Eindhoven University of Technology

MASTER

Condition-based maintenance policies using hidden Markov models

Klaasse, B.

Award date:
2020

Link to publication

Disclaimer
This document contains a student thesis (bachelor's or master's), as authored by a student at Eindhoven University of Technology. Student theses are made available in the TU/e repository upon obtaining the required degree. The grade received is not published on the document as presented in the repository. The required complexity or quality of research of student theses may vary by program, and the required minimum study period may vary in duration.

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain

EINDHOVEN UNIVERSITY OF TECHNOLOGY

MASTER THESIS

Condition-based maintenance policies using hidden Markov models

Author:
BO KLAASSE

Supervisor:
Dr. S. KAPODISTRIA
Dr. J.W. PORTEGIES

A thesis submitted in fulfillment of the requirements for the degree of Master of Science

in

Industrial and Applied Mathematics

July 20, 2020

Acknowledgements

First and foremost, I would like to thank my supervisors Stella Kapodistria and Jim Portegies for giving me the opportunity to work on this project and their help along the way. They have helped me in a great many ways and under their guidance I have grown both as a mathematician and a communicator. In particular, I am grateful for the many valuable and interesting discussions we have had throughout this project, which often lasted for hours.

My sincere thanks goes to Marko Boon, who is not only part of my master thesis committee but also gave me the amazing opportunity to write a paper about my internship in 2019. I would also like to thank Rik Timmerman for travelling with me to a conference in Milan to present this paper and his vital role in the writing process.

I would like to express my sincere gratitude to Onno Boxma, who was the supervisor of my bachelor thesis and sparked my interest in stochastic operations research.

Last but not least, I would like to thanks my family and friends for their support through the years. My special thanks are extended to the friends who suffered through reading earlier versions of this thesis and whose inputs have made it easier to read.

EINDHOVEN UNIVERSITY OF TECHNOLOGY

Abstract

Department of Mathematics and Computer Science

Master of Science

Condition-based maintenance policies using hidden Markov models

by Bo KLAASSE

Objective: Motivated by the large financial burden of maintenance costs, we study the use of multivariate Gaussian hidden Markov models for cost-sensitive condition-based maintenance policies. We consider 5 simulated scenarios of deteriorating component, where, at discrete epochs, one can either replace the component or do nothing. Our goal is to determine a policy that minimizes the average cost per time unit.

Method: Our proposed method can be split into two parts: 1. estimating the parameters of the hidden Markov model (HMM) based on monitoring data; 2. formulating and solving the partially observable Markov decision process (POMDP). The parameters of the HMM are estimated using the Baum-Welch algorithm and the POMDP is solved with approximate point-based value iteration (based on the expected total discounted cost). We test the resulting policy, as well as POMDP-based heuristics such as the QMDP policy (which chooses action based on the state-action value function of the MDP), in a statistically identical environment. In addition, we address several well-known issues of HMMs in maintenance, such as choosing the optimal number of states.

Results and insights: Our numerical experiments indicate that multivariate Gaussian HMMs can produce policies that have an average cost that is between 1% and 15% higher than the average cost of the optimal threshold policy. Additionally, we provide results that indicate that the performance of the POMDP policy is influenced by the number of states of the HMM and that even a HMM with 3 component states and 1 failure state can produce decent overall results (that is, at most 4% higher than the average cost of the POMDP policy with the optimal number of states). We also show that, in our setting, the performance of the POMDP policy is comparable with the QMDP policy.

Committee members:
Dr. S. KAPODISTRIA
Dr. J.W. PORTEGIES
Dr. M.A.A. BOON

Keywords:
Condition-based maintenance, Hidden Markov models, Partially observable Markov decision processes, Markov decision processes, Monitoring data, Maintenance policies, Degradation process simulation

Contents

Nomenclature **ix**

1 Introduction **1**
1.1 Motivation for the study 1
1.2 Organization of the report 3

2 Background **5**
2.1 Related work 5
2.2 Setting 6
2.3 Simulations 9

3 Hidden Markov Models **15**
3.1 Hidden Markov models for component degradation 15
3.2 Literature overview of the EM algorithm and numerical optimization techniques 17
3.3 The Baum-Welch algorithm 18

4 Markov Decision Processes with Partial Information **25**
4.1 Markov Decision Processes 25
4.2 Partially Observable Markov Decision Processes 27

5 Results **39**
5.1 Results from the simulations 39
5.2 Discussion 46

6 Conclusions and outlook **49**

A Additional proofs **51**
A.1 The Banach Fixed-Point theorem 51
A.2 Convexity of the ‘do maintenance’ region 53

Bibliography **55**

Nomenclature

K	Number of features of the monitoring data - page 7
N	Number of cycles in the monitoring data/training data - page 7
n_j	Length of cycle j of the monitoring data/training data - page 7
j	Specific cycle of the monitoring data/training data - page 7
$\mathbf{t}$	Sequence of sampling times of the monitoring data/training data - page 7
$\mathbf{o}$	Sequence of observations of the monitoring data/training data - page 7
$o^{(j)}$	Observations sequence of cycle j of the monitoring data/training data - page 7
$t^{(j)}$	Sampling time sequence of cycle j of the monitoring data/training data - page 7
$\mathcal{S}_X$	State space of the degradation process - page 8
$\mathcal{S}_O$	State space of the observation process - page 8
$\mathcal{F}_O$	Observation that uniquely defines the failure state - page 8
$\mathcal{T}$	Discrete set of decision epochs - page 8
$\mathcal{A}$	Discrete set of actions - page 8
c_{pm}	Cost of preventive maintenance - page 8
c_{cm}	Cost of corrective maintenance - page 8
π	Condition-based maintenance policy - page 8
X_t	Value of the degradation process at time t - page 8
O_t	Value of the observation process at time t - page 8
Z_t	State of Markov chain that describes the degradation process of the component at time t - page 15
T_Z	Transition matrix of the Markov chain that describes the degradation process of the component at time t - page 15
$\mathcal{S}_Z$	State space of the Markov chain that describes the degradation process of the component - page 15
ν	Initial state distribution of the Markov chain that describes the degradation process of the component - page 15
θ	Set of parameters of a HMM - page 15
ω	Conditional distribution functions of the observations of a HMM - page 15
$\mathbf{z}$	Makov chain state sequence of the monitoring data/training data - page 16

$z^{(j)}$	Markov chain state sequence of cycle j of the monitoring data/training data - page 16
$\mathcal{F}_Z$	State of the Markov chain that corresponds to a failed component (failure state/state $m+1$) - page 16
m	Number of non-failure states of the Markov chain that describes the degradation process of the component - page 16
T_M	Transition matrix of an MDP - page 26
R_Z	Immediate cost function - page 26
γ	Discount rate - page 26
π_M^*	Optimal policy of an MDP - page 26
v_M^π	Value function of an MDP with policy π - page 27
v_M^*	Optimal value function of an MDP - page 27
q_M^*	Optimal state-action function of an MDP - page 27
Ω	Probabilistic observation model of a POMDP - page 28
$\hat{\mathcal{S}}_O$	Discretized tate space of the observation process - page 28
v	Number of discretized observations per feature - page 29
$\mathtt{H}_t$	History available at decision epoch t in a POMDP framework - page 29
$b_t\left(z\right)$	Element of the belief state vector b_t at time t - page 29
$\mathcal{B}$	Belief space - page 29
$T_B\left(b,a,o\right)$	Belief state update: in belief b, we took action a and saw observation o - page 30
$v_n^\pi\left(b\right)$	Value function of an n-stage POMDP with policy π and initial belief b - page 31
R_B	The immediate cost function for POMDPs - page 31
$\mathcal{D}$	Dynamic programming operator of POMDPs - page 31
$\Omega_b^a\left(o\right)$	Probability observing o at time t, given we took action a at time $t-1$ in belief state b - page 31
v_n^*	Optimal value function of an n-stage POMDP - page 31
α_n^i	i-th α-vector of an n-stage POMDP - page 32
$\hat{\alpha}_n$	Set of all α-vectors of an n-stage POMDP - page 32
e_z	Zero vector, of appropriate size, with only at index z the element 1 - page 32
α_n	Parsimonious set of α-vectors of an n-stage POMDP - page 33
v^*	Optimal value function of an infinite-horizon POMDP - page 34
`backup`	Backup operator - page 34
$\tilde{\alpha}_n$	Set of α-vectors that describe the approximate optimal value function after n iterations - page 35
$\tilde{\mathcal{B}}$	Proxy belief space - page 35

$\tilde{\mathcal{D}}$	Randomized dynamic programming operator for POMDPs - page 35
π_{MLS}	Policy of the MLS heuristic - page 37
π_{QMDP}	Policy of the QMDP heuristic - page 37

Acronyms and abbreviations

AIC	Akaike information criterion
AICc	Corrected Akaike information criterion
CBM	Condition-based maintenance
EM	Expectation-maximization
HMM	Hidden Markov model
MDP	Markov decision process
MLS	Most likely state
PMF	Probability mass function
POMDP	Partially observable Markov decision process
QMDP	Q-values Markov decision process
RUL	Remaining useful lifetime
Exp	Exponential distribution
Unif	Uniform distribution

Chapter 1

Introduction

Maintenance costs can be a significant cost burden, as these costs can range between 15 to 60 percent of the cost of goods produced [29]. Several different maintenance programs exist, such as corrective maintenance, which is the most expensive [29]. This approach, where a component is replaced upon failure, was common before the 1960s, but has been gradually complemented by an approach where maintenance is performed before a failure occurs [19], known as preventive maintenance. However, performing unnecessary and excessive preventive maintenance is costly in its own right, as it is believed that one-third of the money spent on maintenance in the United States was unnecessary/wasted [49]. Due to rapid technological development, the cost of preventive maintenance has increased to the point that more efficient solutions are required, such as condition-based maintenance (CBM), where maintenance actions are recommended based on collected information [21]. In this study, we propose a multivariate Gaussian hidden Markov model for condition-based maintenance policies.

1.1 Motivation for the study

A hidden Markov model (HMM) is a bivariate discrete-time stochastic process, that consists of an observable process and an unobservable Markov chain. This unobservable Markov chain can be seen as the latent component state process and could serve as a basis for a health index, which makes HMMs quite suitable for industrial applications [9]. In particular because, in practice, the latent component state is often not continuous but jumps between different states [52].

Initially, HMMs were used in CBM due to their success in speech processing and the parallel between speech processing problems and CBM [6]. Also their ease of interpretation in comparison to black-box models (such as artificial neural networks) [1], could have increased their attractiveness.

Another advantage is the existence of efficient algorithms, such as the Baum-Welch algorithm, to estimate the parameters of a HMM (given some training data). In particular, due to the flexibility of the Baum-Welch algorithm, which is an instance of the more general Expectation-maximization (EM) algorithm, one can incorporate specific characteristics of the degradation process. For example, the degradation of a component is typically subject to certain constraints (such as monotone degradation), see e.g. [52], and the Baum-Welch algorithm can easily be adjusted to account for such behaviour, which leads to more robust estimation.

Our choice of multivariate Gaussian HMMs comes from the fact that usually multiple, strongly correlated, sensors are used to monitor a component's condition, where each sensor contains only partial information of component state [52]. In such a setting, assuming the observations to be samples from a multivariate Gaussian distribution (whose parameters depend on the state of the component), could be reasonable. In addition, the parameters of the multivariate Gaussian distributions can efficiently be estimated using the Baum-Welch algorithm.

As a result, there have been numerous studies that use HMMs for diagnostics (such as fault detection) and prognostics (such as predicting remaining useful lifetime, also known as RUL), see e.g. [1], [17] and [52].

HMMs can also be used as a basis for cost-sensitive decision-making under uncertainty, by including actions and costs. The resulting problem can be modelled as a partially observable Markov decision process (POMDP), which could be solved to obtain a cost-optimal policy.

Despite their strong mathematical foundation and suitability to solve decision-making under uncertainty problems, the use of POMDPs for maintenance problems is, possibly due to the limitation that solving a POMDP to optimality is only possible for small problems, not widely recognized [33]. However, so-called *point-based value iteration* algorithms (which can be used to approximate the solution of a POMDP), have shown promising results in maintenance applications (see [34]).

Our goal is to investigate whether such a method can produce near-optimal maintenance policies. To this end, we estimate the parameters of a HMM and formulate a POMDP, based on a discretization of the observation space, which we solve, based on the total expected discounted cost, with a randomized point-based value iteration algorithm.

Instead of using field data, we rely on simulated degradation processes. The advantage of this is threefold:

1) We have full control over the signal-to-noise ratio;

2) We bypass the often time consuming and non-trivial task of data processing (i.e. data cleaning and data analysis);

3) We have access to a statistically identical environment in which we can test the effectiveness of the method.

The latter point shows where our study complements existing research. Indeed, since most papers study the use of POMDP maintenance models using field data, it is difficult to properly assess the quality of the resulting policy. Because our data is simulated, we can use the simulation to construct an environment in which the resulting POMDP policy can be tested and compared to, for example, the optimal threshold policy. This also allows us to investigate the performance of POMDP-based heuristics and see whether the computational burden of (approximately) solving a POMDP leads to significantly better policies. Lastly, we address the well-known issue of choosing the optimal number of states of the HMM (see e.g. [1]).

Our research questions read as follows:

Main question:

Can multivariate Gaussian HMM be used to construct a POMDP policy that is close to the optimal threshold policy, in terms of average cost?

Sub-question 1:

How do POMDP-based heuristics compare to the POMDP policy in terms of average cost?

Sub-question 2:
How does the average cost of POMDP-based policies depend on the number of states of the HMM?

1.2 Organization of the report

The remainder of this report is structured as follows. In Chapter 2 we provide some background material; we discuss several related papers, mathematically describe our setting/assumptions and introduce the five simulations that we consider. Subsequently, we formally introduce HMMs in Chapter 3 and illustrate how we use the Baum-Welch algorithm to estimate the parameters of the HMM, whilst accounting for our assumptions. In Chapter 4, we introduce the POMDP and discuss the approximate value iteration algorithm that we use to solve the POMDP. We also introduce two POMDP-based heuristics. In Chapter 5 we present and discuss the results of our study. Lastly, in Chapter 6 we present our conclusions, state some limitations of our study and suggest several directions for future research.

Chapter 2

Background

2.1 Related work

2.1.1 A collection of HMMs

In [1], HMMs are used both for diagnostics and prognostics. It is explained how HMMs can be used to classify the state of a component by training a collection of HMMs (one for every state – say, one for the 'good' state, one for the 'minor defects' state and one for the 'maintenance required' state). Subsequently, upon receiving monitoring data from a component whose state is unknown, the log-likelihood of each HMM can be calculated and the most likely state of the component can be identified. However, this approach requires the training data to be labelled.

The issue of selecting a suitable number of component states is also discussed. The authors present two options to solve this problem: (1) use domain-knowledge to determine a reasonable number of component states; (2) use cross-validation methods to derive the optimal number of component states.

The authors also provide a method for predicting the RUL using the estimated collection of HMMs. Specifically, say we have monitoring data of N cycles in total and the number of component states is m. This leads to a total of N vectors with each $m - 1$ transition times. The fundamental idea is to assume that these vectors follow some multivariate distribution (say a multivariate Gaussian distribution). By subsequently estimating the parameters of this distribution, the transition times can be estimated. In particular, using the conditional distribution one can construct estimates and confidence intervals for future transition times. For example, say we know when the component transitioned from the 'good' state to the 'minor defects' state, using the conditional distribution we can estimate the time the component enters the 'maintenance required' state.

2.1.2 Estimating the component state using unlabelled monitoring data

In [17], HMMs are used to estimate the current and future component state. To this end, the authors use training data (i.e. monitoring data of multiple completed cycles) to estimate the parameters of the HMM. Specifically, for each cycle in the training data, each (continuous) feature is split into m categories, using uniform discretization. The component states are defined to be exactly these m categories.

The authors study a gradual degradation process and therefore assume that the component can only transition to the right-neighbouring state (i.e. the component can either remain in state i, or transition to state $i + 1$). The optimal number of states of the HMM is determined by investigating the mean squared error of the component states using cross-validation.

The monitoring data of a new, uncompleted, cycle is multidimensional, continuous and unlabelled. The authors assume that this monitoring data is a sequence of multivariate Gaussian random variables, whose parameters depend on the component state, and use the labelled training data to estimate these parameters. This allows for estimating the state of the component, based on a new stream of monitoring data. Using the temporal model (i.e. the estimated Markov chain) and a current estimate of the component state, this method is extended to also predict the future component state.

2.1.3 Adjusting the Baum-Welch algorithm

In [52], the use of HMMs for RUL prediction and cost-sensitive decision-making (using a POMDP) is investigated. The authors assume a 'left-to-right' transition matrix (i.e. an upper triangular transition matrix), where the failure state is the only absorbing state in the system. The initial state and the failure state are assumed to be directly observable. The monitoring data is assumed to be generated by a multivariate Gaussian distribution, whose parameters depend on the component state. Furthermore, the monitoring data is assumed to be monotone.

The parameters of the HMM are estimated using a modification of the commonly used Baum-Welch algorithm. However, this modification largely happens after performing an iteration of the traditional Baum-Welch algorithm; at each iteration the parameters of the HMM are estimated and subsequently the left-to-right structure of the transition matrix is enforced by a 'projection-step' (where the estimated transition matrix is mapped to an upper triangular transition matrix) and the states are ordered based on the estimated means of the multivariate Gaussian distribution (using majority vote), referred to as the 'sorting-step'.

The RUL prediction is performed using the expected absorption time for each state of the Markov chain (i.e. each component state). The RUL prediction for an arbitrary distribution of component states (e.g. at time t the component state is 'good' with probability 0.7 and 'minor defects' with probability 0.3), is obtained by conditioning on the component state.

It follows from [52] that HMMs can indeed be used to identify the system state, specifically when the estimation procedure is made more robust by adjusting the Baum-Welch algorithm based on (reasonable) assumption about the degradation process. However, determining the optimal number of states is not trivial. The authors note that including more states leads to a finer granularity of the component state space. However, including more states may also cause different states to overlap and become indistinguishable. For RUL prediction, the authors note that including more states leads to better estimates.

2.2 Setting

2.2.1 Condition-based maintenance and monitoring data

In [21] condition-based maintenance is described as the following 3-step procedure:

(i) Collecting data;

(ii) Data processing;

(iii) Maintenance decision-making.

Collected data can be split into two categories; event data (which describes what happened, e.g. 'the oil was changed') and monitoring data (which contains the obtained measurements from the component). There are three different categories of monitoring data; value data (e.g. temperature or pressure), waveform data (such as a vibration signal) and multidimensional data (for example images). Once the data is collected, data processing is used for cleaning of the data (to account for errors in the data) and data analysis (e.g. feature extraction). The last step is maintenance decision-making, in which a maintenance policy is optimized according to a certain criteria (such as cost).

In this study, we collect data using a simulation of a degradation process. Event data is not included, although it should be mentioned that, in practice, event data is thought to be equally important as monitoring data for condition-based maintenance [21]. Furthermore, we restrict ourselves to only analysing value data. As a result, there is no need for feature extraction.

The (simulated) monitoring data contains noisy observations of K features that are related to the condition of the component. This can be interpreted as monitoring data gathered by K correlated imperfect sensors. The monitoring data contains a total of N perfect-to-failure cycles, where the first sampling of a cycle occurred immediately after the installation of the component (when the component is in a perfect condition) and the last sampling of a cycle occurred immediately after the component has failed. Each cycle is statistically identical.

The monitoring data of cycle $j \in \{1, 2, ..., N\}$, can be seen as a multidimensional time series of length n_j, containing both the observations $o^{(j)}$ and the corresponding sampling times $t^{(j)}$, where

$$\begin{aligned} o^{(j)} &:= \left(o_0^{(j)}, ..., o_{n_j}^{(j)}\right) \\ t^{(j)} &:= \left(t_0^{(j)}, ..., t_{n_j}^{(j)}\right). \end{aligned} \tag{2.1}$$

We emphasize that $0 = t_0^{(j)} \leq t_1^{(j)} \leq ... \leq t_{n_j}^{(j)}$ and that $t_{n_j}^{(j)}$ corresponds to the exact moment of failure.

The entire monitoring data can be described by the tuple $(\mathbf{o}, \mathbf{t})$, where $\mathbf{o}$ denotes observation sequences and $\mathbf{t}$ denotes the sampling time sequences, i.e.

$$\mathbf{o} := \left(o^{(1)}, ..., o^{(N)}\right) \text{ and } \mathbf{t} := \left(t^{(1)}, ..., t^{(N)}\right). \tag{2.2}$$

2.2.2 Problem statement and assumptions

The monitoring data $(\mathbf{o}, \mathbf{t})$ is generated by simulating N cycles of a scenario. Such a cycle/scenario can be described by four elements:

1. **The sampling regime**: The sampling regime denotes the distribution of the inter-sampling time, i.e. the time between samplings. We emphasize that regardless of the sampling regime, the failures are 'self-announcing'; consider sampling times $(t_0, ..., t_n)$, then $t_0 = 0$ denotes the first sampling (when the

component is in a perfect condition) and t_n denotes the sampling that occurs at the exact moment of failure.

2. **The degradation process:** The degradation process $\{X_t\ ;\ t \geq 0\}$ is the monotonically increasing stochastic process that describes the degradation of the component and induces failures, where $X_t \in \mathcal{S}_X$ and $\mathcal{S}_X := \mathbb{R}$ denotes the degradation space.

3. **The observation process:** The observation process $\{O_t\ ;\ t = t_0, ..., t_n\}$ is the stochastic process that generates the monitoring data, based on the degradation process. That is, it corresponds to noisy and/or partial measurements of the degradation process at sampling times. We emphasize that $O_t \in \mathcal{S}_O$ for $t = t_0, ..., t_n$, where $\mathcal{S}_O$ denotes the observation space, and $O_{t_n} := \mathcal{F}_O$, where $\mathcal{F}_O$ denotes the observation that uniquely characterizes a failed component.

4. **The failure mechanism:** The failure mechanism is described as the first passage time of the degradation process to the failure threshold $\xi \in \mathcal{S}_X$.

These four elements of a cycle/simulation can be used to construct a statistically identical environment in which a maintenance policy can be tested. This means that we extend the simulation to include actions and costs. This leads to a discrete set of decision epochs $\mathcal{T}$, according to the sampling regime, where at each decision epoch $t \in \mathcal{T}$ an action from the action space $\mathcal{A}$ can be chosen. We define this action space $\mathcal{A}$ as follows:

$$\mathcal{A} := \{\text{'do nothing'}, \text{'do maintenance'}\}, \tag{2.3}$$

denoted by 0 and 1 respectively.

In case action 0 is chosen, the cycle continues and the component may fail before the next decision epoch. If the component failed, it is immediately correctively replaced with a new, perfect, component, at cost c_{cm}. In case action 1 is chosen, the component is immediately preventively replaced with a perfect component, at cost c_{pm}. Performing maintenance (be it preventively or correctively) immediately ends the current cycle. This implicitly assumes that the repair time is zero, or negligible in comparison to the inter-sampling time.

The choice between doing nothing (action 0) and replacing the component (action 1) is made by a policy π. This policy is map between a space $\mathcal{B}$ and the action space $\mathcal{A}$. This space $\mathcal{B}$ is the space on which the POMDP and the POMDP-based heuristics operate.

For a policy π, the expected duration of a cycle is finite and can be written as

$$\mathbb{E}^{\pi}\left[CL\right] := \mathbb{E}^{\pi}\left[\min\left\{T_{\text{pm}}, T_{\text{fail}}\right\}\right], \tag{2.4}$$

where T_{pm} denotes the time until preventive maintenance is performed and T_{fail} denotes the time until the component fails.

The expected cost per cycle of a policy π can simply be written as

$$\mathbb{E}^{\pi}\left[CC\right] := c_{\text{pm}}\mathbb{P}^{\pi}\left(T_{\text{pm}} < T_{\text{fail}}\right) + c_{\text{cm}}\mathbb{P}^{\pi}\left(T_{\text{pm}} \geq T_{\text{fail}}\right). \tag{2.5}$$

The average cost (per time unit) of a policy π is given by $\mathbb{E}^{\pi}\left[CL\right] / \mathbb{E}^{\pi}\left[CC\right]$.

2.2.3 Assumptions

Our assumptions can be summarized as follows:

(i) We assume that the degradation process is monotonically increasing and that the average duration of a cycle is finite;

(ii) We assume that the Markov chain of the component state has an upper-triangular transition matrix;

(iii) We assume that the component is always in a perfect condition at the start of a cycle;

(iv) We assume that the failures are obvious, i.e. the failure state is uniquely identified by a single observation;

(v) We assume that the failures are self-announcing, i.e. once the component fails, the cycle immediately ends;

(vi) We assume that the repair time is zero, or negligible in comparison to the inter-sampling time, i.e. once a cycle has ended, the next cycle immediately starts;

(vii) We assume that the error induced by the observation process is normally distributed with mean zero.

2.3 Simulations

We study a total of five scenarios with two different types of degradation: Markovian and autoregressive. The Markovian degradation is simulated using a compound Poisson process and the autoregressive degradation is simulated using the running maximum of an autoregressive model with linear trend.

The compound Poisson process can be used to model damage to a component that has accumulated over time due to random shocks (see e.g. [47]); the exponentially distributed inter-arrival time of shocks, with mean $1/\lambda$, represents the temporal randomness of shocks and the stochastic size of each shock can be described by an arbitrary distribution function F. The resulting compound Poisson process $\{X(t), t \geq 0\}$ can by described as the tuple $\texttt{CP}(\lambda, F)$ and is given by

$$X(t) := \sum_{i=1}^{A(t)} Y_i, \tag{2.6}$$

where $A(t)$ denotes the number of shocks on the interval $[0, t]$ and Y_i denotes the shock sizes (with $Y_i \overset{iid}{\sim} F$).

Since Markovian degradation is not always realistic (see e.g. [35]), we also include an autoregressive part. To account for both the long-term behaviour and the short-term fluctuations, we combine a deterministic linear trend model with an autoregressive model (of order p). The resulting process $\{Q_t \, ; \, t \in \mathcal{T}\}$ is defined as

$$Q_{t_i} := t_i + \phi_1 Q_{t_{i-1}} + \phi_2 Q_{t_{i-2}} + ... + \phi_p Q_{t_{i-p}} + \epsilon_{t_i}^{\phi}, \tag{2.7}$$

where $t_i \in \mathcal{T}, \phi := (\phi_1, ..., \phi_p)$ and $\epsilon_{t_i}^{\phi} \sim \mathcal{N}\left(0, \sigma_{\phi}^2\right)$.

The corresponding degradation process $\{X_t \,;\, t \geq 0\}$ is defined as the running maximum of Equation (2.7). Such a degradation process is referred to as $\texttt{AR}\left(\phi, \sigma_\phi^2\right)$.

Furthermore, it is common that the degradation process of a component can be divided into different 'health stages' (see e.g. [26]). For that reason, we also study a so-called *delay model*, where instead of gradual degradation over time, the degradation process can be split into two parts: a healthy stage and an unhealthy stage. During the healthy stage, the components works perfectly, however, at one point, the unhealthy stage is entered and the component begins to deteriorate. Essentially, the degradation process is zero until time T, where T is a random variable described by an arbitrary distribution function such that $\mathbb{E}\left[T\right] < \infty$.

2.3.1 Simulation 1: Noisy Poisson process

1. The inter-sampling time is always 1, except in case of failure;

$$\mathcal{T} := \{0, 1, ..., \lfloor t_n \rfloor - 1, \lfloor t_n \rfloor, t_n\}, \tag{2.8}$$

   where t_n denotes the arrival time of the shock that induces the failure.

2. The degradation process $\{X_t; t \geq 0\}$ is a Poisson process (i.e. a compound Poisson process where each shock has size 1), with arrival rate 1.

3. The observation process $\{O_t; t \in \mathcal{T}\}$ corresponds to noisy measurements of the degradation process, i.e.

$$O_t := X_t + \epsilon_t \quad \forall t \in \mathcal{T}, \tag{2.9}$$

   where $\epsilon_t \overset{iid}{\sim} \mathcal{N}\left(0, \sigma_\epsilon^2 = 0.2\right)$.

4. The failure mechanism is the first passage time of the degradation process to the failure threshold $\xi = 10$, i.e.

$$t_n := \inf_{t \geq 0} \{X_t \geq 10\}. \tag{2.10}$$

2.3.2 Simulation 2: Autoregressive degradation

1. The inter-sampling time is always 1;

$$\mathcal{T} := \{0, 1, 2, ..., n-1, n\}. \tag{2.11}$$

2. The degradation process $\{X_t; t \geq 0\}$ is described by

$$\texttt{AR}\left(\phi = (0.35, 0.35, 0.125, 0.125), \sigma_\phi^2 = 1\right). \tag{2.12}$$

3. The observation process $\{O_t; t \in \mathcal{T}\}$ corresponds to noisy measurements of the degradation process, i.e.

$$O_t := X_t + \epsilon_t \quad \forall t \in \mathcal{T}, \tag{2.13}$$

   where $\epsilon_t \overset{iid}{\sim} \mathcal{N}\left(0, \sigma_\epsilon^2 = 0.5\right)$.

4. The failure mechanism is the first passage time of the degradation process to the failure threshold $\xi = 10$, i.e.

$$t_n := \inf_{t \geq 0} \{X_t \geq 10\}. \tag{2.14}$$

2.3.3 Simulation setup 3: Memoryless sampling times

1. The inter-sampling time is exponentially distributed with mean 1 (except in case of failure);

$$\mathcal{T} := \{t_0, t_1, ..., t_n\}, \tag{2.15}$$

where $t_i - t_{i-1} \sim \exp(1)$ for $i = 1, ..., n-1$ and t_n denotes the exact moment of failure.

2. The degradation process $\{X_t; t \geq 0\}$ is defined as

$$X_t := 0.3X_t^{(1)} + 0.7X_t^{(2)}, \tag{2.16}$$

where $X_t^{(1)} \sim \texttt{AR}\left(\phi = (0.6, 0.2, 0.1), \sigma_\phi^2 = 1\right)$
and $X_t^{(2)} \sim \texttt{CP}\,(\lambda = 1, F \sim \text{unif}\,(0,1))$.

3. The observation process $\{O_t; t \in \mathcal{T}\}$ corresponds to noisy measurements of the Markovian part of the degradation process, i.e.

$$O_t := X_t^{(2)} + \epsilon_t \quad \forall t \in \mathcal{T}, \tag{2.17}$$

where $\epsilon_t \overset{iid}{\sim} \mathcal{N}\left(0, \sigma_\epsilon^2 = 0.2\right)$.

4. The failure mechanism is the first passage time of the degradation process to the failure threshold $\xi = 10$, i.e.

$$t_n := \inf_{t \geq 0} \{X_t \geq 10\}. \tag{2.18}$$

2.3.4 Simulation setup 4: Two health stages

1. The inter-sampling time is exponentially distributed with mean 1 (except in case of failure);

$$\mathcal{T} := \{t_0, t_1, ..., t_n\}, \tag{2.19}$$

where $t_i - t_{i-1} \sim \exp(1)$ for $i = 1, ..., n-1$ and t_n denotes the exact moment of failure.

2. The degradation process $\{X_t; t \geq 0\}$ is defined as

$$X_t := 0.5X_t^{(1)} + 0.25X_t^{(2)} + 0.25X_t^{(3)}, \tag{2.20}$$

where $X_t^{(1)} \sim \texttt{AR}\left(\phi = (0.6, 0.2, 0.1), \sigma_\phi^2 = 1\right)$,
$X_t^{(2)} \sim \texttt{CP}\,(\lambda = 1, F \sim \exp(1))$,
$X_t^{(3)} \sim \texttt{CP}\,(\lambda = 2, F \sim \exp(2))$ and
the deterioration start at time $T \sim \exp(5)$.

3. The observation process $\{O_t; t \in \mathcal{T}\}$ corresponds to noisy measurements of the autoregressive part of the degradation process, i.e.

$$O_t := X_t^{(1)} + \epsilon_t \quad \forall t \in \mathcal{T}, \tag{2.21}$$

where $\epsilon_t \overset{iid}{\sim} \mathcal{N}\left(0, \sigma_\epsilon^2 = 1\right)$.

4. The failure mechanism is the first passage time of the degradation process to the failure threshold $\xi = 15$, i.e.

$$t_n := \inf_{t \geq 0} \{X_t \geq 15\}. \tag{2.22}$$

2.3.5 Simulation setup 5: Correlated sensors

1. The inter-sampling time is exponentially distributed with mean 1 (except in case of failure);

$$\mathcal{T} := \{t_0, t_1, ..., t_n\}, \tag{2.23}$$

where $t_i - t_{i-1} \sim \exp(1)$ for $i = 1, ..., n-1$ and t_n denotes the exact moment of failure.

2. The degradation process $\{X_t; t \geq 0\}$ is defined as

$$X_t := 0.3X_t^{(2)} + 0.7X_t^{(3)} + 0.3X_t^{(4)} + 0.7X_t^{(1)}, \tag{2.24}$$

where $X_t^{(1)} \sim \texttt{AR}\left(\phi = (0.4, 0.3, 0.2), \sigma_\phi^2 = 1\right)$,
$X_2^{(2)} \sim \texttt{AR}\left(\phi = (0.6, 0.2, 0.1), \sigma_\phi^2 = 1\right)$,
$X_t^{(3)} \sim \texttt{CP}(\lambda = 1, F \sim \exp(1))$,
$X_t^{(4)} \sim \texttt{CP}(\lambda = 2, F \sim \exp(2))$ and
the deterioration start at time $T \sim \mathcal{N}(5, 1)$.

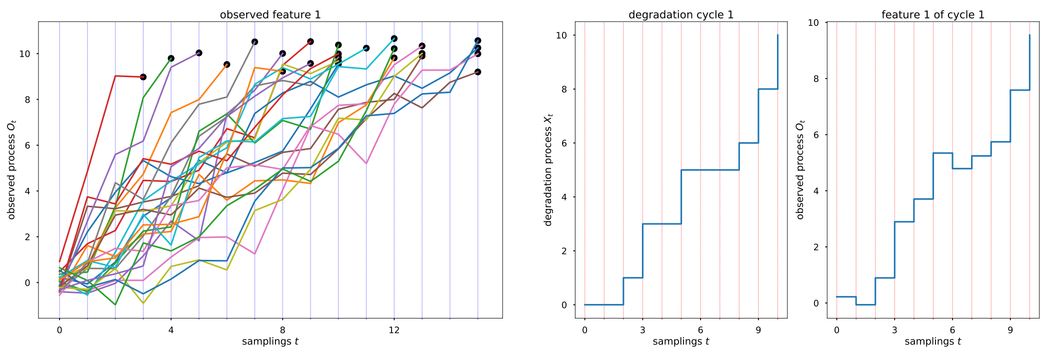
3. The observation process $\{O_t; t \in \mathcal{T}\}$ corresponds to noisy measurements of 2 parts of the degradation process:

$$O_t := \left(X_t^{(1)} + \epsilon_t^{(1)}, X_t^{(3)} + \epsilon_t^{(3)}\right) \quad \forall t \in \mathcal{T}, \tag{2.25}$$

where $\epsilon_t^{(1)}, \epsilon_t^{(3)} \overset{iid}{\sim} \mathcal{N}\left(0, \sigma_\epsilon^2 = 1\right)$.

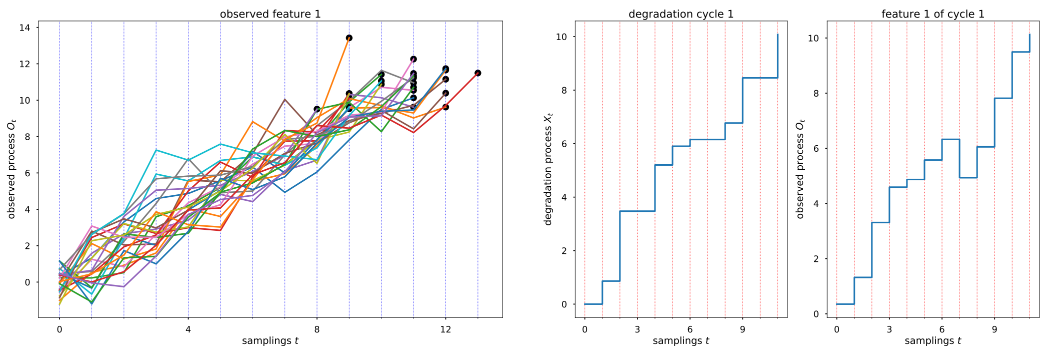
4. The failure mechanism is the first passage time of the degradation process to the failure threshold $\xi = 30$, i.e.

$$t_n := \inf_{t \geq 0} \{X_t \geq 30\}. \tag{2.26}$$



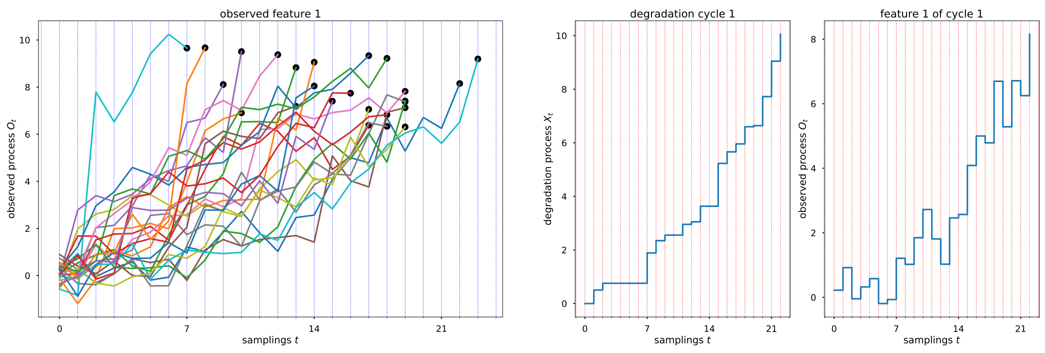
(A) (B)

FIGURE 2.1: (A) The monitoring data **o** used for Simulation 1; (B) Sample path of the degradation process and the observation process for the first cycle.



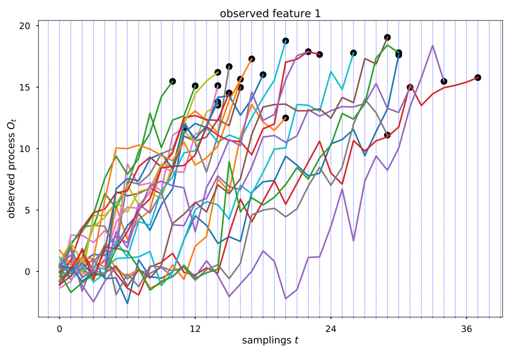
(A) (B)

FIGURE 2.2: (A) The monitoring data **o** used for Simulation 2; (B) Sample path of the degradation process and the observation process for the first cycle.

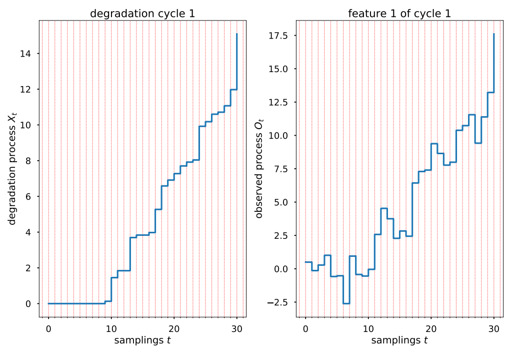


(A) (B)

FIGURE 2.3: (A) The monitoring data **o** used for Simulation 3; (B) Sample path of the degradation process and the observation process for the first cycle.

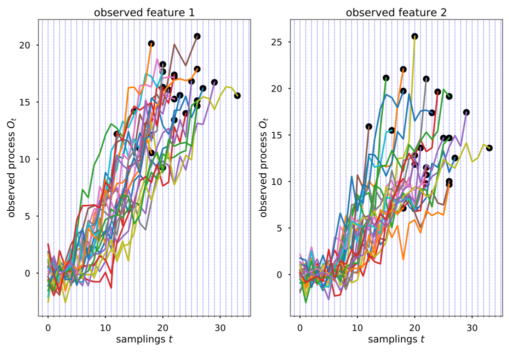


(A)

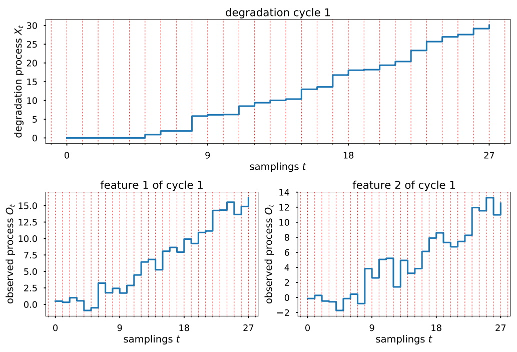


(B)

FIGURE 2.4: (A) The monitoring data **o** used for Simulation 4; (B) Sample path of the degradation process and the observation process for the first cycle.



(A)



(B)

FIGURE 2.5: (A) The monitoring data **o** used for Simulation 5; (B) Sample path of the degradation process and the observation process for the first cycle.

Chapter 3

Hidden Markov Models

3.1 Hidden Markov models for component degradation

In this section we mathematically define hidden Markov models (HMMs). We assume that the degradation process of the component can be described by a Markov chain and that the monitoring data is generated by an observed process, governed by this Markov chain.

A hidden Markov model (HMM) is a bivariate discrete-time stochastic process that consists of an observable process and an unobservable Markov chain. Although the idea of a latent Markov chain was already present in the Gilbert-Elliott channel model (see [18] and [16]), the general idea of a hidden Markov model was proposed in [2] and subsequently further developed in a series of papers by Baum and various co-authors. The core assumption is the existence of a latent (or *hidden*) Markov chain which governs the distribution of the observations (or *emissions*). Conform [8], we use the following intuitive definition of a hidden Markov model:

Definition 1 (Hidden Markov model)**.** *A hidden Markov model (HMM) is a bivariate stochastic process $\{O_t, Z_t \,;\, t \in \mathcal{T}_H\}$ where $\{O_t \,;\, t \in \mathcal{T}_H\}$ is an observed process and $\{Z_t \,;\, t \in \mathcal{T}_H\}$ an unobserved Markov chain, both defined on the discrete time domain $\mathcal{T}_H := \{0, 1, ...\}$.*

The unobserved Markov chain is defined on the finite state space $\mathcal{S}_Z$ and its transition matrix is denoted by T_Z, with elements $T_Z(z, z') := \mathbb{P}(Z_1 = z |\, Z_0 = z)$ for $z, z' \in \mathcal{S}_Z$.

The initial state distribution of the unobserved Markov chain is denoted by ν, where $\nu(z) := \mathbb{P}(Z_0 = z)$ for $z \in \mathcal{S}_Z$.

The observed process is defined on the observation space $\mathcal{S}_O$ and for the distribution function of observation O_t, it holds that

$$\begin{aligned} &F_{O_t}(o_t | O_0 = o_0, ..., O_{t-1} = o_{t-1}, Z_0 = z_0, ..., Z_{t-1} = z_{t-1}, Z_t = z_t) \\ &= F_{O_t}(o_t | Z_t = z_t) \quad \forall \left[o_\cdot \in \mathcal{S}_O \wedge z_\cdot \in \mathcal{S}_Z\right]. \end{aligned} \tag{3.1}$$

We characterize the parameters of a HMM by the set $\theta := \{T_Z, \omega, \nu\}$, where ω denotes the conditional distribution functions of the observations, as presented in Equation (3.1), i.e. $\omega := \{F_{O_t}(o | Z_t = z) \,;\, o \in \mathcal{S}_O, z \in \mathcal{S}_Z\}$.

Consider the monitoring data of the j-th cycle $o^{(j)}$. We assume that this time-series, defined on the discrete time domain $t^{(j)}$, is a realization of the observed process of a HMM with parameters θ. This means that we assume that the underlying Markov chain describes the degradation process of the component.

Specifically, for cycle $j \in \{1, 2, ..., N\}$, we have

$$\begin{aligned} o^{(j)} &= \left(o_0^{(j)}, ..., o_{n_j}^{(j)}\right) \\ t^{(j)} &= \left(t_0^{(j)}, ..., t_{n_j}^{(j)}\right) \equiv \left(0, ..., n_j\right). \end{aligned} \tag{3.2}$$

Whereas each observation before failure is a K-dimensional numerical value, the observation at time n_j, uniquely characterizes the 'failure state', i.e.

$$\begin{aligned} o_t^{(j)} &\in \mathbb{R}^K \text{ for } t \in \{0, 1, ..., n_j - 1\} \\ o_{n_j}^{(j)} &:= \mathcal{F}_O. \end{aligned} \tag{3.3}$$

Correspondingly, the observation space $\mathcal{S}_O$ is defined as $\mathcal{S}_O := \mathbb{R}^K \cup \{\mathcal{F}_O\}$.

The Markov chain state space is given by $\mathcal{S}_Z := \{1, ..., m\} \cup \{\mathcal{F}_Z\}$. This means that we divide the degradation of the component into $m+1$ states, where 1 corresponds to a perfect condition and $\mathcal{F}_Z$ ($\equiv m+1$) is the failure state. We restrict ourselves to a setting where the number of states is an input parameter, i.e. m is given beforehand.

The Markov chain state sequence of cycle j is denoted by

$$z^{(j)} := \left(z_0^{(j)}, ..., z_{n_j}^{(j)}\right), \tag{3.4}$$

where $z_0^{(j)} = 1$, $z_{n_j}^{(j)} := \mathcal{F}_Z$ and $z_t^{(j)} \in \{1, ..., m\}$ for $t \in \{0, 1, ..., n_j - 1\}$. Note that the initial state distribution is therefore known a-priori and given by $\nu(z) = \mathbb{1}\{z = 1\}$.

Additionally, since we assumed that the condition of the component does not improve by itself, the state of the Markov chain is non-decreasing over time, i.e.

$$z_{t+1}^{(j)} \geq z_t^{(j)} \text{ for } t \in \{0, 1, ..., n_j - 2\}. \tag{3.5}$$

The entire monitoring data and Markov chains state sequences are respectively given by

$$\mathbf{o} = \left(o^{(1)}, ..., o^{(N)}\right) \text{ and } \mathbf{z} := \left(z^{(1)}, ..., z^{(N)}\right). \tag{3.6}$$

We assume that the observations before failure are samples from a multivariate Gaussian distribution, whose parameters depend on the state of the Markov chain. Specifically, an observation $O_t \in \mathcal{S}_O$, given the current state of the underlying Markov chain (say $Z_t = z \neq \mathcal{F}_Z$), has the following density function:

$$f_{O_t|Z_t}\left(o_t | Z_t = z\right) = \frac{\exp\left(-\frac{1}{2}\left(o_t - \mu_z\right)^\top \Sigma_z^{-1}\left(o_t - \mu_z\right)\right)}{\sqrt{(2\pi)^K |\Sigma_z|}}, \tag{3.7}$$

where $\mu_z \in \mathbb{R}^K$ and $\Sigma_z \in \mathbb{R}^{(K \times K)}$ denote the mean vector and covariance matrix respectively, in other words $O_t | \left(Z_t = z\right) \sim \mathcal{N}\left(\mu_z, \Sigma_z\right)$.

Since the multivariate Gaussian distribution is uniquely defined by the mean vector and covariance matrix, we write

$$\omega \equiv \{\mu_z, \Sigma_z\}_{z=1}^m. \tag{3.8}$$

3.2 Literature overview of the EM algorithm and numerical optimization techniques

In this section we motivate our choice for using the EM algorithm/Baum-Welch algorithm for estimating the parameters of the HMM. We explain the advantages and disadvantages of this approach and briefly elaborate on other algorithms.

Ideally, one would determine

$$\underset{\theta \in \Theta}{\arg\max} \log f_{\mathbf{O}}^{N}(\mathbf{o}; \theta), \tag{3.9}$$

i.e. the parameters that maximize the likelihood of observing monitoring data $\mathbf{o}$.

However, due to the dependence of the observations on the states of the Markov chain, no analytical solution to the optimization problem described by Equation (3.9) is known [24]. In practice, iterative optimization techniques based on either the Expectation-Maximization algorithm (EM) or numerical optimization techniques (such as the gradient descent algorithm) are used to estimate the parameters of a HMM [24]. In both cases, multiple iterations are performed to maximize a likelihood function.

In this work, we use EM to estimate the parameters of the HMM. The EM algorithm is an iterative heuristic to find maximum-likelihood estimates for general incomplete-data problems, proposed in [14]. Incomplete data refers to the implied existence of two sample spaces ($\mathcal{S}_O$ and $\mathcal{S}_Z$), such that an observation $o \in \mathcal{S}_O$ could be generated by multiple states $z \in \mathcal{S}_Z$. The main idea is to translate an incomplete-data problem to a complete-data problem, whose likelihood has, for many statistical problems, a nice form [28].

The EM algorithm consists of two steps: the Expectation step (E-step) and the Maximization step (M-step). During the E-step, the conditional expectation of the log-likelihood of the complete-data problem, given the monitoring data and some initial/previous estimate of the model parameters, is constructed. Subsequently, this quantity is maximized (M-step) with respect to the model parameters. This procedure is repeated until convergence, resulting in a final estimate of the model parameters.

Estimating the parameters of a HMM based on the observations naturally fits the framework of an incomplete-data problem, as HMMs belong to a subcategory of incomplete data models (known as missing data models) [8]. In fact, several years before the publication of [14], the Baum-Welch algorithm was proposed [3], which is an instance of the more general EM algorithm, specifically for hidden Markov models.

The main advantages of using the Baum-Welch algorithm (or, equivalently, the EM algorithm) for estimating the parameters of the HMM are that

(1) the estimated transition matrix is always a stochastic matrix;

(2) it is numerically more robust against poor initialization values or when the model has a large number of parameters, in comparison to gradient-based techniques [24];

(3) the likelihood monotonically increases and that convergence to a stationary point is guaranateed (see [50], where this is shown for the general EM algorithm).

However, the Baum-Welch algorithm also has some weak points. For example, the convergence guarantee stated above only implies convergence to the global maximum of the sample likelihood if the initial parameter estimates are sufficiently close to the optimum. This means that the algorithm may end up in a local maximum or get stuck at a saddle point. Though [50] provides some theorems with regards to this, verifying the theorem's conditions can be difficult. It is therefore suggested to try different starting values. Additionally, convergence of the sample likelihood does not imply convergence of the parameters. It may for example be possible that the parameters oscillate between two local maxima with the same value.

Another downside of the EM algorithm is its slow convergence [28]. This is particularly the case when the complete data is much more informative than the incomplete data or when the algorithm is near a solution (in the latter case it converges linearly), see [12]. Finally, there is no obvious way of assessing the variability of the parameter estimates, since only point estimates are provided [45].

Standard numerical optimization techniques work with the objective function of the optimization problem described in Equation (3.9) (i.e. the incomplete likelihood), and its derivatives, directly [24]. However, obtaining the gradient (and possibly the Hessian) of the likelihood function generally requires heavy analytical preparatory work and the implementation of these methods may present numerical difficulties, especially when the number of parameters to be estimated is high [12]. For example, the Newton–Raphson method in practice breaks down because the Hessian often eventually becomes singular [45]. In [8], two important advantages of gradient-based methods are mentioned. First, they do not require an M-step, making these methods desirable in case no simple closed-form solution for the M-step can be found. Secondly, they converge faster, as they can reach quadratic convergence.

Our choice for the EM-algorithm/Baum-Welch algorithm is motivated by the availability of exact formulas for performing the M-step in case of multivariate Gaussian observations. Additionally, the convergence rate is not a major concern for us, as the computational bottleneck of our approach is solving the POMDP (see Chapter 4) and not estimating the parameters of the HMM.

3.3 The Baum-Welch algorithm

In this section we explain how, given monitoring data $\mathbf{o}$, the parameters θ of a HMM can be estimated using the Baum-Welch algorithm [3], which is an instance of the more general Expectation Maximization algorithm [14]. For a more fundamental discussion of the EM algorithm in the context of HMMs, the reader is referred to [32] and [8].

Consider the conditional log-likelihood of monitoring data $\mathbf{o}$ of an HMM with parameters θ and Markov chain state sequence $\mathbf{z}$

$$\log f^{N}_{\mathbf{O}|\mathbf{Z}}(\mathbf{o}|\,\mathbf{z};\theta) = \sum_{j=1}^{N} \log f^{(j)}_{\mathbf{O}|\mathbf{Z}}\left(o^{(j)}\middle|\, z^{(j)};\theta\right), \tag{3.10}$$

where $f^{(j)}_{\mathbf{O}|\mathbf{Z}}\left(o^{(j)}\middle|\, z^{(j)};\theta\right)$ denotes the conditional likelihood of the j-th cycle.

We introduce the set $\mathcal{S}^{*}_{Z} \ni \mathbf{z}$, which denotes the set of all possible N sample paths of the Markov chain, with length $n_1, ..., n_j$ respectively. Based on Equations (3.4) and (3.5), we define this set as

$$\mathcal{S}_Z^* := \left\{ \mathbf{z} : \left[z_0^{(j)} = 1 \wedge z_{n_j}^{(j)} = \mathcal{F}_Z \wedge \left[z_{t+1}^{(j)} \geq z_t^{(j)} \; \forall t \in \{0, 1, ..., n_j - 2\} \right] \right] \forall j \in \{1, ..., N\} \right\}. \tag{3.11}$$

The PMF $p_{\mathbf{Z}}^N(\mathbf{z})$ of $\mathbf{z} \in \mathcal{S}_Z^*$ can be written as

$$p_{\mathbf{Z}}^N(\mathbf{z}) = \prod_{j=1}^{N} p_{\mathbf{Z}}^{(j)}\left(z^{(j)}\right), \tag{3.12}$$

where $p_{\mathbf{Z}}^{(j)}\left(z^{(j)}\right)$ denotes the PMF of the j-th cycle.

The log-likelihood of the complete-data problem is simply given by

$$f_{\mathbf{O},\mathbf{Z}}^N(\mathbf{o}, \mathbf{z}; \theta) = \prod_{j=1}^{N} \left[f_{\mathbf{O}|\mathbf{Z}}^{(j)}\left(o^{(j)} \middle| z^{(j)}; \theta \right) \right] \prod_{j=1}^{N} \left[p_{\mathbf{Z}}^{(j)}\left(z^{(j)}\right) \right]. \tag{3.13}$$

The incomplete log-likelihood can, using the law of total probability, be written in terms of the complete log-likelihood. Additionally, by introducing a PMF $\tilde{p} : \mathcal{S}_Z^* \to (0, 1]$ and applying Jensen's inequality [22], a lower bound of the incomplete log-likelihood can be derived, specifically:

$$\begin{aligned} \log f_{\mathbf{O}}^N(o; \theta) &= \log \left[\sum_{\mathbf{z} \in \mathcal{S}_Z^*} \tilde{p}(\mathbf{z}) \frac{f_{\mathbf{O},\mathbf{Z}}^N(\mathbf{o}, \mathbf{z}; \theta)}{\tilde{p}(\mathbf{z})} \right] \\ &\geq \sum_{\mathbf{z} \in \mathcal{S}_Z^*} \tilde{p}(\mathbf{z}) \log \left[\frac{f_{\mathbf{O},\mathbf{Z}}^N(\mathbf{o}, \mathbf{z}; \theta)}{\tilde{p}(\mathbf{z})} \right] =: LB(\theta, \tilde{p}, \mathbf{o}). \end{aligned} \tag{3.14}$$

Note that we can apply Jensen's inequality since $\tilde{p}$ is positive for all $\mathbf{z} \in \mathcal{S}_Z^*$ and $x \mapsto \log(x)$ is concave on $\mathbb{R}^+$.

This lower bound LB is pivotal for the EM algorithm, as the EM algorithm basically iteratively maximizes this lower bound (given some initial estimate of the model parameters $\theta^{(0)}$).

The E-step maximizes LB with respect to $\tilde{p}$, given an initial/previous estimate of the model parameters. It can be shown that, at the u-iteration, this is done by choosing

$$\tilde{p}^{(u)}(\mathbf{z}) := p_{\mathbf{Z}|\mathbf{O}}^N\left(\mathbf{z} \middle| \mathbf{o}; \theta^{(u-1)} \right), \tag{3.15}$$

where $p_{\mathbf{Z}|\mathbf{O}}^N$ denotes the conditional PMF of Markov chain state sequences (given monitoring data $\mathbf{o}$ and model parameters $\theta^{(u-1)}$). See [32] for more details.

Subsequently, the M-step updates the model parameters, at the u-th iteration, by maximizing EL with respect to θ, for fixed $\tilde{p} = \tilde{p}^{(u)}$, i.e.

$$\theta^{(u)} = \arg\max_{\theta \in \Theta} LB\left(\theta, \tilde{p}^{(u)}, \mathbf{o}\right), \tag{3.16}$$

where Θ denotes the set of possible model parameters.

This iterative procedure is illustrated in Figure 3.1.

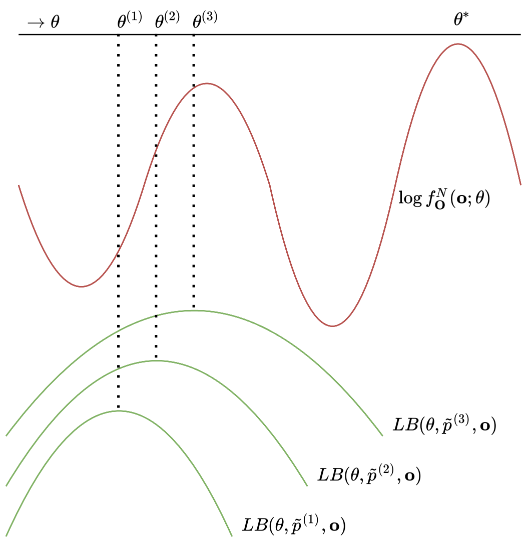

$\to \theta$
$\theta^{(1)}$
$\theta^{(2)}$
$\theta^{(3)}$
θ^*
$\log f_{\mathbf{O}}^N(\mathbf{o};\theta)$
$LB(\theta,\tilde{p}^{(3)},\mathbf{o})$
$LB(\theta,\tilde{p}^{(2)},\mathbf{o})$
$LB(\theta,\tilde{p}^{(1)},\mathbf{o})$

FIGURE 3.1: Illustration of the EM algorithm.

Essentially, this means that, at the u-iteration, one simply updates the model parameters by solving the following optimization problem:

$$\theta^{(u)} := \arg\max_{\theta\in\Theta} W\left(\theta, \theta^{(u-1)}\right), \tag{3.17}$$

where

$$W\left(\theta, \theta^{(u-1)}\right) := \sum_{\mathbf{z}\in\mathcal{S}_Z^*} f_{\mathbf{O},\mathbf{Z}}^N\left(\mathbf{o},\mathbf{z};\theta^{(u-1)}\right) \log\left[f_{\mathbf{O},\mathbf{Z}}^N\left(\mathbf{o},\mathbf{z};\theta\right)\right]. \tag{3.18}$$

Indeed, by dropping terms/factors that are irrelevant for the optimization, we find that

$$\begin{aligned} \theta^{(u)} &= \arg\max_{\theta\in\Theta} LB\left(\theta, \tilde{p}^{(u)}, o\right) \\ &= \arg\max_{\theta\in\Theta} \sum_{\mathbf{z}\in\mathcal{S}_Z^*} \tilde{p}^{(u)}(\mathbf{z}) \log\left[\frac{f_{\mathbf{O},\mathbf{Z}}^N(\mathbf{o},\mathbf{z};\theta)}{\tilde{p}^{(u)}(\mathbf{z})}\right] \\ &= \arg\max_{\theta\in\Theta} \sum_{\mathbf{z}\in\mathcal{S}_Z^*} p_{\mathbf{Z}|\mathbf{O}}^N\left(\mathbf{z}\,\middle|\,\mathbf{o};\theta^{(u-1)}\right) \log\left[f_{\mathbf{O},\mathbf{Z}}^N(\mathbf{o},\mathbf{z};\theta)\right] \\ &= \arg\max_{\theta\in\Theta} \sum_{\mathbf{z}\in\mathcal{S}_Z^*} f_{\mathbf{O},\mathbf{Z}}^N\left(\mathbf{o},\mathbf{z};\theta^{(u-1)}\right) \log\left[f_{\mathbf{O},\mathbf{Z}}^N(\mathbf{o},\mathbf{z};\theta)\right]. \end{aligned} \tag{3.19}$$

For multivariate Gaussian HMMs, Equation (3.13) can easily be expressed in terms of the model parameters θ. Indeed, using the Markov property, it follows that

$$p_{\mathbf{Z}}^{(j)}\left(z^{(j)};\theta\right) = \prod_{i=0}^{n_j-1} T_Z\left(z_i^{(j)}, z_{i+1}^{(j)}\right), \tag{3.20}$$

where T_Z denotes the transition matrix of an HMM with parameters θ.

Given the Markov chain state sequence, the joint density of the observations can simply be expressed as a product of the various densities, i.e.

$$f^{(j)}_{\mathbf{O}|\mathbf{Z}}\left(o^{(j)} \middle| z^{(j)};\theta\right) = \prod_{i=0}^{n_j-1} h^{\langle\omega\rangle}\left(o^{(j)}_i, z^{(j)}_i\right), \tag{3.21}$$

where

$$h^{\langle\omega\rangle}(o,z) := \frac{\exp\left(-\frac{1}{2}(o-\mu_z)^\top \Sigma_z^{-1}(o-\mu_z)\right)}{\sqrt{(2\pi)^K|\Sigma_z|}} \quad \forall\left[o \in \mathbb{R}^K \wedge z \in \{1,...,m\}\right]. \tag{3.22}$$

Since the initial state distribution is always given by $\nu(z) = \mathbb{1}\{z=1\}$, we only need to estimate the transition matrix and the parameters of the multivariate Gaussian distributions.

Consider previous/initial model parameters $\tilde{\theta} := \{\tilde{T}_Z, \tilde{\omega}\}$, where $\tilde{\omega} := \{\tilde{\mu}_z, \tilde{\Sigma}_z\}_{z=1}^m$ and let Θ_T and Θ_ω denote the set of possible transition matrices and parameters of the multivariate Gaussian distributions respectively. The optimization problem from Equation (3.17) can then, due to Equation (3.13), be written as two explicit optimization problems, that is

$$\theta^* := \{T_Z^*, \omega^*\} := \left\{\arg\max_{T_Z \in \Omega_T} \tilde{a}\left(T_Z, \tilde{\theta}\right), \arg\max_{\omega\in\Theta_\omega} \tilde{b}\left(\omega, \tilde{\theta}\right)\right\}, \tag{3.23}$$

where

$$\tilde{a}\left(T_Z, \tilde{\theta}\right) := \sum_{z\in\mathcal{S}_Z^*}\sum_{j=1}^N\sum_{i=1}^{n_j-1}\left[\log T_Z\left(z^{(j)}_i, z^{(j)}_{i+1}\right)\right] f^N_{\mathbf{O},\mathbf{Z}}\left(\mathbf{o},\mathbf{z};\tilde{\theta}\right), \tag{3.24}$$

and

$$\tilde{b}\left(\omega, \tilde{\theta}\right) := \sum_{z\in\mathcal{S}_Z^*}\sum_{j=1}^N\sum_{i=0}^{n_j-1}\left[\log h^{\langle\omega\rangle}\left(o^{(j)}_i, z^{(j)}_i\right)\right] f^N_{\mathbf{O},\mathbf{Z}}\left(\mathbf{o},\mathbf{z};\tilde{\theta}\right). \tag{3.25}$$

Lemma 1 (Update equations M-step, [5]). *The solution of the optimization problem described by Equation (3.23) is $\theta^* = \{T_Z^*, \omega^*\}$, where $\omega^* := \{\mu_z^*, \Sigma_z^*\}_{z=1}^m$, given by*

$$T_Z^*(z,z') = \frac{\sum_{j=1}^N\left[\sum_{t=0}^{n_j-1}\mathbb{P}\left(Z^{(j)}_t = z, Z^{(j)}_{t+1} = z' \middle| \mathbf{o};\tilde{\theta}\right)\right]}{\sum_{j=1}^N\left[\sum_{t=0}^{n_j-1}\mathbb{P}\left(Z^{(j)}_t = z \middle| \mathbf{o};\tilde{\theta}\right)\right]} \tag{3.26}$$

$$\mu_z^* = \frac{\sum_{j=1}^N\left[\sum_{t=0}^{n_j-1} o^{(j)}_t\, \mathbb{P}\left(Z^{(j)}_t = z \middle| \mathbf{o};\tilde{\theta}\right)\right]}{\sum_{j=1}^N\left[\sum_{t=0}^{n_j-1}\mathbb{P}\left(Z^{(j)}_t = z \middle| \mathbf{o};\tilde{\theta}\right)\right]} \tag{3.27}$$

$$\Sigma_z^* = \frac{\sum_{j=1}^N\left[\sum_{t=0}^{n_j-1}\left(o^{(j)}_t - \tilde{\mu}_z\right)^\top\left(o^{(j)}_t - \tilde{\mu}_z\right)\mathbb{P}\left(Z^{(j)}_t = z \middle| \mathbf{o};\tilde{\theta}\right)\right]}{\sum_{j=1}^N\left[\sum_{t=o}^{n_j-1}\mathbb{P}\left(Z^{(j)}_t = z \middle| \mathbf{o};\tilde{\theta}\right)\right]}, \tag{3.28}$$

where $z, z' \in \mathcal{S}_Z$.

Lemma 1 can be proven by rewriting Equations (3.24) and (3.25) and applying the Karush-Kuhn-Tucker conditions to derive the maximizers. Nevertheless, we emphasize the intuitive nature of Equations (3.26) - (3.28): the numerator of Equation (3.26) corresponds to the expected number of transitions from state z to z' and the denominator of Equations (3.26) - (3.28) corresponds to the expected number of visits to state z.

Equations (3.26), (3.27) and (3.28) rely on two distributions that can be recursively determined using the so-called forward-backward algorithm for HMMs. The core idea is to divide the observation sequence in two subsequences, a 'past sequence' and a 'future sequence'. This decomposition was first proposed by [44] and later presented by [3]. Due to the latter being the more common reference and its more general framework, we base our analysis on the approach outlined in [3].[1]

We first define the forward probabilities

$$\alpha\left(t,z,j;\theta\right) := f^{(j)}_{O_0,\dots,O_t,Z_t}\left(o^{(j)}_0,\dots,o^{(j)}_t,z;\theta\right), \tag{3.29}$$

which denotes the joint density of the observation sequence of the j-th cycle up to sampling epoch t and the state of the Markov chain at time t.

Similarly, we define the backward probabilities

$$\beta\left(t,z,j;\theta\right) = f^{(j)}_{O_{t+1},\dots,O_{n_j}|Z_t}\left(o^{(j)}_{t+1},\dots,o^{(j)}_{n_j}|Z^{(j)}_t = z;\theta\right), \tag{3.30}$$

denoting the joint density of the observation sequence of the j-th cycle after time t, given the state at time t.

The forward and backward probabilities can be calculated in a recursive manner, which in our case leads to

$$\alpha\left(t,z,j;\tilde{\theta}\right) = \begin{cases} h^{\langle\tilde{\omega}\rangle}\left(o^{(j)}_0,z\right)\nu\left(z\right) & \text{if } t=0\\ \mathbb{1}\left\{z\neq\mathcal{F}_Z\right\}h^{\langle\tilde{\omega}\rangle}\left(o^{(j)}_t,z\right)\left[\sum_{z'=1}^{m}\alpha\left(t-1,z',j;\tilde{\theta}\right)\tilde{T}_Z\left(z',z\right)\right] & \text{if } t=1,\dots,n_j-1\\ \mathbb{1}\left\{z=\mathcal{F}_Z\right\}\left[\sum_{z'=1}^{m}\alpha\left(n_j-1,z',j;\tilde{\theta}\right)\tilde{T}_Z\left(z',z\right)\right] & \text{if } t=n_j \end{cases} \tag{3.31}$$

and

$$\beta\left(t,z,j;\tilde{\theta}\right) = \begin{cases} \mathbb{1}\left\{z=\mathcal{F}_Z\right\} & \text{if } t=n_j\\ \tilde{T}_Z\left(z,\mathcal{F}_Z\right)\mathbb{1}\left\{z\neq\mathcal{F}_Z\right\} & \text{if } t=n_j-1\\ \sum_{z'=0}^{m}\left[\tilde{T}_Z\left(z,z'\right)h^{\langle\tilde{\omega}\rangle}\left(o^{(j)}_{t+1},z'\right)\beta\left(t+1,z',j;\tilde{\theta}\right)\right] & \text{if } t=n_j-2,\dots,1\\ \mathbb{1}\left[\left\{z=1\right\}\sum_{z'=0}^{m}\tilde{T}_Z\left(z,z'\right)h^{\langle\omega\rangle}\left(o^{(j)}_1,z'\right)\beta\left(1,z',j;\tilde{\theta}\right)\right] & \text{if } t=0. \end{cases} \tag{3.32}$$

Note that, due to the Markovian structure, it holds that

$$\alpha\left(t,z,j;\tilde{\theta}\right)\beta\left(t,z,j;\tilde{\theta}\right) = \mathbb{P}\left(Z^{(j)}_t = z\,\middle|\,o^{(j)}\right)f^{(j)}_{\mathbf{O}}\left(o^{(j)}\right). \tag{3.33}$$

[1]The Baum-Welch algorithm, which supposedly refers to a collaboration between Baum and Welch entitled "A Statistical Estimation Procedure for Probabilisitc Functions of Finite Markov Processes" seems to never have been published. Hence we reference the work of [3], which is the first published description of the approach by Baum.

As a consequence

$$\mathbb{P}\left(Z_t^{(j)} = z \middle| o^{(j)}; \tilde{\theta}\right) = \frac{\alpha\left(t, z, j; \tilde{\theta}\right) \beta\left(t, z, j; \tilde{\theta}\right)}{\sum_{z' \in \mathcal{S}_Z} \left[\alpha\left(t, z', j; \tilde{\theta}\right) \beta\left(t, z', j; \tilde{\theta}\right)\right]}. \tag{3.34}$$

Let us now define

$$\eta\left(t, z, z', j, \tilde{\theta}\right) := \alpha\left(t, z, j; \tilde{\theta}\right) \tilde{T}_Z\left(z, z'\right) \beta\left(t, z', j; \tilde{\theta}\right) h^{\langle\omega\rangle}\left(o_{t+1}^{(j)}, z'\right). \tag{3.35}$$

Again, using the Markovian structure, we find that

$$\mathbb{P}\left(Z_t^{(j)} = z, Z_{t+1}^{(j)} = z' \middle| o^{(j)}; \tilde{\theta}\right) = \frac{\eta\left(t, z, z', j, \tilde{\theta}\right)}{\sum_{z_1, z_2 \in \mathcal{S}_Z} \left[\eta\left(t, z_1, z_2, j, \tilde{\theta}\right)\right]}. \tag{3.36}$$

The initial estimate $\theta^{(0)}$ of the parameters of the HMM is determined based on our assumptions. Specifically, the transition matrix is initialized as a 'decreasing' upper-triangular transition matrix (meaning that transition to states that are far away, are less likely) and the means of the Gaussian distributions are initialized using the k-means algorithm. This completes the description of the Baum-Welch algorithm to derive the parameters of the HMM, a global overview is presented in Algorithm 1.

Algorithm 1 The Baum-Welch algorithm

```
1: function BAUM-WELCH ALGORITHM($\mathbf{o}, \theta^{(0)}$);
2:     $u = 0$
3:     while no convergence of $\log f_{\mathbf{O}}^N\left(\mathbf{o}; \theta^{(u)}\right)$ do
4:         Calculate Equations (3.31) and Equation (3.32)
5:         Calculate Equations (3.34) and (3.36)
6:         Update $\theta^{(u)}$ using Equations (3.26), (3.27) and (3.28)
7:         $u \mathrel{+}= 1$
8: return $\theta^{(u)}$
```

Chapter 4

Markov Decision Processes with Partial Information

4.1 Markov Decision Processes

In this section, we introduce Markov decision processes (MDPs) and illustrate how the CBM problem can be modelled by an MDP, if the component state is always known. Subsequently, we show how this 'full-information MDP' can be solved to determine an optimal policy.

4.1.1 The CBM problem as a full-information MDP

Markov decision processes (MDPs) are a general framework for decision making under uncertainty. MDPs can be seen as Markov chains extended with actions and costs.

Consider an infinite horizon MDP, with initial state $Z_0 = z$ (note that in our case $Z_0 = 1$). At each decision epoch $t \in \{0, 1, 2, ...\}$, we observe the state $Z_t \in \mathcal{S}_Z$ of the Markov chain and subsequently choose an action A_t from the action space $\mathcal{A}$, based on a policy π, which leads to an immediate cost $R_Z(Z_t, A_t)$.[1] Depending on the chosen action and the current state, the state of the Markov chain changes according to a *probabilistic transition model*, which is given by T_M. This, given that the policy π is Markovian with respect to the state of the Markov chain and stationary, induces the bivariate discrete time Markov cost process $\{(Z_t, R_Z(Z_t, \pi(Z_t))); t = 0, 1, 2, ...\}$, which leads to the following definition of an MDP.

Definition 2 (Markov decision process (MDP), conform [48]). *A Markov decision process (MDP) is a tuple $(\mathcal{S}_Z, \mathcal{A}, T_M, R_Z)$, where*

- *$\mathcal{S}_Z$ is the finite set of states of the Markov chain;*
- *$\mathcal{A}$ is the finite set of actions;*
- *T_M is the probabilistic transition model;*
- *R_Z is the immediate cost function.*

The probabilistic transition model is defined as a function

$$T_M : \mathcal{A} \times \mathcal{S}_Z \times \mathcal{S}_Z \to [0, 1], \tag{4.1}$$

[1] The state space of the Markov chain corresponds exactly to the component state space, i.e. the state space of the Markov chain from Chapter 3.

where $T_M(a,z,z') := \mathbb{P}(Z_1 = z' | Z_0 = z, A_0 = a)$.
The immediate cost function is defined as a function

$$R_Z : \mathcal{S}_Z \times \mathcal{A} \to \mathbb{R}, \tag{4.2}$$

where $R_Z\,(z,a) =$ *'immediate cost of action a in state z'.*

A common optimization criteria for infinite-horizon MDPs is the expected total discounted cost, with discount factor $\gamma \in [0,1)$.[2] If we restrict ourselves to the space of deterministic Markovian stationary policies, denoted by Π, the resulting optimization problem is to find

$$\pi_M^* := \arg\min_{\pi\in\Pi} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R_Z\left(Z_t, \pi\left(Z_t\right)\right)\right]. \tag{4.3}$$

Such a policy $\pi \in \Pi$ specifies the decision rule used at each decision epoch. This decision rule, denoted by d, is a deterministic mapping from $\mathcal{S}_Z$ to $\mathcal{A}$ such that $A_t = d\,(Z_t)$. For stationary deterministic policies, the decision rule d defines the entire policy, i.e. $\pi = (d, d, ...) \Leftrightarrow d$.

Recall that in Chapter 3 we assumed that the degradation of the component can be described by an absorbing Markov chain with transition matrix T_Z and state space $\mathcal{S}_Z = \{1, ..., m\} \cup \{\mathcal{F}_Z\}$. As a result, the component state $Z_t \in \mathcal{S}_Z$ at decision epoch t, can be seen as the unique characterization of all information that is important for optimal decision-making. At each decision epoch, one of two actions can be chosen, i.e.

$$\mathcal{A} = \{\text{'do nothing'}, \text{'do maintenance'}\}, \tag{4.4}$$

which we denote by 0 and 1 respectively.

If the component is still operational at decision epoch t and $A_t = 1$, the component is immediately replaced by a new, perfect, component (at cost c_{pm}). If the component is still operational and $A_t = 0$, the component continues to deteriorate according to transition matrix T_Z. If the component was no longer operational, it will be replaced correctly (at cost $c_{\text{cm}} > c_{\text{pm}}$), regardless of the chosen action. This leads to the following probabilistic transition model and immediate cost function:

$$T_{\text{M}}(a,z,z') = \begin{cases} T_Z\,(z,z') & \text{if } a = 0 \wedge z \in \{1,...,m\} \\ T_Z\,(1,z') & \text{if } a = 0 \wedge z = \mathcal{F}_Z \\ T_Z\,(1,z') & \text{if } a = 1 \end{cases} \tag{4.5}$$

and

$$R_Z(a,z) = \begin{cases} 0 & \text{if } a = 0 \wedge z \in \{1,...,m\} \\ c_{\text{pm}} & \text{if } a = 1 \wedge z \in \{1,...,m\} \\ c_{\text{cm}} & \text{if } z = \mathcal{F}_Z. \end{cases} \tag{4.6}$$

The MDP, as defined in Definition 2, with a probabilistic transition model as given by Equation (4.5) and an immediate cost function as given by Equation (4.6), is referred to as the *full-information MDP* for the CBM problem, as it assumes that we always

[2]A discounted cost MDP behaves like an average cost MDP when the discount factor is close to 1, see e.g. [25].

observe the state of the component. The dynamics of this full-information MDP are summarized in Algorithm 2.

Algorithm 2 Dynamics of a full-information MDP

```
Input: policy $\pi$;
1: Initial state $Z_0 = 1$
2: for $t = 0, 1, ..., \infty$ do
3:     Choose action $A_t = \pi(Z_t)$
4:     Incur cost $R_Z(A_t, Z_t)$
5:     State transition $Z_t \to Z_{t+1}$ according to $T_M(A_t, Z_t, \cdot)$
```

4.1.2 Optimal policies for the full-information MDP

Given an initial state $Z_0 = z \in \mathcal{S}_Z$, we evaluate policies by their value function; the value function $v_M^\pi(z)$ of policy π for the full-information MDP is the expected total discounted cost, with discount factor γ, when starting in state z and executing policy π, i.e.

$$v_M^\pi(z) := \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R_Z(Z_t, \pi(Z_t)) \middle| Z_0 = z\right]. \tag{4.7}$$

The optimal value function v_M^* is the unique solution of

$$v_M^*(z) = \min_{a \in \mathcal{A}} \left\{ R_Z(z,a) + \gamma \sum_{z' \in \mathcal{S}_Z} T_M(a,z,z') \, v_M^*(z') \right\} \quad \forall z \in \mathcal{S}_Z, \tag{4.8}$$

and the deterministic Markovian stationary defined as

$$\pi_M^*(z) = \arg\min_{a \in \mathcal{A}} \{ q_M^*(z,a) \} \quad \forall z \in \mathcal{S}_Z, \tag{4.9}$$

where $q_M^*(z,a) := R_Z(z,a) + \gamma \sum_{z' \in \mathcal{S}_Z} T_M(a,z,z') \, v_M^*(z')$, constitutes optimal behaviour in the sense of Equation (4.3), see e.g. [37].

Equation (4.8) can be explained intuitively by the *principle of optimality* (see [4]), which roughly states that any optimal policy consists of optimal 'sub-policies'. Furthermore, in [37] is shown that for MDPs with a finite state space, finite action space and bounded costs, there exists an optimal policy π^* in the class of deterministic Markovian stationary policies. This means that there is no policy in the class of history-dependent randomized policies (i.e. the broadest class of policies) that outperforms policy π^*, as defined in Equation (4.9).

4.2 Partially Observable Markov Decision Processes

In this section, we introduce partially observable Markov decision processes (POMDPs) as an extension of MDPs and illustrate how POMDPs can be seen as continuous-state MDPs. Subsequently, we elaborate on the structure of the optimal value function and explain how the PERSEUS algorithm can be used to approximate the optimal value function. Lastly, we present two simple POMDP-based heuristics.

The full-information MDP described in Section 4.1, assumes that we always observe the current state of the Markov chain. A partially observable Markov decision process (POMDP) (introduced in [15]) extends this model to incorporate uncertainty

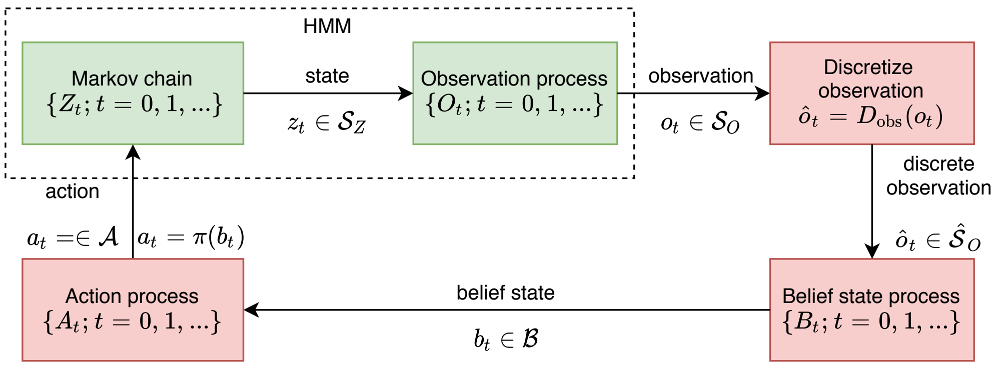

HMM
Markov chain $\{Z_t; t = 0, 1, ...\}$
state
$z_t \in \mathcal{S}_Z$
Observation process $\{O_t; t = 0, 1, ...\}$
observation
$o_t \in \mathcal{S}_O$
Discretize observation $\hat{o}_t = D_{\text{obs}}(o_t)$
discrete observation
$\hat{o}_t \in \hat{\mathcal{S}}_O$
Belief state process $\{B_t; t = 0, 1, ...\}$
belief state
$b_t \in \mathcal{B}$
Action process $\{A_t; t = 0, 1, ...\}$
action
$a_t = \in \mathcal{A}$
$a_t = \pi(b_t)$

FIGURE 4.1: Schematic overview of the POMDP framework.

regarding the state of the Markov chain. This gives rise to a *probabilistic observation model*, which relates the observations to states of the Markov chain. As a result, POMDPs can also be seen as an extension of HMMs with actions and costs.

4.2.1 Extending MDPs to partial information

We define POMDPs based on the corresponding full-information MDP by extending Definition 2 with an observations space and a probabilistic observation model.

Definition 3 (Partially observable Markov decision process (POMDP), based on [48]). *A partially observable Markov decision process (POMDP) is a tuple* $(\mathcal{S}_Z, \mathcal{A}, T_M, R_Z, \hat{\mathcal{S}}_O, \Omega)$, *where*

- $\mathcal{S}_Z$ *is the finite set of states of the Markov chain;*
- $\mathcal{A}$ *is the finite set of actions;*
- T_M *is the probabilistic transition model;*
- R_Z *is the immediate cost function;*
- $\hat{\mathcal{S}}_O$ *is the finite set of observations;*
- Ω *is the probabilistic observation model.*

The probabilistic observation model is defined as a stochastic matrix, with elements

$$\Omega(z, o) := \mathbb{P}(O_1 = o | Z_1 = z) \quad \forall \left[o \in \hat{\mathcal{S}}_O \wedge z \in \mathcal{S}_Z\right]. \tag{4.10}$$

In Figure 4.1, a schematic overview of the POMDP framework is presented. Instead of directly observing the Markov chain, we only have access to an observation process. This observation process depends on the state of Markov chain and combined these stochastic processes form a HMM (see Chapter 3 and Definition 1).

Recall that in Chapter 3 we assumed that the observations are samples from a multivariate Gaussian distribution whose parameters depend on the state of the Markov chain. However, since problems with a continuous observations space, or a very large discrete observation space, cannot be properly solved using model-based algorithms (see e.g. [20]), we restrict ourselves to POMDPs with a finite observation space.[3] As a result, a finite observation space needs to be constructed from the monitoring data $\mathbf{o}$. We employ an unsupervised clustering-based discretization method using k-means (based on [13]), where the monitoring data is reduced to a total of $v^K = \left(|\hat{\mathcal{S}}_O| - 1\right)$ observations. Specifically, using the k-means algorithm, we discretise the monitoring data for each feature to a total of v clusters. The finite observations space $\hat{\mathcal{S}}_O$ is then constructed by calculating the Cartesian product between these K sets of clusters and adding the observation $\mathcal{F}_O$ that uniquely defines the failure state. When executing the POMDP policy, new monitoring data is discretised based on the Euclidean distance.

The probabilistic observation model of the POMDP is given by the following stochastic matrix:

$$\Omega(z,o) = \begin{cases} \frac{f_{O_t|Z_t}(o|Z_t=z)}{\sum_{o'\in\hat{\mathcal{S}}_O\setminus\{\mathcal{F}_O\}} f_{O_t|Z_t}(o'|Z_t=z)} & \text{if } \left[z \in \mathcal{S}_Z \setminus \{\mathcal{F}_Z\} \wedge o \in \hat{\mathcal{S}}_O \setminus \{\mathcal{F}_O\}\right] \\ 1 & \text{if } z = \mathcal{F}_Z \wedge o = \mathcal{F}_O \\ 0 & \text{else,} \end{cases} \tag{4.11}$$

where $f_{O_t|Z_t}(o|\, Z_t = z)$ denotes the conditional density of the observations (see Equation (3.7)).

When operating in a POMDP environment, our history at decision epoch t comprises of the entire sequence of observations and chosen actions up to that point, i.e.

$$\mathtt{H}_t := (a_0, o_1, a_1, ..., o_t) \equiv (\mathtt{H}_{t-1}, a_{t-1}, o_t)\,. \tag{4.12}$$

Based on this history, we choose the next action. However, computing a policy based on the entire history, is impractical. At the same time, directly mapping the observation o_t to an action A_t is not suboptimal (see e.g. [39]). We therefore summarize the entire history into a posterior probability distribution over the states of the Markov chain. This distribution is commonly referred to as the *belief state*, or simply the *belief*, defined as:

$$b_t(z) := \mathbb{P}\left(Z_t = z|\, \mathtt{H}_t\right) \quad \forall z \in \mathcal{S}_Z. \tag{4.13}$$

The belief is the basis for our decision-making (as displayed in Figure 4.1), which means that we are effectively operating on a different state space, the so-called belief space $\mathcal{B}$, which is the m-simplex, i.e.

$$\mathcal{B} := \left\{b \in \mathbb{R}^{m+1} : \left[\sum_{z\in\mathcal{S}_Z} b(z) = 1\right] \wedge \left[b(z) \geq 0\, \forall z \in \mathcal{S}_Z\right]\right\}. \tag{4.14}$$

Consequently, a policy π for the POMDP is now a map from the belief space $\mathcal{B}$ to the action space (see Figure 4.1), which means that the POMDP can be interpreted as continuous-state MDP (see e.g. [51]).

[3]Model-free approaches, on the other hand, tend to require extensive simulation or sufficient a priori knowledge to restrict the policy search space (see [20]).

At decision epoch t, the posterior belief is updated using Bayes' theorem:

$$\begin{aligned}
b_t\left(z'\right) &= \mathbb{P}\left(Z_t = z' \middle| \mathtt{H}_t\right) = \mathbb{P}\left(Z_t = z' | \mathtt{H}_{t-1}, A_{t-1} = a, O_t = o\right) \\
&= \frac{\mathbb{P}\left(O_t = o | Z_t = z', A_{t-1} = a, \mathtt{H}_{t-1}\right) \mathbb{P}\left(Z_t = z' | A_{t-1} = a, \mathtt{H}_{t-1}\right)}{\mathbb{P}\left(O_t = o | \mathtt{H}_{t-1}, A_{t-1} = a\right)} \\
&= \frac{\Omega\left(z', o\right) \sum_{z \in \mathcal{S}_Z} \mathbb{P}\left(Z_t = z' | A_{t-1} = a, Z_{t-1} = z\right) \mathbb{P}\left(Z_{t-1} = z | \mathtt{H}_{t-1}\right)}{\mathbb{P}\left(O_t = o | \mathtt{H}_{t-1}, A_{t-1} = a\right)} \\
&= \frac{\Omega\left(z', o\right) \sum_{z \in \mathcal{S}_Z} T_M\left(a, z, z'\right) b_{t-1}\left(z\right)}{\mathbb{P}\left(O_t = o | \mathtt{H}_{t-1}, A_{t-1} = a\right)}.
\end{aligned} \tag{4.15}$$

Since the denominator of Equation (4.15) is a normalizing constant, it follows that b_{t-1} summarizes the entire history $\mathtt{H}_{t-1}$ and that the belief can simply be updated using the previous belief b_{t-1}, the chosen action a and the resulting observation o. This shows that the belief is a sufficient statistic for the entire history, in the sense that

$$b_t\left(z'\right) = \mathbb{P}\left(Z_t = z' \middle| \mathtt{H}_{t-1}, a_{t-1}, o_t\right) = \mathbb{P}\left(Z_t = z' \middle| b_{t-1}, a_{t-1}, o_t\right), \tag{4.16}$$

as also shown in e.g. [40].

We now define the following diagonal matrix:

$$\bar{\Omega}_o := \operatorname{diag}\left(\Omega\left(1, o\right), ..., \Omega\left(m, o\right), \Omega\left(\mathcal{F}_Z, o\right)\right). \tag{4.17}$$

This allows us to compactly describe the function T_B, which updates the belief:

$$T_B : \mathcal{B} \times \mathcal{A} \times \hat{\mathcal{S}}_O \rightarrow \mathcal{B} \text{ with } T_B\left(b, a, o\right) := \frac{\bar{\Omega}_o T_M^\top\left(a\right) b}{\mathbb{1}_{(m+1)}^\top \bar{\Omega}_o T_M^\top\left(a\right) b}, \tag{4.18}$$

where $T_M\left(a\right) \equiv T_M\left(a, \cdot, \cdot\right)$ denotes the probabilistic transition model of the full-information MDP for action $a \in \mathcal{A}$ and $\mathbb{1}_{(m+1)}$ denotes an $(m+1)$-dimensional vector with entries 1.

The dynamics of the POMDP are summarized in Algorithm 3.

Algorithm 3 Dynamics of a POMDP

```
Input: policy $\pi$;
1: Initial state $Z_0 = 1$, i.e. $b_0 = e_1$
2: for $t = 0, 1, ..., \infty$ do
3:     Choose action $A_t = \pi\left(B_t\right)$
4:     Incur cost $R_B\left(A_t, B_t\right)$
5:     State transition $Z_t \rightarrow Z_{t+1}$ based on $T_M\left(A_t, Z_t, \cdot\right)$
6:     Register observation $O_{t+1}$ based on $\Omega\left(Z_{t+1}, \cdot\right)$
7:     Update belief state $B_{t+1} = T_B\left(B_t, A_t, O_{t+1}\right)$
```

4.2.2 The optimal value function of POMDPs

The value function $v_n^\pi(b)$ of an arbitrary policy π of an n-stage POMDP (i.e. a POMDP with $n+1$ decision epochs) with initial belief state b is given by

$$v_n^\pi(b) := \mathbb{E}\left[\sum_{t=0}^{n} \gamma^t R_B(B_t, \pi(B_t)) \middle| B_0 = b\right] \quad \forall b \in \mathcal{B}, \tag{4.19}$$

where $R_B(B_t, \pi(B_t)) := \sum_{z \in \mathcal{S}_Z} R_M(z, \pi(B_t)) B_t(z)$.

Note that, since $\mathbb{P}(Z_0 = 1) = 1$, we have that $B_0 = e_1$.[4]

We now introduce the dynamic programming operator $\mathcal{D} : U \to U$ for POMDPs, where U denotes the set of bounded and real-valued functions on $\mathcal{B}$ (i.e. $v \in U$ is a bounded function from $\mathcal{B}$ to $\mathbb{R}$), defined as:

$$\mathcal{D}v(b) := \min_{a \in \mathcal{A}} \left\{ R_B(b,a) + \gamma \sum_{o \in \hat{\mathcal{S}}_O} v(T_B(b,a,o)) \Omega_b^a(o) \right\}, \tag{4.20}$$

where $b \in \mathcal{B}$ and $\Omega_b^a(o) := \mathbb{P}(O_1 = o | B_0 = b, A_0 = a)$.

Notice that

$$\mathbb{P}(O_1 = o | B_0 = b, A_0 = a) = \sum_{z,z' \in \mathcal{S}_Z} \Omega(z', o) T_M(a, z, z') b(z). \tag{4.21}$$

The optimal value function v_n^* of an n-stage POMDP satisfies the following recursive relation [40]:

$$\begin{aligned} v_n^*(b) &= \mathcal{D}v_{n-1}^*(b) \\ &= \min_{a \in \mathcal{A}} \{q_n^*(b,a)\}, \end{aligned} \tag{4.22}$$

where $q_n^*(b,a) := R_B(b,a) + \gamma \sum_{o \in \hat{\mathcal{S}}_O} v_{n-1}^*(T_B(b,a,o)) \Omega_b^a(o)$.

One could use so-called *policy trees* to describe a policy of a finite-horizon POMDP (see e.g. [23]). Let $g \in \mathcal{G}_n$ denote a policy tree of an n-stage POMDP, where $\mathcal{G}_n$ denotes the set of all possible policy trees. We recursively define g as a tuple $g := (a, \rho_n)$, where $a \in \mathcal{A}$ denotes the immediate action and $\rho_n : \hat{\mathcal{S}}_O \to \mathcal{G}_n$ denotes the *conditional plan*. In case $n = 0$, we can only perform a single (final) action, hence $\rho_1 : \hat{\mathcal{S}}_O \to \mathcal{A}$ and $\mathcal{G}_0 := \{a : a \in \mathcal{A}\}$.

An illustration of a policy tree is given in Figure 4.2. The policy tree $g := (a, \rho_3) \in \mathcal{G}_3$ consist of the action $a \in \mathcal{A}$ (executed at time t) and the conditional plan ρ_3. The conditional plan specifies a subtree for each observation $o \in \hat{\mathcal{S}}_O$.

[4]Where e_1 denotes the $(m+1)$-dimensional zero vector, with only at index 1 the element 1.

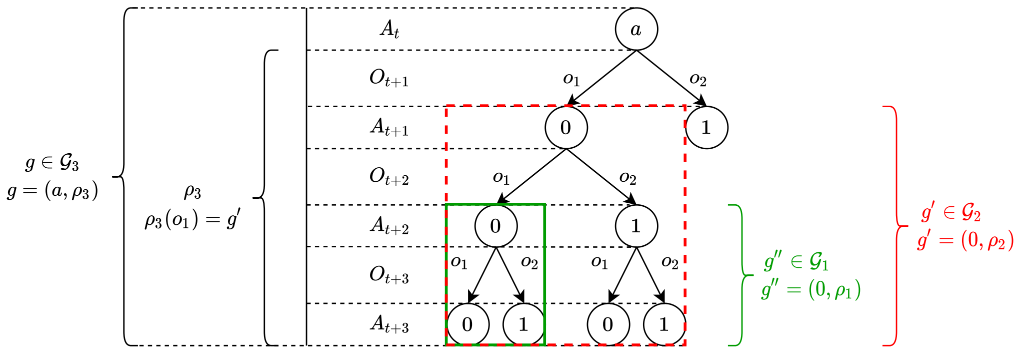


FIGURE 4.2: Illustration of a policy tree for a 3-stage POMDP, with decision epochs $\{t, t+1, t+2, t+3\}$. The action space is $\mathcal{A} := \{0, 1\}$ and the observation space is $\hat{\mathcal{S}}_O := \{o_1, o_2\}$.

When the state of the Markov chain is $z \in \mathcal{S}_Z$, the expected total discounted cost of executing policy tree $g := (a, \rho_n) \in \mathcal{G}_n$ at time t is given by

$$\begin{aligned} v_n^{\langle g \rangle}(e_z) &:= R_Z(z,a) + \gamma \mathbb{E}\left[\text{'future cost'} \middle| A_t = a, Z_t = z\right] \\ &= R_Z(z,a) + \gamma \sum_{o \in \hat{\mathcal{S}}_O} \sum_{z' \in \mathcal{S}_Z} \Big[\mathbb{E}\left[\text{'future cost'} \middle| O_{t+1} = o, Z_{t+1} = z'\right] \\ &\quad \cdot \mathbb{P}\left(O_{t+1} = o, Z_{t+1} = z' \middle| A_t = a, Z_t = z\right)\Big] \\ &= R_Z(z,a) + \gamma \sum_{o \in \hat{\mathcal{S}}_O} \sum_{z' \in \mathcal{S}_Z} v_{n-1}^{\langle \rho_n(o) \rangle}(e_{z'})\, T_M(a, z, z')\, \Omega(z', o), \end{aligned} \tag{4.23}$$

with $v_0^{\langle \rho_1(o) \rangle}(e_z) = R_Z(z, \rho_1(o))$.

For an arbitrary belief state $b \in \mathcal{B}$, it naturally holds that

$$v_n^{\langle g \rangle}(b) := \sum_{z \in \mathcal{S}_Z} b(z)\, v_n^{\langle g \rangle}(e_z). \tag{4.24}$$

The expected total discounted cost of executing a policy tree $g_i \in \mathcal{G}_n$ is denoted by a so-called α-vector.

Definition 4 (α-vector). *Let $g_i \in \mathcal{G}_n$ denote a policy tree of an n-stage POMDP. We call $\alpha_n^i \in \mathbb{R}^{(m+1)}$, defined as*

$$\alpha_n^i(z) := v_n^{\langle g_i \rangle}(e_z) \quad \forall z \in \mathcal{S}_Z, \tag{4.25}$$

an α-vector of the corresponding POMDP. The set of all α-vectors is denoted by

$$\hat{\alpha}_n := \left\{\alpha_n^i : g_i \in \mathcal{G}_n\right\}. \tag{4.26}$$

The optimal value function can now simply be written as

$$v_n^*(b) = \min_{\alpha_n^i \in \hat{\alpha}_n} \sum_{z \in \mathcal{S}_Z} b(z)\, \alpha_n^i(z). \tag{4.27}$$

Equation (4.27) illustrates the fact that the optimal value function of a finite-horizon POMDP is piecewise linear and concave (PWLC), which was first shown in [40]. Figure 4.3 illustrates what this means. It can be seen that the optimal value function

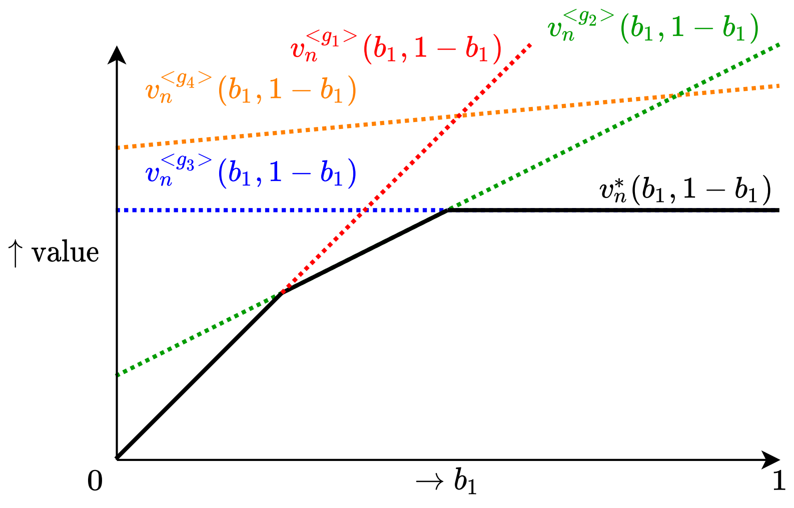


FIGURE 4.3: Illustration of the optimal value function $v_n^*(b)$ of an n-stage POMDP, with two states (i.e. $\mathcal{S}_Z = \{1, 2\}$), where b_1 denotes the probability that the Markov chain is in the first state.

consists of the value functions of three different policy trees. Which policy tree is optimal, depends on the belief. However, there is no belief $b \in \mathcal{B}$ such that policy tree $g_4 \in \mathcal{G}_n := \{g_1, g_2, g_3, g_4\}$ is optimal.

Indeed, certain policy trees $g \in \mathcal{G}_n$ will be suboptimal for every belief $b \in \mathcal{B}$. This means that the corresponding α-vectors are not really part of the optimal value function. As a result, the set of α-vectors $\hat{\alpha}_n$ can be reduced to a parsimonious subset α_n, such that

$$v_n^*(b) = \min_{\alpha_n^i \in \alpha_n} \sum_{z \in \mathcal{S}_Z} b(z)\, \alpha_n^i(z)\,. \tag{4.28}$$

Computing the optimal value function of an $(n+1)$-stage POMDP using the value function of the corresponding n-stage POMDP, is known as exact value iteration. However, this procedure is computationally quite expensive, since the number of possible policy trees (and therefore the number of α-vectors) grows exponentially with the number of stages, specifically

$$|\mathcal{G}_{n+1}| = |\mathcal{A}| \cdot |\mathcal{G}_n|^{|\hat{\mathcal{S}}_O|}\,. \tag{4.29}$$

Additionally, reducing this set to a parsimonious subset (using for example Lark's pruning algorithm, see e.g. [10]) is, especially in higher dimensions, computationally expensive. As a result, the consensus is that exact value iteration is intractable and that it is unlikely that efficient algorithms to find optimal policies for general POMDPs exist (see e.g. [30]).

However, updating the value function for a single belief $b \in \mathcal{B}$ is of smaller computational cost. In fact, given the optimal value function v_n^* of the n-stage POMDP (described by α_n), one can efficiently calculate $v_{n+1}^*(b)$, using the so-called *backup operator*. We now illustrate how the formulas for this backup operator can be derived, conform [48].

First, recall the following relation:

$$v_{n+1}^*(b) = \min_{a \in \mathcal{A}} \left\{ R_B(b,a) + \gamma \sum\nolimits_{o \in \hat{\mathcal{S}}_O} v_n^*(T_B(b,a,o))\, \Omega_b^a(o) \right\}. \tag{4.30}$$

Due to Equation (4.15), the updated belief state can be written as

$$(T_B(b,a,o))(z) = \frac{\Omega(z',o) \sum_{z \in \mathcal{S}_Z} T_M(a,z,z')\, b(z)}{\Omega_b^a(o)} \quad \forall s \in \mathcal{S}_Z, \tag{4.31}$$

where we used that $\mathbb{P}(O_{t+1} = o | \mathtt{H}_t, A_t = a) = \Omega_b^a(o)$ (note that b denotes the belief calculated from the entire history $\mathtt{H}_t$).

Using Equation (4.31), one can rewrite $v_n^*(T_B(b,a,o))$ as follows:

$$\begin{aligned} v_n^*(T_B(b,a,o)) &= \min_{\alpha_n^i \in \alpha_n} \frac{\sum_{z' \in \mathcal{S}_Z} \left[\left(\Omega(z',o) \sum_{z \in \mathcal{S}_Z} T_M(a,z,z')\, b(z) \right) \alpha_n^i(s') \right]}{\Omega_b^a(o)} \\ &= \min_{\alpha_n^i \in \alpha_n} \frac{\sum_{z \in \mathcal{S}_Z} \left[b(z) \sum_{z' \in \mathcal{S}_Z} \Omega(z',o)\, T_M(a,z,z')\, \alpha_n^i(z') \right]}{\Omega_b^a(o)} \\ &= \frac{\min_{\alpha_n^i \in \alpha_n} b \cdot g_{ao}^i}{\Omega_b^a(o)}, \end{aligned} \tag{4.32}$$

with $g_{ao}^i(z) := \sum_{z' \in \mathcal{S}_Z} \Omega(z',o)\, T_M(a,z,z')\, \alpha_n^i(z')$.

As a result, Equation (4.30) can be written as

$$\begin{aligned} v_{n+1}^*(b) &= \min_{a \in \mathcal{A}} \left\{ R_B(b,a) + \gamma \sum\nolimits_{o \in \hat{\mathcal{S}}_O} \min_{\alpha_n^i \in \alpha_n} b \cdot g_{ao}^i \right\} \\ &= \min_{a \in \mathcal{A}} \left\{ b \cdot \alpha_0^{a+1} + b \cdot \gamma \sum\nolimits_{o \in \hat{\mathcal{S}}_O} \arg\min_{\{g_{ao}^i\}_{i=1,\dots,|\alpha_n|}} b \cdot g_{ao}^i \right\} \\ &= \min_{a \in \mathcal{A}} b \cdot g_a^b. \end{aligned} \tag{4.33}$$

where $g_a^b := \alpha_0^{a+1} + \gamma \sum_{o \in \hat{\mathcal{S}}_O} \arg\min_{\{g_{ao}^i\}_{i=1,\dots,|\alpha_n|}} b \cdot g_{ao}^i$.

This leads to the following definition of the backup operator:

$$\mathtt{backup}(b) := \arg\min_{\{g_a^b\}_{a \in \mathcal{A}}} b \cdot g_a^b, \tag{4.34}$$

i.e. the backup operator calculates, using the optimal value function of the n-stage POMDP (described by α_n), the optimal α-vector for a given belief $b \in \mathcal{B}$ of the $(n+1)$-stage POMDP.

4.2.3 Approximate value iteration for infinite-horizon POMDPs

The optimal value function of an infinite-horizon POMDP, must satisfy the following relation [41]:

$$\begin{aligned} v^*(b) &= \mathcal{D} v^*(b) \\ &= \min_{a \in \mathcal{A}} \{ q^*(b,a) \}, \end{aligned} \tag{4.35}$$

where $q^*(b,a) := R_B(b,a) + \gamma \sum_{o \in \hat{\mathcal{S}}_O} v^*(T_B(b,a,o))\, \Omega_b^a(o)$.

Although the optimal value function of an infinite-horizon POMDP need not be piecewise linear (but is still always concave), it can be approximated arbitrarily closely by a function that is concave and piecewise linear [41].

In particular, from the Banach Fixed-Point theorem (see e.g. [37]), which can be applied because function space U with the supremum norm forms a Banach space and operator $\mathcal{D}$ is a contraction mapping (see Appendix A.1), it follows that

(i) there exists a unique function $v^* \in U$ such that $v^* = Dv^*$;

(ii) for arbitrary $v_0 \in U$, the sequence $\{v_n\}_{n=0}^{\infty}$, defined by Equation (4.22) converges uniformly to v^*.

The optimal value function of an infinite-horizon POMDP can therefore be approximated arbitrarily closely by successively solving the finite-horizon POMDPs, using Equation (4.22). Or, by starting from some initial value function and iteratively applying the dynamic programming operator for POMDP, as defined in Equation (4.20), until convergence.

However, since performing exact value iteration is intractable, we use approximate point-based value iteration (PBVI) to solve the POMDP. This means that the POMDP is solved approximately through value iteration over a proxy belief space $\tilde{\mathcal{B}} \subseteq \mathcal{B}$. PBVI algorithms exploit the intuition that for many problems the set of 'reachable beliefs' (i.e. beliefs that can be reached by following an arbitrary policy starting from the initial belief state $b_0 \in \mathcal{B}$) forms a low dimensional manifold in the belief space $\mathcal{B}$, and thus can be covered densely enough by a relatively small number of belief points [42]. These algorithms are computationally desirable because the backup operator, as defined in Equation (4.34), can be efficiently executed for a single belief $b \in \mathcal{B}$ (see e.g. [38]).

In this study, we use the PERSEUS algorithm [42], which uses a randomized dynamic programming operator $\tilde{\mathcal{D}}$. This leads to a sequence $\{\tilde{\alpha}_n\}_{n=0}^{\infty}$, where $\tilde{\alpha}_n := \{\tilde{\alpha}_n^0, \tilde{\alpha}_n^1, ...\}$ denotes the α-vectors that describe the approximate value function $\tilde{v}_n$ (after n iterations). A global overview of the method is presented in Algorithm 4.

Algorithm 4 PERSEUS

```
1: function PERSEUS(POMDP);
2:     $n = 0$
3:     Initialize $\tilde{\alpha}_0 := \{\tilde{\alpha}_0^0\}$, with $\tilde{\alpha}_0^0(z) = (c_{\text{cm}} / (1-\gamma))$ for $z = 1, ..., m+1$
4:     Construct proxy belief space $\tilde{\mathcal{B}}$
5:     while no convergence of $\tilde{v}_n(e_1)$ do
6:         $n$ += 1
7:         $\tilde{\alpha}_n = \tilde{\mathcal{D}}(\tilde{\mathcal{B}}, \tilde{\alpha}_{n-1})$
8: return $\tilde{\alpha}_n$
```

Algorithm 4 can be divided into three parts:

1) Initializing $\tilde{\alpha}_0$;

2) Constructing the proxy belief space $\tilde{\mathcal{B}}$;

3) Executing the randomized dynamic programming operator $\tilde{\mathcal{D}}$.

Whereas most PBVI algorithms alternate between updating the proxy belief space $\tilde{\mathcal{B}}$ and updating the value function (see [38]), the PERSEUS algorithm does not update the proxy belief space. As a consequence, convergence to the optimal value function v^* is not guaranteed [46].

The value function $\tilde{v}_0$ is initialized as a single α-vector which is an upper bound of the optimal value function v^*, conform [42]. Alternatively, one could initialize the value function as close as possible to the optimal value function v^*. This would minimize the number of required iterations, but requires some knowledge of the optimal value function.

Many methods for constructing the proxy belief space exist, see e.g. [42] and [36]. In this study, we construct the proxy belief space based on the available monitoring data **o**. That is, we calculate the belief state of each non-trivial sampling and subsequently extend this set by, for each belief, simulating the next belief. An overview of this procedure can be found in Algorithm 5.

Algorithm 5 Proxy belief space construction

```
1:  function CONSTRUCT-PROXY-BELIEF-SPACE($\mathbf{o}$);
2:      $\tilde{\mathcal{B}} = \emptyset$
3:      for $j = 1, ..., N$ do
4:          $b = e_1$
5:          for $t = 1, ..., n_j - 1$ do
6:              $a = 0$
7:              $b = T_B\left(b, a, o_t^{(j)}\right)$
8:              Add $b$ to $\tilde{\mathcal{B}}$
9:      $\hat{\mathcal{B}} := \tilde{\mathcal{B}}$
10:     for $b \in \hat{\mathcal{B}}$ do
11:         Sample random current state $z \in \mathcal{S}_Z$ based on $b$
12:         Sample random next state $z' \in \mathcal{S}_Z$ based on $T_M(0, z, \cdot)$
13:         Sample random next observation $o \in \hat{\mathcal{S}}_O$ based on $\Omega(z', \cdot)$
14:         $b' = T_B(b, 0, o)$
15:         Add $b'$ to $\tilde{\mathcal{B}}$
16:     Add $e_1$ to $\tilde{\mathcal{B}}$
17:     Add $e_{m+1}$ to $\tilde{\mathcal{B}}$
18: return $\tilde{B}$
```

The randomized dynamic programming operator $\tilde{\mathcal{D}}$ is explained in Algorithm 6.

Algorithm 6 The randomized dynamic programming operator $\tilde{\mathcal{D}}$

```
1:  function $\tilde{\mathcal{D}}(\tilde{\mathcal{B}}, \tilde{\alpha}_n)$;
2:      $\tilde{\alpha}_{n+1} := \emptyset$ and $\hat{\mathcal{B}} := \tilde{\mathcal{B}}$
3:      while $\hat{\mathcal{B}} \neq \emptyset$ do
4:          $b \overset{R}{\leftarrow} \hat{\mathcal{B}}$
5:          Compute $\tilde{\alpha}_b :=$ backup$(b, \tilde{\alpha}_n)$
6:          if $b \cdot \tilde{\alpha}_b \leq \tilde{v}_n(b)$ then
7:              Add $\tilde{\alpha}_b$ to $\tilde{\alpha}_{n+1}$
8:          else
9:              $\tilde{\alpha}'_b := \arg\min_{\tilde{\alpha}_n^i \in \tilde{\alpha}_n} b \cdot \tilde{\alpha}_n^i$
10:             Add $\tilde{\alpha}'_b$ to $\tilde{\alpha}_{n+1}$
11:         Compute $\hat{\mathcal{B}} := \left\{b \in \tilde{\mathcal{B}} : \tilde{v}_{n+1}(b) < \tilde{v}_n(b)\right\}$
    return $\tilde{\alpha}_{n+1}$
```

Essentially, what happens is that at each iteration, a random belief $b \in \tilde{\mathcal{B}}$ is chosen, for which the backup operator, see Equation (4.34), is performed. This results into a vector α_b. Subsequently, we check whether this vector α_b improves the value function at belief b (compared to $\tilde{\alpha}_n$). If this is the case, we add α_b to the set $\tilde{\alpha}_{n+1}$. If this is not the case, we get the best vector from the set $\tilde{\alpha}_n$ and add this vector to the set $\tilde{\alpha}_{n+1}$. This procedure is repeated until the set $\tilde{\alpha}_{n+1}$ leads to an improved (or equally good) value function for the entire belief space. This also implies the convergence of the PERSEUS algorithm to some set $\tilde{\alpha}^*$ [46].

4.2.4 Other policies

Since solving a POMDP, even approximately, poses a significant computational burden, multiple POMDP-based heuristics have been developed (see e.g. [11]). In this study, we include two of these heuristics: the MLS heuristic and the QMDP heuristic. Executing the resulting MLS policy and QMDP policy only requires solving the underlying MDP and keeping track of the belief.

The MLS heuristic, which stand for 'most likely state', is an intuitive heuristic that simply picks the optimal action associated with the most likely state (according to the MDP). The resulting policy is simply given by

$$\pi_{\text{MLS}}\left(b\right) := \pi_M^*\left(\arg\max_{z \in \mathcal{S}_Z} b\left(s\right)\right). \tag{4.36}$$

However, this policy essentially completely ignores the uncertainty.

The QMDP heuristic, which is slightly more sophisticated, does account for the uncertainty and has shown promising results in some small-scale POMDP problems [27]. The QMDP policy is defined as

$$\pi_{\text{QMDP}}\left(b\right) := \arg\min_{a \in \mathcal{A}} \sum_{z \in \mathcal{S}_Z} b\left(s\right) q_M^*\left(z, a\right). \tag{4.37}$$

Chapter 5

Results

In this section, we present the results of our study. In Section 5.1, an overview of the most important results of the 5 scenarios (see Section 2.3) is presented, specifically:

(i) We present the relative average cost of the POMDP, QMDP and MLS policies for an increasing number of states, i.e. the average cost per time unit of the POMDP, QMDP and MLS policies (see Section 2.2.2) divided by the average cost per time unit of the optimal threshold policy (with respect to the degradation space);

(ii) We present the discounted cost of the POMDP and MDP value functions, that is

$$v_M^* (1) = \mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t R_Z \left(Z_t, \pi_{\mathrm{M}}^* \left(Z_t \right) \right) \middle| Z_0 = 1 \right] \tag{5.1}$$

$$v^* (e_1) = \mathbb{E} \left[\sum_{t=0}^{n} \gamma^t R_B \left(B_t, \pi^* \left(B_t \right) \right) \middle| B_0 = e_1 \right] . \tag{5.2}$$

(iii) We present the AIC and AICc of each HMM (see e.g. [7]);

(iv) We present the average cost of various threshold policies (with respect to the degradation process) and the average cost of the best POMDP policy.

The first point answers both our main question and sub-question 1. The second and third point, which relate to sub-question 2, assess whether it is possible to identify the best model based on the discounted costs and the AIC/AICc respectively. This is relevant because in practice one cannot always test different policies. The fourth point provides more perspective on the performance of the best POMDP model relative to other threshold policies.

A more in-depth discussion of these results is presented in Section 5.2.

For each simulation, we chose $N = 25$, $c_{\mathrm{pm}} = 25$, $c_{\mathrm{cm}} = 100$ and $\gamma = 0.99$.

5.1 Results from the simulations

First, we investigate Simulation 1, which has a relatively simple structure. The results of this simulation are shown in Figure 5.1 and Table 5.1. Choosing a HMM with $m = 2$ states, leads to an average cost that is more than 56% higher than the optimal average cost. This average cost can be significantly reduced by increasing the number of states to $m = 3$, in which case the average cost is only 2.7% above the optimal average cost. However, including more states does not significantly improve the policy. In fact, increasing the number of states to $m = 10$ leads to a slightly worse policy (4.8% above the optimal average cost). This is interesting, because Simulation 1 is exactly a HMM with $m = 10$ states. We also observe that the MLS policy is often the worst policy. This is to be expected, as this policy essentially does not take into account the uncertainty. Perhaps more surprising is that the POMDP policy does not consistently outperform the QMDP policy.

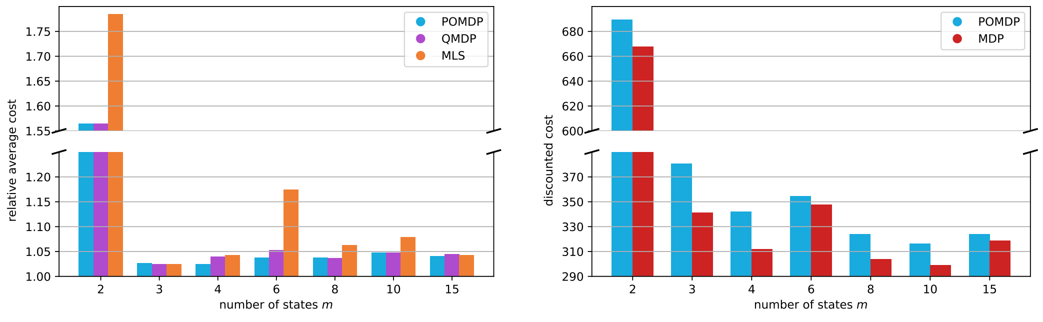
In Figure 5.1b, we see that the discounted cost of the POMDP model is consistently higher than the discounted cost of the MDP model. This does not come as a surprise, as the MDP model assumes that the states of the Markov chain are observed, which is not the case in practice. The POMDP takes this uncertainty into account and therefore shows a higher discounted cost. We also observe that Figure 5.1a and Figure 5.1b are not in synch; Figure 5.1b suggests that $m = 8$ or $m = 10$ are the best choices, whereas Figure 5.1a points to $m = 3$ or $m = 4$. Also the AIC and AICc identify a different model as better ($m = 8$ and $m = 6$ respectively, see Figure 5.2a).

Let us now consider simulation 2, for which the results are presented in Figure 5.3 and Table 5.2. The best policy is achieved by choosing $m = 10$, in which case the average cost per time unit is only 1.4% higher than the average cost of the optimal threshold policy. Further increasing the number of states to $m = 15$, leads to a clearly worse policy. Indeed, the POMDP policy for $m = 15$ has an excess cost (that is, the additional average cost per time unit compared to the optimal threshold policy) that is almost three times higher compared to $m = 10$. This pattern is not reflected in the corresponding discounted costs however, shown in Figure 5.3b.

The results for Simulation 3 are presented in Figure 5.5 and Table 5.3. Interestingly enough, increasing the number of states of the HMM beyond $m = 3$ seems to make little or no difference. Also, the various policies yield more or less the seem average cost; the average cost is consistently around 14% above the average cost of the optimal threshold. Note that this gap is significantly larger than the gap found for Simulations 1 and 2. This is because a large part of the degradation process is entirely unobserved in Simulation 3. Hence, a worse performance is to be unexpected.

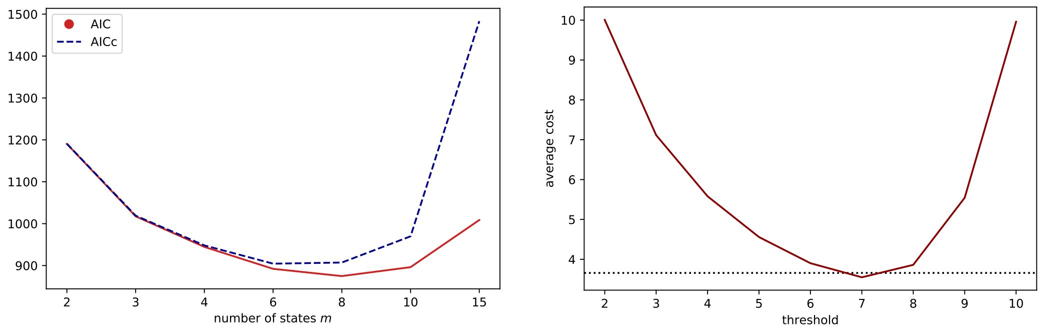
For Simulation 4, see Figure 5.7 and Table 5.4, we find that the optimal number of states is $m = 10$, with a gap of 11.4% compared to the optimal threshold policy (keep in mind that, again, a large part of the degradation process is not observed). Choosing $m = 3$ leads to a very similar performance (a gap of 11.7%).

Lastly, we consider Simulation 5, for which the monitoring data consists of 2 features. The result are presented in Figure 5.9 and Table 5.5. We observe that choosing $m = 10$ states leads to the best policy, which has an average cost that is 3.3% higher than the average cost of the optimal threshold policy. Such a small gap is slightly surprising, since again a major part of the degradation process is not observed. We also note that the MLS policy generally performs the worst (aside from $m = 6$) and that the there is no major or consistent difference between the POMDP policies and the QMDP policies. Furthermore, we see that, in contrast to the results of the other simulations, Figures 5.9b and 5.10a successfully identify the best choice.



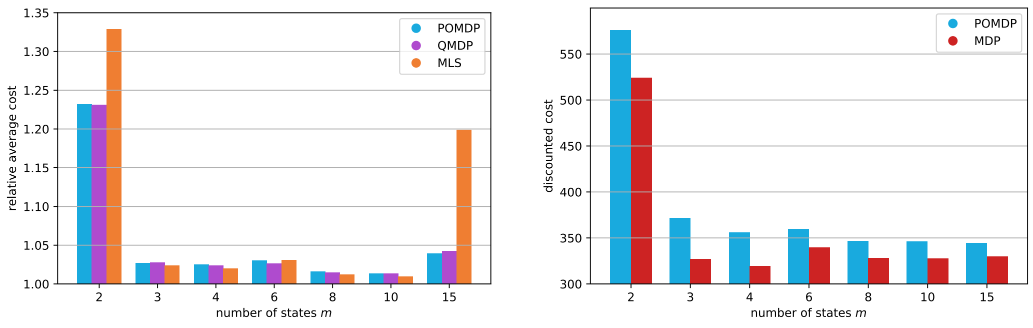
(A) (B)

FIGURE 5.1: Results of Simulation 1 for an increasing number of states of the HMM. (A) The relative average cost of the POMDP, QMDP and MLS policies; (B) The discounted cost of the POMDP and MDP value functions.



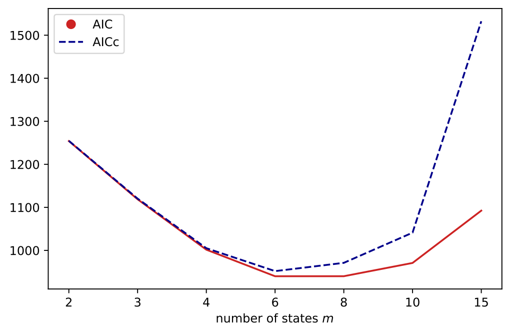
(A) (B)

FIGURE 5.2: Results of Simulation 1 for an increasing number of states of the HMM. (A) The AIC and AICc of the HMMs; (B) The average cost of various threshold policies (with respect to the degradation space) and the average cost of the best POMDP policy (dotted line).

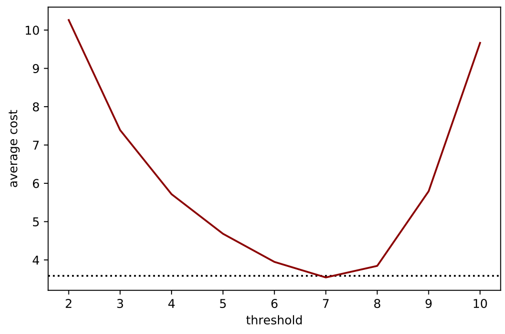


(A) (B)

FIGURE 5.3: Results of Simulation 2 for an increasing number of states of the HMM. (A) The average cost of the POMDP, QMDP and MLS policies; (B) The discounted cost of the POMDP and MDP value functions.

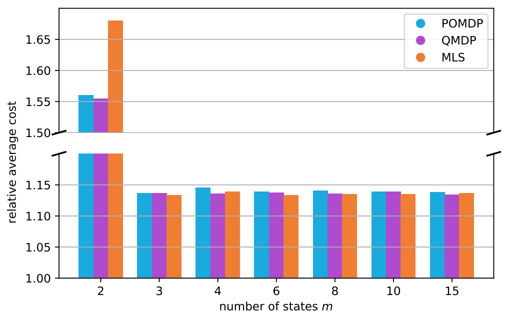


(A)

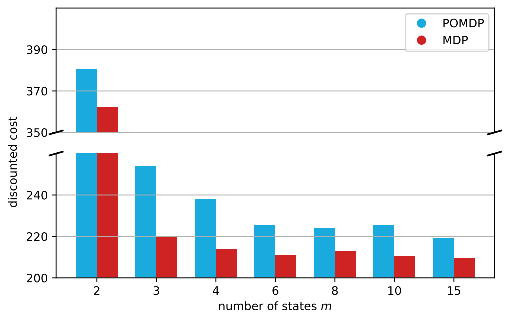


(B)

FIGURE 5.4: Results of Simulation 2 for an increasing number of states of the HMM. (A) The AIC and AICc of the HMMs; (B) The average cost of various threshold policies (with respect to the degradation space) and the average cost of the best POMDP policy (dotted line).

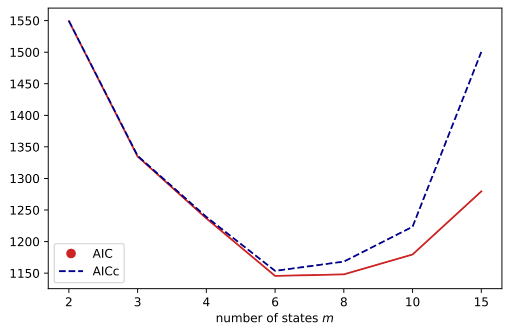


(A)

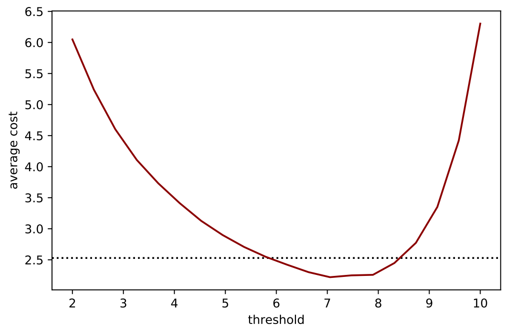


(B)

FIGURE 5.5: Results of Simulation 3 for an increasing number of states of the HMM. (A) The average cost of the POMDP, QMDP and MLS policies; (B) The discounted cost of the POMDP and MDP value functions.



(A)



(B)

FIGURE 5.6: Results of Simulation 3 for an increasing number of states of the HMM. (A) The AIC and AICc of the HMMs; (B) The average cost of various threshold policies (with respect to the degradation space) and the average cost of the best POMDP policy (dotted line).

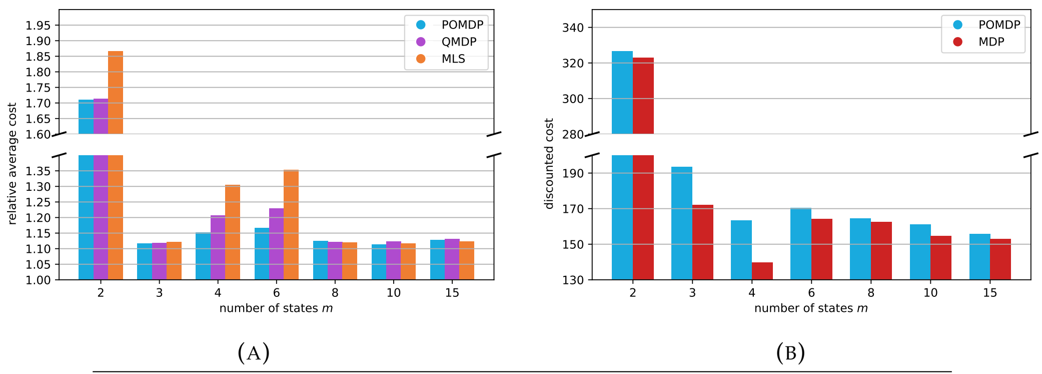


FIGURE 5.7: Results of Simulation 4 for an increasing number of states of the HMM. (A) The average cost of the POMDP, QMDP and MLS policies; (B) The discounted cost of the POMDP and MDP value functions.

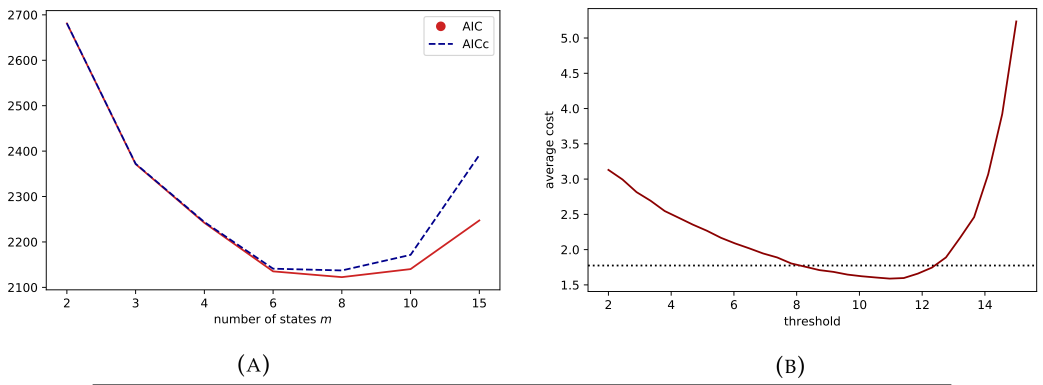


FIGURE 5.8: Results of Simulation 4 for an increasing number of states of the HMM. (A) The AIC and AICc of the HMMs; (B) The average cost of various threshold policies (with respect to the degradation space) and the average cost of the best POMDP policy (dotted line).

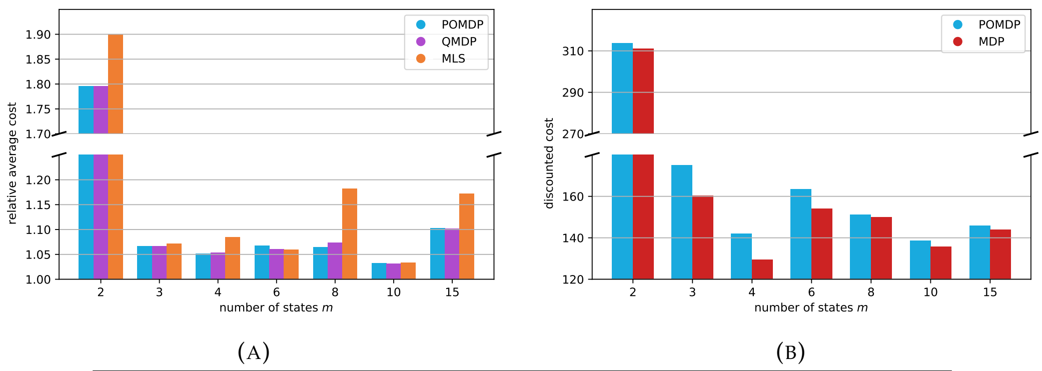
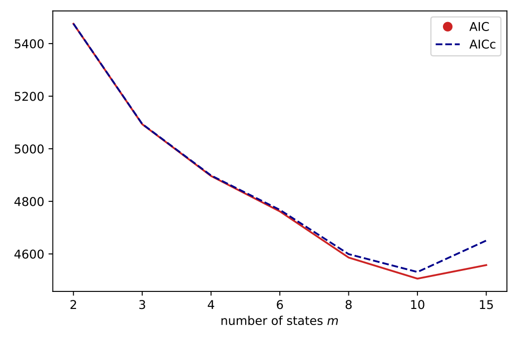
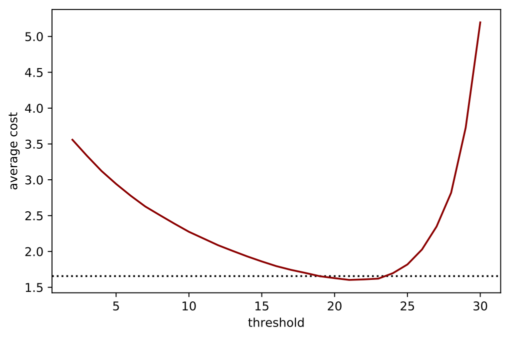


FIGURE 5.9: Results of Simulation 5 for an increasing number of states of the HMM. (A) The average cost of the POMDP, QMDP and MLS policies; (B) The discounted cost of the POMDP and MDP value functions.



(A)



(B)

FIGURE 5.10: Results of Simulation 5 for an increasing number of states of the HMM. (A) The AIC and AICc of the HMMs; (B) The average cost of various threshold policies (with respect to the degradation space) and the average cost of the best POMDP policy (dotted line).

TABLE 5.1: Results of Simulation 1 for an increasing number of states of the HMM. The relative average cost of the POMDP, QMDP and MLS policies.

	POMDP	QMDP	MLS
$m = 2$	1.564	1.565	1.785
$m = 3$	1.027	1.025	1.025
$m = 4$	1.025	1.040	1.043
$m = 6$	1.037	1.053	1.175
$m = 8$	1.038	1.037	1.063
$m = 10$	1.048	1.048	1.079
$m = 15$	1.040	1.045	1.043

TABLE 5.2: Results of Simulation 2 for an increasing number of states of the HMM. The relative average cost of the POMDP, QMDP and MLS policies.

	POMDP	QMDP	MLS
$m = 2$	1.232	1.231	1.329
$m = 3$	1.027	1.028	1.024
$m = 4$	1.025	1.024	1.020
$m = 6$	1.031	1.027	1.031
$m = 8$	1.016	1.015	1.012
$m = 10$	1.014	1.014	1.009
$m = 15$	1.039	1.043	1.199

TABLE 5.3: Results of Simulation 3 for an increasing number of states of the HMM. The relative average cost of the POMDP, QMDP and MLS policies.

	POMDP	QMDP	MLS
$m = 2$	1.561	1.555	1.680
$m = 3$	1.137	1.136	1.134
$m = 4$	1.145	1.136	1.139
$m = 6$	1.139	1.137	1.134
$m = 8$	1.141	1.136	1.135
$m = 10$	1.139	1.139	1.135
$m = 15$	1.138	1.134	1.137

TABLE 5.4: Results of Simulation 4 for an increasing number of states of the HMM. The relative average cost of the POMDP, QMDP and MLS policies

	POMDP	QMDP	MLS
$m = 2$	1.710	1.713	1.867
$m = 3$	1.117	1.118	1.121
$m = 4$	1.153	1.207	1.305
$m = 6$	1.167	1.230	1.354
$m = 8$	1.124	1.122	1.119
$m = 10$	1.114	1.123	1.116
$m = 15$	1.129	1.132	1.123

TABLE 5.5: Results of Simulation 5 for an increasing number of states of the HMM. The relative average cost of the POMDP, QMDP and MLS policies

	POMDP	QMDP	MLS
$m = 2$	1.796	1.795	1.900
$m = 3$	1.066	1.067	1.071
$m = 4$	1.051	1.054	1.085
$m = 6$	1.068	1.060	1.060
$m = 8$	1.065	1.074	1.182
$m = 10$	1.033	1.031	1.034
$m = 15$	1.103	1.102	1.173

5.2 Discussion

In Section 5.1, we consistently observed that the simplest HMM (i.e. $m = 2$), is not a good choice. Increasing the number of states (to e.g. $m = 3$), has shown to produce significantly better policies. This result is to be expected, since the parameters of the HMM are estimated solely based on the likelihood of the monitoring data/training data, thus not taking into account the decision-making process. As a result, policies with $m = 2$ might do preventive maintenance too often, resulting in a higher average cost. This argument could also explain why $m = 3$ leads to generally decent performance; perhaps choosing $m = 3$ provides a sufficient granularity to isolate the 'ideal maintenance region'.

Also, we have seen that making the model more complex could produce suboptimal policies, as increasing the number of states led to an increase of the average cost on multiple occasions. It may be that estimating a large number of parameters (say $m = 15$), leads to an overfitted model, which causes the policy to generalize poorly. Another reason why increasing the number of states of the HMM sometimes increased the average cost, may be that the EM algorithm converged to a suboptimal stationary point. Indeed, the search space of the EM algorithm depends on the number of states of the HMM; due to the structure of the likelihood, it could be that the EM algorithm for, say, $m = 6$ states converges to a stationary point that is worse (in terms of the average cost of the resulting policies) than the stationary point it converges to for $m = 3$ states.

Naturally, the MLS policy, in general, led to the highest average cost. Only a few times, the MLS policy outperformed the QMDP policy and never by a large margin. However, no major or consistent difference was observed between the POMDP and the QMDP policy. This is interesting and a bit surprising, since in [34] the POMDP policy clearly outperforms the QMDP policy in a maintenance setting.[1] The key observation to understanding the strong performance of the QMDP policy in our setting is that it is a heuristic which assumes that the uncertainty will disappear after executing the next action (see e.g. [11]). As a result, it will not choose informative actions (such as 'inspect the component'). Since these actions are absent from our model, this might explain why the QMDP heuristic performs so well.

It could also be that the POMDP was not always solved to optimality. For example, it might be that the proxy belief space was not a sufficient representation of the belief space one would encounter when executing the policy in practice, or it could be that the PERSEUS algorithm had not yet converged. To check whether the algorithm has converged, one could monitor the discounted cost and the number of α-vectors (see Figure 5.11). In Figure 5.11a, we see that the discounted cost decreased rapidly at first and then slowly converged. Also, the number of α-vectors is rather low in the beginning (see Figure 5.11b). This is because during the first iterations a single backup usually improves the value function for many, if not all, beliefs in the proxy belief space. As more iterations are performed, the number of α-vectors gently increases, indicating that the value function is gradually improving.

Let us now consider the POMDP policy for Simulation 5 with $m = 3$. In this case, the component, while operational, can be in one of three states. As a result, the belief space $\mathcal{B}$ can be presented in a 2-dimensional way (see e.g. [25]), as shown in Figure 5.12. We see that the policy chooses 'do maintenance' in state 3 and 'do nothing'

[1]The model studied in [34] is slightly different though. For example, it accounts for a total of three preventive maintenance actions: 'minor repair', 'major repair' and 'preventive replacement'.

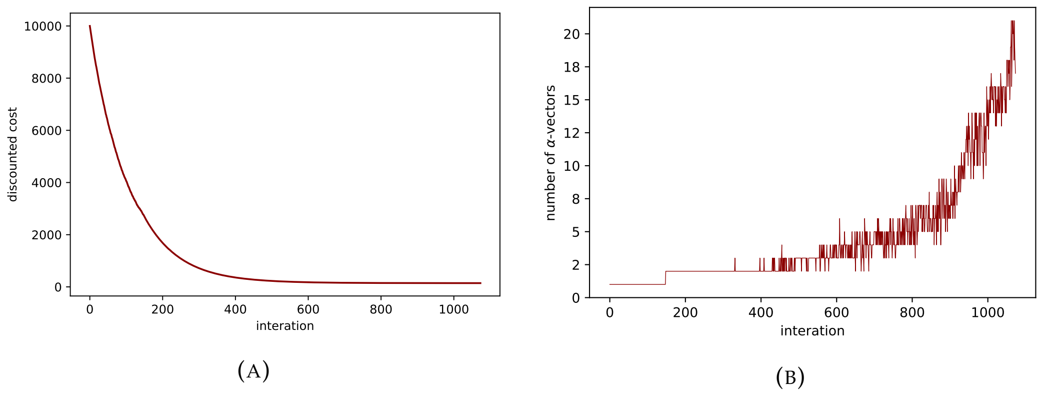


FIGURE 5.11: The discounted cost (A) and the number of α-vectors (B) for each iteration of the PERSEUS algorithm when solving the POMDP for Simulation 5 with $m = 10$.

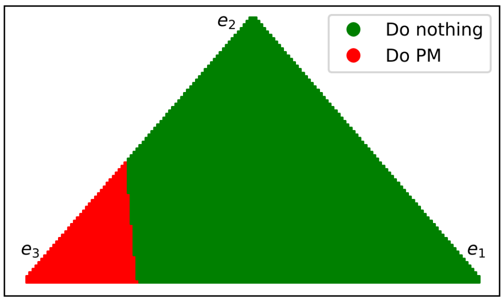


FIGURE 5.12: The belief space when the component is operational (i.e. $b \in \mathcal{B}$ such that $b\,(m+1) = 0$) for the POMDP policy of Simulation 5 with $m = 3$.

in state 1 and 2. Additionally, we note that both the 'do maintenance' and the 'do nothing' region, restricted to this subset of the belief space, seem convex. In general, one can easily prove that the 'do maintenance' region is convex (see Appendix A.2). However, the 'do nothing' region need not be convex. Nevertheless, in our experience the obtained POMDP policies are generally quite simple in structure; in some cases the policy is a hyperplane (as is the case in Figure 5.12) and often the policy can be described by a small set of α-vectors (even the POMDP policy of Simulation 5 with $m = 10$ contains only 17 relevant α-vectors, see Figure 5.11b).

We have studied four different criteria that could possibly identify the optimal number of states of the HMM, specifically: the discounted cost of the POMDP/MDP and the AIC/AICc. Figures 5.2a, 5.4a, 5.6a and 5.8a show a very similar shape; the AIC and AICc both consistently point to $m = 6$ or $m = 8$ as the optimal number of states. On the hand, the discounted costs (see Figures 5.1b, 5.3b, 5.5b and 5.7b), do not show a consistent pattern. Nevertheless, the simulation results show that none of these criteria identified the best number of states. Only for Simulation 5 have the AIC, AICc and the discounted cost of the POMDP identified the optimal choice.

Lastly, we note that the POMDP policies have shown to produce results that, for $m \geq 3$, were consistently less than 5% higher than the optimal threshold policy in case all parts of the degradation process were observed (that is Simulations 1 and 2). This is not trivial, as can be seen by studying Figures 5.2b and 5.4b. In case only a part of the degradation process was observed, the resulting average cost was still less 15% higher than the average cost of the optimal threshold policy. Keep in mind that since we do not observe a major part of the degradation process, the optimal threshold policy (as depicted in Figures 5.6b and 5.8b) is likely not attainable in practice. Finally, for Simulation 5, which has both a more complex structure and unobserved parts, the average cost of the POMDP policy was at most 10% higher. The non-triviality of this performance can be understood by studying Figure 5.10b. In addition, we notice that choosing $m = 3$, despite the simplicity of the resulting model, seems to produce decent results for each simulation.

Chapter 6

Conclusions and outlook

In this study, we investigated the use of multivariate Gaussian HMMs for CBM policies. We used various simulated degradation processes (e.g. Markovian degradation, non-Markovian degradation and unobserved features) to generate monitoring data. We described the degradation process of the component by a latent Markov chain that governs the monitoring data, i.e. the monitoring data contains observations generated by a multivariate Gaussian distribution, whose parameters depend on the state of the underlying Markov chain. The parameters of this HMM were estimated using the well-known Baum-Welch algorithm. Subsequently, a maintenance policy was obtained by formulating and solving the corresponding POMDP, using approximate point-based value iteration. The performance of the resulting policy, as well as several POMDP-based heuristics, was tested in a statistically identical environment and compared with the optimal threshold policy.

We have found that, despite their simplistic nature, HMMs can be used to derive policies that perform reasonable compared to the optimal threshold policy in both a Markovian and non-Markovian setting, despite the noisy measurements and unobserved features. However, the chosen number of states of the Markov chain does affect the average cost of the resulting policy. We presented two ways to determine the number of states; using the discounted cost of the POMDP (or the MDP) and using the AICc (or the AIC). However, in most cases, both metrics pointed to a suboptimal number of states. Nevertheless, we observed that the simplest model ($m = 2$) leads to suboptimal policies and that in most cases choosing a small model (say $m = 3$) produces decent results.

Furthermore, we observed no major or consistent difference between the POMDP policy and the QMDP policy. We argued that this may be due to the simplicity of the model under consideration. This would mean that in this setting one could simply solve the corresponding MDP and subsequently execute the QMDP policy, without a loss of performance. However, in more advanced settings (in particular when the action space is extended with exploration actions), the QMDP policy will likely perform significantly worse than the POMDP policy.

Our study also has some limitations. For example, we have only studied value data. Whereas, in practice, the monitoring data may also include waveform data and multidimensional data. We also did not include event data and simulated only perfect-to-failure cycles (whereas including left-truncated and/or right-censored data might be more realistic). Additionally, we recognize that the construction of the POMDP can be improved. For example, the current formulation requires a discretization of the observation space, which imposes a trade-off between accuracy and speed. In our simulations this is likely not problematic, but for high-dimensional monitoring data, this issue could be more apparent. As an alternative, one could explore the possibility of allowing a continuous multidimensional observation space (see e.g. [20]). Also, we note that our current solution method lacks a way to verify whether the POMDP has converged to optimality. Since point-based value iteration algorithms might suddenly improve after seemingly having reached a plateau for some time (see e.g. [33]), implementing an upper bound to the optimal value function might be beneficial. Lastly, we emphasize that we solved the POMDP based on the discounted reward, whereas our end goal was to minimize the average cost per time unit. This choice was motivated by the technical difficulty associated with solving an average cost POMDP (see e.g. [25]) and compensated for by choosing a discount factor of 0.99.

There are several interesting extensions and topics for future research. For example, in this work we studied a HMM with an upper triangular transition matrix, however, many different types of HMMs exists (see e.g. [31]). Seeing whether similar, or better, performance can be obtained using different types of HMMs, might make HMMs a more versatile tool. Additionally, one may want to explore different techniques to estimate the parameters of the HMM. In particular, techniques that allow for online learning of the parameters may be interesting, as this allows the model to adapt to newly-acquired training data.

Another interesting extension might be in the direction of semi-Markov models. That is, instead of a latent Markov chain, the degradation process is described by a stochastic process that is only Markovian at transition times (the time between transitions is random). This could lead to much more realistic models, as it is well-known that the Markovian assumptions is not always accurate. However, a partially observable semi-Markov decision process (see e.g. [43]), leads to complex solution procedures. Nevertheless, since the QMDP heuristic showed such promising results in our study, the extension to semi-Markov models might be less troublesome in a similar setting to ours.

Appendix A

Additional proofs

A.1 The Banach Fixed-Point theorem

Consider the supremum norm $\|\cdot\|_\infty$, defined as:

$$\|\cdot\|_\infty := \sup_{b\in\mathcal{B}} |v(b) - \tilde{v}(b)|\,. \tag{A.1}$$

Lemma 2. *The normed space $(U, \|\cdot\|_\infty)$ is a Banach space*

Proof. Let $\{v_n\}$ be a Cauchy sequence in U, i.e.

$$\forall_{\epsilon>0}\exists_{N>0} : \|v_n - v_m\|_\infty < \epsilon \quad \forall_{n,m>N}. \tag{A.2}$$

We show that the Cauchy sequence $\{v_n\}_{n=0}^{\infty} =: \{v_n\}$ contains a limit point $v \in U$.

(i) First, we show that $\{v_n\}$ converges uniformly to v on $\mathcal{B}$. Let $b \in \mathcal{B}$ and consider

$$\begin{aligned}|v_n(b) - v(b)| &= |v_n(b) - \lim_{m\to\infty} v_m(b)| \\ &= \lim_{m\to\infty} |v_n(b) - v_m(b)| < \epsilon.\end{aligned} \tag{A.3}$$

The last statement follows from the observation that

$$|v_n(b) - v_m(b)| \leq \|v_n - v_m\|_\infty < \epsilon \quad \forall_{b\in\mathcal{B}}, \tag{A.4}$$

which is an immediate consequence of $\{v_n\}$ being a Cauchy sequence.

From Equation (A.3) it follows that

$$\|v_n - v\|_\infty < \epsilon, \tag{A.5}$$

which implies that $\{v_n\}$ converges uniformly to v on $\mathcal{B}$.

(ii) Next, we show that the resulting limit point v is a real-valued function. To this end, we note that for all $b \in \mathcal{B}$, the sequence $\{v_n(b)\}$ is a Cauchy sequence in $\mathbb{R}$. Since $v(b) = \lim_{n\to\infty} v_n(b)$ and $\mathbb{R}$ is complete, it follows that $v(b)$ is a real-valued function for all $b \in \mathcal{B}$.

(iii) Lastly, we show that v is bounded. Using the triangle inequality and the definition of the supremum norm, we find that for all $b \in \mathcal{B}$

$$\begin{aligned} |v(b)| &= |v(b) - v_{N+1}(b) + v_{N+1}(b)| \\ &\leq |v(b) - v_{N+1}(b)| + |v_{N+1}(b)| \\ &\leq \|v - v_{N+1}\|_\infty + \|v_{N+1}\|_\infty . \end{aligned} \tag{A.6}$$

Since $\{v_n\}$ is Cauchy sequence, $\|v_{N+1}\|_\infty < c$ (for some $c \in \mathbb{R}$). Furthermore,

$$\begin{aligned} \|v - v_{N+1}\|_\infty &= \sup_{b\in\mathcal{B}} |v(b) - v_{N+1}(b)| \\ &= \sup_{b\in\mathcal{B}} \left| \lim_{n\to\infty} v_n(b) - v_{N+1}(b) \right| \\ &= \sup_{b\in\mathcal{B}} \lim_{n\to\infty} |v_n(b) - v_{N+1}(b)| < \epsilon. \end{aligned} \tag{A.7}$$

The last statement is a consequence of Equation (A.4).

From Equation (A.6) we now conclude that for all $b \in \mathcal{B}$, it follows that $|v(b)| \leq c + \epsilon$. Which shows that v is bounded.

We conclude that $\{v_n\}$ converges uniformly to a limit point v, which is a bounded real-valued function. □

Lemma 3 ([37]). *The dynamic programming operator D, as defined in Equation (4.20) is a contraction mapping, i.e.*

$$\exists_{\lambda\in[0,1)} : \|Dv - D\tilde{v}\|_\infty \leq \lambda \|v - \tilde{v}\|_\infty \quad \forall_{v,\tilde{v}\in U}. \tag{A.8}$$

Proof. Fix $b \in \mathcal{B}$ and suppose that $Dv(b) \geq D\tilde{v}(b)$. We define

$$a^* := \arg\min_{a\in\mathcal{A}} \left\{ R_B(b,a) + \gamma \sum_{o\in\hat{\mathcal{S}}_O} v(T_B(b,a,o))\, \Omega^a_b(o) \right\}. \tag{A.9}$$

It follows that

$$\begin{aligned} 0 &\leq Dv(b) - D\tilde{v}(b) \\ &\leq R_B(b,a) + \gamma \textstyle\sum_{o\in\hat{\mathcal{S}}_O} v(T_B(b,a,o))\, \Omega^a_b(o) \\ &\qquad - R_B(b,a) + \gamma \textstyle\sum_{o\in\hat{\mathcal{S}}_O} \tilde{v}(T_B(b,a,o))\, \Omega^a_b(o) \\ &= \gamma \textstyle\sum_{o\in\hat{\mathcal{S}}_O} (v(T_B(b,a,o)) - \tilde{v}(T_B(b,a,o)))\, \Omega^a_b(o) \\ &\leq \gamma \textstyle\sum_{o\in\hat{\mathcal{S}}_O} \|v - \tilde{v}\|_\infty \Omega^a_b(o) \\ &= \gamma \|v - \tilde{v}\|_\infty . \end{aligned} \tag{A.10}$$

Similarly, it can be shown that if $Dv(b) \leq D\tilde{v}(b)$, it holds that

$$0 \leq D\tilde{v}(b) - Dv(b) \leq \gamma \|v - \tilde{v}\|_\infty . \tag{A.11}$$

Hence,

$$|Dv(b) - D\tilde{v}(b)| \leq \gamma \|v - \tilde{v}\|_\infty \quad \forall_{b\in\mathcal{B}}, \tag{A.12}$$

which implies that

$$\|Dv - D\tilde{v}\|_\infty \leq \gamma \|v - \tilde{v}\|_\infty . \tag{A.13}$$

□

A.2 Convexity of the 'do maintenance' region

Lemma 4 (Convexity of the 'do maintenance' region restricted to $\tilde{\mathcal{B}}$)**.** *The maintenance region, restricted to $\tilde{\mathcal{B}}$, defined as $\tilde{\mathcal{B}} := \{b \in \mathcal{B} : b\,(m+1) \in \{0,1\}\}$, is a convex set.*

Proof. Let e_i denote the belief state where we are in state i with probability 1.

$$\begin{aligned}\Omega_b^1\,(o) &= \sum\nolimits_{z,z' \in \mathcal{S}_Z} \Omega\,(z',o)\,T_M\,(1,z,z')\,b\,(z)\\ &= \sum\nolimits_{s,s' \in \mathcal{S}_Z} \Omega\,(z',o)\,T_M\,(0,1,z')\,b\,(z)\\ &= \sum\nolimits_{z' \in \mathcal{S}_Z} \Omega\,(z',o)\,T_M\,(0,1,z')\\ &= \mathbb{P}\,(O_{t+1} = o | B_{t+1} = 1, A_t = 0)\\ &= \Omega_{e_1}^0\,(o)\,.\end{aligned} \tag{A.14}$$

Consider $b' := T_B\,(b,1,o)$, then

$$\begin{aligned}b'\,(z) &= \frac{\Omega\,(z',o)}{\Omega_b^a\,(o)} \sum\nolimits_{z \in \mathcal{S}_Z} T_M\,(1,z,z')\,b\,(z)\\ &= \frac{\Omega\,(z',o)}{\Omega_{e_1}^0\,(o)} \sum\nolimits_{z \in \mathcal{S}_Z} T_M\,(0,1,z')\,b\,(z)\\ &= \frac{\Omega\,(z',o)}{\Omega_{e_1}^0\,(o)} T_M\,(0,\cdot,z') \cdot e_1\end{aligned} \tag{A.15}$$

We conclude that $T_B\,(b,1,o) = T_B\,(e_1,0,o)$.

Hence for $b \neq \mathcal{F}_Z$

$$\begin{aligned}q^*\,(b,1) &= c_{\text{pm}} + \gamma \sum\nolimits_{o \in \hat{\mathcal{S}}_O} v^*\,(T_B\,(b,1,o))\,\Omega_b^a\,(o)\\ &= c_{\text{pm}} + \gamma \sum\nolimits_{o \in \hat{\mathcal{S}}_O} v^*\,(T_B\,(e_1,0,o))\,\Omega_b^a\,(o)\\ &= c_{\text{pm}} + q^*\,(e_1,0)\,.\end{aligned} \tag{A.16}$$

and

$$q^*\,(\mathcal{F}_Z,1) = c_{\text{cm}} + q^*\,(e_1,0)\,. \tag{A.17}$$

Let us denote the 'do maintenance' region as $\mathcal{A}_1 \subseteq \mathcal{B}$ and consider $b_1, b_2 \in \mathcal{A}_1$, we show that for $\lambda \in [0,1]$ it holds that $\lambda b_1 + (1-\lambda)\,b_2 \in \mathcal{A}_1$:

$$\begin{aligned}v^*\,(\lambda b_1 + (1-\lambda)\,b_2) &\geq \lambda v^*\,(b_1) + (1-\lambda)\,v^*\,(b_2)\\ &= \lambda q^*\,(b_1,1) + (1-\lambda)\,q^*\,(b_2,1)\\ &= \lambda\,(c_{\text{pm}} + q^*\,(e_1,0)) + (1-\lambda)\,(c_{\text{pm}} + q^*\,(e_1,0))\\ &= c_{\text{pm}} + q^*\,(e_1,0)\end{aligned} \tag{A.18}$$

The first statement is true because of the concavity of v^* (see e.g. [25]). The second statement holds because $b_1, b_2 \in \mathcal{A}_1$. The third statement follows from the derivation above. Now since $v^*\,(b) > c_{\text{pm}} + q^*\,(e_1,0)$ is impossible, it follows that $v^*\,(b) = c_{\text{pm}} + q^*\,(e_1,0)$, which means that $\lambda b_1 + (1-\lambda)\,b_2 \in \mathcal{A}_1$. □

Bibliography

[1] P. Baruah and R. B. Chinnam. “HMMs for diagnostics and prognostics in machining processes”. In: *International Journal of Production Research* 43.6 (2005), pp. 1275–1293. DOI: 10.1080/00207540412331327727.

[2] L.E. Baum and T. Petrie. “Statistical Inference for Probabilistic Functions of Finite State Markov Chains”. In: *The Annals of Mathematical Statistics* 37.6 (1966), pp. 1554–1563. DOI: 10.1214/aoms/1177699147.

[3] Leonard E. Baum et al. “A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains”. In: *Institute of Mathematical Statistics Stable* 41.1 (1970), pp. 164–171. DOI: 10.1214/aoms/1177697196.

[4] Richard Bellman. *Dynamic Programming*. Princeton, New Jersey: Princeton University Press, 1957.

[5] Jeff A. Bilmes. “A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models”. In: (1998).

[6] Carey Bunks, Dan McCarthy, and Tarik Al-Ani. “Condition-based maintenance of machines using hidden Markov models”. In: *Mechanical Systems and Signal Processing* 14.4 (2000), pp. 597–612. DOI: 10.1006/mssp.2000.1309.

[7] Kenneth P. Burnham and David R. Anderson. *Model Selection and Inference: A Practical Information-Theoretic Approach*. Springer-Verlag New York, 1998. DOI: 10.1007/978-1-4757-2917-7.

[8] Olivier Cappe, Eric Moulines, and Tobias Ryden. *Inference in Hidden Markov Models*. 1st ed. Springer-Verlag New York, 2005. DOI: 10.1007/0-387-28982-8.

[9] Francesco Cartella et al. “Hidden Semi-Markov Models for Predictive Maintenance”. In: *Mathematical Problems in Engineering* (2015). DOI: 10.1155/2015/278120.

[10] Anthony Cassandra, Michael L. Littman, and Nevin L. Zhang. “Incremental Pruning: A Simple, Fast, Exact Method for Partially Observable Markov Decision Processes”. In: *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence* (1997), pp. 54–61.

[11] Anthony R. Cassandra. “Exact and approximate algorithms for partially observable markov decision processes”. Doctoral thesis. Brown University, 1998.

[12] Christophe Couvreur. “The EM Algorithm: A Guided Tour”. In: *Computer Intensive Methods in Control and Signal Processing: The Curse of Dimensionality*. Ed. by K. Warwick and M. Kárný. Birkhäuser Boston, 1997, pp. 209–222. DOI: 10.1007/978-1-4612-1996-5_12.

[13] Rajashree Dash, Rajib Lochan Paramguru, and Rasmita Dash. “Comparative analysis of supervised and unsupervised discretization techniques”. In: (2011).

[14] A. P. Dempster, N. M. Laird, and D. B. Rubin. "Maximum Likelihood from Incomplete Data via the EM Algorithm". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 39.1 (1977), pp. 1–22. DOI: 10.1111/j.2517-6161.1977.tb01600.x.

[15] A. Drake. "Observation of a Markov Process Through a Noisy Channel". PhD thesis. Massachusetts Institute of Technology, 1962.

[16] E. O. Elliott. "Estimates of Error Rates for Codes on Burst-Noise Channels". In: *Bell System Technical Journal* 42.5 (1963), pp. 1977–1997. ISSN: 15387305. DOI: 10.1002/j.1538-7305.1963.tb00955.x.

[17] Omid Geramifard et al. "A Physically Segmented Hidden Markov Model Approach for Continuous Tool Condition Monitoring: Diagnostics and Prognostics". In: *IEEE Transactions on Industrial Informatics* 8.4 (2012), pp. 964–973. DOI: 10.1109/TII.2012.2205583.

[18] E. N. Gilbert. "Capacity of a Burst-Noise Channel". In: *Bell System Technical Journal* 39.5 (1960), pp. 1253–1265. DOI: 10.1002/j.1538-7305.1960.tb03959.x.

[19] Rafael Gouriveau, Kamal Medjaher, and Noureddine Zerhouni. *From Prognostics and Health Systems Management to Predictive Maintenance 1: Monitoring and Prognostics*. John Wiley and Sons, 2016. DOI: 10.1002/9781119371052.

[20] Jesse Hoey and Pascal Poupart. "Solving POMDPs with continuous or large discrete observation spaces". In: *Proceedings of the 19th International Joint Conference on Artificial Intelligence*. 2005, pp. 1332–1338.

[21] Andrew K.S. Jardine, Daming Lin, and Dragan Banjevic. "A review on machinery diagnostics and prognostics implementing condition-based maintenance". In: *Mechanical Systems and Signal Processing* 20.7 (2006), pp. 1483–1510. ISSN: 08883270. DOI: 10.1016/j.ymssp.2005.09.012.

[22] J. L. W. V. Jensen. "Sur les fonctions convexes et les inégualités entre les valeurs Moyennes". In: *Acta Mathematica* 30.1 (1906), pp. 175–193. DOI: 10.1007/BF02418571.

[23] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. "Planning and acting in partially observable stochastic domains". In: *Artificial Intelligence* 101.1-2 (1998), pp. 99–134. ISSN: 00043702. DOI: 10.1016/s0004-3702(98)00023-x.

[24] Wael Khreich et al. "A survey of techniques for incremental learning of HMM parameters". In: *Information Sciences* 197 (2012), pp. 105–130. ISSN: 00200255. DOI: 10.1016/j.ins.2012.02.017. URL: http://dx.doi.org/10.1016/j.ins.2012.02.017.

[25] Vikram Krishnamurthy. *Partially Observed Markov Decision Processes: From Filtering to Controlled Sensing*. Cambridge University Press, 2016. DOI: 10.1017/cbo9781316471104.

[26] Yaguo Lei et al. "Machinery health prognostics: A systematic review from data acquisition to RUL prediction". In: *Mechanical Systems and Signal Processing* 104 (2018), pp. 799–834. ISSN: 10961216. DOI: 10.1016/j.ymssp.2017.11.016.

[27] Michael L. Littman, Anthony R. Cassandra, and Leslie Pack Kaelbling. "Learning policies for partially observable environments: Scaling up". In: *Machine Learning Proceedings 1995*. Elsevier, 1995, pp. 362–370. DOI: 10.1016/b978-1-55860-377-6.50052-9.

[28] Geoffrey J. McLachlan and Thriyambakam Krishnan. *The EM Algorithm and Extensions*. 2nd. John Wiley & Sons, 2008. ISBN: 9780471201700. DOI: 10.2307/1271189.

[29] R. Keith Mobley. *An Introduction to Preventive Maintenance*. 2nd. Elsevier, 2002. ISBN: 9780750675314.

[30] Martin Mundhenk et al. "Complexity of Finite-Horizon Markov Decision Process Problems". In: *Journal of the ACM* 47.4 (2000), pp. 681–720. DOI: 10.1145/347476.347480.

[31] Kevin Murphy. "Dynamic Bayesian Networks: Representation, Inference and Learning". PhD Thesis. University of California, Berkeley, 2002.

[32] R.M. Neal and G.E. Hinton. "A View of the Em Algorithm that Justifies Incremental, Sparse, and other Variants". In: *Learning in Graphical Models*. Ed. by M.I. Jordan. Dordrecht: Springer, 1998, pp. 355–368. DOI: 10.1007/978-94-011-5014-9_12.

[33] K. G. Papakonstantinou and M. Shinozuka. "Planning structural inspection and maintenance policies via dynamic programming and Markov processes. Part I: Theory". In: *Reliability Engineering and System Safety* 130 (2014), pp. 202–213. DOI: 10.1016/j.ress.2014.04.005.

[34] K. G. Papakonstantinou and M. Shinozuka. "Planning structural inspection and maintenance policies via dynamic programming and Markov processes. Part II: POMDP implementation". In: *Reliability Engineering and System Safety* 130 (2014), pp. 214–224. DOI: 10.1016/j.ress.2014.04.006.

[35] Ying Peng, Ming Dong, and Ming Jian Zuo. "Current status of machine prognostics in condition-based maintenance: A review". In: *International Journal of Advanced Manufacturing Technology* 50.1-4 (2010), pp. 297–313. ISSN: 02683768. DOI: 10.1007/s00170-009-2482-0.

[36] J. Pineau, G. Gordon, and S. Thrun. "Anytime Point-Based Approximations for Large POMDPs". In: *Journal of Artificial Intelligence Research* 27 (2006), pp. 335–380. DOI: 10.1613/jair.2078.

[37] M.L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. New York: John Wiley & Sons, 1994.

[38] Guy Shani, Joelle Pineau, and Robert Kaplow. "A survey of point-based POMDP solvers". In: *Autonomous Agents and Multi-Agent Systems* 27.1 (2012), pp. 1–51. DOI: 10.1007/s10458-012-9200-2.

[39] Satinder Singh, Tommi Jaakkola, and Michael Jordan. "Learning Without State-Estimation in Partially Observable Markovian Decision Processes". In: *Machine Learning Proceedings 1994*. Elsevier, 1994, pp. 284–292. DOI: 10.1016/b978-1-55860-335-6.50042-8.

[40] Richard D. Smallwood and Edward J. Sondik. "The Optimal Control of Partially Observable Markov Processes over a Finite Horizon". In: *Operations Research* 21.5 (1973), pp. 1071–1088. ISSN: 0030364X. DOI: 10.1287/opre.21.5.1071.

[41] Edward J. Sondik. "The Optimal Control of Partially Observable Markov Processes over the Infinite Horizon: Discounted Costs". In: *Operations Research* 26.2 (1978), pp. 282–304. DOI: 10.1287/opre.26.2.282.

[42] Matthijs T.J. Spaan and Nikos Vlassis. "Perseus: Randomized point-based value iteration for POMDPs". In: *Journal of Artificial Intelligence Research* 24 (2005), pp. 195–220. ISSN: 10769757. DOI: 10.1613/jair.1659.

[43] Rengarajan Srinivasan and Ajith Kumar Parlikad. "Semi-Markov Decision Process With Partial Information for Maintenance Decisions". In: *IEEE Transactions on Reliability* 63.4 (2014), pp. 891–898. ISSN: 00189529. DOI: 10.1109/TR.2014.2338811.

[44] R.L. Stratonovich. "Conditional Markov Processes". In: *Theory of Probability and Its Applications* 5.2 (2015), pp. 172–195. DOI: 10.1137/1105015.

[45] Rolf Turner. "Direct maximization of the likelihood of a hidden Markov model". In: *Computational Statistics and Data Analysis* 52.9 (2008), pp. 4147–4160. DOI: 10.1016/j.csda.2008.01.029.

[46] Nikos Vlassis and MTJ Spaan. "A fast point-based algorithm for POMDPs". In: *Benelearn 2004: Proceedings of the Annual Machine Learning Conference of Belgium and the Netherlands.* (2004), pp. 170–176.

[47] Qi Wang et al. "Failure Modeling and Maintenance Decision for GIS Equipment Subject to Degradation and Shocks". In: *IEEE Institute of Electrical and Electronics Engineers* 32.2 (2017), pp. 1079–1088. DOI: 10.1109/TPWRD.2017.2655010.

[48] Marco Wiering and Martijn Van Otterlo, eds. *Reinforcement Learning*. Springer Berlin Heidelberg, 2012. DOI: 10.1007/978-3-642-27645-3.

[49] Wireman T. *Benchmarking Best Practices for Maintenance, Reliability and Asset Management*. 3rd. Industrial Press, 2014.

[50] C. F. Jeff Wu. "On the Convergence Properties of the EM Algorithm". In: *The Annals of Statistics* 11.1 (1983), pp. 95–103. DOI: 10.1214/aos/1176346060.

[51] A. A. Yushkevich. "Blackwell optimal policies in a Markov decision process with a Borel state space". In: *ZOR Zeitschrift für Operations Research Mathematical Methods of Operations Research* 40.3 (1994), pp. 253–288. ISSN: 03409422. DOI: 10.1007/BF01432969.

[52] Xinyu Zhao et al. "Semi-supervised constrained hidden markov model using multiple sensors for remaining useful life prediction and optimal predictive maintenance". In: *Annual Conference of the PHM Society*. Vol. 11. 1. 2019. DOI: 10.36001/phmconf.2019.v11i1.851.