 57–66. ISSN: 0925-7535. DOI: 10.1016/j.ssci.2015.11.018.

Jeung, M., J. Jang, K. Yoon, and S.-S. Baek (2023). "Data assimilation for urban stormwater and water quality simulations using deep reinforcement learning". In: *Journal of Hydrology* 624, p. 129973. ISSN: 0022-1694. DOI: https://doi.org/10.1016/j.jhydrol.2023.129973.

João A. Zeferino, A. P. A. and M. C. Cunha (2010). "Multi-objective model for regional wastewater systems planning". In: *Civil Engineering and Environmental Systems* 27.2, pp. 95–106. DOI: 10.1080/09540250802658988.

Kerkkamp, D., Z. Bukhsh, Y. Zhang, and N. Jansen (2022). "Grouping of Maintenance Actions with Deep Reinforcement Learning and Graph Convolutional Networks". English. In: *Proceeding of the 14th International Conference on Agents and Artificial Intelligence*. Vol. 2, pp. 574–585. DOI: 10.5220/0000155600003116.

Khurelbaatar, G., B. Al Marzuqi, M. Van Afferden, R. A. Müller, and J. Friesen (2021). "Data reduced method for cost comparison of wastewater management scenarios–case study for two settlements in Jordan and Oman". In: *Frontiers in Environmental Science* 9, p. 626634. DOI: 10.3389/fenvs.2021.626634.

Laakso, T., T. Kokkonen, I. Mellin, and R. Vahala (2019). “Sewer Life Span Prediction: Comparison of Methods and Assessment of the Sample Impact on the Results”. In: *Water* 11.12. ISSN: 2073-4441. DOI: 10.3390/w11122657.

Lee, J., C. Y. Park, S. Baek, S. H. Han, and S. Yun (2021). “Risk-Based Prioritization of Sewer Pipe Inspection from Infrastructure Asset Management Perspective”. In: *Sustainability* 13.13. ISSN: 2071-1050. DOI: 10.3390/su13137213.

Li, C., P. Zheng, Y. Yin, B. Wang, and L. Wang (2023). “Deep reinforcement learning in smart manufacturing: A review and prospects”. In: *CIRP Journal of Manufacturing Science and Technology* 40, pp. 75–101. ISSN: 1755-5817. DOI: https://doi.org/10.1016/j.cirpj.2022.11.003.

Lin, P., X.-X. Yuan, and E. Tovilla (2019). “Integrative modeling of performance deterioration and maintenance effectiveness for infrastructure assets with missing condition data”. In: *Computer-Aided Civil and Infrastructure Engineering* 34.8, pp. 677–695. DOI: 10.1111/mice.12452.

M.A. Cardoso, M. A. and M. S. Silva (2016). “Sewer asset management planning – implementation of a structured approach in wastewater utilities”. In: *Urban Water Journal* 13.1, pp. 15–27. DOI: 10.1080/1573062X.2015.1076859.

Marc Ribalta Ramon Bejar, C. M. and E. Rubión (2023). “Machine learning solutions in sewer systems: a bibliometric analysis”. In: *Urban Water Journal* 20.1, pp. 1–14. DOI: 10.1080/1573062X.2022.2138460.

Marugán, A. P. (2023). “Applications of Reinforcement Learning for maintenance of engineering systems: A review”. In: *Advances in Engineering Software* 183, p. 103487. ISSN: 0965-9978. DOI: 10.1016/j.advengsoft.2023.103487.

Marzouk, M. and M. Ibrahim (Nov. 2013). “Multiobjective optimisation algorithm for sewer network rehabilitation”. In: *Structure and Infrastructure Engineering* 9, pp. 1094–1102. DOI: 10.1080/15732479.2012.666254.

Montserrat, A., L. Bosch, M. Kiser, M. Poch, and L. Corominas (2015). “Using data from monitoring combined sewer overflows to assess, improve, and maintain combined sewer systems”. In: *Science of The Total Environment* 505, pp. 1053–1061. ISSN: 0048-9697. DOI: https://doi.org/10.1016/j.scitotenv.2014.10.087.

Mullapudi, A., M. J. Lewis, C. L. Gruden, and B. Kerkez (2020). “Deep reinforcement learning for the real time control of stormwater systems”. In: *Advances in Water Resources* 140, p. 103600. ISSN: 0309-1708. DOI: 10.1016/j.advwatres.2020.103600.

Obradović, D., M. Šperac, and S. Marenjak (2019). “Possibilities of using expert methods for sewer system maintenance optimisation”. In: *Građevinar* 71.9, pp. 769–779. DOI: 10.14256/JCE.2589.2018.

Ogunfowora, O. and H. Najjaran (2023). “Reinforcement and deep reinforcement learning-based solutions for machine maintenance planning, scheduling policies, and optimization”. In: *Journal of Manufacturing Systems* 70, pp. 244–263. ISSN: 0278-6125. DOI: 10.1016/j.jmsy.2023.07.014.

Puterman, M. L. (1990). “Markov decision processes”. In: *Handbooks in Operations Research and Management Science* 2, pp. 331–434. DOI: 10.1016/S0927-0507(05)80172-0.

Qasem, A. and R. Jamil (2021). “GIS-Based Financial Analysis Model for Integrated Maintenance and Rehabilitation of Underground Pipe Networks”. In: *Journal of Performance of Constructed Facilities* 35.5, p. 04021046. DOI: 10.1061/(ASCE)CF.1943-5509.0001623.

Ramos-Salgado, C., J. Muñuzuri, P. Aparicio-Ruiz, and L. Onieva (2022). “A comprehensive framework to efficiently plan short and long-term investments in water supply and sewer networks”. In: *Reliability Engineering & System Safety* 219, p. 108248. ISSN: 0951-8320. DOI: https://doi.org/10.1016/j.ress.2021.108248.

Real Torres, A. del, D. S. Andreiana, Á. Ojeda Roldán, A. Hernández Bustos, and L. E. Acevedo Galicia (2022). "A Review of Deep Reinforcement Learning Approaches for Smart Manufacturing in Industry 4.0 and 5.0 Framework". In: *Applied Sciences* 12.23. ISSN: 2076-3417. DOI: 10.3390/app122312377.

Schulman, J., F. Wolski, P. Dhariwal, A. Radford, and O. Klimov (2017). "Proximal policy optimization algorithms". In: *arXiv preprint arXiv:1707.06347*. DOI: 10.48550/arXiv.1707.06347.

Stable-Baselines3 Contributors (2024). *PPO - Proximal Policy Optimization*. https://stable-baselines3.readthedocs.io/en/master/modules/ppo.html. Accessed: 2024-08-28.

Tian, W., G. Fu, K. Xin, Z. Zhang, and Z. Liao (2024). "Improving the interpretability of deep reinforcement learning in urban drainage system operation". In: *Water Research* 249, p. 120912. ISSN: 0043-1354. DOI: https://doi.org/10.1016/j.watres.2023.120912.

Tian, W., Z. Liao, G. Zhi, Z. Zhang, and X. Wang (2022). "Combined Sewer Overflow and Flooding Mitigation Through a Reliable Real-Time Control Based on Multi-Reinforcement Learning and Model Predictive Control". In: *Water Resources Research* 58.7. e2021WR030703 2021WR030703, e2021WR030703. DOI: https://doi.org/10.1029/2021WR030703.

"Urban stormwater drainage management: The development of a multicriteria decision aid approach for best management practices" (2007). In: *European Journal of Operational Research* 181.1, pp. 338–349. ISSN: 0377-2217. DOI: 10.1016/j.ejor.2006.06.019.

Wang, X., S. Wang, X. Liang, D. Zhao, J. Huang, X. Xu, B. Dai, and Q. Miao (2024). "Deep Reinforcement Learning: A Survey". In: *IEEE Transactions on Neural Networks and Learning Systems* 35.4, pp. 5064–5078. DOI: 10.1109/TNNLS.2022.3207346.

Wirahadikusumah, R. and D. M. Abraham (2003). "Application of dynamic programming and simulation for sewer management". In: *Engineering, Construction and Architectural Management* 10.3, pp. 193–208. DOI: 10.1108/09699980310478449.

Wirahadikusumah, R., D. M. Abraham, and J. Castello (1999). "Markov decision process for sewer rehabilitation". In: *Engineering, Construction and Architectural Management* 6.4, pp. 358–370. DOI: 10.1108/eb021124.

Yang, M.-D. and T.-C. Su (2007). "An optimization model of sewage rehabilitation". In: *Journal of the Chinese Institute of Engineers* 30.4, pp. 651–659. DOI: 10.1080/02533839.2007.9671292.

Yao, L., Q. Dong, J. Jiang, and F. Ni (2020). "Deep reinforcement learning for long-term pavement maintenance planning". In: *Computer-Aided Civil and Infrastructure Engineering* 35.11, pp. 1230–1245. DOI: https://doi.org/10.1111/mice.12558.

Yin, Z., A. S. Leon, A. Sharifi, and M. H. Amini (n.d.). "Optimal Control of Combined Sewer Systems to Minimize Sewer Overflows by Using Reinforcement Learning". In: *World Environmental and Water Resources Congress 2023*, pp. 711–722. DOI: 10.1061/9780784484852.067.

Zhang, Z., W. Tian, and Z. Liao (2023). "Towards coordinated and robust real-time control: a decentralized approach for combined sewer overflow and urban flooding reduction based on multi-agent reinforcement learning". In: *Water Research* 229, p. 119498. ISSN: 0043-1354. DOI: https://doi.org/10.1016/j.watres.2022.119498.

Chapter 7

Maintenance Strategies for Sewer Pipes with Multi-State Deterioration and Deep Reinforcement Learning

Paper published at **L. A. Jimenez-Roa**, T. D. Simão, Z. Bukhsh, T. Tinga, H. Molegraaf, N. Jansen, M. Stoelinga, *"Maintenance Strategies for Sewer Pipes with Multi-State Degradation and Deep Reinforcement Learning"*, in *PHM Society European Conference*, vol. 8, no. 1, pp. 14, 2024, doi:10.36001/phme.2024.v8i1.4091.

Abstract

Large-scale infrastructure systems are crucial for societal welfare, and their effective management requires strategic forecasting and intervention methods that account for various complexities. Our study addresses two challenges within the Prognostics and Health Management (PHM) framework applied to sewer assets: modeling pipe deterioration across severity levels and developing effective maintenance policies. We employ Multi-State Deterioration Model (MSDM) to represent the stochastic deterioration process in sewer mains and use Deep Reinforcement Learning (DRL) to devise maintenance strategies. A case study of a Dutch sewer network exemplifies our methodology. Our findings demonstrate the model's effectiveness in generating intelligent, cost-saving maintenance strategies that surpass heuristics. It adapts its management strategy based on the pipe's age, opting for a passive approach for newer pipes and transitioning to active strategies for older ones to prevent failures and reduce costs. This research highlights DRL's potential in optimizing maintenance policies. Future research will aim improve the model by incorporating partial observability, exploring various reinforcement learning algorithms, and extending this methodology to comprehensive infrastructure management.

7.1 Introduction

Sewer network systems, crucial for public health, population well-being, and environmental protection, require maintenance to ensure their reliability and availability (M.A. Cardoso and Silva, 2016). This maintenance is challenged by limited budgets, environmental changes, ageing infrastructure, and hard-to-predict system deterioration (Tscheikner-Gratl, Caradot, Cherqui, et al., 2019).

Optimizing maintenance policies for sewer networks requires methodologies that can efficiently explore a broad solution space while adapting to the system's dynamic constraints and complexities. MPO addresses these needs by developing and analysing mathematical models to derive maintenance strategies (de Jonge and Scarf, 2020) that reduce maintenance costs, extend asset life, maximize availability, and ensure workplace safety (Ogunfowora and Najjaran, 2023).

This research explores the potential of DRL for MPO of sewer networks, first focusing on a component-level (i.e., pipe-level) analysis. DRL is a framework that merges neural network representation learning capabilities with RL, a branch of machine learning known for its effectiveness in sequential decision-making problems. RL is increasingly recognized for its role in developing cost-effective policies in MPO across diverse domains such as transportation, manufacturing, civil infrastructure and energy systems. It is emerging as a prominent paradigm in the search for optimal maintenance policies (Marugán, 2023).

This chapter aims to achieve two primary objectives: first, to present a comprehensive model for pipe-level MPO analysis facilitated by DRL, considering deterioration over the pipe length and employing inhomogeneous-time Markov chain models to simulate the non-linear stochastic behaviour associated with sewer main deterioration; second, to assess the efficacy of the model's policy through a case study of a large-scale sewer network in the Netherlands, comparing it with heuristics, including condition-based, scheduled, and reactive maintenance.

We acknowledge as limitations in our approach the focus on *fully observable* state spaces, which means that inspection actions are not necessary, and our analysis is at the *component-level.* Future research will aim to broaden this scope to include partially observable state spaces and system-level analysis.

Contributions. This work's primary contributions include:

(i) We propose a framework to carry out maintenance policy optimization for sewer mains considering the deterioration along the pipe length. This framework integrates Multi-State Deterioration Models (MSDMs) and Deep Reinforcement Learning (DRL).

(ii) Our framework introduces a novel approach by encoding the prediction of the MSDM into the state space, aiming to harness prognostics that describe the deterioration pattern of sewer mains.

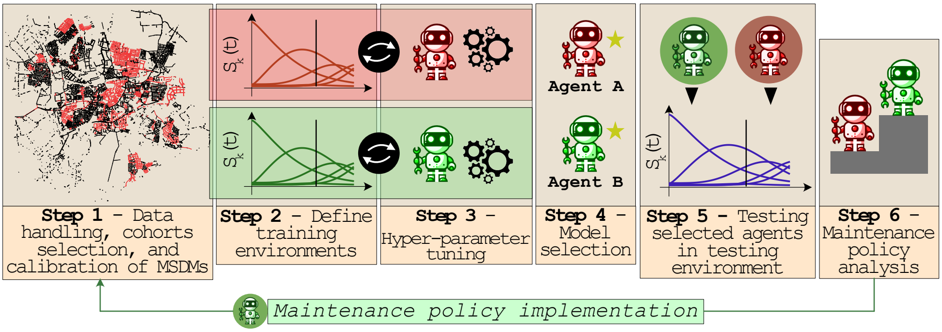

S_k(t)
S_k(t)
Agent A
Agent B
S_k(t)
Step 1 - Data handling, cohorts selection, and calibration of MSDMs
Step 2 - Define training environments
Step 3 - Hyper-parameter tuning
Step 4 - Model selection
Step 5 - Testing selected agents in testing environment
Step 6 - Maintenance policy analysis
Maintenance policy implementation

Figure 7.1: Methodology overview for sewer main maintenance policy optimization using Deep Reinforcement Learning and Multi-State Deterioration models.

(iii) We demonstrate that DRL has the potential to devise intelligent strategic maintenance strategies adaptable to various conditions, such as pipe age.

(iv) We provide our framework in Python and all data used in this study at zenodo.org/records/11258904.

Chapter outline. Section 7.2 presents the technical background. Section 7.3 outlines our research methodology. Section 7.4 formulates the MSDM. Section 7.5 details the framework for maintenance policy optimization via DRL. Section 7.6 presents our experimental setup. Section 7.7 analyses the results. Section 7.8 discusses findings, concludes, and suggests future research.

7.2 Technical background

Refer to the following sections for the technical background of this chapter: Markov chains (Section II.4.1, page 95); Multi-State Deterioration Models (MSDMs) (Section 6.2.1, page 126); and Markov Decision Process (Section III.4.1, page 143).

We aim to use Deep Reinforcement Learning (DRL) (Section III.4.2, page 143) to train agents in virtual environments with degradation patterns determined by the MSDM, as described in Section 7.5. For this, we apply Proximal Policy Optimisation (PPO) (Section III.4.4, page 145), a policy-based method in DRL.

7.3 Methodology

Our methodology, illustrated in Figure 7.1, comprises six steps, detailed below.

Step 1. Perform data handling of historical inspection records, selecting subsets (cohorts) of interest, and calibrating the MSDM on this data. This step is beyond the scope of this chapter; for details, see Jimenez-Roa, Heskes, Tinga, et al., 2022; Jimenez-Roa, Tinga, Heskes, et al., 2024. The results of this step are given in Section 7.4.

Step 2. After calibrating the MSDM, integrate these models into an environment suitable for RL applications. We present the details of our environment integrating MSDM in Section 7.5. In addition, we define environments for training RL agents. This is to test different MSDM hypotheses; details on this can be found in Section 7.6.

Step 3. Train DRL agents with PPO. Use `optuna` for hyper-parameter tuning and `Stable Baselines3` for RL implementation. Details are in Section 7.7.1.

Step 4. Train and select the RL agents with the optimal hyper-parameters on the *training* environments. In essence, these agents learn the dynamics described by the MSDM encoded in the environment.

Step 5. Compare the maintenance policies advised by the RL agents using the *test* environment against the heuristics: Condition-Based Maintenance (CBM), Scheduled Maintenance (SchM), and Reactive Maintenance (RM). Find the definition of these heuristics in Section 7.6.2.

Step 6. Analyse and compare the behaviour of the maintenance strategies for the different RL models and heuristics. Reflect on the policies advantages and disadvantages. Find in Section 7.7.2 the overview of this comparison, and in Section 7.7.3 are the details along episodes.

7.4 Multi-state deterioration models

7.4.1 Case study

The case study is detailed in Section II.4.3 on page 100. In this chapter, we focused on the the damage code `BAF`, which signifies *surface damage* and was observed in 35.3% of the inspections.

7.4.2 Parametrisation

We consider three hazard rate distributions: Exponential, Gompertz, and Weibull. The hazard rates $\lambda(t|\cdot)$ are as follows: The Exponential distribution (Eq. (7.1a)) has a constant hazard rate, implying a *homogeneous* time with *memoryless* properties. In contrast, the Gompertz (Eq. (7.1b)) and Weibull (Eq. (7.1c)) distributions exhibit varying hazard rates, indicating *inhomogeneous* time.

$$\textit{Exponential}\text{ hazard function:}\quad \lambda^E(t|\epsilon) = \epsilon, \tag{7.1a}$$

$$\textit{Gompertz}\text{ hazard function:}\quad \lambda^G(t|\alpha,\beta) = \alpha\beta e^{\beta t} \tag{7.1b}$$

$$\textit{Weibull}\text{ hazard function:}\quad \lambda^W(t|\eta,\rho) = \frac{\rho}{\eta}\left(\frac{t}{\eta}\right)^{\rho-1} \tag{7.1c}$$

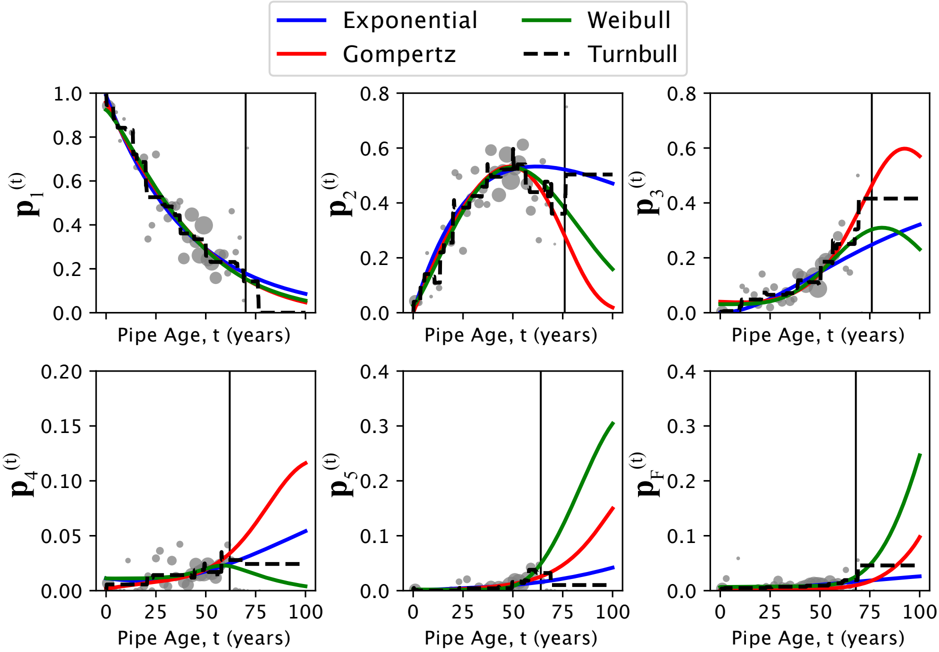


Figure 7.2: Probability of being in state $k \in \mathbb{S}$ at pipe age. The hazard functions are parametrised using the Exponential, Gompertz, and Weibull probability density functions. The Turnbull non-parametric estimator indicates the ground truth. The grey circles indicate the frequency based on the inspection dataset.

7.4.3 Solving the Multi-State Deterioration Model

Figure 6.1 (page 126) defines the Markov chain structure to model deterioration in a sewer main, and Section 7.4.2 introduced the hazard rate functions. The corresponding system of differential equations is presented in Eq. 6.2 (page 127) and is solved numerically using the `LSODA` algorithm from the FORTRAN `odepack` library implemented in SciPy (Virtanen, Gommers, Oliphant, et al., 2020), which employs the `Adams/BDF` method with automatic stiffness detection.

7.4.4 Parametric Multi-State Deterioration Models

We extract a subset from our case study dataset to construct a cohort with concrete sewer mains carrying *mixed and waste content* (cohort `CMW`), representing 37.1% of the sewer network. The model parameters for this cohort are detailed in Appendix D in Tables D.1 and D.2.

Figure 7.2 illustrates the MSDMs predictions, detailing the stochastic dynamics of sewer main deterioration for pipes in cohort `CMW`. As Figure 6.1 describes, this deterioration is segmented into five sequentially ordered severity levels ($k = 1$ to $k = 5$), plus a functional failure state ($k = F$). Differences in the y-axis scales are

Table 7.1: RMSE with respect Turnbull estimator, per severity level k and total RMSE, cohort: `CMW`.

	Exponential	Gompertz	Weibull
$\boldsymbol{p}_{k=1}(t)$	3.38E-02	3.27E-02	3.34E-02
$\boldsymbol{p}_{k=2}(t)$	7.04E-02	3.70E-02	3.57E-02
$\boldsymbol{p}_{k=3}(t)$	6.27E-02	2.81E-02	4.38E-02
$\boldsymbol{p}_{k=4}(t)$	4.28E-03	1.13E-02	5.06E-03
$\boldsymbol{p}_{k=5}(t)$	8.33E-03	1.09E-02	3.04E-02
$\boldsymbol{p}_{k=F}(t)$	9.19E-03	1.17E-02	3.62E-03
Total	4.13E-02	2.45E-02	2.96E-02

intentional, to emphasise details and behaviours that various deterioration models express across severity levels.

Gray circles represent the frequency per severity level from the inspection dataset. Jimenez-Roa, Heskes, Tinga, et al., 2022 details how these frequencies are computed. Vertical black lines in Figure 7.2 mark the last available data point for each severity level.

Additionally, Figure 7.2 presents the *Turnbull* non-parametric estimator, which assumes no specific distribution for survival times (Turnbull, 1976). In our context, this estimator represents the ground truth of stochastic deterioration behaviour in sewer mains.

Tables 7.1 presents the Root Mean Square Error (RMSE) computed with respect to the Turnbull estimator, for each MSDM assumption, for cohorts `CMW`. These results show that models employing Gompertz and Weibull distributions yield smaller RMSEs compared to the one using the Exponential distribution.

These MSDMs serve two crucial roles within our environment: first, they drive the deterioration behaviour of sewer mains, effectively emulating how sewer mains degrade over time. Second, the output from the MSDMs is incorporated as prognostic information, available to the agent to support decisions at any time point. This latter aspect is considered a novel feature of our framework. Details on the MDP are provided in the section below.

7.5 Definition of Markov Decision Process for Maintenance Policy Optimisation of a Sewer Main considering pipe length deterioration

Figure 7.3 provides the workflow that the RL agent uses to learn maintenance policies for sewer mains, considering deterioration along the pipe length. In the

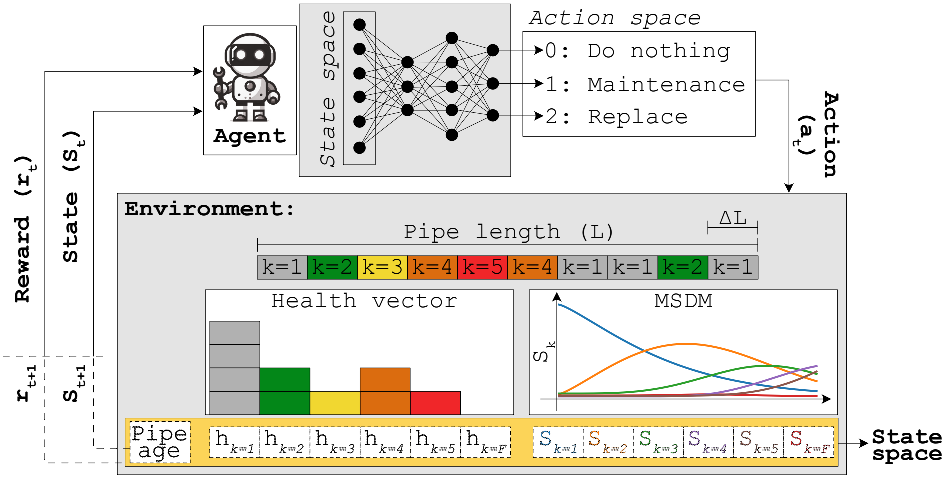


Figure 7.3: Environment for maintenance policy optimisation of a sewer main via Deep Reinforcement Learning, considering deterioration along the pipe length.

following sections, we provide the details of the environment, namely the state and action spaces, as well as the transition probability and reward functions.

7.5.1 State space $\mathbb{S}$

Our approach focuses on developing age-based maintenance policies, incorporating the sewer main's age into the state representation. Our state space is *continuous* and it is structured to include three key components: (i) the age of the pipe, (ii) the *health vector*, and (iii) the stochastic prediction of severity levels. We next describe the last two components.

Health vector (h)

In modelling the deterioration of linear structures like sewer mains, it is important to represent changes accurately along their length. For this purpose, we define a *health vector* (**h**), which quantitatively measures the deterioration at various points along the pipe. The vector is crucial in our framework, particularly influencing the reward function as described in Section 7.5.4.

Construction of h: We discretise the pipe into segments of equal length ΔL, with $\Delta L < L$, where L is the total length of the pipe. The number of segments, $\boldsymbol{\nu}$, is calculated using the ceiling function to ensure it remains an integer even if L is not perfectly divisible by ΔL:

$$\boldsymbol{\nu} = \left\lceil \frac{L}{\Delta L} \right\rceil \tag{7.2}$$

Each segment's deterioration level is initially assessed and categorised into *severity levels* according to the MSDM. As the deterioration progresses, the state of each segment changes following the transition probabilities described by the matrix $\mathbf{P}_{ij}$ where i is the current severity level, and j is the subsequent severity level, as described by the forward Kolmogorov equation (Eq. 4.2, page 98).

Notice that by doing this, we assume there is no statistical dependency between segments, which is a strong assumption that needs further research. However, for simplicity, we maintain this assumption in our deterioration model.

Quantifying Deterioration: The distribution of severity levels across the pipe is captured in vector $\mathbf{d}$, with each element indicating the severity level of a segment. To quantify this distribution in the health vector $\mathbf{h}$, we first count the number of segments at each severity level k using the following expression:

$$\boldsymbol{\nu}_k = \sum_{i=1}^{|\boldsymbol{\nu}|} \mathbf{1}_{\{\mathbf{d}_i = k\}}, \tag{7.3}$$

where $\mathbf{1}$ is the indicator function that is 1 if the condition is true and 0 otherwise. The health vector $\mathbf{h}$ is then determined by normalising these counts to reflect the proportion of segments at each severity level:

$$\mathbf{h}_k = \frac{\boldsymbol{\nu}_k}{\boldsymbol{\nu}}, \tag{7.4}$$

where $\boldsymbol{\nu}_k$ is the number of segments at severity level k. Thus, $\mathbf{h}_k$ becomes part of the state space indicating the *level of deterioration* present in the pipe.

Stochastic prediction of severity levels

To enable the agent to access information provided by the MSDM, we incorporate the prediction of severity levels into the state space. This is accomplished by solving Eq. 4.3, yielding a distribution $\boldsymbol{p}_k(t)$.

Finally, our state space is defined as a tuple with 13 elements:

$$\mathcal{S} = \langle \text{Pipe Age}, \mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3, \mathbf{h}_4, \mathbf{h}_5, \mathbf{h}_F, \boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3, \boldsymbol{p}_4, \boldsymbol{p}_5, \boldsymbol{p}_F \rangle$$

7.5.2 Action space $\mathcal{A}$

Our action space $\mathcal{A}$ is *discrete* with dimensionality $|\mathcal{A}| = 3$. At each time step t, the agent selects an action a. If the decision at time t is *do nothing*, a_t is set to 0. To perform *maintenance*, a_t is set to 1, and to *replace* the pipe, a_t is set to 2. The outcomes of these actions are discussed in Section 7.5.3.

7.5.3 Transition probability function P

Our transition function $\mathbf{P}(s_{t+1}|s_t, a)$ is *stochastic*, dependent on time t, and considers both the actions $a \in \mathcal{A}$ and the current s_t and next state s_{t+1} dynamics described by the MSDM. We illustrate the behaviour of $\mathbf{P}$ with the following example.

For a 30-year-old pipe with length $L = 40$ meters and discretised in segments of length $\Delta L = 1$, let the current state space be $s_{t=30} \in \mathcal{S}$:

$$s_{t=30} = \langle 30, 0.60, 0.35, 0.025, 0.025, 0.0, 0.0, \\ 0.475, 0.436, 0.069, 0.010, 0.005, 0.005\rangle\,.$$

$s_{t=30}$ indicates the age of the pipe is 30 years. From Eq. 7.4, the number of segments at severity k is determined by multiplying the health vector ($\mathbf{h}_k$):

$$\mathbf{h}_k = [0.60, 0.35, 0.025, 0.025, 0.0, 0.0]$$

by 40 meters, yielding $\boldsymbol{\nu}_k = [24, 14, 1, 1, 0, 0]$, indicating that, out of the 40 meters of pipe length, 24 segments of 1 meter are at severity $k = 1$, 14 at severity $k = 2$, and so forth.

The distribution $\boldsymbol{p}_k(t = 30.0)$ predicts the probability of being in a severity level k at age $t = 30$. This is achieved by evaluating $t = 30.0$ in the corresponding MSDM.

$$\boldsymbol{p}_k(t = 30.0) = [0.475, 0.436, 0.069, 0.010, 0.005, 0.005]$$

Assuming the agent takes an action every half year, we illustrate the effect of each action in $\mathcal{A}$ below.

- **If** $a_t = 0$: the agent decides to "do nothing", the pipe's degradation evolves in line with the MSDM progression. Here the new state space becomes $s^{a=0}_{t=30.5}$.

  $$s^{a=0}_{t=30.5} = \langle 30.5, 0.575, 0.35, 0.05, 0.025, 0.0, 0.0, \\ 0.470, 0.439, 0.071, 0.010, 0.05, 0.05\rangle$$

  Notice that the pipe age increased to 30.5, and $\boldsymbol{\nu}_k = [23, 14, 2, 1, 0, 0]$, where a segment with severity $k = 1$ progressed to $k = 2$, and one segment with $k = 2$ advanced to $k = 3$. Additionally, $\boldsymbol{p}_k(t)$ is updated by evaluating $t = 30.5$.
- **If** $a_t = 1$: the agent decides to "perform maintenance," all damage points with severity levels $k \in \{3, 4, 5\}$ are moved to $k = 2$. Consequently, this action does not affect damage points with severity levels $k \in \{1, 2, F\}$. The new state space becomes $s^{a=1}_{t=30.5}$.

$$
\begin{aligned}
s^{a=1}_{t=30.5} = \langle & 30.5, 0.60, 0.40, 0.0, 0.0, 0.0, 0.0, \\
& 0.47, 0.439, 0.071, 0.010, 0.05, 0.05\rangle
\end{aligned}
$$

Notice that the pipe age increased to 30.5, and $\boldsymbol{\nu}_k = [24, 16, 0, 0, 0, 0]$. However, $\boldsymbol{p}_k(t)$ is updated by evaluating $t = 30.5$, same as when $a_t = 0$.

- **If** $a_t = 2$: the agent decides to "replace" the pipe, resetting its condition to as good-as-new. The new state space is $s^{a=2}_{t=0.0}$:

$$
\begin{aligned}
s^{a=2}_{t=0.0} = \langle & 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, \\
& 0.986, 0.014, 0.0, 0.0, 0.0, 0.0\rangle.
\end{aligned}
$$

The pipe age is reset to 0.0, with $\boldsymbol{\nu}_k = [40, 0, 0, 0, 0, 0]$, and $\boldsymbol{p}_k(t)$ is updated for $t = 0.0$.

7.5.4 Reward function $\mathcal{R}$

Our reward function $\mathcal{R}(s_t, a_t, s_{t+1})$ assigns a reward r_t at every decision point t, determined by the current state s_t and action a_t. This function integrates the costs of maintenance (C_M), replacement (C_R), and failures (C_F). $\mathcal{R}$ is *sparse* because it issues a non-zero value only when failures occur or interventions are undertaken.

Maintenance cost C_M is calculated as per Eq. 7.5, where it combines a variable cost based on severity k with a fixed logistic cost of €500, covering the expenses related to maintenance.

Table 7.2: Maintenance costs per severity k per segment (c^k_M)

	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=F$
$c^k_M =$	0	0	-€500	-€700	-€900	N.A.

These costs vary with the severity level k, as detailed in Table 7.2. Note that no maintenance costs are associated with $k = F$ because maintenance cannot be performed on a segment that has already failed. In this case, the agent must replace. Replacement costs (C_R) is computed with Eq. 7.6:

$$C_M = -\sum_{i \in k} (\boldsymbol{\nu}_k \cdot c^k_M) - 500 \tag{7.5}$$

$$C_R = -(450 + 0.66D + 0.0008D^2)L \tag{7.6}$$

Here, L and D denote the pipe's length in meters and diameter in millimetres, respectively. C_R is in Euros (€).

The cost of failure, denoted by C_F, entails assigning a substantial penalty when the agent allows a segment of the pipe to achieve a failure state ($k = F$). This penalty cost is established at €-100,000. Our reward function is then:

$$r_t = \frac{C_M + C_R + C_F}{100,000 + 900 \times 40} = \frac{C_M + C_R + C_F}{136,000}, \tag{7.7}$$

where r_t represents the reward obtained at time t, the normalisation constant $136,000$ corresponds to the most expensive penalty possible at time t. Thus, r_t is defined within the interval $[-1, 0]$. This reward function aims for the agent to balance maintenance actions with the prevention of undesirable pipe conditions.

7.6 Experimental setup

7.6.1 Setup

We will evaluate our framework with a single pipe of constant length (40 meters) and diameter (200 mm) from the cohort `CMW`, which carries mixed and waste content. Given the constant dimensions, the replacement cost C_R, as defined in Eq. 7.6, is €24,560. The pipe age, when initialising the episode, is randomly sampled from the uniform distribution $U \sim [0, 50]$, allowing the agent to learn the behaviour of pipes within this age range. Additionally, we evaluate the policy in steps of half a year and $\Delta L = 1$ meter.

In the methodology section, we describe the training of two agents: **Agent-E** and **Agent-G**. **Agent-E** is trained in an environment where sewer main deterioration follows the MSDM parametrised with an *Exponential* probability density function, while **Agent-G** is trained in an environment where deterioration follows the MSDM parametrised with a *Gompertz* probability density function.

Both agents are tested in an environment where sewer main deterioration follows the MSDM parametrised with the *Weibull* probability density function. During training, each agent follows a specific state space, defined as follows:

$$\mathcal{S}^{\textbf{Agent-E}}_{\text{Training}} = \langle \text{Pipe Age}, \mathbf{h}^{E}_{k}, \boldsymbol{p}^{E}_{k}(t) \rangle \tag{7.8a}$$

$$\mathcal{S}^{\textbf{Agent-G}}_{\text{Training}} = \langle \text{Pipe Age}, \mathbf{h}^{G}_{k}, \boldsymbol{p}^{G}_{k}(t) \rangle \tag{7.8b}$$

Here, $\mathcal{S}$ represents the state space for each agent during training. The subscripts E and G denote the *Exponential* and *Gompertz* probability density functions, respectively. Each agent's objective is to learn an optimal maintenance strategy based on their environment's dynamics.

For testing, both agents are evaluated in the same environment, with the state space defined as follows:

$$\mathcal{S}^{\textbf{Agent-E}}_{\text{Testing}} = \langle \text{Pipe Age}, \mathbf{h}^{W}_{k}, \boldsymbol{p}^{E}_{k}(t) \rangle \tag{7.9a}$$

$$\mathcal{S}^{\textbf{Agent-G}}_{\text{Testing}} = \langle \text{Pipe Age}, \mathbf{h}^{W}_{k}, \boldsymbol{p}^{G}_{k}(t) \rangle \tag{7.9b}$$

In both cases, $\boldsymbol{p}^{E}_{k}(t)$ and $\boldsymbol{p}^{G}_{k}(t)$ remain consistent with the training phase, reflecting the MSDM predictions. However, the health vector $\mathbf{h}_k$ follows the deterioration behaviour described by the *Weibull* probability density function, indicated by the subscript W.

7.6.2 Comparison of maintenance strategies

We compare the RL agent's performance against maintenance policies based on heuristics. For this, we define the following:

- **Condition-Based Maintenance (`CBM`)**: Maintenance actions are based on the sewer main's condition. Specifically, replacement ($a_t = 2$) is performed if `pipe_age` ≥ 70 or $\mathbf{h}_{k=F} \geq 0.0$; maintenance ($a_t = 1$) is conducted if $\mathbf{h}_{k=4} \geq 0.1$ or $\mathbf{h}_{k=5} \geq 0.05$; otherwise, no action ($a_t = 0$) is taken.
- **Scheduled Maintenance (`SchM`)**: Actions are time-based. Replacement ($a_t = 2$) is executed if $\mathbf{h}_{k=F} \geq 0.0$; maintenance ($a_t = 1$) occurs every 10 years; otherwise, no action ($a_t = 0$) is taken.
- **Reactive Maintenance (`RM`)**: Replacement is undertaken only upon pipe failure, i.e., replacement ($a_t = 2$) is performed if $\mathbf{h}_{k=F} \geq 0.0$; otherwise, no action ($a_t = 0$) is taken.

Note that `CBM` and `SchM` are defined based on plausible values. However, these heuristics can be further calibrated for enhanced performance, which is beyond the scope of this chapter.

7.7 Results

7.7.1 Implementation and hyper-parameter tuning

Our framework uses `Stable Baselines3` (Raffin, Hill, Gleave, et al., 2021), comprising robust implementations of RL algorithms in PyTorch (Ansel, Yang, He, et al., 2024). Specifically, we utilise the PPO algorithm. Hyper-parameter optimisation is performed using `optuna` (Akiba, Sano, Yanase, et al., 2019), a framework dedicated to automating the optimisation of hyper-parameters.

The search space encompasses: exponentially-decaying learning rate with a decay rate of 0.05, with an initial learning rate ranging from 10^{-5} to 10^{-2}, discount factor (γ) from 0.8 to 0.9999, entropy coefficient from 0.0001 to 0.01, steps per update (`n_steps`) from 250 to 3000, batch sizes from 16 to 256, activation functions (σ)

Table 7.3: Optimal hyper-parameters found using `optuna` (Akiba, Sano, Yanase, et al., 2019).

Hyper-parameter	Value
Learning rate	0.0003
Discount factor (γ)	0.995
Entropy coefficient	0.008
Steps per update (`n_steps`)	2,080
Batch size	104
Activation function (σ)	Sigmoid
Policy network architecture	[32, 32, 32]
Training epochs (`n_epochs`)	50

('tanh', 'relu', 'sigmoid'), policy network architectures ([16, 16], [32, 32], [64, 64], [32, 32, 32]), and training epochs (`n_epochs`) from 5 to 100.

We set up `optuna` to conduct 500 trials, aiming to maximise cumulative reward in 100 episodes. Table 7.3 details the optimal hyper-parameters identified. These parameters are used to obtain the results discussed in Sections 7.7.2 and 7.7.3, where our agents are trained over a total of 5 million time steps.

7.7.2 Policy analysis: overview

This section offers a broad evaluation of the policies, with a detailed analysis over episodes presented in Section 7.7.3. We compare the agents' performances with the heuristics detailed in Section 7.6.2 across 100 simulations in the **test** environment (Eq. 7.9), considering pipe ages of 0, 25, and 50 years, aiming to evaluate policy efficacy concerning deterioration over varying pipe ages.

Table 7.4: Policy cost comparison: Mean and standard deviation (Std.) of costs for Agent-E, Agent-G, CBM, SchM, and RM, evaluated over 100 episodes in the test environment. Costs, in thousands of Euros (€), for pipe ages of 0, 25, and 50 years.

	Pipe age: 0		Pipe age: 25		Pipe age: 50	
Policy	Mean	Std.	Mean	Std.	Mean	Std.
Agent-E	51.3	80.8	116.5	97.7	156.8	121.2
Agent-G	**39.7**	66.2	**78.7**	96.6	**127.1**	128.3
CBM	51.3	107.2	112.3	88.5	**110.7**	86.6
SchM	**42.5**	70.9	**78.9**	96.4	159.8	95.9
RM	48.6	76.6	135.8	86.5	165.7	80.8

Table 7.4 presents the *mean policy cost* for Agent-E, Agent-G, `CBM`, `SchM`, and `RM`, highlighting the best and second-best policies in blue and red, with corresponding means and standard deviations from the simulations.

Table 7.5: Percentage (%) of actions per policy obtained with Agent-E, Agent-G, CBM, SchM, and RM, evaluated over 100 episodes in the test environment, for different pipe ages.

Pipe age	Action	Agent-E	Agent-G	CBM	SchM	RM
0	$a_t = 0$	99.5%	97.51%	99.54%	94.76%	99.61%
	$a_t = 1$	0.0%	2.21%	0.05%	4.95%	0.00%
	$a_t = 2$	0.5%	0.28%	0.41%	0.29%	0.39%
25	$a_t = 0$	98.81%	94.96	% 98.14%	94.56%	98.92%
	$a_t = 1$	0.00%	4.50%	0.62%	4.94%	0.00%
	$a_t = 2$	1.19%	0.53%	1.24%	0.50%	1.08%
50	$a_t = 0$	98.4%	94.52%	98.05%	93.99%	98.68%
	$a_t = 1$	0.0%	4.43%	0.67%	4.88%	0.00%
	$a_t = 2$	1.6%	1.05%	1.28%	1.13%	1.32%

From these results, we observe that Agent-G's policy generally outperforms others for pipe ages of 0 and 25 years, securing a second-best position for pipes aged 50 years. It is noted that the cost of all policies increases with pipe age, which aligns with expectations as older pipes require more interventions.

After reviewing the mean policy costs, our focus shifts to the specific actions involved in each policy. Table 7.5 provides a summary of the actions executed by each policy across simulations for different pipe ages. For new pipes, the SchM policy leads in maintenance activities ($a_t = 1$), with Agent-G following. In terms of replacements ($a_t = 2$), Agent-E is the foremost in implementing this action, with CBM in second place. Both Agent-G and SchM exhibit lower replacement frequencies, explaining the mean policy costs since maintenance actions incur lower expenses compared to the penalties and replacement costs resulting from pipe failures.

For pipes aged 25 years, Agent-G executes more maintenance actions ($a_t = 1$), similar to SchM. Agent-E opts for no maintenance, aligning more with RM's strategy. Although CBM carries out some maintenance actions, replacement actions predominate, indicating a greater tendency to permit pipe failures, which explains the observed differences in mean policy costs.

For pipes aged 50 years, CBM offers the most cost-effective policy, with Agent-G's following. CBM conducts fewer maintenance actions and more replacements than Agent-G, accounting for the cost disparity. The policies of Agent-E, RM, and SchM have similar costs. Despite SchM conducting more maintenance, its high number of replacements suggests the maintenance interval requires adjustment. These results indicate that the strategies of CBM, SchM, and RM are less efficient for older pipes due to their higher failure probability.

Regarding the *mean pipe severity level* to assess the impact of various policies on pipe deterioration, as shown in Table 7.6. Our analysis reveals a notable correlation

Table 7.6: Percentage (%) of severity level per policy obtained with Agent-E, Agent-G, CBM, SchM, and RM, evaluated over 100 episodes in the test environment for different pipe ages.

Pipe age	Severity	Agent-E	Agent-G	CBM	SchM	RM
0	$k = 1$	59.77%	58.75%	59.94%	59.84%	58.88%
	$k = 2$	33.27%	39.14%	32.67%	38.05%	33.15%
	$k = 3$	5.39%	1.70%	6.00%	1.79%	6.36%
	$k = 4$	1.38%	0.28%	1.13%	0.26%	1.30%
	$k = 5$	0.18%	0.13%	0.25%	0.04%	0.31%
	$k = F$	0.01%	0.01%	0.01%	0.01%	0.01%
25	$k = 1$	50.49%	41.72%	46.88%	39.07%	46.62%
	$k = 2$	38.96%	55.27%	43.09%	55.55%	40.86%
	$k = 3$	8.37%	2.63%	8.48%	4.85%	9.80%
	$k = 4$	1.37%	0.29%	1.18%	0.41%	1.51%
	$k = 5$	0.78%	0.07%	0.36%	0.10%	1.18%
	$k = F$	0.02%	0.01%	0.02%	0.01%	0.03%
50	$k = 1$	57.93%	44.65%	55.01%	40.92%	54.36%
	$k = 2$	32.58%	51.40%	36.14%	50.46%	33.09%
	$k = 3$	7.50%	3.29%	7.20%	7.34%	9.32%
	$k = 4$	1.31%	0.39%	1.19%	0.59%	1.64%
	$k = 5$	0.65%	0.25%	0.43%	0.67%	1.55%
	$k = F$	0.03%	0.02%	0.02%	0.03%	0.03%

between the average actions per policy, detailed in Table 7.5, and the mean pipe severity level. Specifically, the Agent-G control strategy tends to maintain pipes within a severity level of $k \in [1, 2, 3]$, whereas the Agent-E, CBM, SchM, and RM policies often result in higher severity levels $k \in [4, 5, F]$, which correlates with increased policy costs.

To summarise, our findings indicate that the Agent-G's policy, derived using DRL, implements a dynamic management strategy that varies with the pipe's age. This strategy encompasses a more passive approach with new pipes, transitioning to active intervention as the pipes age. This indicates the agent's preference for more frequent maintenance actions rather than allowing pipe failures, which incur higher penalties and replacement costs.

Moreover, Agent-G outperforms Agent-E, illustrating the impact of the deterioration model assumption. Specifically, Agent-G's prognostic model used during training aligns more closely with the test environment's deterioration pattern than Agent-E's, potentially explaining why Agent-G is better equipped to navigate and understand the deterioration pattern. This, in turn, enables it to devise a more effective maintenance policy by leveraging a more accurate deterioration model.

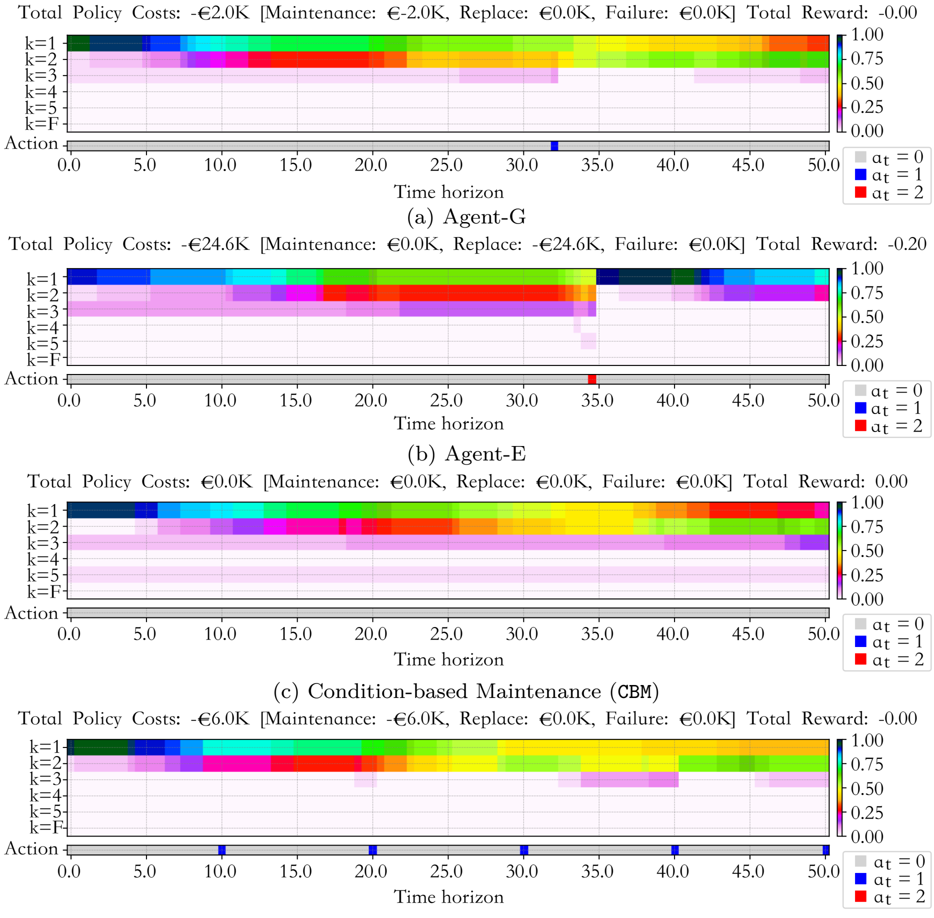
7.7.3 Policy analysis over episode

In Section 7.7.2, we present an overview of policy performances. This section delves into the details per episode to provide further understanding on these policies. Figures 7.4, 7.5, and 7.6 detail the performance of the Agent-E, Agent-G, `CBM`, and `SchM` policies for pipes with ages 0, 25 and 50, respectively. The `RM` heuristic is excluded from this analysis due to its straightforward approach: allowing the pipe to fail before replacing it.

Figure 7.4 shows that for a brand new pipe: (a) Agent-G performs maintenance on the pipe at approximately 32 years old; (b) Agent-E opts to replace the pipe when it is around 35 years old, which may be attributed to the presence of elements with higher severity levels in that specific episode; (c) `CBM` chooses not to act, which results in the least expensive policy in this comparison. However, it is observed that some pipe sections reach severity level $k = 5$ throughout the episode. Not taking any action is deemed risky since progressing to $k = F$ becomes more likely and incurs higher costs; (d) `SchM` effectively controls severity levels but is more expensive than Agent-G's policy due to more frequent maintenance actions.

Figure 7.5 shows that for a pipe aged 25: (a) Agent-G exhibits increased activity, indicating more frequent maintenance actions, especially as the pipe ages to 50, shortening the maintenance intervals; (b) Agent-E postpones any action until the pipe fails, at which point it replaces the pipe with a new one, akin to `RM`; (c) `CBM` also initiates maintenance around the pipe's 50-year mark. However, deterioration escalates from age 60, leading to failure at 66. The inability to manage this increased severity results in significant penalty costs, diminishing the effectiveness of this policy; (d) Similarly, `SchM` manages severity levels effectively until the pipe reaches approximately 70 years of age, at which point deterioration accelerates, resulting in failure at 73.

Figure 7.6 shows that for a pipe aged 50: (a) Agent-G opts to replace the pipe at age 50, followed by maintenance in the subsequent time step. This decision is likely influenced by parts of the pipe being at severity levels $k \in 3, 4$. Such a scenario is plausible, as new pipes can exhibit high severity levels at a young age due to defects in the material or errors during the construction and installation process. This concept is represented in the MSDM by the initial probability state vector (S_k^0). Additionally, Agent-G recommends maintenance at the interval when the pipe reaches the age of 26 years; (b) Agent-E suggests replacement at approximately 62 years, without recommending further maintenance; (c) `CBM` advocates for maintenance at about 65 years, followed by replacement at 70 years, in line with heuristics described in Section 7.6.2; (d) `SchM` consistently performs maintenance at regular intervals, yet faces significant deterioration, culminating in failure around 97 years.



(a) Agent-G

(b) Agent-E

(c) Condition-based Maintenance (`CBM`)

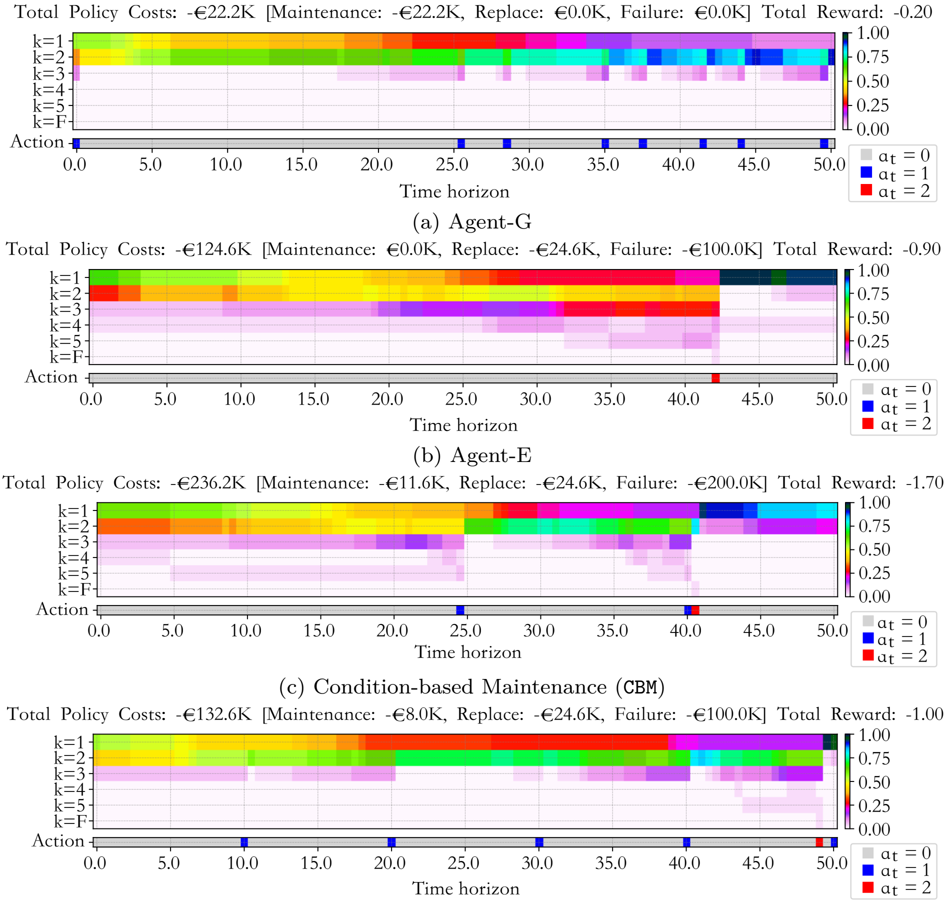
(d) Scheduled Maintenance (`SchM`)

Figure 7.4: Policies behaviour over an episode for a **new pipe**, showing the health vector over the pipe age and actions per policy: (a) Agent-G, (b) Agent-E, (c) Condition-based Maintenance (`CBM`), and (d) Scheduled Maintenance (`SchM`).

7.8 Discussion and Conclusions

In this chapter, we explore the applications of Prognostics and Health Management (PHM) in sewer main asset management. Our study focuses on component-level (i.e., pipe-level) maintenance policy optimisation by integrating stochastic multi-state deterioration modelling and Deep Reinforcement Learning (DRL). The goal is to assess the effectiveness of DRL in deriving cost-effective maintenance strategies tailored to the specific conditions and requirements of sewer mains.

A key contribution of our work is the integration of prognostics models with a maintenance policy optimisation framework. We utilise a tailored reward function that aligns with damage severity levels, enabling a more complex and realistic



(a) Agent-G

(b) Agent-E

(c) Condition-based Maintenance (CBM)

(d) Scheduled Maintenance (SchM)

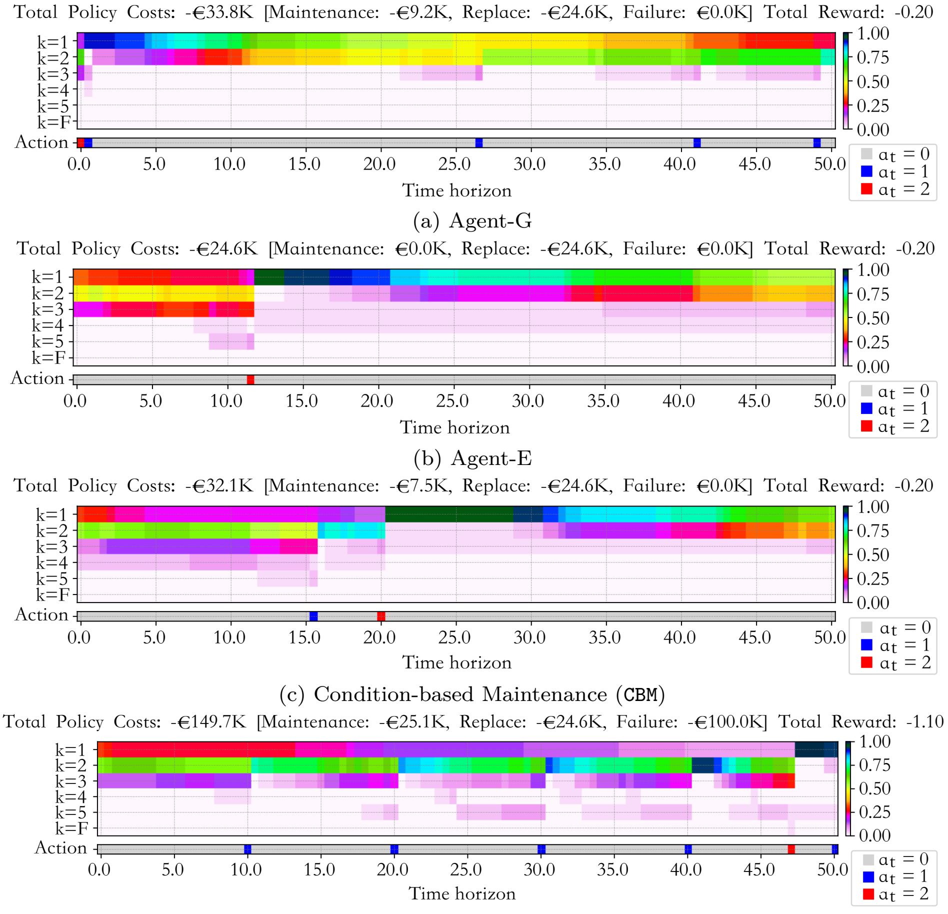
Figure 7.5: Policies behaviour over an episode for a **pipe aged 25**, showing the health vector over the pipe age and actions per policy: (a) Agent-G, (b) Agent-E, (c) Condition-based Maintenance (CBM), and (d) Scheduled Maintenance (SchM).

maintenance optimisation setup.

Our methodology includes a real-world case study from a Dutch sewer network, which provides historical inspection data. Through hyper-parameter tuning and policy analysis, we benchmark our optimised policies against traditional heuristics, including condition-based, scheduled, and reactive maintenance.

Our findings suggest that agents trained with the Proximal Policy Optimisation (PPO) algorithm are highly capable of developing strategic maintenance policies, adapting to pipe age, and surpassing heuristic baselines by learning cost-effective dynamic management strategies.

To evaluate the impact of deterioration model assumptions, we trained one agent



(a) Agent-G

(b) Agent-E

(c) Condition-based Maintenance (`CBM`)

(d) Scheduled Maintenance (`SchM`)

Figure 7.6: Policies behaviour over an episode for a **pipe aged 50**, showing the health vector over the pipe age and actions per policy: (a) Agent-G, (b) Agent-E, (c) Condition-based Maintenance (`CBM`), and (d) Scheduled Maintenance (`SchM`).

using the Gompertz probability density function and another using the Exponential probability density function.

During testing, both agents were assessed in an environment parametrised with the Weibull probability density function. The Gompertz-trained agent, whose behaviour more closely resembled the Weibull model, demonstrated better generalisation, resulting in more effective maintenance policies compared to the Exponential-trained agent.

Future work: The following directions are identified:

- Advancing toward partially observable state spaces with the introduction of inspection actions, considering context, and leveraging deep learning capabilities.

- Utilising knowledge acquired by agents to develop explainable and robust heuristics.
- Although this chapter focused on a single cohort of pipes, studies in Jimenez-Roa, Heskes, Tinga, et al., 2022; Jimenez-Roa, Tinga, Heskes, et al., 2024 show different cohorts exhibit varied dynamics, highlighting the importance of understanding how RL agents adapt.
- Comparing RL-based approaches with other policy optimisation algorithms to better understand the capacity of RL methods to achieve global-optima maintenance strategies.
- Investigating various reward functions (e.g., dense) and RL algorithms to determine the most effective for devising maintenance policies.
- Extent to system-level analysis and evaluate scalability.
- Moving toward multi-infrastructure asset management to promote coordinated management for optimising costs and minimising disruption from interventions.

7.9 References

Akiba, T., S. Sano, T. Yanase, T. Ohta, and M. Koyama (2019). "Optuna: A Next-generation Hyperparameter Optimization Framework". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. KDD '19. Anchorage, AK, USA: Association for Computing Machinery, pp. 2623–2631. ISBN: 9781450362016. DOI: 10.1145/3292500.3330701.

Ansel, J., E. Yang, H. He, N. Gimelshein, A. Jain, M. Voznesensky, B. Bao, P. Bell, D. Berard, E. Burovski, G. Chauhan, A. Chourdia, W. Constable, A. Desmaison, Z. DeVito, E. Ellison, W. Feng, J. Gong, M. Gschwind, B. Hirsh, S. Huang, K. Kalambarkar, L. Kirsch, M. Lazos, M. Lezcano, Y. Liang, J. Liang, Y. Lu, C. K. Luk, B. Maher, Y. Pan, C. Puhrsch, M. Reso, M. Saroufim, M. Y. Siraichi, H. Suk, S. Zhang, M. Suo, P. Tillet, X. Zhao, E. Wang, K. Zhou, R. Zou, X. Wang, A. Mathews, W. Wen, G. Chanan, P. Wu, and S. Chintala (2024). "PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph Compilation". In: *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*. ASPLOS '24. La Jolla, CA, USA: Association for Computing Machinery, pp. 929–947. DOI: 10.1145/3620665.3640366.

de Jonge, B. and P. A. Scarf (2020). "A review on maintenance optimization". In: *European Journal of Operational Research* 285.3, pp. 805–824. ISSN: 0377-2217. DOI: 10.1016/j.ejor.2019.09.047.

Jimenez-Roa, L. A., T. Heskes, T. Tinga, H. J. A. Molegraaf, and M. Stoelinga (2022). "Deterioration modeling of sewer pipes via discrete-time Markov chains: A large-scale case study in the Netherlands". In: *Proceedings of the 32nd European Safety and Reliability Conference, ESREL 2022 - Understanding and Managing Risk and Reliability for a Sustainable Future*, pp. 1299–1306. DOI: 10.3850/978-981-18-5183-4_R22-13-482-cd.

Jimenez-Roa, L. A., T. Tinga, T. Heskes, and M. Stoelinga (2024). "Comparing Homogeneous and Inhomogeneous Time Markov Chains for Modelling Degradation in Sewer Pipe Networks". In: *Proceedings of the European Safety and Reliability Conference (ESREL 2024)*. DOI: 10.48550/arXiv.2407.12557.

M.A. Cardoso, M. A. and M. S. Silva (2016). "Sewer asset management planning – implementation of a structured approach in wastewater utilities". In: *Urban Water Journal* 13.1, pp. 15–27. DOI: 10.1080/1573062X.2015.1076859.

Marugán, A. P. (2023). "Applications of Reinforcement Learning for maintenance of engineering systems: A review". In: *Advances in Engineering Software* 183, p. 103487. ISSN: 0965-9978. DOI: 10.1016/j.advengsoft.2023.103487.

Ogunfowora, O. and H. Najjaran (2023). "Reinforcement and deep reinforcement learning-based solutions for machine maintenance planning, scheduling policies, and optimization". In: *Journal of Manufacturing Systems* 70, pp. 244–263. ISSN: 0278-6125. DOI: 10.1016/j.jmsy.2023.07.014.

Raffin, A., A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann (2021). "Stable-Baselines3: Reliable Reinforcement Learning Implementations". In: *Journal of Machine Learning Research* 22.268, pp. 1–8. URL: http://jmlr.org/papers/v22/20-1364.html.

Tscheikner-Gratl, F., N. Caradot, F. Cherqui, J. P. Leitão, M. Ahmadi, J. G. Langeveld, Y. Le Gat, L. Scholten, B. Roghani, J. P. Rodríguez, et al. (2019). "Sewer asset management–state of the art and research needs". In: *Urban Water Journal* 16.9, pp. 662–675. DOI: 10.1080/1573062X.2020.1713382.

Turnbull, B. W. (1976). "The Empirical Distribution Function with Arbitrarily Grouped, Censored and Truncated Data". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 38.3, pp. 290–295. DOI: https://doi.org/10.1111/j.2517-6161.1976.tb01597.x.

Virtanen, P., R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors (2020). "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods* 17, pp. 261–272. DOI: 10.1038/s41592-019-0686-2.

Discussion, Conclusions & Recommendations

Chapter 8

Discussion

This doctoral thesis, divided into three parts, addresses relevant aspects of the Prognostics and Health Management (PHM) paradigm for engineering applications. This chapter discusses: Reliability Modelling, specifically the data-driven inference of fault tree models (Section 8.1); Markov Process-based Prognostics, particularly multi-state deterioration modelling (Section 8.2); and Maintenance Optimisation using Deep Reinforcement Learning (Section 8.3).

8.1 Reliability Modelling: Data-driven Inference of Fault Tree models

Overview of the research problem in Part I

One of the main challenges in reliability modelling is building the model itself, and this is particularly true for graph-based methods such as Fault Tree Analysis (FTA), where constructing Fault Tree (FT) models typically involves an iterative process between experts and FT modellers, which may be prone to human error. Developing algorithms to automate this process and identify overlooked patterns is key.

Recap on key contributions in Part I

In Part I of this dissertation, we explored, for the first time, Multi-Objective Evolutionary Algorithms (MOEAs) to automatically infer FTs from failure datasets. In the domain of reliability modelling, our contributions are three-fold:

1. The `FT-MOEA` algorithm (Chapter 2), based on a MOEA, accounts for three optimisation metrics, including minimising FT size and accuracy-related error metrics. With `FT-MOEA`, we can consistently obtain compact FT structures. Data and implementations are available at `zenodo.org/record/5536431`.
2. The `SymLearn` toolchain (Chapter 3) employs a ‘divide and conquer’ strategy, exploiting symmetries and modules that may be present in the failure dataset.

With SymLearn, we can handle larger problems and thus improve scalability. Data and implementations are available at zenodo.org/record/5571811.

3. The FT-MOEA-CM extension (Chapter 4) expands the multi-objective optimisation function by incorporating metrics computed from the confusion matrix. With FT-MOEA-CM, we improved robustness by consistently achieving global optima for larger problems. Data and implementations are available at https://gitlab.utwente.nl/fmt/fault-trees/ft-moea.

Our findings suggest that using MOEAs for the inference of FT models generally has a positive impact in terms of robustness, scalability, and convergence speed.

Why are algorithms for FT inference important?

In the digital era, vast amounts of data are collected, offering opportunities to manage complex systems and processes better. Graph learning algorithms capture the underlying intricate relationships within data by modelling essential connections between vertices through edges while adhering to the properties of the graph model. These models are valuable for human interpretation, as they translate complex data relationships into more comprehensible forms. Moreover, their rich mathematical properties make them important tools for managing complex systems and processes.

In terms of FTs, this is particularly relevant, as FT models are applicable in many industries. Effective algorithms for learning FT models from data offer benefits such as accounting for updated FT structures as changes occur and generating more efficient structures. For example, our COVID-19 case study originally presented a FT structure of 33 elements, and after applying our algorithms, an equivalent structure with 13 elements was revealed. Risk asset managers may find that smaller and more compact equivalent FT could aid in redefining the meaning of intermediate events.

What are the implications of using MOEAs in the inference of FTs?

Inspired by the FT-EA algorithm proposed in Linard, Bucur, and Stoelinga, 2019, we observed that while FT-EA introduces a novel approach and opens up an interesting research direction, there are areas in need of improvement. One of these is the consistency of the FT structure, which involves obtaining the same (or equivalent) graph model from identical failure datasets. This consistency was not always achieved with FT-EA, especially in larger problems.

Why is this consistency important? A consistent FT model reveals patterns that aid in understanding the relationships between basic and intermediate events. In other words, it automatically uncovers the complex logical connections related to how failures propagate through the system. This is particularly valuable for end-users like risk asset managers, where both expressiveness and interpretability are crucial.

By implementing MOEAs, we successfully achieved this consistency. Furthermore, we observed improvements in convergence speed and scalability, which is critical for real-world engineering challenges involving numerous basic events. In hindsight, this proved to be an effective approach.

However, applying MOEAs to larger datasets presents additional challenges. Evolutionary algorithms rely heavily on random mutations, and in large solution spaces, this randomness can require substantial computational power, making convergence difficult. This limitation is a key drawback of using MOEAs for inferring FTs. For instance, most of our validation cases are derived from real-world engineering problems which, while representative, are relatively small, involving up to 24 Basic Events and 26 Minimal Cut Sets. Other real-world problems could be much larger, with incomplete and noisy datasets.

Other type MOEAs, or advanced methods that rely less on randomness, combined with improvements in computational power and strategic implementations such as parallelisation, could enhance the effectiveness of MOEAs for inferring FTs from data.

8.2 Markov Process-based Prognostics: Multi-state deterioration modelling

Overview of the research problem in Part II

Accounting for reliable deterioration models is key in Prognostics and Health Management (PHM), as they help predict failures and undesirable states, enabling timely optimal actions. Various types of deterioration models exist depending on the system/component/process of interest, and the literature is extensive on this subject. Part II of this dissertation focuses on deterioration modelling in sewer main systems, which are complex due to their scale, numerous components, and dependencies.

We centre our attention on Multi-State Deterioration Models (MSDMs) using Markov chains, which aim to predict damage severity levels. These models have been addressed in the literature, but several aspects remain open for discussion, including model assumptions, and data availability.

Recap on key contributions in Part II

In Part II, we used Markov chains to model Multi-State Deterioration (MSD) in sewer mains. Our contributions are three-fold:

1. We present a real-world case study from the Netherlands (Section II.4.3). Part of the data is publicly available at `zenodo.org/record/6535853`.
2. We evaluate two types of Markov chain structures (Chapter 5) typically used for MSDM in sewer mains, discussing their benefits and drawbacks. Additionally, we extend and propose a Markov chain structure (Chapter 6) that accounts for functional failure states.
3. We compare the assumptions of homogeneous and inhomogeneous time Markov chains (Chapter 6), identifying inhomogeneous-time Markov chains as more suitable for long-lived assets like sewer mains. Data and implementations are available at `https://gitlab.utwente.nl/fmt/degradation-models/ihctmc`.

Our findings suggest that using Markov chains for MSDM in sewer mains has the potential to infer the severity level across populations of sewer mains.

Data availability for deterioration modelling of sewer mains: the source of (many) challenges

The goal of deterioration modelling is to create a mathematical abstraction that adequately links context with the system's/component's degradation behaviour. Data-driven approaches' success relies heavily on data quality, which is particularly challenging in sewer asset management. High-quality inspection datasets collected from sewer mains (if any) are rare let alone public, hindering methods comparison and analyses (see Table II.1).

To address this gap, we have made our case study available, sharing relevant damage codes and cohorts (Section II.4.3). A high-level comparison with other case studies shows that our dataset, featuring sewer covariates and historical inspections is fairly typical. Though this is a step forward, it is far from closing the gap.

We now must question whether this information alone suffices to build robust deterioration models for long-term sewer condition assessment. Likely the answer is *no*, for several reasons. For starters, a sewer network, in its simplest form—ignoring dependencies—is a population of pipes with *covariates* such as material, content, and age, and *conditions* like damage severity—data gathered from inspections.

Data-driven models for long-term condition assessment require 'rich' and 'sufficient' data, which is difficult to obtain from sewer inspection datasets. This is mainly because inspections in sewer mains are performed to assess pipe conditions to support maintenance or replacement actions and planning, **not** to *feed* degradation models.

Sewer mains often represent heterogeneous populations. As shown in Figures II.2 and II.3, the datasets are highly unbalanced, missing key contextual data like soil properties and road usage. This leads to poor statistics for some covariates, lowering model performance. Moreover, deterioration in sewer mains is a slow process spanning decades. During this time, context may significantly change as cities evolve e.g., demands in the sewer mains shift, sewer main properties improve due to advances in material science and construction techniques, etc. Effective deterioration models, among others, must consider the context and its evolution.

Known additional challenges include the absence of maintenance records in sewer inspection data, which complicates the assessment of their impact on degradation behaviour. Functional failures are infrequent, making it difficult to construct models that can accurately predict them. The interval-censored nature of the data (Duchesne, Beardsell, Villeneuve, et al., 2013) introduces biases and uncertainties, which must be considered in effective models. Synthetic data, as proposed by Scheidegger, Hug, Rieckermann, et al., 2011, may offer a promising alternative to cope with some of these limitations. However, improving the quality of inspection data remains an open issue. Data management and quality control for sewer assets are discussed in detail in Auger, Besnier, Bijnen, et al., 2024.

Having acknowledged all these challenges in the inspection data, there are also mathematical assumptions in deterioration models that must be discussed, some of which address dataset limitations, while others prioritise simplicity. We elaborate on this in the next section.

Assumptions on deterioration modelling of sewer mains via Markov chains

Among the various models for long-term sewer main condition assessment, we focus on Markov chains due to their intuitive representation of severities and well-established mathematical properties. Markov chains assume that the degradation process follows the Markov property, where future states depend solely on the current state, disregarding previous states. The suitability of this assumption for sewer mains has been discussed in Timashev and Bushinskaya, 2015 and is not addressed in this dissertation; however, corroborating this property for larger schemes using multiple case studies is still worth exploring.

This part of the dissertation focuses on the structure of Markov chains, the assumption of homogeneity, interval-censored data, and parameter inference. We assess typical Markov chain structures from the literature and observe that, under regular conditions—excluding rare events such as explosions or accidents—their performance is comparable. However, more complex structures, with additional parameters, tend to exhibit undesired behaviours, including the formation of unwanted absorbing states. Adding a functional failure state is necessary, as high severity does not indicate failure. Although the architecture we proposed including functional failure state produced reasonable results, further evaluation is necessary due to the scarcity of functional failure data, hindering validation.

Among the modelling choices when implementing Markov chains to model deterioration in sewer mains is the assumption that time transitions are either homogeneous or inhomogeneous. Both approaches are used in the literature (see Table II.3, page 106). Homogeneous-time Markov chains are simpler due to their constant transition rate, but our results indicate that inhomogeneous-time Markov chains better capture non-linear stochastic behaviours. However, this adds degrees of freedom, sometimes leading to overfitting. Future work should focus on more robust calibration methods to enhance predictability. Properly calibrated, inhomogeneous models should capture homogeneous behaviours, but not vice versa.

As pointed out before, the inspection data in sewer mains, due to the way data is collected, are interval-censored (Duchesne, Beardsell, Villeneuve, et al., 2013), which introduces uncertainty about the exact moment of transition between severity levels. This complicates the calibration of degradation models as it requires a more complex loss function, often overlooked. In this dissertation, we assess this issue using the non-parametric Turnbull estimator (Chapter 6) as a reference. Interestingly, despite not accounting for interval censoring during calibration, the predictions of our degradation models closely aligned with those from the Turnbull estimator in terms of the probability distribution over severity levels at different pipe ages. This suggests, for this dataset, that the effect of interval censoring may be negligible, though further investigation is required to evaluate the extent of generalisation of this observation, as this could significantly simplify the model calibration process.

What is the current usefulness of MSDMs?

Having acknowledged the limitations in data and model assumptions, let's now discuss the usefulness of MSDMs via Markov chains for sewer main degradation modelling. We believe that these models, when provided with 'sufficient' data, are useful to *approximate* the distribution of damage severities across a *population* of sewer mains. However, two scenarios where these models are less reliable—apart from covariates with data scarcity—are (i) predicting the condition at a pipe level and (ii) predicting the condition of a pipe beyond the observed data. The first issue may be improved by larger and richer datasets (e.g., synthetic data and data integration), and the second by implementing an effective calibration process, with a strong focus on improving prognostic capabilities.

8.3 Maintenance Optimisation: Maintenance optimisation of multi-state components

Overview of the research problem in Part III

Building on the findings from Part II, the next logical step was to apply Multi-State Deterioration Models (MSDMs) to design maintenance policies for sewer mains, which is the focus of Part III. Obtaining optimal maintenance policies is challenging, especially in large state spaces, where computing the optimal policy can be inefficient or infeasible within time constraints. This motivated the exploration of alternatives like Deep Reinforcement Learning (DRL). While DRL does not guarantee globally optimal solutions, we aimed to assess its potential for Maintenance Policy Optimisation (MPO), focusing on how degradation model assumptions affect agent performance.

Recap on key contributions in Part III

In Part III, we used Multi-State Deterioration Model (MSDM) and Deep Reinforcement Learning to devise component-level strategic maintenance planning with applications in sewer mains. Our contributions are two-fold:

1. In Chapter 7, we propose a DRL framework for devising maintenance policies at the pipe level, considering MSDM. We detail model calibration and have made our models and dataset publicly available in the repository: `zenodo.org/records/11258904`.
2. We evaluate the influence of homogeneous and inhomogeneous MSDM on devising strategic maintenance, comparing agent behaviours against well-known maintenance policy heuristics.

Our findings suggest that DRL offers a flexible framework with untapped potential for maintenance strategies, and it is crucial to integrate degradation model assumptions, as they significantly influence policy behaviour.

When does it make sense to use DRL for maintenance optimisation?

DRL extends Reinforcement Learning (RL), providing a flexible framework to tackle sequential decision-making problems, producing virtual agents with enforced behaviours through ‘trial and error’. This approach is particularly suitable for maintenance optimisation when the state space is *large*.

For systems like sewer networks, large state spaces arise easily due to the number of discrete and continuous variables involved (e.g., covariates, sensor data, etc.). Thus, effective techniques to ‘solve’ these optimisation problems are crucial.

Approaches that seek globally optimal solutions through exhaustive search can be computationally expensive. This is where alternative approaches, such as DRL, can explore larger state spaces, though without guaranteeing global optimality. Both approaches are valuable, and further research with applications on these type of systems remains important. Notably, a key challenge in DRL is the time-consuming hyper-parameter tuning needed to achieve agents with ‘good behaviour’.

The potential of DRL is significant, highlighted by growing interest evidenced in recent reviews (Real Torres, Andreiana, Ojeda Roldán, et al., 2022; Siraskar, Kumar, Patil, et al., 2023; Marugán, 2023; Li, Zheng, Yin, et al., 2023) and the unexplored applications in infrastructure systems such as sewer networks.

What are the implications of using DRL for the maintenance of large multi-component systems?

DRL creates virtual agents that, when well-trained, act as virtual 'experts' exposed to diverse scenarios described by the environment. The knowledge encoded in these agents can potentially improve strategic maintenance. Though challenging, this could eventually become key in managing infrastructure and resource constraints, enabling timely actions and optimal resource allocation.

In Chapter 7, we demonstrate how these agents outperform heuristics. An implication of this work is for instance, that the knowledge acquired by the trained agents potentially could improve heuristics, or even at a higher level, understanding the agent's behaviour could offer valuable insights for strategic maintenance. Additionally, we observed that agents developed distinct behaviours in line with the dynamics of their environments, emphasising the importance of properly integrating prognostics within the policy optimisation framework.

8.4 Moving towards comprehensive Prognostics and Health Management: Closing Thoughts

To recap, this dissertation explored several key facets of the Prognostics and Health Management (PHM) paradigm, deploying advanced methodologies to support reliability analysis such as Multi-Objective Evolutionary Algorithms for automatically constructing Fault Tree models from data, and Markov Chains for Multi-State Deterioration Models (MSDMs) applied to sewer mains. We also delved into strategic maintenance planning using Deep Reinforcement Learning (DRL), showcasing the versatility yet complexity of DRL in crafting optimal maintenance strategies.

Reflecting on my research journey, I find that effective and comprehensive PHM remains a significant challenge. A key issue is the gap between the *expectation* of what PHM can offer and what is achievable with current technology. During my PhD, I observed many companies showing interest in predictive maintenance systems but lacking the actionable data needed for effective implementation. To close this gap, it is crucial, among others, to align data collection strategies with specific *model* requirements, especially in prognostics, where abstracting long-term degradation profiles is difficult. Collecting data without clear objectives is inefficient and wastes valuable resources.

The integration of PHM elements was challenging but enlightening. Initially, I focused on isolated tasks within PHM, from data collection and processing to model development and deployment. However, this often led to siloed research outcomes that, while effective in narrow contexts, struggled to translate into broader system-level applications. Narrow-scoped research is challenging, and maintaining focus is essential to avoid unmanageable scopes. Nevertheless, combining these tasks effectively within PHM is crucial for developing integrated, comprehensive methodologies applicable to real-world engineering problems.

Recognising this, I shifted towards methodologies aimed at bridging these gaps. An example of better integration between *prognostics* and *maintenance optimisation* (see Figure 1.2) is our work in Chapter 7, where our DRL framework for strategic maintenance of sewer mains incorporated MSDMs (Chapter 6), enabling a more realistic maintenance optimisation setup. However, this also revealed challenges, such as whether standard RL algorithms, which assume stationary transition probabilities, would remain effective given the inhomogeneous behaviours introduced by MSDMs. An extension of this example to integrating PHM tasks (not covered in this dissertation) would involve deploying approaches from Part I, advancing towards system-level analysis by incorporating additional dependencies between subsystems and their components.

Clearly, this attempt does not fully bridge the remaining gaps across the PHM paradigm. However, by improving *cohesion* and scalability within the PHM stages, we are making significant progress toward higher technological maturity, which is crucial for building *trust* in these technologies. I remain optimistic about future prospects, particularly with the integration of AI technologies such as Large Language Models. These tools provide valuable insights by efficiently generalising across tasks and have the potential to serve as the crucial *link* for a more seamless and integrated PHM life-cycle.

My personal growth throughout this doctoral program has been profoundly shaped by these challenges and realisations. I have learned the immense value of a collaborative and multi-disciplinary approach, which not only enriches the research itself but also enhances its applicability in real-world scenarios. The PrimaVera project (`https://primavera-project.com`), which funded my doctoral studies, exemplifies this collaborative spirit, bringing together experts from academia, industry, and public institutions to pioneer innovative PHM solutions.

Finally, while this dissertation does not solve all the complex problems within PHM, it contributes to a deeper understanding and presents a pathway forward through collaborative, data-informed, and technologically integrated approaches. As we continue to build on this foundation, I am optimistic about the potential of emerging technologies to make PHM not just a theoretical ideal but a practical tool for sustaining the health and reliability of critical systems worldwide.

8.5 References

Auger, S., J.-B. Besnier, M. van Bijnen, F. Cherqui, G. Chuzeville, F. Clemens-Meyer, M. G. Jaatun, J. Langeveld, Y. Le Gat, S. Moin, G. E. Oosterom, W. van Riel, B.

Roghani, M. M. Rokstad, J. Røstum, F. Tscheikner-Gratl, and R. Ugarelli (June 2024). “Data management and quality control”. In: *Asset Management of Urban Drainage Systems: If anything exciting happens, we’ve done it wrong!* IWA Publishing. ISBN: 9781789063059. DOI: 10.2166/9781789063059_0299.

Duchesne, S., G. Beardsell, J.-P. Villeneuve, B. Toumbou, and K. Bouchard (2013). “A survival analysis model for sewer pipe structural deterioration”. In: *Computer-Aided Civil and Infrastructure Engineering* 28.2, pp. 146–160. DOI: 10.1111/j.1467-8667.2012.00773.x.

Li, C., P. Zheng, Y. Yin, B. Wang, and L. Wang (2023). “Deep reinforcement learning in smart manufacturing: A review and prospects”. In: *CIRP Journal of Manufacturing Science and Technology* 40, pp. 75–101. ISSN: 1755-5817. DOI: https://doi.org/10.1016/j.cirpj.2022.11.003.

Linard, A., D. Bucur, and M. Stoelinga (2019). “Fault Trees from Data: Efficient Learning with an Evolutionary Algorithm”. In: *International Symposium on Dependable Software Engineering: Theories, Tools, and Applications.* Vol. 11951 LNCS, pp. 19–37. DOI: 10.1007/978-3-030-35540-1_2.

Marugán, A. P. (2023). “Applications of Reinforcement Learning for maintenance of engineering systems: A review”. In: *Advances in Engineering Software* 183, p. 103487. ISSN: 0965-9978. DOI: 10.1016/j.advengsoft.2023.103487.

Real Torres, A. del, D. S. Andreiana, Á. Ojeda Roldán, A. Hernández Bustos, and L. E. Acevedo Galicia (2022). “A Review of Deep Reinforcement Learning Approaches for Smart Manufacturing in Industry 4.0 and 5.0 Framework”. In: *Applied Sciences* 12.23. ISSN: 2076-3417. DOI: 10.3390/app122312377.

Scheidegger, A., T. Hug, J. Rieckermann, and M. Maurer (2011). “Network condition simulator for benchmarking sewer deterioration models”. In: *Water Research* 45.16, pp. 4983–4994. ISSN: 0043-1354. DOI: 10.1016/j.watres.2011.07.008.

Siraskar, R., S. Kumar, S. Patil, A. Bongale, and K. Kotecha (2023). “Reinforcement learning for predictive maintenance: A systematic technical review”. In: *Artificial Intelligence Review* 56.11, pp. 12885–12947. DOI: 10.1007/s10462-023-10468-6.

Timashev, S. and A. Bushinskaya (2015). “Markov approach to early diagnostics, reliability assessment, residual life and optimal maintenance of pipeline systems”. In: *Structural Safety* 56, pp. 68–79. ISSN: 0167-4730. DOI: 10.1016/j.strusafe.2015.05.006.

Chapter 9

Conclusions & Recommendations

9.1 Conclusions

In Part I of this thesis, we investigated *how to obtain efficient and compact Fault Tree models from failure datasets in a robust and scalable manner* and concluded that, with available failure datasets, this task can be effectively addressed using `FT-MOEA-CM`, our proposal based on `NSGA-II`, a Multi-Objective Evolutionary Algorithm. Our findings indicate that six metrics from the confusion matrix, along with the size of the Fault Tree, are crucial for consistently achieving compact and efficient Fault Tree structures, ensuring effective algorithm convergence. This is crucial for unveiling useful patterns associated with failure propagation through basic and intermediate events in a Fault Tree.

The evidence shows that `FT-MOEA-CM` is more robust and scalable than state-of-the-art algorithms such as `FT-EA` and its predecessor `FT-MOEA`. Furthermore, in cases involving symmetries in the failure dataset, coupling `FT-MOEA-CM` with toolchains such as `SymLearn` may further improve convergence and scalability.

In Part II of this dissertation, we investigated *how and to what extent it is possible to accurately model Multi-State Deterioration with applications in sewer mains.* For this, we explore the use of Markov Chains and examined both *data availability* and *model assumptions.*

High-quality, publicly available inspection datasets for sewer mains are scarce, limiting predictive modelling and validation efforts. To address this, we shared part of the data from a real-world case study in the Netherlands, contributing to the understanding of sewer mains' degradation behaviour. We evaluated two typical Markov chain structures to model damage severity levels, concluding that the simpler structure suffices for typical sewer main degradation. Comparing homogeneous and inhomogeneous time Markov chains, our results suggest that inhomogeneous chains better capture non-linear stochastic behaviours. However, inhomogeneous chains introduced additional hyper-parameters, leading to overfitting and hindering predictability. For calibration, we used a process combining the Metropolis-Hastings algorithm with Sequential Least Squares Programming.

Although we did not account for interval censoring during calibration, our results aligned well with the non-parametric Turnbull estimator (which accounts for interval-censoring), suggesting that the effects of interval censoring are negligible for this case study.

Answering this research question more comprehensively, Markov chains offer a suitable approach for modelling damage severity levels by representing them as discrete states within the chain. However, the applicability of these models, particularly as implemented in this dissertation, is entirely data-driven, meaning their usability and effectiveness heavily rely on data quality. In the case of sewer mains, as previously noted, data quality remains a significant concern. While Markov chain models provide a solid initial step, more advanced degradation modelling techniques are necessary to overcome the limitations imposed by data quality.

Finally, in Part III of this dissertation, we investigated *how to devise optimal maintenance strategies for components with Multi-State Deterioration such as sewer mains using Deep Reinforcement Learning*. We evaluated the effectiveness of Deep Reinforcement Learning (DRL) in developing cost-effective maintenance strategies for sewer mains. A key contribution is the integration of Multi-State Deterioration models within a DRL optimisation framework. We proposed a novel fully-observable Markov Decision Process (MDP) with a reward function aligned to damage severity levels.

We benchmarked agent-based policies against traditional heuristics, including condition-based, scheduled, and reactive maintenance. Our results suggest that agents trained with the Proximal Policy Optimisation (PPO) algorithm excel in developing dynamic, cost-effective strategies, surpassing heuristic baselines.

Our experiments also compared the impact of homogeneous and inhomogeneous assumptions in the deterioration models on the agent's behaviour. For this, one agent was trained with the Gompertz (inhomogeneous) function and another with the Exponential (homogeneous) function. In testing, the Gompertz-trained agent outperformed the Exponential-trained agent, likely due to the Weibull-distributed deterioration model in the test environment, which aligns more closely with the Gompertz function.

9.2 Recommendations

9.2.1 Automatic Inference of Fault Tree Models

While the algorithm and extensions proposed in Chapters 2, 3, and 4 improve FT model inference, several challenges remain.

1. **Local optima**: Minimising FT model size can result in local optima, as irrelevant structures may satisfy Pareto criteria. For instance, an FT with

a single gate and event might be optimal in size but poor in accuracy. The `FT-MOEA-CM` extension (Chapter 4) addresses this by introducing extra metrics, but other solutions could be explored.

2. **Scalability**: Scalability is a major issue as complexity increases exponentially with the number of basic events. Although `SymLearn` (Chapter 3) and `FT-MOEA-CM` (Chapter 4) demonstrated improvements, performance with larger datasets remains uncertain. Evolutionary algorithms struggle with the vast solution space and require excessive computation time for larger FT populations, complicating convergence. More efficient methods are needed for larger problems.
3. **Noisy and incomplete data**: Real-world datasets are often noisy or incomplete, lacking records of rare events and containing inaccuracies. While this was briefly discussed in Chapter 2, more research is needed to address these issues effectively.
4. **Comparison benchmark**: A systematic comparison of algorithms for FT model inference is lacking. Such a benchmark could highlight strengths and weaknesses, suggesting improvements.
5. **Additional logic gates**: We focused on static FTs (using AND and OR gates). Including more gate types (e.g., voting gates) and considering dynamic FTs (Ruijters and Stoelinga, 2015) could lead to more efficient and representative models.
6. **Application to other formalisms**: Our approach, based on Multi-Objective Evolutionary Algorithm, may also apply to other formalisms with similar challenges (i.e., automatically inferring the graph structure from data), such as Attack trees (Kordy, Piètre-Cambacédès, and Schweitzer, 2014), dynamic FTs, and Reliability Block Diagrams (Guo and Yang, 2007). Research in this direction is valuable.

9.2.2 Multi-State Deterioration Modelling of Sewer Mains

In Chapters 5 and 6, we explored the use of Markov chains to model damage severity in sewer mains. Below, we outline the remaining challenges:

1. **Data collection**: A key challenge in data collection is the ‘data-loop problem’ (Auger, Besnier, Bijnen, et al., 2024), where the lack of evidence for the *benefits* of data collection discourages further investment. This leads to a cycle of declining data quality and quantity, reducing the effectiveness of systematic data collection. This problem is not exclusive to sewer systems but is common in civil infrastructure. Breaking this cycle is crucial to address data scarcity and build trust in the methods.
2. **Extending the inspection goal**: Currently, sewer main inspections focus on assessing pipe conditions for maintenance or replacement, rather than feeding degradation models. To address this, inspection campaigns should

incorporate requirements set by degradation models, shifting the goal from merely assessing conditions to also generating data for these models.

3. **Cohort-based approach**: Dividing inspection datasets into cohorts for Markov chain calibration led to challenges, often resulting in small, unrepresentative populations of sewer mains, which undermined model reliability. Future work should develop systematic methodologies or guidelines to define cohorts more appropriately for this application. The assumption of cohort homogeneity may not hold, so addressing data heterogeneity is critical.
   An alternative to cohort-based approaches is to develop models that directly incorporate covariates, such as pipe location, which can generate tailored degradation curves while preserving properties such as monotonicity and damage severity levels.
4. **Parameter inference**: The methods used for parameter inference often converged prematurely, getting trapped in local optima. Sampling-based methods were computationally expensive, and the likelihood-based loss functions may have been insufficient to prevent overfitting and to ensure reliable predictions. Future work should explore more robust parameter inference techniques that can avoid these pitfalls.
5. **Include context**: "It is implicitly assumed that the conditions between inspections were identical and that all processes occurred at a constant rate over time" (Cherqui, Clemens-Meyer, Tscheikner-Gratl, et al., 2024). This assumption is unrealistic, especially for large-scale systems like sewer mains. A potential solution, as previously mentioned, is to develop or implement methods that account for covariates such as pipe location, which would enable more realistic modelling of varying conditions.
6. **Uncertainty quantification**: Prognostic models ideally should incorporate uncertainty bounds to reflect confidence in their predictions, which is essential for decision-making. Degradation models based on Markov chains are no exception. For this, we initially employed sampling-based methods to estimate confidence intervals, successfully applied to Discrete-Time Markov Chains (Figure 5.2, page 118). However, for Inhomogeneous-time Markov Chains, the computational cost made these methods impractical. An alternative approach could use *interval Markov chains* (Kozine and Utkin, 2002), where transition probabilities lie within intervals rather than being fixed values (Sproston, 2023). The key challenge here is efficiently infer the parameters of the interval Markov chain to represent the lower and upper confidence bounds.
7. **Interval-censored data**: Extending the challenges mentioned in parameter inference and uncertainty quantification, an open challenge is the consideration of *interval-censoring* during model calibration. This aspect is inherent in the sewer network inspection dataset, as the exact transition time between severity levels is unknown. To address this, a more complex loss function is required to properly handle interval censoring; see Hout, 2016 for further details. Additional testing of this aspect is necessary across multiple case

studies.

8. **System-level dependencies**: Future research should incorporate system-level dependencies—such as stochastic and structural (de Jonge and Scarf, 2020)—as treating the sewer network as a collection of 'disconnected' elements is not realistic. The influence of these dependencies on degradation assessment and modelling is largely unexplored, and their inclusion could significantly enhance the accuracy of the models.
9. **Hybrid methods**: Data-driven approaches alone may not suffice, especially considering the persistence of data quality issues. Hybrid methods that combine data-driven models with other types of models (e.g., physics-based or expert knowledge) could enhance performance and help overcome data limitations. These hybrid approaches could be especially useful when data is scarce or of poor quality.

9.2.3 Strategic Maintenance Planning for Sewer Mains using Reinforcement Learning

In Chapter 7, we explored Deep Reinforcement Learning (DRL) for component-level strategic maintenance in sewer mains. Key challenges and future research directions include:

1. **Account for context**: Context refers to static variables in the state space that influence the reward and transition functions. For example, different materials used in sewer mains may result in varying degradation profiles. The maintenance strategy should adapt to such contextual factors. For large systems with variable *context*, frameworks like Contextual Markov Decision Process (Hallak, Di Castro, and Mannor, 2015), which extend standard Markov Decision Process to incorporate contextual variability, could be beneficial.
2. **Partial observability**: The assumption of a fully observable state space is unrealistic, as not all component states are visible at all times. Shifting to partially observable state spaces, where inspections are required to reveal component states, expands the optimisation problem. This allows the planning of inspections to gather data and improve decision-making. Frameworks such as *Partially Observable Markov Decision Processes* (Kıvanç, Özgür-Ünlüakın, and Bilgiç, 2022) offer potential solutions for addressing this issue.
3. **Hyper-parameter tuning**: A significant challenge in DRL is tuning the RL algorithm's hyper-parameters, which can greatly influence agent behaviour. This includes both the parameters of the MDP and the hyper-parameters specific to DRL algorithms. Selecting the 'right' hyper-parameters is crucial, as they impact the agents' ability to adopt the desired maintenance strategies. Research into more effective tuning methods is needed to ensure agents exhibit optimal behaviour.

4. **Explainability**: Unlike traditional heuristics, DRL policies can be complex and difficult to interpret. Research is needed to translate these policies into robust heuristics, aligning with the principles of *explainable AI* (XAI) (Linardatos, Papastefanopoulos, and Kotsiantis, 2021). XAI techniques aim to make the actions of AI agents understandable, helping to answer questions like: why is the agent taking this sequence of actions, and how can this be used to improve maintenance decisions? This is key for building trust in DRL-based methods.
5. **System-level analysis**: While much of our focus is on component-level analysis, transitioning to system-level analysis is crucial. Sewer networks are not isolated components, and system dependencies may play an important role in strategic maintenance planning. Developing frameworks that incorporate these dependencies and evaluate their impact is an important research direction.
6. **Integration with prognostics models**: Integrating predictions from degradation models into the MDP state space would equip agents with information on future deterioration. This would allow more efficient exploration of the solution space and potentially improve decision-making. Further research is needed to assess how integrating prognostic models can enhance agent performance.

9.3 References

Auger, S., J.-B. Besnier, M. van Bijnen, F. Cherqui, G. Chuzeville, F. Clemens-Meyer, M. G. Jaatun, J. Langeveld, Y. Le Gat, S. Moin, G. E. Oosterom, W. van Riel, B. Roghani, M. M. Rokstad, J. Røstum, F. Tscheikner-Gratl, and R. Ugarelli (June 2024). "Data management and quality control". In: *Asset Management of Urban Drainage Systems: If anything exciting happens, we've done it wrong!* IWA Publishing. ISBN: 9781789063059. DOI: 10.2166/9781789063059_0299.

Cherqui, F., F. Clemens-Meyer, F. Tscheikner-Gratl, and B. van Duin (June 2024). *Asset Management of Urban Drainage Systems: If anything exciting happens, we've done it wrong!* IWA Publishing. ISBN: 9781789063059. DOI: 10.2166/9781789063059.

de Jonge, B. and P. A. Scarf (2020). "A review on maintenance optimization". In: *European Journal of Operational Research* 285.3, pp. 805–824. ISSN: 0377-2217. DOI: 10.1016/j.ejor.2019.09.047.

Guo, H. and X. Yang (2007). "A simple reliability block diagram method for safety integrity verification". In: *Reliability Engineering & System Safety* 92.9. Critical Infrastructures, pp. 1267–1273. ISSN: 0951-8320. DOI: 10.1016/j.ress.2006.08.002.

Hallak, A., D. Di Castro, and S. Mannor (2015). "Contextual Markov Decision Processes". In: *arXiv preprint arXiv:1502.02259*. DOI: 10.48550/arXiv.1502.02259.

Hout, A. van den (2016). *Multi-State Survival Models for Interval-Censored Data*. CRC Press, pp. 1–238. ISBN: 978-146656841-9. DOI: 10.1201/9781315374321.

Kıvanç, İ., D. Özgür-Ünlüakın, and T. Bilgiç (2022). "Maintenance policy analysis of the regenerative air heater system using factored POMDPs". In: *Reliability Engineering & System Safety* 219, p. 108195. ISSN: 0951-8320. DOI: https://doi.org/10.1016/j.ress.2021.108195.

Kordy, B., L. Piètre-Cambacédès, and P. Schweitzer (2014). “DAG-based attack and defense modeling: Don’t miss the forest for the attack trees”. In: *Computer Science Review* 13-14.C, pp. 1–38. DOI: `10.1016/j.cosrev.2014.07.001`.

Kozine, I. and L. Utkin (Apr. 2002). “Interval-Valued Finite Markov Chains”. In: *Reliable Computing* 8, pp. 97–113. DOI: `10.1023/A:1014745904458`.

Linardatos, P., V. Papastefanopoulos, and S. Kotsiantis (2021). “Explainable AI: A Review of Machine Learning Interpretability Methods”. In: *Entropy* 23.1. ISSN: 1099-4300. DOI: `10.3390/e23010018`.

Ruijters, E. and M. Stoelinga (2015). “Fault tree analysis: A survey of the state-of-the-art in modeling, analysis and tools”. In: *Computer Science Review* 15-16, pp. 29–62. ISSN: 1574-0137. DOI: `https://doi.org/10.1016/j.cosrev.2015.03.001`.

Sproston, J. (2023). “Qualitative reachability for open interval Markov chains”. In: *PeerJ Computer Science* 9, e1489. DOI: `10.7717/peerj-cs.1489`.

Appendices

Appendix A

Appendix: Introduction

A.1 Example of a Multi-State Deterioration model with two states

Here we demonstrate that in a two-state Markov chain model with a single transition (from nominal to non-nominal behaviour) the state probability of being in the nominal state $S_N(t)$ is equivalent to the reliability function $R(t)$. For this, Figure A.1 presents a two-states Markov chain, where N represents *nominal* state and $\neg N$ is the *non-nominal* state. By using the master equation of Markov chains (Eq. 4.3), we define the corresponding system of differential equations in Eq. A.1.

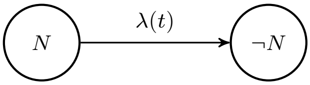


Figure A.1: Markov chain with 2 states. N represents *nominal* state and $\neg N$ is the *non-nominal* state. $\lambda(t)$ is the hazard rate function.

$$\frac{\partial S_N(t)}{dt} = -\lambda(t)S_N(t) \tag{A.1a}$$

$$\frac{\partial S_{\neg N}(t)}{dt} = \lambda(t)S_N(t) \tag{A.1b}$$

In Eq. A.1, $S_N(t)$ and $S_{\neg N}(t)$ represent respectively the state probability of being in the nominal and non-nominal states; and $\lambda(t)$ is the hazard rate. In Appendix C.1, more fundamental relations commonly used in reliability are presented. There, the following relations are derived:

$$f(t) = -R'(t) \tag{A.2a}$$

$$\lambda(t) = \frac{f(t)}{R(t)} \tag{A.2b}$$

$$R(t) = \exp\Big(-\int_0^t \lambda(s)ds\Big) \tag{A.2c}$$

Where $t \geq 0$ represents time and it is positively defined; $f(t)$ is the probability density function; and $R(t)$ is the reliability function. Now, by replacing these relations in Eq. A.1, we get:

$$\frac{\partial S_N(t)}{dt} = \frac{R'(t)}{R(t)} S_N(t) \tag{A.3a}$$

$$\frac{\partial S_{\neg N}(t)}{dt} = -\frac{R'(t)}{R(t)} S_N(t) \tag{A.3b}$$

Let's focus in Eq. A.3a, we solve it as follows:

$$\int_0^t \frac{dS_N(s)}{S_N(s)} = \int_0^t \frac{\frac{dR(s)}{ds}}{R(s)} ds = \int_0^t \frac{dR(s)}{R(s)} \tag{A.4a}$$

$$\Big[\ln|S_N(s)|\Big]_0^t = \Big[\ln|R(s)|\Big]_0^t \tag{A.4b}$$

$$\ln|S_N(t)| - \ln|S_N(0)| = \ln|R(t)| - \ln|R(0)| \tag{A.4c}$$

In Eq. A.4.c, $S_N(0)$ is known as the initial state probability, and by definition $R(0) = 1$, then:

$$\ln|S_N(t)| = \ln|R(t)| + \ln|S_N(0)| \tag{A.5a}$$

$$S_N(t) = S_N(0)R(t) \tag{A.5b}$$

From Eq. A.5, we observe that if $S_N(0) = 1$, then $S_N(t) = R(t)$. This means that when the initial state probability in the nominal condition is 1, the probability of being in the nominal state $S_N(t)$ is equivalent to the reliability function $R(t)$. Let's now consider Eq. A.3b.

$$dS_{\neg N}(t) = -\frac{R'(t)}{R(t)} S_N(t)dt, \tag{A.6}$$

Replacing Eq. A.5b in Eq. A.6, and making a change of variable, we get:

$$\int_0^t dS_{\neg N}(s) = -S_N(0)\int_0^t dR(s) \tag{A.7a}$$

$$\Big[S_{\neg N}(s)\Big]_0^t = -S_N(0)\Big[R(s)\Big]_0^t \tag{A.7b}$$

$$S_{\neg N}(t) - S_{\neg N}(0) = -S_N(0)\big(R(t) - R(0)\big) \tag{A.7c}$$

$$S_{\neg N}(t) = S_N(0)\big(1 - R(t)\big) + S_{\neg N}(0) \tag{A.7d}$$

Where $S_{\neg N}(0)$ is the probability of being in the non-nominal state at $t = 0$. Since $S_N(0) + S_{\neg N}(0) = 1$, we see that if $S_N(0) = 1$, from Eq. A.7 we get that $S_{\neg N}(t) = 1 - R(t) = 1 - S_N(t)$.

A different way to demonstrate this is by taking Eq. A.1 and directly integrate:

$$\int_0^t \frac{\partial S_N(t)}{S_N(t)} = -\int_0^t \lambda(t)dt \tag{A.8a}$$

$$\int_0^t \frac{\partial S_{\neg N}(t)}{S_N(t)} = \int_0^t \lambda(t)dt \tag{A.8b}$$

Now, using Eq. A.2c in Eq. A.8a, we get:

$$\Big[\ln|S_N(s)|\Big]_0^t = \ln\big(R(t)\big) \tag{A.9a}$$

$$\ln|S_N(t)| - \ln|S_N(0)| = \ln\big(R(t)\big) \tag{A.9b}$$

$$\ln|S_N(t)| = \ln\big(R(t)\big) + \ln|S_N(0)| \tag{A.9c}$$

$$S_N(t) = S_N(0)R(t) \tag{A.9d}$$

Replacing Eq. A.9d in Eq. A.8b, we get:

$$\int_0^t S_{\neg N}(t) = \int_0^t \lambda(t)S_N(t)dt \tag{A.10a}$$

$$\Big[S_{\neg N}(t)\Big]_0^t = S_N(0)\Big[-R(t)\Big]_0^t \tag{A.10b}$$

$$S_{\neg N}(t) - S_{\neg N}(0) = S_N(0)\big[-R(t) + 1\big] \tag{A.10c}$$

$$S_{\neg N}(t) = S_N(0)\big(1 - R(t)\big) + S_{\neg N}(0) \tag{A.10d}$$

A.2 Example of a Multi-State Deterioration model with three states

Figure A.2 presents a Markov chain with three states. The corresponding system of differential equations formulated from the master equation of the Markov chains (Eq. 4.3) is presented in Eq. A.11.

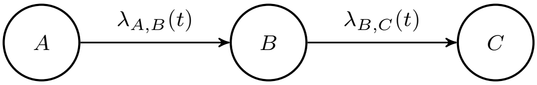


Figure A.2: Markov chain with 3 states.

$$\frac{\partial S_A(t)}{dt} = -\lambda_{A,B}(t)S_A(t) \tag{A.11a}$$
$$\frac{\partial S_B(t)}{dt} = \lambda_{A,B}(t)S_A(t) - \lambda_{B,C}(t)S_B(t) \tag{A.11b}$$
$$\frac{\partial S_C(t)}{dt} = \lambda_{B,C}(t)S_B(t) \tag{A.11c}$$

To solve the system of differential equations in Eq. A.11, we follow a similar approach to the two-state Markov chain described in Section A.1. Let's start first with Eq. A.11a.

$$\frac{\partial S_A(t)}{dt} = -\lambda_{A,B}(t)S_A(t)$$
$$S_A(t) = S_A(0)R_{A,B}(t) \tag{A.12a}$$

To get the solution of Eq. A.11b, we first make use of the following expression:

$$u(t) = \exp\left(\int_0^t \lambda(\tau)d\tau\right) \tag{A.13a}$$
$$u'(t) = \lambda(t)u(t) \tag{A.13b}$$

By multiplying $u(t)$ to Eq. A.11b, and making use of the relation $u'_{B,C}(t) = \lambda_{B,C}(t)u_{B,C}(t)$, we get:

$$\frac{dS_B(t)}{dt}u_{B,C}(t) + S_B(t)\frac{du_{B,C}(t)}{dt} = u_{B,C}(t)\lambda_{A,B}(t)S_A(t)$$
$$\frac{d}{dt}\Big[S_B(t)u_{B,C}(t)\Big] = u_{B,C}(t)\lambda_{A,B}(t)S_A(t)$$
$$S_B(t) = \frac{1}{u_{B,C}(t)}\left[\int_0^t u_{B,C}(\tau)\lambda_{A,B}(\tau)S_A(\tau)d\tau + C^*\right]$$

By replacing $u(t) = 1/R(t)$, and considering that $R(t=0) = 1$ and $S_B(t=0) = S_B(0)$ we find that $C^* = S_B(0)$. Also, considering $\lambda_{A,B}(t) = -R'_{A,B}(t)/R_{A,B}(t)$ and $S_A(t) = S_A(0)R_{A,B}(t)$, we get:

$$S_B(t) = R_{B,C}(t)\Big[\int_0^t \frac{1}{R_{B,C}(\tau)}\lambda_{A,B}(\tau)S_A(\tau)d\tau + S_B(0)\Big]$$

$$S_B(t) = R_{B,C}(t)\Big[-S_A(0)\int_0^t \frac{1}{R_{B,C}(\tau)}\frac{R'_{A,B}(\tau)}{R_{A,B}(\tau)}R_{A,B}(\tau)d\tau + S_B(0)\Big]$$

$$S_B(t) = R_{B,C}(t)\Big[S_B(0) - S_A(0)\int_0^t \frac{R'_{A,B}(\tau)}{R_{B,C}(\tau)}d\tau\Big] \tag{A.15a}$$

Finally, we find $S_C(t)$ following a similar process, we provide below the result:

$$S_C(t) = \int_0^t R'_{B,C}(\tau)\Big(S_A(0)\int_0^t \frac{R'_{A,B}(u)}{R_{B,C}(u)}du - S_B(0)\Big)d\tau + S_C(0) \tag{A.16}$$

Appendix B

Appendixes: FT-MOEA

B.1 Data-driven methods to infer FTs from data

Table B.1 (divided in two parts) contain references associated to data-driven algorithms to infer FTs, the name of the algorithm (if any), whether it is publicly available (if 'yes' the table provides a hyperlink redirecting to the respective online repository), the key aspects of the methodology, the input data, the benefits, and drawbacks.

B.2 Applying NSGA-II and Crowding-Distance to infer FTs

B.2.1 Applying NSGA-II to infer FTs

We provide a conceptual visualisation in Figure B.1 that explains our implementation of the `NSGA-II` and `Crowding-Distance` in the context of the automatic inference of FTs. To ease the visualisation, we consider the bi-dimensional case where the multi-objective function is *sd* (see Section 2.5.5, Table 2.4). After computing the metrics for a population of FTs within a given generation, one can depict the FTs with circles as in Figure B.1(a).

The output of the `NSGA-II` algorithm is a set of *Pareto fronts*, represented in Figure B.1(b) with different colours (red, blue, green and purple). Figure B.1(c) shows some details related to the FTs in the first front (red). Note that these FTs have different structures. Here the top FTs has a higher error based on the failure dataset (ε_d) compared to the others, but it is the smallest FTs in the first front. On the contrary, the bottom FTs has a smaller error in ε_d, but with the trade-off of having a larger size.

Table B.1: (Part I) *Data-driven* approaches for the automatic inference of FT models. In the columns the references, algorithm name, whether the algorithm is publicly available or not, the key aspects of the methodology, the input data, the benefits and drawbacks.

Reference(s)	*Name*	*Aval?*	*Methodology*	*Input*	*Benefits*	*Drawbacks*
M. G. Madden and P. J. Nolan, 1994; M. G. Madden, 1970; M. Madden and P. Nolan, 1999	`IFT`	No	Based on the ID3 algorithm Quinlan, 1986 to induce Decision Trees (DTs).	TS	Purely data-driven approach. Provides insights to carry out rules-based diagnostics.	It is unclear whether the `IFT` algorithm guarantees the encoding of cause-effect relationships, which is a requirement for FT models.
Berikov, 2004	-	No	They build FT models based on DTs.	TS	They build the FT model in layers from the DT, this is an interesting way to exploit DTs capabilities.	It is difficult to determine automatically the non-terminal nodes of the FT without expert advice.
Mukherjee and Chakraborty, 2007	-	No	Based on text mining and natural language processing.	Text	Makes use of data such as maintenance reports.	It requires a manually built (partial) FT, which is then refined through their method.
Roth, Wolf, and Lindemann, 2015	-	No	Based on various matrix analysis methods commonly used to model dependencies.	SDR	The approach helps to identify critical failures or elements from a structural point of view. Also enables handling alternative input data.	It requires functional analysis and expert domain support.
Nauta, Bucur, and Stoelinga, 2018	`LIFT`	Yes	Based on the Mantel-Haenszel statistical test.	BD	It works well for small problems and with low noise levels in the data.	It requires as input information about the intermediate events. Moreover, the algorithm does an exhaustive search, and thus has exponential time complexity.
Waghen and Ouali, 2019	`ILTA`	No	Knowledge Discovery in Dataset (KDD) + Interpretable Logic Trees (ILT)	TS	It does not require human expertise in the construction stage. It generates intuitive logic trees structures that encode hidden system's causal relations.	The `ILTA` algorithm is insufficient when several subsystems and characteristic variables may have interdependent relationships with each other.

Abbreviations: Decision Trees (**DTs**), Time Series (**TS**), Binary Data (**BD**), System Domain and Relations (**SDR**).

Table B.2: (Part II) *Data-driven* approaches for the automatic inference of FT models. In the columns the references, algorithm name, whether the algorithm is publicly available or not, the key aspects of the methodology, the input data, the benefits and drawbacks.

Reference(s)	*Name*	*Aval?*	*Methodology*	*Input*	*Benefits*	*Drawbacks*
Linard, Bucur, and Stoelinga, 2019	`FT-EA`	Yes	Based on evolutionary algorithms and a one-dimensional cost function based on accuracy computed from the labelled binary fault dataset.	BD	Achieves FT models with small error based on the fault dataset.	The cost function does not consider the FT model size, leading to massive, difficult-to-interpret structures in large problems, increasing computational time to converge. It can only handle AND and OR gates.
Linard, Bueno, Bucur, et al., 2020	`FT-BN`	Yes	First learns the parameters of a Bayesian Network (BN) and then translates the BN into a FT.	BD	The method is particularly robust in handling noisy data.	Requires prior assumptions based on *white-* and *black-listing*, defining arcs that are respectively missing or present in the BN.
Lazarova-Molnar, Niloofar, and Barta, 2020	`DDFTA`	No	Utilises binarization techniques, MCSs, and Boolean algebra	TS	Efficient in terms of computational time. The algorithm infers VoT logic gates.	Unclear how the algorithm performs under noisy data.
Waghen and Ouali, 2021	`MILTA`	No	Uses an iterative burn-and-build algorithm to identify causal relations between the fault event and its intermediate causes, level after level until the root causes are uncovered.	TS	The algorithm does not require human expert participation to build the model. It focuses on finding data patterns indicating the logical relationships of the FT model.	The authors note that as the system degrades over time, the `MILTA` model must be improved to capture the evolution of the failure event.
Ch. 2	`FT-MOEA`	Yes	Multi-objective evolutionary algorithms based on Non-dominated Sorting Genetic Algorithms and Crowding-Distance	BD	The multi-target function is highly customizable. Minimising the size of the FT model provides small, easy-to-interpret structures, functioning as a compression technique. They often converge to the same FT structure.	Computation time increases exponentially with problem complexity. It can only handle labelled binary data, inferring FT models using AND and OR gates. Depending on the cost function, it may fall into local optima.

Abbreviations: Bayesian Network (**BN**),Time Series (**TS**), Binary Data (**BD**).

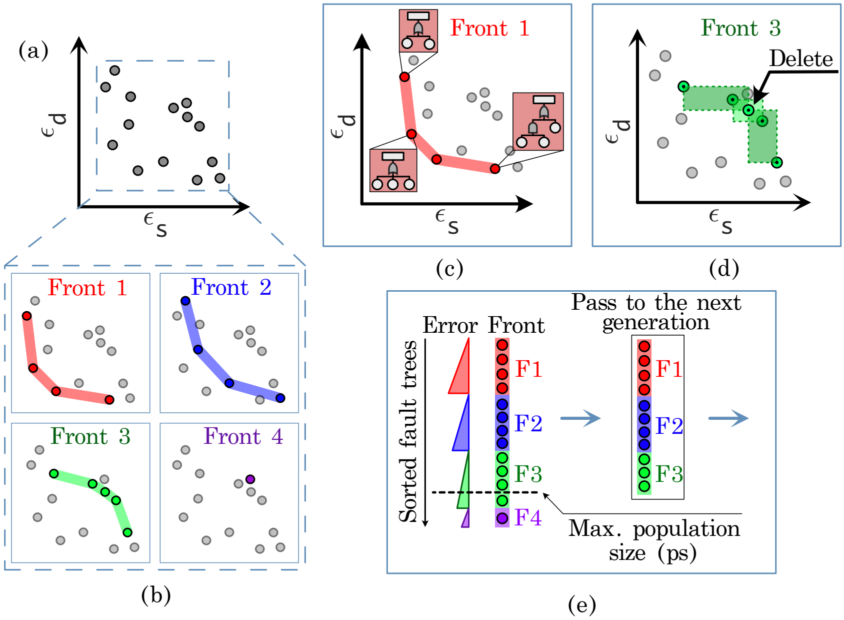


Figure B.1: Conceptual visualisation of the *Non-Dominated Sorting Genetic Algorithms* (`NSGA-II`) and `Crowding-Distance` in the context of automatic inference of FTs. In (a) error based on the failure dataset (ε_d) versus fault tree size (ε_s), in (b) Pareto fronts, (c) details of the first front, (d) influence of the Crowding-Distance, and (e) criteria for acceptance and rejection of FTs between generations.

Figure B.1(d) shows the effect of the `Crowding-Distance`. Suppose that only four of the five FTs of the third front can pass to the next generation. Therefore, it is necessary to "break" the front. To do so, we compute the *Crowding-Distance metric* (d_i) (Section B.2.2). Those solutions in the front with a large d_i value have priority to pass to the next generation. Conversely, those with a small d_i value have a lower priority because it means that they are similar to other solutions in the front. Therefore, in Figure B.1(d) the FTs marked with the arrow "Delete" must have similar features in ε_s and ε_d compared to its neighbours, thus becoming the candidate to be deleted from the third front.

Figure B.1(e) represents the process to select the FTs that pass to the next generation, where first the FTs within each front are ordered from the minimum to the maximum error (or errors when considering the minimisation of both ϕ_c and ϕ_d, here we sum them up and sort them up), then only the first *ps* FTs pass to the next generation. Here we can observe that one FTs from the third front and the only FTs from the fourth front did not manage to pass to the next generation.

B.2.2 Crowding-Distance

This process is based on the *Crowding-Distance metric* (d_i) which makes that individuals with a large d_i wins, the latter to avoid solutions to be similar (i.e., to maintain diversity). This metric estimates the density of a particular solution in the non-dominated front based on the neighbour solutions. We provide below a summary of the steps necessary to compute d_i. For details, we suggest the reader to consult Deb, 2005.

- Step 1: For each solution in the non-dominated set ($\mathscr{F}$) assign $d_i = 0$.
- Step 2: for all the objective functions $m = 1, 2, ..., M$, sort the set in worse order of f_m and obtain the sorted index vector as $I^m = sort(f_m, >)$.
- Step 3: For all the objective functions, assign a large distance to the boundary solutions (i.e., $d_{I_1^m} = d_{I_{|F|}^m} = \infty$), and for the rest of the solutions $j = 2$ to $(|\mathscr{F}| - 1)$, assign
$$d_{I_j^m} = d_{I_j^m} + \frac{f_m^{I_{j+1}^m} - f_m^{I_{j-1}^m}}{f_m^{max} - f_m^{min}} \tag{B.1}$$
Here I_j denotes the solution index of the j-th member sorted in the vector.
- Step 4: Pass the solution with the largest d_i, then the solution with the second largest d_i, until the maximum population size is met.

B.3 Example of inferred fault trees

Here we provide as an example a pair of FTs obtained with the FT-MOEA. Figure B.2 shows the inferred FT associated with the *Mono-propellant Propulsion System* (MPPS) case study, the example that was used several times to exemplify our results. Here the right branch of the FT (i.e., the one associated with the intermediate event IE2), both sets of BEs and IEs coincide with the ground truth FT. In contrast, the intermediate events of the left branch were inferred by FT-MOEA (red boxes), to which we gave an interpretation.

Similarly, in Figure B.3 we present the inferred FT of the COVID-19 infection risk. This FT originally has 33 elements, but after applying FT-MOEA, most of the intermediate events were replaced by more efficient logics, resulting in a FT with 13 elements. Again, we provide an interpretation to the intermediate events found by the FT-MOEA (red boxes). First, it is interesting to see that all *transmission modes* were clustered under an OR-gate (IE3). We interpret the other two intermediate as *Transmissibility of COVID-19 pathogen* (AND-gate, IE2) and *Existence of COVID-19* (OR-gate, IE1).

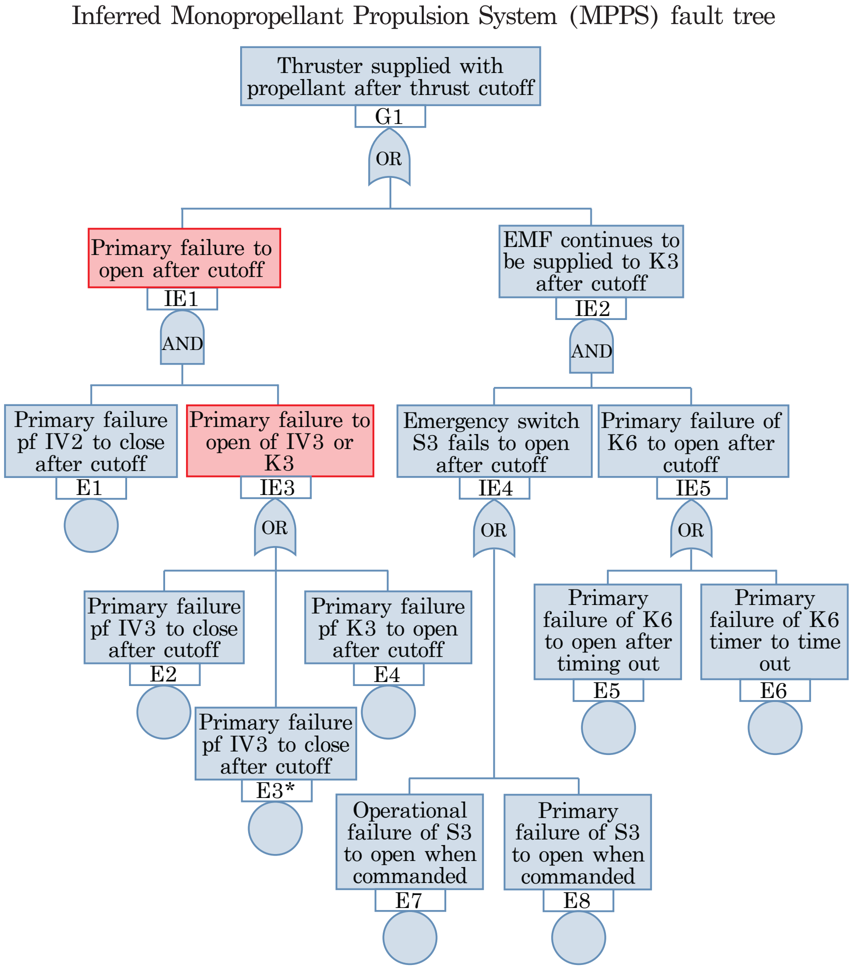

Inferred Monopropellant Propulsion System (MPPS) fault tree
Thruster supplied with propellant after thrust cutoff
G1
OR
Primary failure to open after cutoff
IE1
AND
EMF continues to be supplied to K3 after cutoff
IE2
AND
Primary failure pf IV2 to close after cutoff
E1
Primary failure to open of IV3 or K3
IE3
OR
Emergency switch S3 fails to open after cutoff
IE4
OR
Primary failure of K6 to open after cutoff
IE5
OR
Primary failure pf IV3 to close after cutoff
E2
Primary failure pf K3 to open after cutoff
E4
Primary failure of K6 to open after timing out
E5
Primary failure of K6 timer to time out
E6
Primary failure pf IV3 to close after cutoff
E3*
Operational failure of S3 to open when commanded
E7
Primary failure of S3 to open when commanded
E8

Figure B.2: Inferred Mono-propellant Propulsion System (MPPS) fault tree after applying `FT-MOEA`, source: NASA, 2002.

B.4 Details of Convergence of Metrics Over Generations

Figure B.4 depicts the convergence of the metrics ε_s, ε_d, and ε_c over the generations for the same example discussed in Figure 2.6 in Section 2.6.3. In this instance, the *distribution* within the population of each generation is also visualised. The aim is to provide a better understanding of the convergence of the metrics between generations. The common elements in these figures are grey shades associated with the percentiles 25% (darkest shade), 50%, 75%, and 100% (lightest shade), a horizontal blue line indicating the size of the ground truth FT, red dots representing

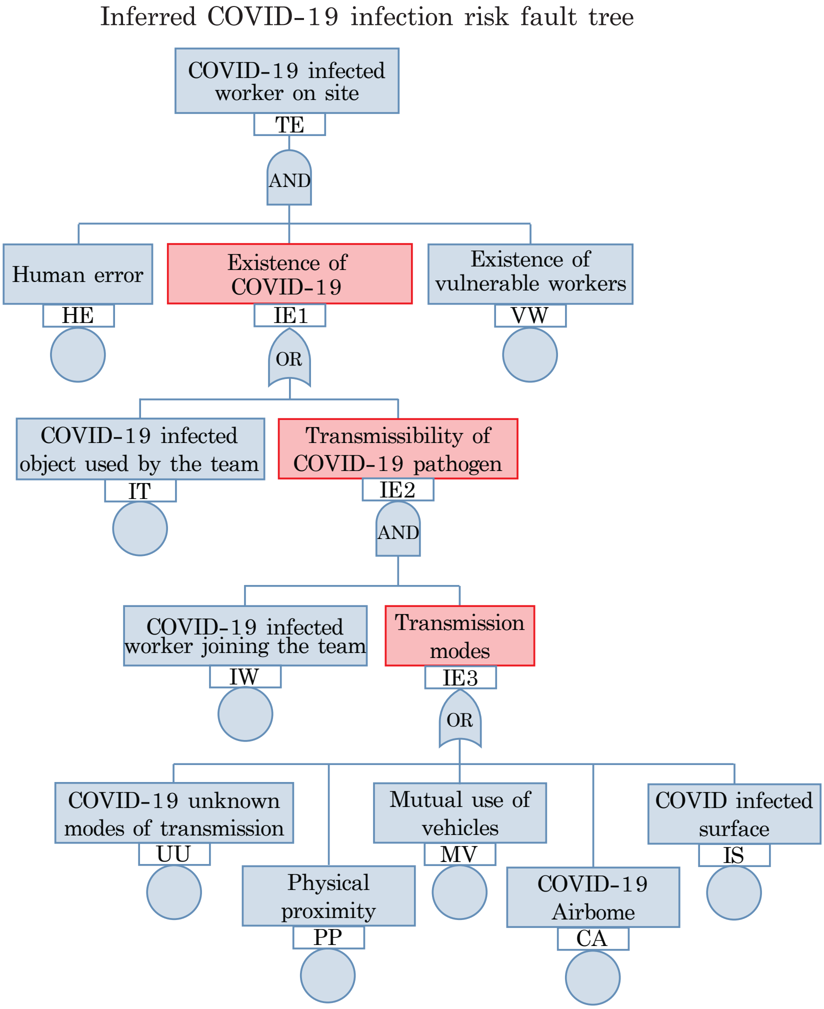
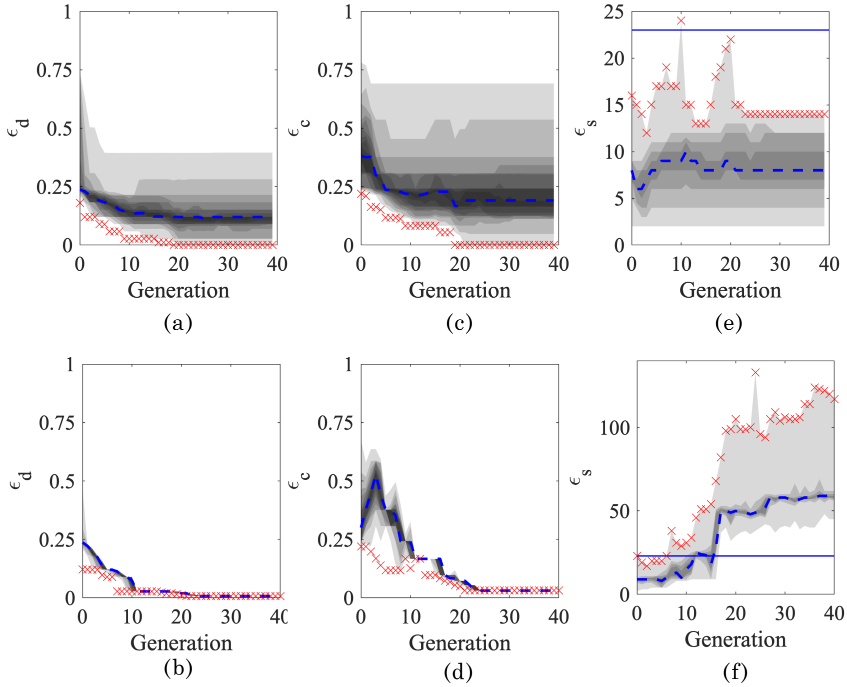

Inferred COVID-19 infection risk fault tree
COVID-19 infected worker on site
TE
AND
Human error
HE
Existence of COVID-19
IE1
Existence of vulnerable workers
VW
OR
COVID-19 infected object used by the team
IT
Transmissibility of COVID-19 pathogen
IE2
AND
COVID-19 infected worker joining the team
IW
Transmission modes
IE3
OR
COVID-19 unknown modes of transmission
UU
Physical proximity
PP
Mutual use of vehicles
MV
COVID-19 Airbome
CA
COVID infected surface
IS

Figure B.3: Inferred COVID-19 infection risk fault tree after applying `FT-MOEA`, source: Bakeli, Hafidi, et al., 2020.

the extreme values in that generation for a given metric, and a white dashed line indicating the mean value of the metric.

Figures B.4(a) and B.4(b) illustrate the convergence of ε_d. By using the multi-objective function (m.o.f.) *sdc*, we observe higher variance compared to the m.o.f. *d* throughout the generations. This is due to the fact that some FTs are Pareto optimal in other aspects, e.g., FTs with a small size, which often have higher error. In contrast, Figure B.4(b) shows less variance, indicating that FTs within a generation have a similar error based on the failure dataset (ε_d).



(a) (c) (e)

(b) (d) (f)

Figure B.4: Visualization of metrics (ε_s, ε_d, ε_c) over the generations considering the percentiles 25%, 50%, 75% and 100% for the case study MPPS ($ps = 400$, $ng = 100$, $uc = 20$). In (a), (c) and (e) using the m.o.f. `sdc`, and in (b), (d), and (f) using the m.o.f. `d`.

Figures B.4(c) and B.4(d) illustrate the convergence of ε_c. A similar pattern is observed, where our approach maintains more "variety" of FT structures between generations, with each structure being Pareto efficient in at least one metric. In Figure B.4(d), we observe that variance decreases with generations, indicating that FTs tend to become more similar in their MCS matrices in later generations.

Figures B.4(e) and B.4(f) illustrate the convergence of ε_s. Here, the most significant differences between both approaches can be seen. In Figure B.4(e), we observe that the FTs tend to be small. It is important to note that just before finding the global optimum, the FTs within a generation tend to increase in size. Once the global optimum is found (i.e., $\varepsilon_d = \varepsilon_c = 0$), the remaining part of the process naturally focuses on minimising the FT size, resulting in a *compressed* version of the found global optimum. On the other hand, Figure B.4(f) shows that not controlling the size of the FT results in a *structural explosion*, yielding massive structures that are not beneficial for the inference process.

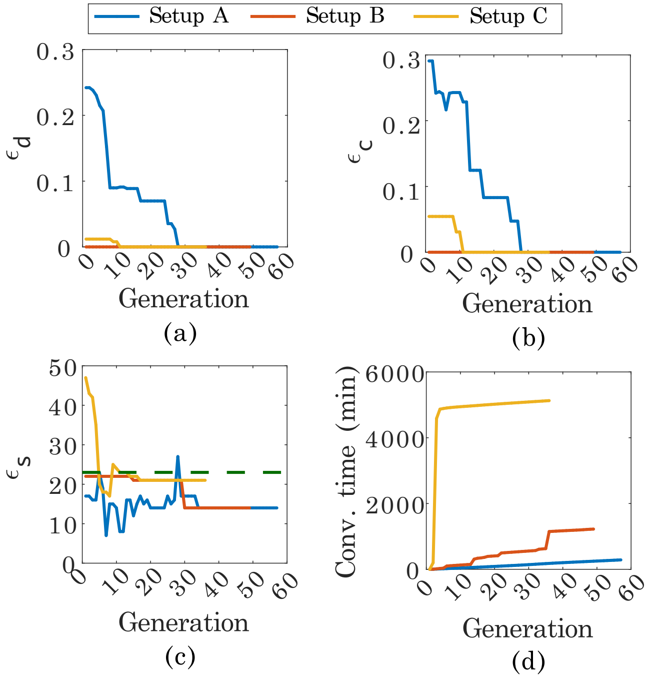


Figure B.5: Influence of varying the parent FTs on (a) ε_d, (b) ε_d, (c) ε_s (green dashed line is the ground truth FT size), (d) convergence time. For the m.o.f *sdc*, the case study MPPS (ps = 400, ng = 100, uc = 20, ρ = 6). *Setup A*: as described in Section 2.5.3; *Setup B*: using as parent FT the *disjunctive normal form*; *Setup C*: using as parent FT a sub-optimal FT of ϕ_s = 98 obtained with the m.o.f. d in Figure 2.9(d).

B.5 Varying parent Fault Trees (details)

We defined in Section 2.5.3 what a parent FT is, and here we evaluate the effects of varying this parameter. We use the case study MPPS, the m.o.f. *sdc* with the following parameters ps = 400, ng = 100, uc = 20, ρ = 6. We consider three setups. *Setup A* is our reference, we have been using it throughout this paper (see Section 2.5.3). *Setup B* consists of a single parent FT based on the Disjunctive Normal Form. *Setup C* takes an inferred sub-optimal FT model with ε_s = 98 previously obtained using the m.o.f. d in Figure 2.9(a).

We already discussed in detail the results of Setup A in Figure 2.6, Section 2.6.3. Figure B.5(a) and (b) shows that Setup B has an error of zero since the onset (i.e., $\varepsilon_d = \varepsilon_d = 0.0$). This is expected because Setup B considers as parent FT the Disjunctive Normal Form (DNF). However, this does not mean the parent FT has the optimal structure, this is what we observe in Figure B.5(c) where around the 30th generation the `FT-MOEA` found a smaller structure with the same performance. The latter structure is the same as the one where Setup A converged.

Setup C started with a sub-optimal parent FT of $\varepsilon_s = 98$ but in Figure B.5(c) we observe that right from the first generation, the `FT-MOEA` algorithm found a better solution of $\varepsilon_s = 47$, and it converged in $\varepsilon_s = 21$. Figure B.5(d) shows that it took more than 3 days to get through the first three generations, the reason being that computing the MCSs of many large FTs was expensive, but as soon as the FTs of the population reached a smaller size, the process became faster.

From our results we conclude that Setup A is faster than Setups B and C, although it took more generations to find the global optimum. On the contrary, Setup C found the global optimum with fewer generations, but both, Setup B and C are slower, the reason most likely is because obtaining MCSs from large FTs is computationally expensive.

B.6 Comparing the performance of m.o.f.'s for the case studies CSD, PT, and SMS'

These figures add to the discussion in Section 2.6.4. Figure B.6 compares the performance of multi-objective functions (m.o.f.'s) for the case studies CSD, PT, and SMS in Table 2.5, using $ps = 400$, $ng = 100$, $uc = 20$.

B.7 Noise Effects in the Inference of Fault Trees with `FT-MOEA`

To illustrate the influence of noise on the inference of FTs, Figure B.7 presents the results based on the COVID-19 case study using *box charts*, constructed as described in Section 2.6.4.

The noise was modelled similarly to the method used for generating the failure dataset via the Monte Carlo method (see Section 2.6.1), but here the probability of success p_i corresponds to the level of noise added to the failure dataset.

We evaluated noise levels ranging from 0% (i.e., noise-free failure dataset) to 30%. Figure B.7 illustrates the *reference* (orange boxes), which represents the inferred FT metrics calculated with respect to the failure dataset **without noise**. In contrast, the blue boxes show the same inferred FT metrics calculated with respect to the failure dataset **with noise**.

Figure B.7(a) shows that for the COVID-19 case study, the `FT-MOEA` sometimes manages to infer the global optimum (i.e., $\varepsilon_d = 0$ in the orange boxes) up to a noise level of 15%. At higher noise levels, the algorithm fails to find the global optimum. We observe that the values in the blue boxes increase with the level of noise, as expected, since ε_d will never be zero in the presence of noise.

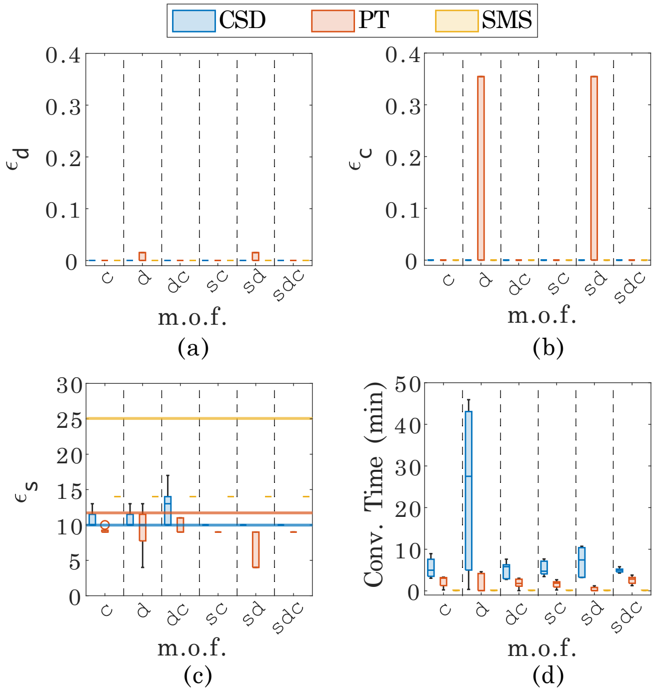


Figure B.6: Comparing the performance of all m.o.f.'s using $ps = 400$, $ng = 100$, $uc = 20$ and the case studies CSD, PT, and SMS. In (a) ε_d, (b) ε_c, (c) ε_s, (d) convergence time.

A similar reasoning applies to the results in Figure B.7(b). Notice that we use the m.o.f. *sd*, meaning that we are not considering the MCSs in the cost function because MCSs cannot be computed from noisy failure datasets. Here, we observed that some inferred FTs have $\varepsilon_c = 0$ up to noise levels of 15% (i.e., global optimum). For higher noise levels, ε_c seems to increase.

Figure B.7(c) shows that despite the level of noise, all the inferred FTs are smaller than the ground truth (i.e., $\varepsilon_s < 33$, red line). Regarding convergence time (Figure B.7(d)), we do not observe a particular trend.

In conclusion, `FT-MOEA` can handle noisy data, at least to a certain extent. Further research should explore whether it is possible to make `FT-MOEA` even more robust.

B.8 References

Bakeli, T., A. A. Hafidi, et al. (2020). "COVID-19 infection risk management during construction activities: An approach based on Fault Tree Analysis (FTA)". In: *Journal of Emergency Management* 18.7, pp. 161–176. DOI: 10.5055/jem.0539.

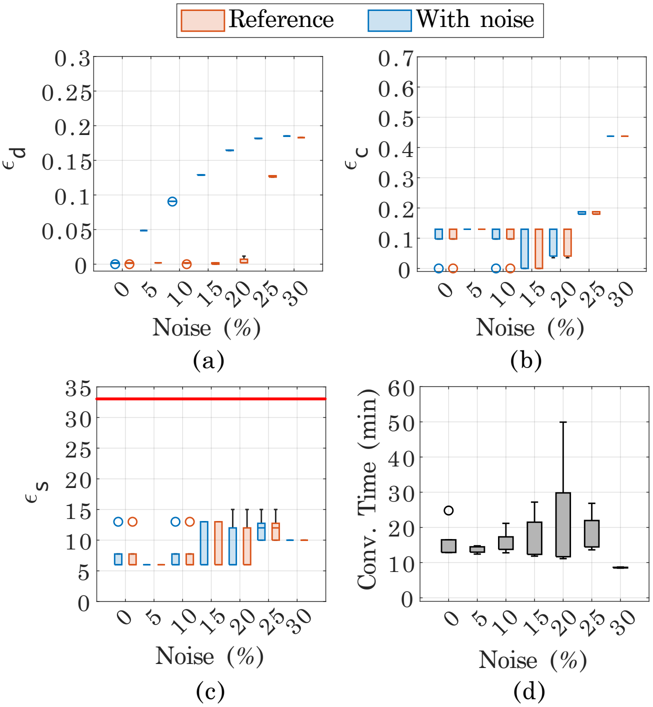


Figure B.7: Assessment of noise on the inference of FTs with m.o.f. *sd* using $ps = 400$, $ng = 100$, $uc = 20$ for the COVID-19 case study. In (a) ε_d, (b) ε_c, (c) ε_s (red line indicates the size of the ground truth FT), (d) convergence time.

Berikov, V. (2004). “Fault tree construction on the basis of multivariate time series analysis”. In: *Proceedings. The 8th Russian-Korean International Symposium on Science and Technology, 2004. KORUS 2004.* Vol. 2, 103–106 vol. 2. DOI: 10.1109/KORUS.2004.1555557.

Deb, K. (2005). “Multi-Objective Optimization”. In: *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques.* Ed. by E. K. Burke and G. Kendall. Boston, MA: Springer US, pp. 273–316. ISBN: 978-0-387-28356-2. DOI: 10.1007/0-387-28356-0_10.

Lazarova-Molnar, S., P. Niloofar, and G. K. Barta (2020). “Data-Driven Fault Tree Modeling for Reliability Assessment of Cyber-Physical Systems”. In: *2020 Winter Simulation Conference (WSC)*, pp. 2719–2730. DOI: 10.1109/WSC48552.2020.9383882.

Linard, A., M. Bueno, D. Bucur, and M. Stoelinga (2020). “Induction of fault trees through Bayesian networks”. In: *Proceedings of the 29th European Safety and Reliability Conference, ESREL 2019*, pp. 910–917. DOI: 10.3850/978-981-11-2724-3_0596-cd.

Linard, A., D. Bucur, and M. Stoelinga (2019). “Fault Trees from Data: Efficient Learning with an Evolutionary Algorithm”. In: *International Symposium on Dependable Software Engineering: Theories, Tools, and Applications.* Vol. 11951 LNCS, pp. 19–37. DOI: 10.1007/978-3-030-35540-1_2.

Madden, M. and P. Nolan (1999). “Monitoring and diagnosis of multiple incipient faults using fault tree induction”. In: *IEE Proceedings: Control Theory and Applications* 146.2, pp. 204–212. DOI: 10.1049/IP-CTA:19990088.

Madden, M. G. (1970). “Hierarchically structured inductive learning for fault diagnosis”. In: *WIT Transactions on Information and Communication Technologies* 20. DOI: 10.2495/AI980411.

Madden, M. G. and P. J. Nolan (1994). “Generation of fault trees from simulated incipient fault case data”. In: *WIT Transactions on Information and Communication Technologies* 6. DOI: 10.2495/AI940611.

Mukherjee, S. and A. Chakraborty (2007). “Automated fault tree generation: bridging reliability with text mining”. In: *2007 Annual Reliability and Maintainability Symposium*. IEEE, pp. 83–88. DOI: 10.1109/RAMS.2007.328096.

NASA (2002). *Fault Tree Handbook with Aerospace Applications*. Handbook. U.S. National Aeronautics and Space Administration.

Nauta, M., D. Bucur, and M. Stoelinga (2018). “LIFT: Learning fault trees from observational data”. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 11024 LNCS, pp. 306–322. DOI: 10.1007/978-3-319-99154-2_19.

Quinlan, J. (1986). “Induction of Decision Trees”. In: *Machine Learning* 1.1, pp. 81–106. DOI: 10.1023/A:1022643204877.

Roth, M., M. Wolf, and U. Lindemann (2015). “Integrated Matrix-based Fault Tree Generation and Evaluation”. In: *Procedia Computer Science* 44. 2015 Conference on Systems Engineering Research, pp. 599–608. ISSN: 1877-0509. DOI: 10.1016/j.procs.2015.03.027.

Waghen, K. and M.-S. Ouali (2019). “Interpretable logic tree analysis: A data-driven fault tree methodology for causality analysis”. In: *Expert Systems with Applications* 136, pp. 376–391. DOI: 10.1016/j.eswa.2019.06.042.

Waghen, K. and M.-S. Ouali (2021). “Multi-level interpretable logic tree analysis: A data-driven approach for hierarchical causality analysis”. In: *Expert Systems with Applications* 178, p. 115035. ISSN: 0957-4174. DOI: 10.1016/j.eswa.2021.115035.

Appendix C

Appendix: Multi-State Deterioration

C.1 Relations in reliability analysis

From *probability theory*, in an absolutely continuous univariate distribution, a random variable T has a *density function* f_T, where f_T is a non-negative Lebesgue-integrable function:

$$\mathbb{P}[a \leq T \leq b] = \int_a^b f_T(t)dt \tag{C.1}$$

The *cumulative distribution function* $F_T(t)$ is:

$$\mathbb{P}[T \leq t] = F_T(t) = \int_{-\infty}^t f_T(u)du \tag{C.2}$$

The *survival function* $\mathbf{S}_T(t)$, mathematically equivalent to the *reliability function* $R(t)$, is:

$$\mathbb{P}[T \geq t] = \mathbf{S}_T(t) = R(t) = 1 - F_T(t) = \int_t^\infty f_T(u)du \tag{C.3}$$

Now, let the *hazard function* $\lambda(t)$ be defined as:

$$\lambda(t) = \lim_{\Delta t \to 0} \frac{\mathbb{P}[t \leq T < t + \Delta t | T \geq t]}{\Delta t} \tag{C.4}$$

When applying Bayes theorem in Eq. C.4, we get:

$$\mathbb{P}[t \leq T < t + \Delta t | T \geq t] = \frac{\mathbb{P}[(t \leq T < t + \Delta t) \cap (T \geq t)]}{\mathbb{P}[T \geq t]}$$

Since $T \geq t$ is part of the event $t \leq T < t + \Delta$, we simplify:

$$\mathbb{P}[(t \leq T < t + \Delta) \cap (T \geq t)] = \mathbb{P}[t \leq T < t + \Delta t]$$

Finally, we get:

$$\mathbb{P}[t \leq T < t + \Delta t | T \geq t] = \frac{\mathbb{P}[t \leq T < t + \Delta t]}{\mathbb{P}[T \geq t]} \tag{C.5}$$

By replacing Eq. C.5 in Eq. C.4 we get:

$$\lambda(t) = \lim_{\Delta t \to 0} \frac{\mathbb{P}[t \leq T < t + \Delta t]}{\Delta t \mathbb{P}[T \geq t]} \tag{C.6}$$

For $\Delta_t \to 0$, from Eq. C.1 we get that:

$$\mathbb{P}[t \leq T \leq t + \Delta t] = \int_t^{t+\Delta t} f_T(t) dt \approx f(t) \cdot \Delta t \tag{C.7}$$

By plugging-in Eq. C.7 and Eq. C.3 into Eq. C.6, we get:

$$\lambda(t) = \frac{f(t)}{\mathbf{S}(t)} \tag{C.8}$$

From Eq. C.3 we get that:

$$R'(t) = -f(t) \tag{C.9}$$

Then by replacing Eq. C.9 in Eq. C.8 we get:

$$\frac{R'(t)}{R(t)} = -\lambda(t) \tag{C.10a}$$

$$\frac{dR(t)}{R(t)} = -\lambda(t) dt \tag{C.10b}$$

$$\int_0^t \frac{dR(t)}{R(t)} = -\int_0^t \lambda(s) ds \tag{C.10c}$$

$$\Big[\ln |R(s)| \Big]_0^t = -\int_0^t \lambda(s) ds \tag{C.10d}$$

$$\ln |R(t)| - \ln |R(0)| = -\int_0^t \lambda(s) ds \tag{C.10e}$$

$$R(t) = \exp\left(-\int_0^t \lambda(s) ds \right) \tag{C.10f}$$

Appendix D

Parameters of multi-state deterioration models

Table D.1: MSDM hyper-parameters for cohort `CMW`, using hazard functions modelled with the *exponential* ($\lambda^E(t|\epsilon)$), *Gompertz* ($\lambda^G(t|\alpha,\beta)$), and *Weibull* ($\lambda^W(t|\eta,\rho)$) probability density functions.

	$\lambda^E(t\|\epsilon)$	$\lambda^G(t\|\alpha,\beta)$		$\lambda^W(t\|\eta,\rho)$	
$i \rightarrow j$	ϵ	α	β	η	ρ
$1 \rightarrow 2$	2.4E-02	2.3E+00	8.4E-03	1.3E+00	4.4E+01
$2 \rightarrow 3$	9.4E-03	2.1E-02	5.5E-02	2.9E+00	7.7E+01
$3 \rightarrow 4$	5.7E-03	3.3E+00	2.8E-03	3.5E+00	8.1E+01
$4 \rightarrow 5$	1.8E-02	2.4E+00	8.7E-03	7.0E+00	5.5E+01
$1 \rightarrow F$	3.0E-18	1.4E-01	3.1E-04	4.1E-06	4.6E+01
$2 \rightarrow F$	6.0E-04	8.8E-01	7.0E-19	2.7E-04	4.6E+01
$3 \rightarrow F$	1.0E-18	2.2E-03	4.5E-02	3.0E-05	4.7E+01
$4 \rightarrow F$	1.0E-18	9.8E-05	8.6E-03	1.1E-03	4.5E+01
$5 \rightarrow F$	1.0E-18	7.0E-19	3.8E-01	1.7E+00	5.9E+01

Table D.2: Initial state vector S_k^0 for MSDM of cohort `CMW`.

S_k^0	Exponential	Gompertz	Weibull
$k = 1$	9.89E-01	9.58E-01	9.23E-01
$k = 2$	1.26E-17	0.00E+00	2.59E-02
$k = 3$	3.70E-23	4.00E-02	3.10E-02
$k = 4$	1.11E-02	1.61E-03	1.13E-02
$k = 5$	2.11E-22	2.00E-15	2.07E-03
$k = F$	3.87E-22	1.56E-04	6.40E-03

About the author

Lisandro Arturo Jimenez Roa was born on 22 September 1991 in Cali, Colombia. His academic journey started in 2009 at *Del Valle University*, earning a BSc in Civil Engineering in 2014. His undergraduate thesis focused on Structural Health Monitoring (SHM) for dynamic analysis of buildings, for which he received *Summa Cum Laude*. Upon graduation, he was recognised as the top student in the Engineering Faculty and also received the *Roberto Caicedo Douat Award* for achieving the highest GPA in Structural Engineering subjects.

In 2016, Lisandro carried out his MSc thesis in Civil Engineering at the *École Polytechnique Fédérale de Lausanne* in Switzerland. There, he participated in a large-scale experimental campaign at the *Earthquake Engineering and Structural Dynamics Laboratory*. He contributed to research investigating the boundary condition behaviour of single-mesh layer slender reinforced concrete walls under cyclic loads and carried out the modelling and calibration of Finite Element Models to replicate experimental behaviours.

In 2017, Lisandro briefly started a second master's degree at the *Università di Bologna* in Italy, which he interrupted to move to the Netherlands and start his Engineering Doctorate (EngD) at the *University of Twente* (UT) in 2018, in the *Construction Management and Engineering* group. Sponsored by Rijkswaterstaat, his EngD project focused on bridge asset management, employing SHM, Machine Learning, and Data Science to detect global anomalies in bridges.

In 2020, Lisandro extended his research domain to the field of Computer Science and embarked on a PhD sponsored by the PrimaVera project, a collaborative research initiative that uses big data and AI to develop predictive maintenance technologies. The project aimed to optimise maintenance timing, reduce downtime, and extend component lifespan, thereby contributing to sustainable and reliable critical infrastructure. He joined the UT's *Formal Methods and Tools* group. During his PhD, Lisandro participated in multiple conferences, workshops, and visits all over Europe to enrich his research in the field of *Prognostics and Health Management* (PHM). His main contributions range from the inference of reliability models to agent-based strategic maintenance. By the end of his PhD in 2024, Lisandro became a Dutch citizen.

Lisandro's ambitions include leading development, advancements, and collaborations to effectively make PHM a reality in different industrial domains, including the integration of AI, trustworthy prognostics, and smart maintenance. Outside work, Lisandro enjoys being active in sports such as CrossFit and sharing time with his friends and family.