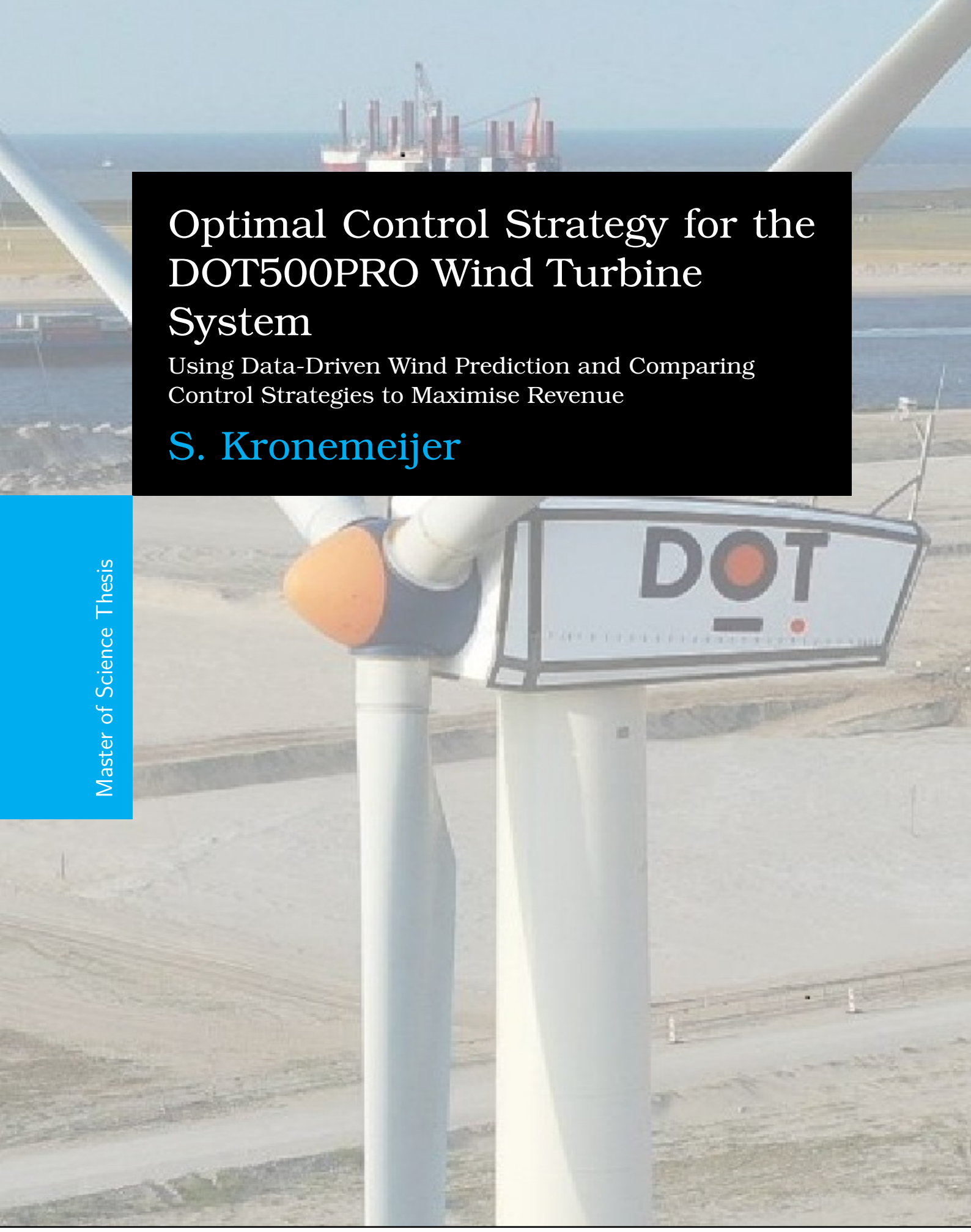


Optimal Control Strategy for the DOT500PRO Wind Turbine System

Using Data-Driven Wind Prediction and Comparing Control Strategies to Maximise Revenue

S. Kronemeijer

Master of Science Thesis

TU Delft Delft University of Technology

Delft Center for Systems and Control

Optimal Control Strategy for the DOT500PRO Wind Turbine System

Using Data-Driven Wind Prediction and Comparing Control Strategies to Maximise Revenue

Master of Science Thesis

For the degree of Master of Science in Systems and Control at Delft University of Technology

S. Kronemeijer

June 16, 2025

Faculty of Mechanical Engineering (ME) · Delft University of Technology

The work in this thesis was supported by the Delft Offshore Turbine B.V (DOT). Their cooperation is hereby gratefully acknowledged.

Abstract

This thesis aims to develop an optimal control strategy for the DOT 500kW Pilot Reverse Osmosis (DOT500PRO) turbine system. The system integrates a 500 kW wind turbine with a reverse osmosis (RO) module to produce freshwater. The primary goal is to maximise revenue generation by optimising the turbine's state transitions based on wind predictions.

The thesis begins with an analysis of the DOT500PRO and its state machine, identifying operational states, transitions, and constraints. A Markov model is used to model and predict wind speeds, which fits nicely with the Markov Decision Process (MDP) framework. The problem is formulated as an MDP, and multiple control strategies, including Threshold Control, Model Predictive Control (MPC), Stochastic Dynamic Programming (SDP), and Approximate Dynamic Programming (ADP), are evaluated.

MPC is found to be computationally intensive, making it less feasible for real-time control. SDP shows promising results, but is limited by the curse of dimensionality, restricting the use to higher-order models. ADP, which approximates SDP solutions, can offer a potential controller for higher-order models but requires further tuning and optimisation.

Simulations are conducted to compare the performance of these control strategies in several scenarios. While SDP demonstrates slight improvements over threshold control on the training dataset, its performance on different wind patterns is less consistent. The study concludes that while proactive control strategies such as SDP and ADP can offer improvements over reactive methods, their performance is dependent on the accuracy of wind predictions and the specific operational conditions.

Future work suggestions include refining the turbine and wind models, exploring adaptive control methods, and conducting real-life experiments to validate the control strategies, which are crucial for practical implementation and optimisation.

Table of Contents

List of Figures

Preface and acknowledgements

After nearly eight years of studying, this thesis is the final project that will complete my higher education. The result of 9 months of work, performed at the DOT company, lies before you. The process was long and sometimes tiring, but overall, the project was very enjoyable and I learned a lot. I am grateful for the opportunity to do my thesis at DOT and will look back at this time fondly. Having reached the end of my thesis, I would like to thank everyone who contributed to its completion.

First and foremost, I would like to thank my university supervisor Koty McAllister. I am truly grateful for our weekly meetings where I was able to discuss my progress and ask questions about everything I didn't yet understand. You made me feel at ease from the start, and I believe we had some great discussions. Thank you for patiently explaining the difference between SDP and ADP for what must have been at least 3 times. I am really glad to have such an understanding and friendly supervisor, who always made time for me.

Secondly, I want to thank the people at DOT, who have explained to me everything there is to explain about offshore wind, making me much more interested in that field. Special thanks to David and Casper. You have been a motivating force, making me feel at home and involving me in all of the exciting projects you guys are doing. I enjoyed your company very much and wish you all the best.

Lastly, I want to sincerely thank my girlfriend for her unwavering support throughout this journey. I also want to thank my family, roommates, and friends for all the support over the last months and also for making my years at the TU Delft so much fun.

Thank you all,
Sander Kronemeijer
Delft, June 2025

Chapter 1

Introduction

This introductory chapter provides the context and relevance of the study. In the first section, some background information on wind-powered desalination is given. Afterwards, the case of the DOT500PRO system is presented. Following this, the objective of the thesis is given. Lastly, the chapter concludes with an outline of the rest of the thesis.

1-1 Background

This section describes the problem of global water scarcity and the role of desalination technologies. It highlights the relevance of wind-powered desalination and provides a brief overview of the current state of this technology.

1-1-1 Water scarcity

The surface of the Earth is covered for about seventy percent by water [1], but still, half of the world population experiences severe water scarcity for at least part of the year [6]. Due to urbanisation and climate change, this number is projected to keep growing in the future [7]. The United Nations has been warning of a water crisis since 1977 [8] and has set *"Clean drinking water and sanitation"* as their sixth goal for sustainable development since 2015.

According to Igor Shiklomanov, only 2.5 percent of water on Earth is freshwater, and the majority is not easily attainable because it is trapped in glaciers or at the poles [1]. Currently, freshwater is primarily used in agriculture and is collected from rivers, lakes and aquifers. These water reserves are becoming increasingly scarce and are unevenly distributed across the world [9], resulting in a growing need for new solutions to support water demand. Desalination has become an essential part of these solutions.

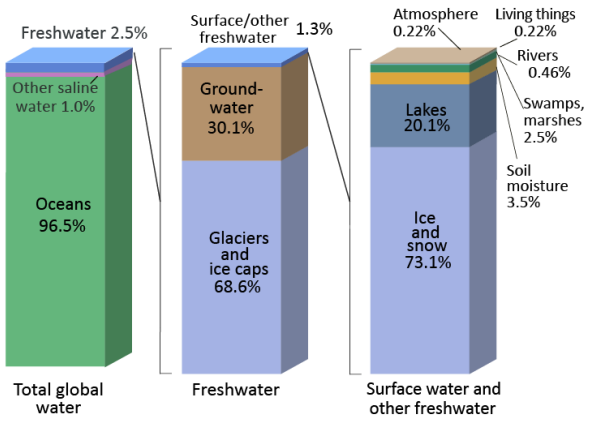


Figure 1-1: The distribution of water on Earth [1]

1-1-2 Desalination

Desalination is the process of converting saline seawater into freshwater. Desalination can be carried out using multiple techniques: Reverse Osmosis (RO) and Electrodialysis (ED) use semi-permeable membranes, while Multi-Stage Flash (MSF) distillation, Multi-Effect Distillation(MED) and Vapour Compression (VC) involve phase changes. Out of these methods, RO is the most commonly used technique [10] because of its efficiency and ease of use [11]. The principle of RO relies on applying high pressure on saline water, pushing the water through a semi-permeable membrane to separate the salt from the water, as illustrated in Figure 1-2.

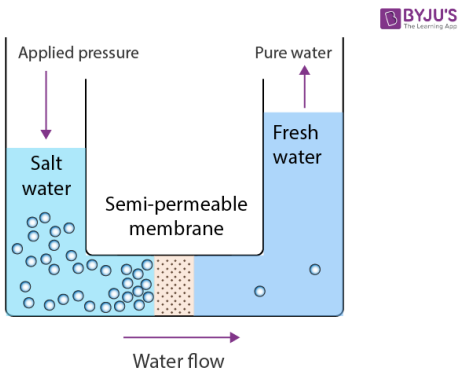


Figure 1-2: The principle of Reverse Osmosis (RO) [2]

1-1-3 The connection to wind power

Desalination is very energy-intensive, accounting for 26 percent of the energy in the global water sector [6]. Currently, most desalination plants are connected to the electricity grid or are coupled with a fossil-fuelled power plant. As the world is shifting away from fossil fuels, alternative ways are being investigated to power the desalination process. Multiple renewable power sources have been considered and wind energy has emerged as the most promising option [12, 4]. Wind energy

provides a technologically mature and relatively inexpensive setup that is easy to scale. Wind-powered RO is the most compelling, as RO presents a developed and widespread method which can best handle the intermittency of the wind [4].
Wind-powered RO is most economically viable for coastal regions where the price of freshwater is high, especially when there is no strong (green) electricity grid available. This makes the coastal areas of Africa and the Middle East, as well as many island regions, among the most promising locations for this technology [13].
In recent years, interest in wind-powered desalination has grown significantly [14]. There have been several simulated concepts [13], but just a few experimental setups have been put into operation. One of the biggest operational wind-powered desalination plants is located on the Canary Islands, consisting of an R&D setup using two 230 kW turbines coupled with a flywheel for energy storage [15]. The conclusions from this test setup show feasibility and reaffirm RO as the most suitable desalination technique. Considering the increased worldwide efforts on sustainability and the increasing efficiency of wind turbines, there is an expanding market for innovations in wind-powered desalination. One particularly innovative project is being developed at the Delft Offshore Turbine (DOT), aptly named the *DOT500kW Pilot Reverse Osmosis (DOT500PRO)* project [3].

1-2 The DOT500PRO

The DOT500PRO consists of a 500 kW turbine connected to a Sea Water RO (SWRO) module capable of producing up to 25000 litres of fresh water per hour. A schematic overview of the DOT500PRO system is shown in Figure 1-3. Unlike conventional wind turbines, which typically house a generator in the nacelle, this turbine is retrofitted with a High-Pressure Pump (HPP). The HPP is used to transform the rotational energy from the rotor to hydraulic power in the form of pressure and flow. This hydraulic power can be used in two ways: to produce electricity using a Pelton turbine or to desalinate water using an RO module. The use of hydraulic transmission removes the medium of electric power between the turbine and the RO module that exists in a conventional wind-powered RO setup. The DOT500PRO is the first turbine aimed at proving the concept on this scale and with this configuration.

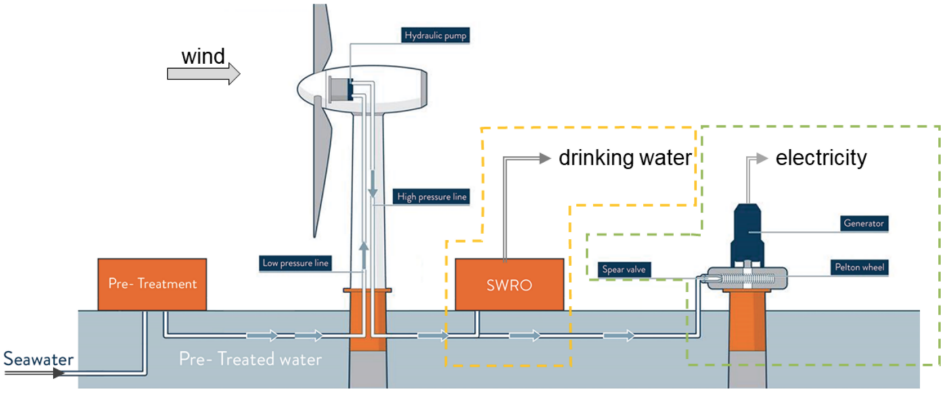


Figure 1-3: Schematic showing the DOT500PRO system[3]

1-3 Thesis Objective

Simplistically speaking, the DOT 500kW Pilot Reverse Osmosis (DOT500PRO) can be off (producing nothing), generating electricity, or simultaneously producing both electricity and freshwater. Moving between states takes both time and power, making frequent switching undesirable. The turbine's production, and hence revenue, depends on the state, the applied control, and the wind speed at that time. Although the state of the system is fully controllable, the future wind speed is uncertain. Therefore, accurate wind prediction is a crucial part of making the best decision for the future. With a smart control strategy, the system can be controlled proactively rather than reactively. This brings us to the goal of the thesis:

How can the state machine of the DOT500PRO be controlled to operate for maximum revenue?

In order to answer the main research question, several key challenges must be addressed, including the design of a wind model and the selection of a suitable control method. Accordingly, the thesis objective is subdivided into the following sub-questions:

1. *Which wind model is most appropriate for predicting wind speeds in this application?*
2. *How do Model Predictive Control (MPC), Stochastic Dynamic Programming (SDP) and Approximate Dynamic Programming (ADP) compare in terms of optimising the DOT500PRO turbine's operation?*
3. *Does the integrated prediction-controller system significantly outperform a simple threshold policy in terms of revenue generation?*
4. *How do variations in wind prediction accuracy affect the performance of the prediction-controller system?*

1-4 Thesis Outline

This thesis is structured to answer the research questions systematically. To begin, chapter 2 gives a comprehensive description of the DOT500PRO system and analyses the statemachine. In this chapter, an explanation of the system states, transitions, constraints and objective revenue function are presented. At the end of the chapter, these four elements are combined in a Markov Decision Process (MDP) framework. Following this, chapter 3 focuses on comparing different wind models. After testing several models, the most suitable one is chosen, answering the first sub-question. Chapter 4 explores the different control methods for the DOT500PRO system: MPC, SDP and ADP. These methods will be compared in a simulation, for which a framework is described in Chapter 5. In this chapter, different scenarios and test metrics are presented to compare the control methods. Subsequently, chapter 6 reveals the results of the simulations and allows us to answer the second, third and fourth research question. Finally, chapter 7 concludes the findings, answers the main research question, and provides recommendations for future work.

Chapter 2

Problem Description

This chapter provides an overview of the state machine of the DOT 500kW Pilot Reverse Osmosis (DOT500PRO), followed by an introduction to the system's constraints and the objective function. By understanding these elements, an optimisation problem is formulated in the form of an Markov Decision Process (MDP).

2-1 Components of the DOT500PRO

What makes the DOT500PRO unique is that, rather than simply having a turbine and generator, it integrates the turbine into a much larger system, as shown in Figure 2-1. This system includes pre-treatment equipment, a High-pressure pump (HPP), a Pelton turbine, a Reverse Osmosis (RO) unit and an Energy Recovery Device (ERD). A detailed explanation of each component is provided below.

- During *pre-treatment*, the seawater is filtered multiple times to remove any sand, metals or plastic waste. A small pump is used for saltwater intake. After pre-treatment, a boost pump sends the filtered water to the nacelle or the ERD.
- Within the nacelle, the kinetic power from the wind turbine is converted into hydraulic power by use of a *High-pressure pump (HPP)*. Between the rotor and the HPP is a gearbox to increase the rotational velocity from the rotor to the HPP.
- The high-pressure water can be led to the *Pelton turbine* by opening the Spear Valve (SV). This creates a water jet that spins the Pelton wheel, which is connected to a generator to produce electricity.
- Alternatively, the high-pressure water can be fed into the *RO unit* by adjusting the Flow Control Valve (FCV). The RO unit uses semi-permeable membranes to separate the freshwater (permeate) and the brine. Brine is water consisting of a high concentration of salts.

- The brine still contains a lot of hydraulic energy when it leaves the RO unit. In the *Energy Recovery Device (ERD)*, this energy is transferred to low-pressure water coming from the pre-treatment. The low-pressure water is pressurised and merged with the feed of the SWRO. The low-pressure brine is disposed of back into the sea.

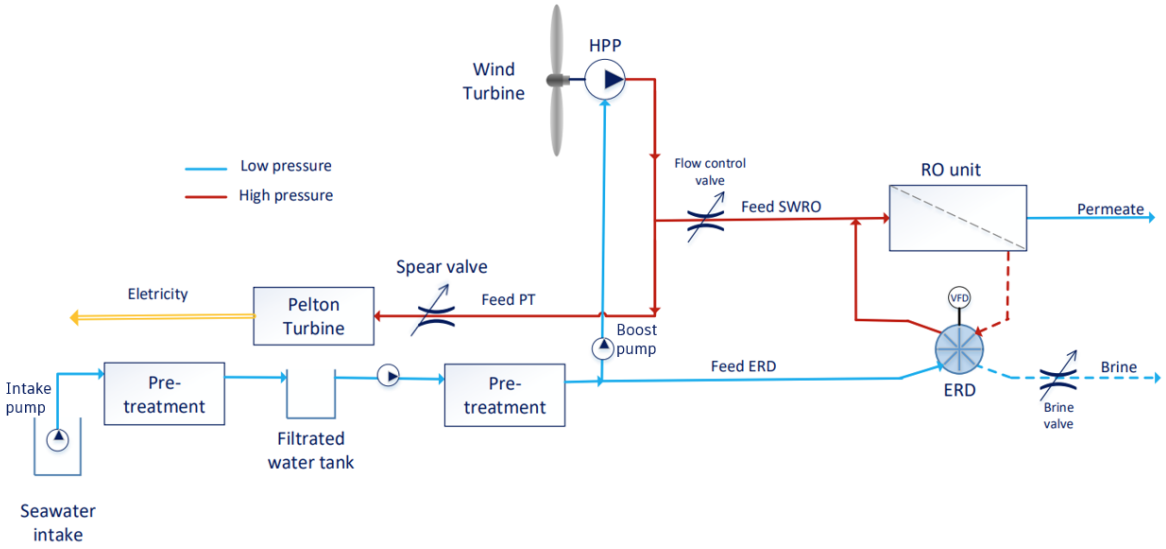


Figure 2-1: Schematics of DOT500PRO [4]

High-level system control is managed at four places:

- The boost pump delivering seawater up into the nacelle
- The pitch of the blades
- The SV controlling the flow towards the Pelton
- The FCV controlling the flow towards the RO unit.

While each of the controllers is managed by a dedicated low-level controller, the coordination and activation of these controllers is to be optimised. For a detailed explanation of the theory behind the optimal control of the low-level controllers themselves, see Appendix A.
In the scope of this research, the system can be switched into an electricity production mode by activating the booster pump and pitching the blades to extract energy from the wind. The SV is used to control the flow towards the Pelton turbine, which is where the system produces electricity. Afterwards, the system is also able to go into a water production mode by activating both the ERD and RO module and opening the FCV.

2-2 The State Machine

To further understand the control problem, it is important to identify the system's state machine. As described earlier, there are multiple modes in which the system can operate. Six such operational modes are defined below.

1. **Hibernation.** The system uses as little energy as possible. Both the ERD and all the pumps are off. This state is intended to reduce energy usage as much as possible for longer periods of little to no wind.
2. **Standby (clean).** The system is ready to start electricity production, but the turbine is not spinning. The pumps are on and can begin to provide pressure to the system.
3. **Standby (contaminated).** Identical to Standby (clean), but the RO module is contaminated.
4. **Electricity Production (clean).** The booster pump delivers water to the nacelle, where the rotor blades are spinning, and the HPP pressurises the water. The water flows to the Pelton turbine in a constant stream, controlled by the spear valve. The Pelton turbine spins a generator, resulting in electricity production.
5. **Electricity Production (contaminated).** Identical to Electricity Production (clean), but the RO module is contaminated.
6. **Water Production.** The FCV is opened, meaning that part of the water flow is directed to the RO module, producing permeate. The flow towards the RO is stabilised by the Pelton turbine. The ERD is also on. The system generates both freshwater and electricity.

Both the *Standby* and *Electricity Production* modes include a contaminated version. This is a design choice made to keep track of the cleanliness of the membranes in the RO module. These six operational modes, combined with two timers for the contamination and hibernation modes, form the state s of the wind turbine. These states form the state space $\mathbb{S}$. Thus, a state $s \in \mathbb{S}$ contains three points of information: The operational mode, a contamination timer, and a hibernation timer. The reason for these timers is elaborated in the next section on constraints. A flow diagram of the state machine, including transitions between states, is further illustrated in Figure 2-2.

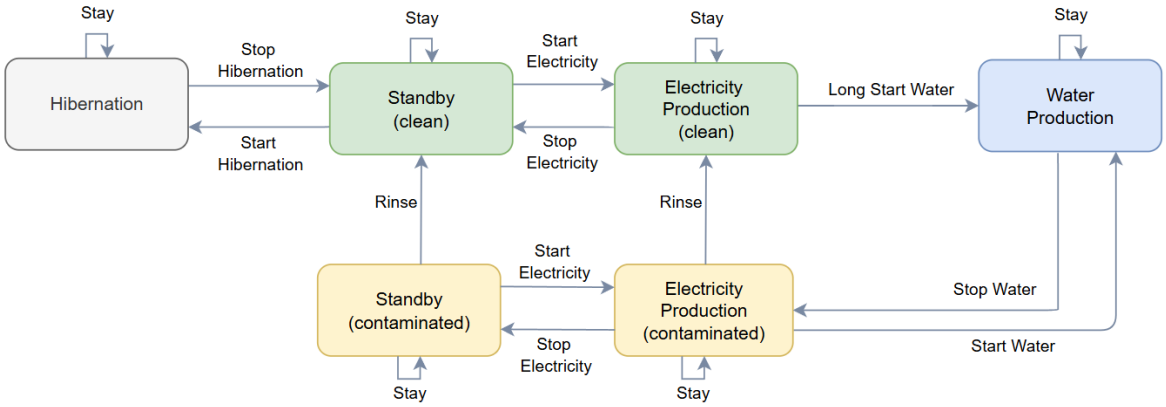


Figure 2-2: The state machine of the DOT500PRO

The transitions between the states are controlled by a set of 9 defined actions: *Start/Stop Hibernation, Start/Stop Electricity, Long Start Water, Start/Stop Water, Rinse*, and *Stay*. These actions form the action space $\mathbb{U}$. Each of these actions corresponds to a change in the system's configuration. For instance, *Start Electricity* initiates turbine rotation and pressurisation at the

HPP, while *Stop Electricity* shuts down these processes. Only performing the action *Stay* does nothing.

Due to the potential contamination in the RO module, the available actions differ slightly between the clean and contaminated states. For example, when the state is in the operational mode *Electricity Production (clean)*, the startup procedure of turning on the water production (*Long Start Water*) takes more time compared to its contaminated counterpart. This prolonged transition is due to a careful ramp-up sequence required to protect the membranes. If the system is still contaminated, it means that the RO membranes are still fouled and therefore the RO process can be started again sooner, using the action *Start Water*. Regardless of this benefit of the contamination, the system may not be contaminated for too long, so a dedicated *Rinse* action can transfer the contaminated state to its clean counterpart. Note that as the water production is stopped, the system always gets contaminated.

While the real system takes different amounts of time to transition between states, inducing some sort of 'transition state', the transition times are omitted in this thesis. The assumption of instantaneous transitions is made to simplify the modelling and allow for a more straightforward optimisation. However, the time and energy associated with these transitions are accounted for in the objective function, which will be explained later.

The transition between states is represented by the discrete state update equation.

$$s_{k+1} = f(s_k, u_k) \qquad \text{(State update)} \tag{2-1}$$

Here, the function $f : \mathbb{S} \times \mathbb{U} \to \mathbb{S}$ represents a mapping that indicates the transition from the original state $s_k \in \mathbb{S}$ at time k, to the new state $s_{k+1} \in \mathbb{S}$ using one of the possible actions $u_k \in \mathbb{U}(s_k)$. In this thesis, a timestep of 18 seconds is used. Here, $\mathbb{U}(s_k) \subseteq \mathbb{U}$ denotes the set of possible actions available in state s_k. Generally, the set of possible actions available is as indicated in Figure 2-2. However, certain constraints can sometimes restrict the availability of specific actions, thereby forcing other actions. These constraints are explained in the next section.

2-3 Constraints

There are constraints on the system to keep the operation of the DOT500PRO safe and efficient. The reasoning behind these constraints is explained in this section.

1. **Contamination:** The first constraint concerns the amount of time the system is allowed to be contaminated. After stopping water production, the water that remains in the RO module stagnates, causing contaminants like salts to settle on the membrane surface, also known as fouling. This can decrease efficiency and potentially damage the membranes. Therefore, the RO system has to be rinsed using a backflush procedure if it has been idling for more than 30 minutes. Once this contamination time threshold is reached, the only available action is to *Rinse*. The contamination time is tracked using the contamination timer. After rinsing, the timer is reset.

2. **Hibernation mode:** A similar constraint is given regarding the hibernation mode. When the system enters hibernation mode, it shuts down all non-essential electronics, allowing the system to stabilise its thermal and pressurised components over the course of 20 minutes.

This helps to maintain safe and stable conditions and reduce the wear and tear of excessive cycling between standby and hibernation mode. As a result, when the system enters the hibernation state, it must remain in this state for at least 20 minutes. During this time, the only available action is to *Stay*. The hibernation time is tracked using the hibernation timer. Performing the action *Stop Hibernation* resets the timer to zero.

3. **Cut-out wind:** The turbine rotor blades are not allowed to spin above 20 m/s (cut-out) because of safety constraints. This constraint is being handled by a different controller and thus not considered in this thesis.

4. **Delicate membranes:** The RO module can be severely damaged if engaged while the pressure in the system is lower than the osmotic pressure. Therefore, the system is prohibited from being in water production when the wind speed is below 5.6 m/s. Instead of restricting the action space directly, this constraint has been modelled using a penalty in the objective function to capture the cost of operating under these conditions.

These constraints do not just limit which actions are available, but also significantly increase the number of system states to consider when controlling. The core system logic can be described in six operational modes from the previous section, but the introduction of time-dependent constraints means that, in practice, the effective state space grows to several hundred unique combinations as the time spent in contamination or hibernation mode needs to be included.[1] A table with the available actions per state is given in Appendix B

2-4 Wind update

The future wind speed $w_{k+1} \in \mathbb{W}$ is modelled as a stochastic variable that is updated every timestep. Because of its sequential nature, the probability distribution is conditioned on the previous n wind speeds $w_k, w_{k-1}, \ldots, w_{k-n+1}$, as is expressed in Equation 2-2. The statistical modelling of wind is addressed in the next chapter, as well as the justification for this choice.

$$w_{k+1} \sim P(\cdot | w_k, \ldots, w_{k-n+1}) \qquad \text{(wind update)} \tag{2-2}$$

2-5 Objective function

Having defined the system's capabilities and constraints, as well as the way that the wind is modelled, the next step is to establish the system's objective: maximising the revenue generated by the DOT500PRO. The amount of revenue (in euros) is dependent on both the state s_k, the wind speed w_k and the action u_k at time k. Mathematically, this is represented by the function $R : \mathbb{S} \times \mathbb{U} \times \mathbb{W} \to \mathbb{R}$

$$\text{Revenue at time } k \text{ (in €)} = R(s_k, u_k, w_k) \tag{2-3}$$

[1] In practice, this results in over 200 unique states, depending on how time-dependent conditions are discretised. For example, using an 18-second timestep, the hibernation and contamination timers span approximately 66 and 100 steps, respectively.

The system can generate positive revenue when producing electricity or water, but when inactive, the power consumption of the system leads to a negative revenue. Additionally, transitioning between states comes with its own costs, such as energy use and the time delays, which are also incorporated in the function R.
The goal is to maximise the total amount of revenue over time. This is expressed by taking the infinite summation of (2-3). Because of the stochastic nature of the wind speed w_k, $R(s_k, u_k, w_k)$ is also expressed as a random variable. Taking the expectation of R, a measurable value of the total revenue is obtained, which can be used to assess performance.

$$\text{Total Revenue } = \sum_{k=0}^{\infty} \mathbb{E}[R(s_k, u_k, w_k)] \qquad \text{(Objective function)} \tag{2-4}$$

2-6 Markov Decision Process

In this section, all components presented in the previous sections of this chapter are combined to formulate the problem as an MDP. This framework allows the discrete states and transitions to be modelled together with the stochasticity of the wind model. Before setting up the full MDP, the objective function is first reformulated from maximising the revenue to minimising the loss $\ell : \mathbb{S} \times \mathbb{U} \times \mathbb{W} \to \mathbb{R}$, defined as

$$\ell(s_k, u_k, w_k) := -R(s_k, u_k, w_k)$$

This formulation is mathematically equivalent, but follows the standard MDP conventions. The MDP is defined by the following set of spaces, functions, and probabilities:

- $\mathbb{S}$ represents the state space, which includes the operational modes but also the time spent in hibernation or contamination.
- $\mathbb{U}(s_k)$ is the action space, which defines the possible actions available to the controller at each state s_k, such as starting or stopping electricity or water production.
- $\mathbb{W}$ represents the wind states, which evolve over time in a stochastic manner. The wind speed at each timestep is taken from a distribution which is conditioned on previous wind states, with probabilities modelled by $P(w_{k+1}|w_k, \ldots, w_{k-n+1})$.
- The transition function $f(s_k, u_k)$ describes how the system evolves from state s_k to s_{k+1} when an action u_k is taken.
- The loss function $\ell(s_k, u_k, w_k)$ represents the cost or loss associated with being in state s_k, taking action u_k, and experiencing wind speed w_k.

Together, this forms the optimisation problem of Equation 2-5

$$\begin{aligned} &\min_{\pi \in \Pi} \sum_{k=0}^{\infty} \mathbb{E}[\ell(s_k, u_k, w_k)] \\ s.t. \quad &s_{k+1} = f(s_k, u_k), \\ &w_{k+1} \sim P(\cdot | w_k, \ldots, w_{k-n+1}), \\ &u_k = \pi(s_k, w_k, \ldots, w_{k-n+1}) \end{aligned} \tag{2-5}$$

Here, π represents a policy, characterised by a function in the function space $\Pi : \mathbb{S} \times \mathbb{W}^n \to \mathbb{U}$. The policy π provides an action u_k dependent on the current state s_k and the n past wind speeds w_k to w_{k-n+1}. The goal is to find the best policy π^* that provides the actions to minimise the objective function. The process of finding this best policy or approximating it, is explored in Chapter 4. Assuming the model assumptions hold, the policy π^* enables the DOT500PRO to minimise losses and maximise long-term revenue performance.

Chapter 3

Wind Modelling

This chapter contains the process of selecting a suitable wind model for the application at hand. First, the requirements of the model are determined, followed by a review of the possible models available. Based on a qualitative evaluation, the most appropriate method is then chosen. Subsequently, the core principle of this method is then introduced, before describing the implementation using real-world wind data. The chapter concludes with a discussion of the limitations and assumptions corresponding to this approach.

3-1 Introduction and Modelling Requirements

For the design of the wind model, it is crucial to look at the requirements that correspond to the application of the DOT 500kW Pilot Reverse Osmosis (DOT500PRO). In the previous chapter, it is found that the dynamics of the statemachine of the DOT500PRO operate on a relatively short timescale. For instance, the constraints in the model result in the requirement to distinguish between the 29th and 31st minute of contamination. To make the best decision, the controller requires a wind prediction that also behaves on a short timescale. The wind model should therefore operate on a very short timescale, with a resolution of less than 1 minute.

The speed of wind is determined by multiple factors, like pressure zones, climate change and even the rotation of the planet. However, for the operation of the DOT500PRO, these physical relations are either far too complex or are not measured at the turbine itself. For the sake of simplicity and practicality, the wind speed is modelled as a time series, relying solely on recent wind speeds.

Moreover, as wind speeds are highly variable, the model needs to be able to describe the uncertainty. Instead of assuming one exact value for the wind speed, the model should provide a distribution of possible values and give a probabilistic forecast.

Lastly, it is crucial to note that the primary objective of this thesis is not to develop the best wind prediction model but rather to create a model that allows good control. Therefore, a model is

preferred that fits well within the Markov Decision Process (MDP) framework. This means that the model should be predictable and transitions should be well-defined.

In summary, the wind model needs to meet several criteria: it should operate at a very short timescale, provide probabilistic forecasts, rely on recent wind measurements and allow predictable and well-defined transitions suitable for MDP. Using these criteria as a selection method, an appropriate model can be chosen.

3-2 Review of Wind Models

For centuries, humanity has tried to predict the weather [16]. From early farmers and sailors scanning the skies to today's sophisticated weather stations, forecasting has come a long way. Nowadays, wind forecasting models come in a broad range of variations. In [5], these wind forecasting methods are classified according to Figure 3-1.

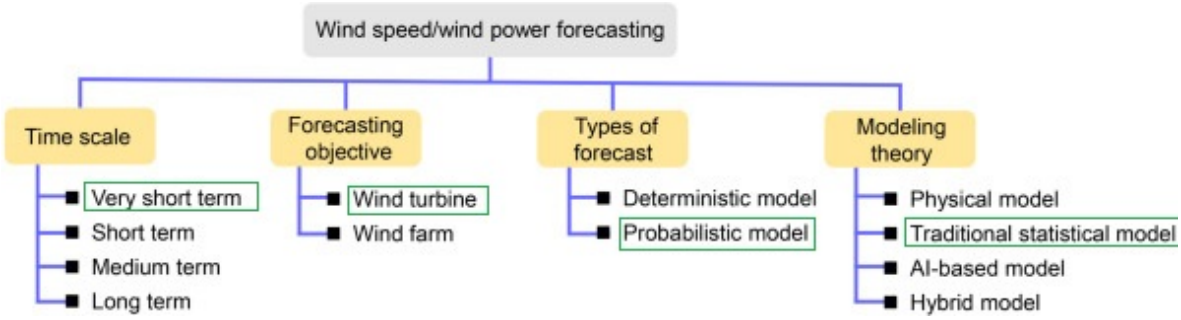


Figure 3-1: Classification of Wind Forecasting methods according to [5]

Knowing the requirements of the model, the focus is on a timescale in the very short term (≤ 30 min) range. The forecasting objective is a single wind turbine, rather than a wind farm, and the model should be probabilistic.

In terms of modelling theory, physical models rely on atmospheric physical relations and use Numerical Weather Prediction (NWP) to model weather over large scales. Although useful for long-term weather forecasts, this is not very reliable for very short-term predictions [17]. Machine learning and hybrid models, although increasingly popular, are also avoided due to their computational cost, data requirements, and limited interpretability in real-time systems [18, 19, 20]. The focus is placed on traditional statistical models, with the scope narrowed to three widely studied methods: ARIMA, Kalman Filtering and Markov processes. These methods are well-supported in the literature for very short-term wind forecasting. A brief overview of each method's application is provided below.

3-2-1 ARIMA

In [18], the ARIMA model is compared with several machine learning and deep learning methods in predicting wind power plant performance. The ARIMA model showed a higher Mean Squared Error (MSE) than the machine learning methods, indicating comparatively weaker performance. However, Liu et al. [21] makes a similar comparison with Seasonal ARIMA (SARIMA) and two

machine learning methods: Gated Recurrent Unit (GRU) and Long-Short Term Memory (LSTM). In this setup, the SARIMA model outperformed the advanced ML methods. This seems surprising as in [18] the LSTM is performing much better than ARIMA. While the addition of the seasonality could be a cause of this, it does show that the performance of ARIMA and ML methods is dependent on the specific situation. One is not simply better than the other. Nonetheless, ARIMA models remain more interpretable and easier to implement than their ML counterparts.

3-2-2 Kalman Filtering

After the initial formulation of the Kalman Filter (KF) by Rudolf Kalman in 1960 [22], Bossanyi [23] was one of the first to apply the KF for short-term wind prediction. This resulted in a reduction of 10 percent in one-minute-average wind speed forecasting error. In 1995, Huang [24] combined the AutoRegressive(AR) model with the KF to improve wind predictions for horizons ranging from one hour to several hours ahead. Louka et al. [25] showed that the KF can also be used for post-processing two atmospheric models, to provide a wind forecast, which was described as a 'remarkable improvement' to the original model. Liu [26] created a hybrid ARIMA-Kalman model, where the ARIMA model is used to initialize the Kalman Measurement and the state equations. This also showed the suitability of the KF for multi-step, non-stationary, wind speed prediction.

Chen [27] proposes a novel hybrid method combining Support Vector Regression (SVR) and an Unscented KF (UKF) for short-term wind speed prediction. The UKF is an extension of the normal KF that can be used to work with non-gaussian uncertainties. In the study, the SVR-UKF model performed better than an AR-Kalman model for one-step and multi-step prediction across three USA test sites. Shukur [28] compares AR-ANN and AR-KF on wind data from both Malaysia and Iraq. The AR-KF outperformed both ARIMA and the machine-learning based AR-ANN models. More recently, Hur [29] proposed the use of the Extended KF (EKF) together with machine learning to predict strong gusts near wind turbines. As an alternative to expensive LiDaR-based systems, this method showed promise in improving the turbine control performance of the wind turbine, being able to track actual wind more closely for 2 and 3 seconds ahead.

Finally, Kalman Filtering is being noted as a good option in a 2024 review by Tuncar [30]. In the paper, he discusses the growing trend of hybrid forecasting methods, combining the strengths of multiple methods. However, it also acknowledges the limited research on offshore wind farms compared to land-based wind facilities.

3-2-3 Markov Processes

Markov processes have been widely used in wind speed time series generation, as shown by Shamshad et al. [31], who demonstrated that these processes can synthetically generate time series while preserving key statistical characteristics. In the context of wind power prediction, Carpinone et al. [32] applied first- and second-order Markov processes for very short-term wind forecasting (10-minute intervals). Although the higher-order method yielded better results, the study acknowledged that shorter time intervals led to even more accurate predictions. In [33], Jacobsen demonstrated the effectiveness of Markov processes in increasing wind power yield on a Norwegian wind farm. The study also highlighted the advantage of clustering states, rather than linearly spacing them. In a different application, Markov processes were utilised not for direct prediction, but to select the best chaotic prediction model [34, 35]. Building on this, Zhao et

al. [36] enhanced the model by incorporating a temporal Markov process that accounts for the interactions between nearby turbines in a wind farm.

3-2-4 Comparison

The three aforementioned models were tested numerically for very short term wind speed forecasting. The process of this comparison is found in Appendix D. Although ARIMA models showed slightly better numerical performance in terms of the Root Mean Squared Error (RMSE) and Mean Interval Score (MIS), the difference with the other two models was marginal and not decisive for this application. Experiments with the Kalman Filter did not result in any meaningful results. Given ARIMA's assumption of Gaussian noise, combined with the added interpretability and compatibility of the Markov model with the MDP, the decision is made to model the wind as a Markov Process. The comparison based on the application requirements set in Section 3.1 can be seen in Table 3-1. Markov Processes are fully data-driven, probabilistic by design, and operate on discrete states. This makes it an intuitive and fitting model for the purpose of control. For these reasons, the wind is modelled as a Markov process in the remainder of this thesis.

Requirement	**ARIMA**	**Kalman Filter**	**Markov Process**
High time resolution (<1 min)	✓	✓	✓
Probabilistic forecast	∼	∼	✓
Uses only recent wind measurements	✓	✓	✓
Predictable, well-defined transitions for MDP integration	×	∼	✓
Overall fit for this application	Medium	Medium-High	**High**

Table 3-1: Comparison of Wind Forecasting Models Based on Application Requirements

3-3 Markov Processes

A Markov Process is defined by its order n. A first-order Markov process is a stochastic model where the probability of an event depends only on the previous event. This means that previous events are not contributing to the probabilities of the next event[1]. The order of a Markov process notes the number of historical events relevant. For example, the first-order and second-order Markov processes satisfy this property

$$P(w_t|w_{t-1}) = P(w_t|w_{t-1}, w_{t-2}, \cdots) \quad \text{(first-order)} \tag{3-1}$$

$$P(w_t|w_{t-1}, w_{t-2}) = P(w_t|w_{t-1}, w_{t-2}, w_{t-3}, \cdots) \quad \text{(second-order)} \tag{3-2}$$

As seen in Equation 3-1, the addition of more, older information (i.e., $w_{t-2}, \cdots$) does not change the probability for the first-order process. This property is also known as the *Markovian Property*.

[1] In this thesis, only discrete-time Markov Processes on a finite space are considered.

3-3-1 First-order example

A first-order Markov process is illustrated with an example. Let w_t be a random variable at timestep t which can take on one of m discrete states $\mathbb{W} = (W_1, W_2, \cdots, W_m)$. Assuming a Markov process of the first order, the probability of transitioning from W_i to W_j can be defined as

$$p_{i,j} = P(w_t = W_j | w_{t-1} = W_i) \tag{3-3}$$

A transition table T of size $m \times m$ can be formed by collecting all $p_{i,j}$ and placing the value in the corresponding row and column.

$$T = \begin{bmatrix} p_{1,1} & \cdots & p_{m,1} \\ \vdots & \ddots & \vdots \\ p_{1,m} & \cdots & p_{m,m} \end{bmatrix} \tag{3-4}$$

Note that

$$\sum_{j=1}^{m} p_{i,j} = 1 \quad \forall i$$

that is, each column of T sums up to a total probability of 1.

Let $\vec{z}_t$ be a vector consisting of the probability of the random variable w_t being in every state W_1 to W_m.

$$\vec{z}_t := \begin{bmatrix} P(w_t = W_1) \\ P(w_t = W_2) \\ \vdots \\ P(w_t = W_m) \end{bmatrix}$$

Here, $\vec{z}_t$ represents a discrete probability distribution, and the sum of its values is one. By the law of total probability, $P(A) = \sum_{n=1}^{m} P(A|B_n) \cdot P(B_n)$ as long as the set of $(B_n : n = 1, 2, 3, \ldots, m)$ is finite. Based on this, the relation of $\vec{z}_t$ with $\vec{z}_{t-1}$ can be expressed using T.

$$\begin{aligned} \vec{z}_t &= \begin{bmatrix} P(w_t = W_1) \\ \vdots \\ P(w_t = W_m) \end{bmatrix} \\ &= \begin{bmatrix} \sum_{i=1}^{m} p_{i,1} \cdot P(w_{t-1} = W_i) \\ \vdots \\ \sum_{i=1}^{m} p_{i,m} \cdot P(w_{t-1} = W_i) \end{bmatrix} \\ &= \begin{bmatrix} p_{1,1} & \cdots & p_{m,1} \\ \vdots & \ddots & \vdots \\ p_{1,m} & \cdots & p_{m,m} \end{bmatrix} \begin{bmatrix} P(w_{t-1} = W_1) \\ \vdots \\ P(w_{t-1} = W_m) \end{bmatrix} \\ &= T\vec{z}_{t-1} \end{aligned}$$

This allows predictions to be made by multiplying the transition table by an initial probability distribution to get the new distribution at time t.

This way, the future distribution at time t can be predicted as

$$\begin{aligned}\vec{z}_t &= T\vec{z}_{t-1} \\ &= T^2\vec{z}_{t-2} \\ &= T^t\vec{z}_0\end{aligned}$$

3-3-2 Higher orders

The process can be extended for second- or higher-order Markov processes as well. This involves creating a higher-dimensional matrix, also known as a tensor, instead of the two-dimensional transition table T. For a Markov process of order n, a tensor with $n+1$ dimensions is used. The value of a cell in this tensor represents the probability of transitioning to this state, conditioned on the sequence of n states. As an example, for an order 3 transition table, the value of $p_{1,2,2,3}$ represents $P(w_t = W_1|w_{t-1} = W_2, w_{t-2} = W_2, w_{t-3} = W_3)$, i.e. the probability that the sequence $\{W_3, W_2, W_2\}$ is followed by W_1. The size of a transition table is m^{n+1}.

3-4 Implementation and Data Setup

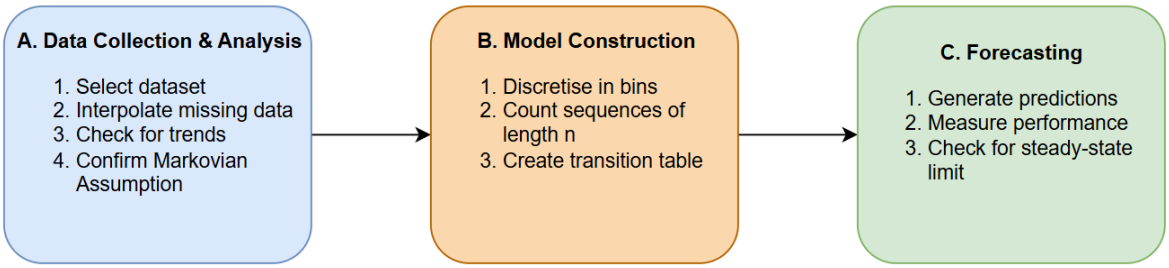


Figure 3-2: The workflow of this section

The implementation of a Markov Process as a wind model follows the workflow shown in Figure 3-2. The first step is to collect wind speed data and perform an analysis of the properties of the dataset. Following this, a model is constructed. To do this, the data is discretised into a finite number of bins. Then, given an order of the Markov chain n, a transition table can be constructed. The last step is to perform forecasting and gain insight into the dynamics of the wind model.

3-4-1 Data Preparation and Analysis

Data is gathered by KNMI from the Borssele windpark 1 + 2, from 1-10-2024 to 28-2-2025 [37]. The location of the windpark is shown in Figure 3-3. Wind speed data is collected every 18 seconds using a LiDAR system, measuring horizontal wind velocity at a height of 38m. [2] The raw wind speed data contains missing values (NaNs) and some unrealistic high values. These data points are removed and replaced using linear interpolation.

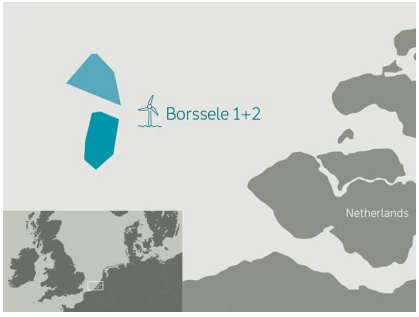
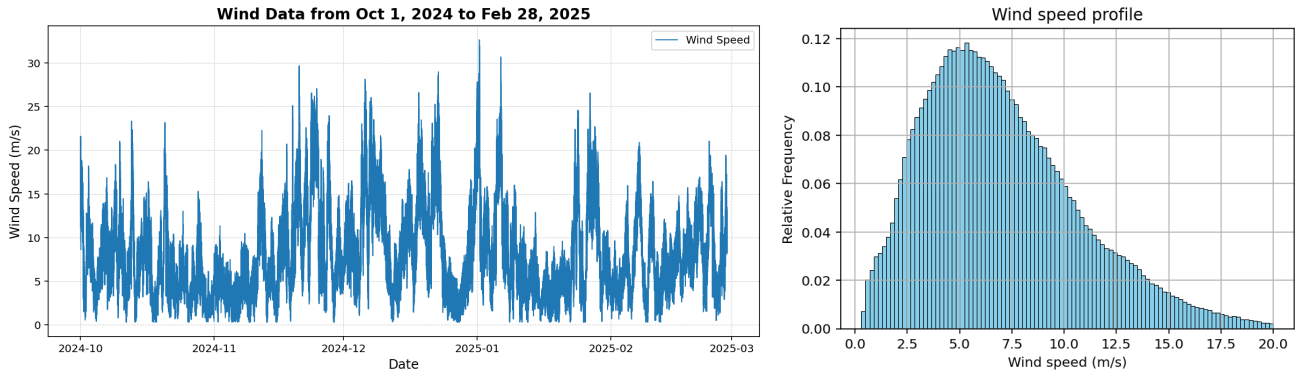


Figure 3-3: Location of the windpark Borssele 1+2

The months October to February were deliberately chosen, as these months showed the most consistent behaviour. Additionally, this dataset is quite recent and relatively complete, unlike earlier years. The full data set, shown in Figure 3-4, consists of more than 720.000 measurements, providing a reliable basis for analysis and model construction.



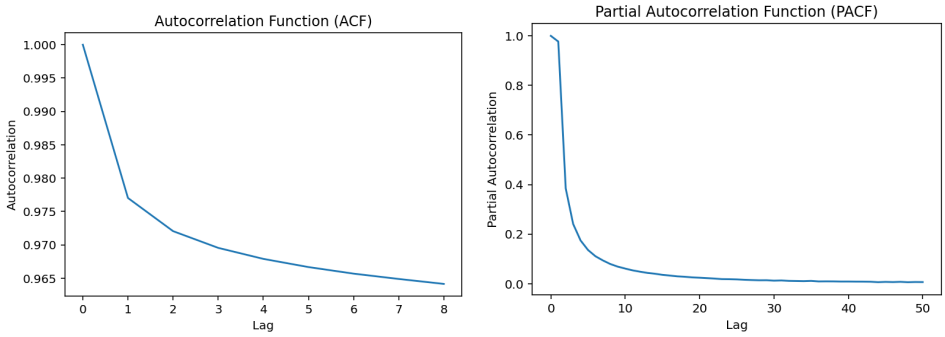
(a) Wind data

(b) Wind speed profile of the data

Figure 3-4: The dataset and its wind profile

After analysing the data, no clear (daily) trends were discovered. On the other hand, there is a clear temporal dependency between sequential datapoints. This can be seen by looking at the graphs of the AutoCorrelation Function (ACF) and Partial AutoCorrelation Function (PACF) of the data in Figure 3-5. The high values corresponding to the first few lags indicate a strong correlation of recent data, strengthening the claim of the Markovian Property of Equation 3-1 and 3-2. The rapid decline of the PACF after approximately five lags confirms the diminishing relevance of older data.

[2] For reference, the DOT500PRO has a nacelle height of 44 meters.



(a) ACF of the data **(b)** PACF of the data

Figure 3-5: ACF and PACF of the training data

3-4-2 Model construction

For the construction of the Markov model, the data first needs to be discretised. The data can be discretised in several ways. The most straightforward method would be to divide the data into intervals of the same length, for instance, using intervals of 0.5 m/s, such as [0.3,0.8], [0.8, 1.3], and so on. The downside of this method relates to the distribution of the wind. As the wind is not distributed uniformly, this would result in some intervals having much more datapoints than others, potentially resulting in bins with only a few datapoints to determine the behaviour from. Instead, the data is discretised using an equal frequency approach, resulting in the same number of occurrences for every interval bin (see Figure 3-6). The added benefit of this approach is that the algorithm behaves more precise around the important transition points of the DOT500PRO statemachine, whilst reducing complexity on the windspeeds that are less relevant. For example, wind speed data points below 1.6 m/s are all in the same bin.

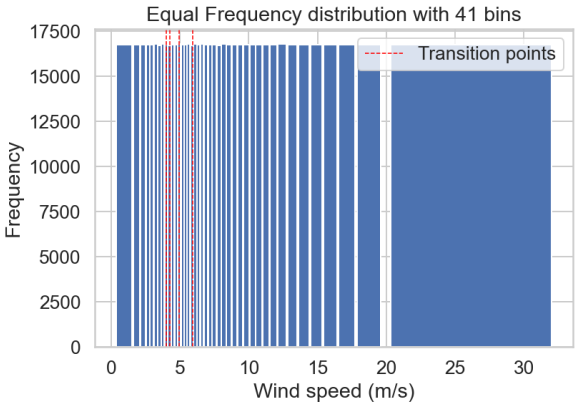


Figure 3-6: Wind distribution using an equal frequency discretisation. Every bin has equal height, but the interval is scaled.

Choosing the number of bins is somewhat arbitrary; more bins give a finer-grained result, but comes at the cost of a larger state-space and less data points per interval. Choosing 41 bins provides a sweet spot, maintaining over 16 thousand datapoints per bin. Based on this discretisation, all wind data is mapped to these 41 states in $\mathbb{W} = \{W_1, \ldots, W_{41}\}$, where the mean value of the

interval is used later for calculations.

Apart from discretisation, choosing an appropriate order n of the Markov process is much more crucial, as this determines the number of unique state sequences and thus the size of the transition table. As discussed in section 3-3, each n-length sequence is paired with the following state to create the transition table T.

This table is constructed from the discretised data. The value of $p_{i,j}$ can be estimated by counting the number of transitions $n_{i,j}$ from W_i to W_j in the data. Dividing $n_{i,j}$ by the total number of transitions $\sum_{k=1}^{m} n_{i,k}$ coming from W_i gives the probability of transitioning from state W_i to W_j.

$$p_{i,j} = \frac{n_{i,j}}{\sum_{k=1}^{m} n_{i,k}} \tag{3-5}$$

3-4-3 Forecasting

With the transition table in place, predictions can now be made. This is done by multiplying an initial state by the transition table, as discussed in section 3-3. An interesting observation is that the further in the future predictions are made, the more the predicted distribution resembles the wind speed profile in Figure 3-4b. This indicates that the model converges towards a steady-state distribution coinciding with the wind speed profile as one looks further into the future.

As the forecast reaches this steady state, further propagation with the transition table does not change the distribution anymore. This point is crucial because a prediction past this steady-state point is no longer dependent on the initial state or model, and might as well be picked from the distribution of Figure 3-4b.

It was observed that both a higher model order n and a rare or extreme initial wind sequence will increase the number of timesteps before convergence. An example of the propagation is shown in Figure 3-7. Here, a model of order 3 is initialised with three windspeed measurements of 15 m/s. After approximately 300 timesteps, or 1.5 hours, the probability density reaches this steady state. This reveals a key limitation of this approach: forecasts are useful only for short-term predictions.

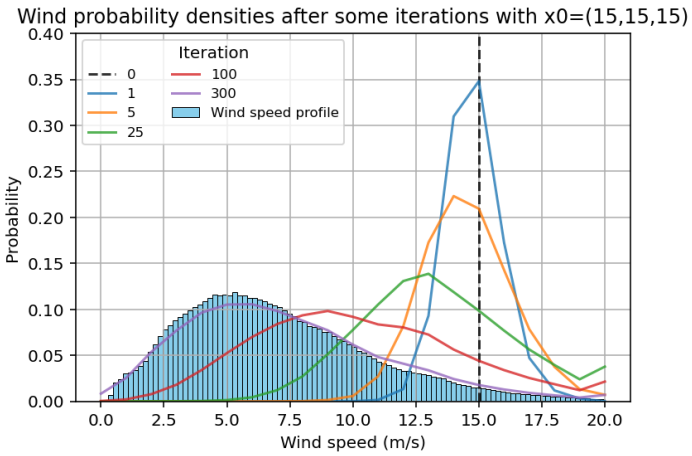


Figure 3-7: Wind probability forecast after several iterations

3-5 Limitations

While Markov processes offer a simple and interpretable way to model wind speed, there are a few limitations to consider:

1. **Convergence to Steady State.** As explained in the previous section, the forecast is restricted to very-short term predictions. This is due to the convergence of the prediction to a steady state.
2. **State Space Explosion for Higher Orders.** As the order n increases, the number of states grows exponentially, leading to an even bigger transition table. This quickly results in problems with memory and computation time, limiting the use of higher orders.
3. **Data Sparsity** Even with a large dataset, many transitions may very rarely or never occur. These can cause unreliable or wrong predictions in the transition table, resulting in poor performance if they do occur.
4. **Stationarity.** The model is based on the assumption of stationarity, providing one transition table for all data. This rejects potential daily, monthly or seasonal trends that could occur. By constraining the dataset to the five months mentioned, this is offset for the most part, but it also means that the transition table might not perform well at other times of the year.
5. **Markov Assumption.** The PACF does show a big drop in correlation after the first few lags, but a fully Markovian system would have absolutely zero dependence on events happening more than n times in the past.
6. **Discretisation** The Markov process requires a discrete state space, forcing the discretisation of the wind data. This loses granularity and forces us to make trade-offs in the number of bins and their size. This means that results can differ depending on the discretisation strategy
7. **Data-Driven model.** One of the biggest advantages of the Markov Process is that it is purely data-driven, without any physical assumptions or complicated weather models. However, this is also a limitation as it ignores all meteorological studies on wind behaviour and correlations with other factors such as pressure and temperature. Instead, this model treats the wind speed as a purely statistical data point, solely dependent on past winds.

Chapter 4

Control Strategies

In this chapter, different control strategies are explored. First, the problem is reformulated to fit more naturally into the MDP framework. Afterwards, four control algorithms are presented, each time increasing in complexity. With every method, the limitations of the approach are included in the text.

4-1 Problem formulation

In Chapter 2, the optimisation problem of the MDP is defined like this:

$$\begin{aligned} &\min_{\pi \in \Pi} \sum_{k=0}^{\infty} \mathbb{E}[\ell(s_k, u_k, w_k)] \\ s.t. \quad &s_{k+1} = f(s_k, u_k), \\ &w_{k+1} \sim P(\cdot | w_k, \ldots, w_{k-n+1}), \\ &u_k = \pi(s_k, w_k, \ldots, w_{k-n+1}) \end{aligned} \tag{4-1}$$

Here, s_k represents the state of the turbine, which is updated by the function f using the action u_k, which is provided by the policy π of the controller. w_{k+1} is a random variable representing the wind speed at time $k+1$. This random variable is taken from a discrete probability density, based on the transition table from an n-th order Markov Process, as described in Chapter 3.

The objective function is expressed using the expectation operator. This is defined as a summation over the m possible wind states in $\mathbb{W} = (W_1, W_2, \ldots, W_m)$[1]. For every $W_i \in \mathbb{W}$, the probability $P(w_k = W_i)$ is determined and multiplied by the loss function ℓ using the corresponding value of W_i.

$$\mathbb{E}[\ell(s_k, u_k, w_k)] = \sum_{i=1}^{m} \Big\{ P(w_k = W_i) \cdot \ell(s_k, u_k, W_i) \Big\} \tag{4-2}$$

[1] In Chapter 3, we have determined $m = 41$.

4-1-1 Definition of the state vector x_k

To combine both the state and the wind update, a state vector $\vec{x}_k$ is defined containing both s_k and the wind speed values w_k to w_{k-n+1}. The vector $\vec{x}$ exists in the space $\mathbb{X} := \mathbb{S} \times \mathbb{W}^n$. From now on, this vector is denoted simply as x_k, omitting the vector arrow.

$$x_k = \begin{bmatrix} s_k \\ w_k \\ \vdots \\ w_{k-n+1} \end{bmatrix} \tag{4-3}$$

The full state update then becomes the following.

$$x_{k+1} = \begin{bmatrix} f(s_k, u_k) \\ w_{k+1} \\ \vdots \\ w_{k-n+2} \end{bmatrix} \quad \text{with } w_{k+1} \sim P(\cdot|w_k, \dots, w_{k-n+1}) \tag{4-4}$$

The state evolution of x_k is defined with a probabilistic transition function $\mathcal{F} : \mathbb{X} \times \mathbb{U} \to [0,1]^m$ corresponding with the update as in Equation 4-4

$$x_{k+1} \sim \mathcal{F}(\cdot|x_k, u_k)$$

After this reformulation, the MDP can be written as:

$$\begin{aligned} &\min_{\pi \in \Pi} \sum_{k=0}^{\infty} \mathbb{E}[\ell(x_k, u_k)] \\ s.t. \quad &x_{k+1} \sim \mathcal{F}(\cdot|x_k, u_k) \\ &u_k = \pi(x_k) \end{aligned} \tag{4-5}$$

where the function $\ell(s_k, u_k, w_k)$ is adjusted to $\ell(x_k, u_k)$ accordingly. The problem can now fully be described with the 4-tuple $(\mathbb{X}, \mathbb{U}, \mathcal{F}, \ell)$.

4-2 Threshold Control

One of the simplest ways to control the DOT 500kW Pilot Reverse Osmosis (DOT500PRO) state machine would be to use a rule-based transition controller. In this type of application, rule-based control is typically implemented in the form of a threshold controller. A threshold control applies actions for transitions if the wind reaches above or below a certain threshold.

While this method is often used in the industry, it is a reactive heuristic, meaning it will often perform suboptimally. For instance, imagine the threshold control is implanted to switch to water production if the wind speed is above 6 m/s. This would result in the system to do a lot of frequent switches when the wind oscillates around this threshold. This problem can be omitted by setting very conservative thresholds, for instance, 7 m/s to turn water production on, 5 m/s to turn it off. However, this would clearly come at the cost of optimal control as well.

4-3 Model Predictive Control

Model Predictive Control (MPC) is a control strategy that finds the optimal input for a finite-horizon version of the problem. Instead of finding a policy π that works for all cases of x_k, the problem focuses on a specific trajectory of inputs of length N in the future. This transforms the problem formulation to Equation 4-6

$$\begin{aligned} \min_{u_0,\dots,u_{N-1}} & \sum_{k=0}^{N-1} \mathbb{E}[\ell(x_k, u_k)] \\ s.t. \quad & x_{k+1} \sim \mathcal{F}(\cdot|x_k, u_k) \\ & u_k \in \mathbb{U}(x_k) \quad k = 0, \dots, N-1 \\ & x_0 \in \mathbb{X} \end{aligned} \tag{4-6}$$

The MPC approach of solving this problem can be split into three steps:

1. This algorithm first predicts the N wind speed distributions of the finite horizon based on the initial wind sequence. This provides the distributions of $w_{k+1}, w_{k+2}, \dots, w_{k+N}$.
2. Then, the value of being in a state at the last timestep is calculated based on the wind prediction of w_{k+N}. After calculating this cost $V(s_N)$ for every state $s \in \mathbb{S}$, the algorithm performs a backwards iteration step , calculating the best action for every state and the resulting cost-to-go $V(s_k)$ corresponding to the sequence of actions until timestep N. This is known as dynamic programming and is based on the Bellman equation:
$$V(s_k) = \min_{u_k \in \mathbb{U}} \mathbb{E}[\ell(s_k, u_k, w_k) + V(s_{k+1})] \tag{4-7}$$
3. Having iterated back to the initial timestep, the action u_0 corresponding to the minimisation of the value function of the initial state $V(s_0)$ is determined as the best action. The controller then applies the control action u_0 to the system. When the following measurement is received, the state s_0 is updated, and the entire problem is solved again over a shifted horizon. This is known as a receding horizon approach over horizon N.

4-3-1 Limitations

While MPC is flexible and used very often in highly constrained, continuously spaced systems, it does come with limitations that prevent its usage for the DOT500PRO system.

- The MPC controller works in an on-line optimisation setting. This means that it is required to solve the problem again for every timestep, leading to a high computation cost. This computation cost can become a bottleneck if it prohibits the system to act fast enough before the dynamics of the system update.
- Apart from that, the MPC formulation of the problem provides an approximation of the actual problem in Equation 4-5. It simplifies the problem by reducing the horizon and finding an open-loop solution, rather than a policy for the infinite horizon.
- The performance of the control is highly dependent on the choice of horizon N. This is always going to involve trade-offs between performance and computation cost.

4-4 Stochastic Dynamic Programming

Stochastic Dynamic Programming (SDP) is intended to find a policy for the Markov Decision Process (MDP) presented in Equation 4-5. However, because of the infinite horizon, the sum in the objective function can diverge. To resolve this issue, the SDP algorithm is built around a slight modification in the problem statement. By solving for the minimum average-cost, instead of total cost, the problem is rephrased as Equation 4-8. As the function ℓ is bounded, the average cost remains bounded and finite as well, thereby addressing the issue of a diverging objective.

$$\begin{aligned} \min_{\pi \in \Pi} \lim_{N \to \infty} \frac{1}{N} \sum_{k=0}^{N-1} \mathbb{E}[\ell(x_k, u_k)] \\ s.t. \quad x_{k+1} \sim \mathcal{F}(\cdot|x_k, u_k) \\ u_k = \pi(x_k) \end{aligned} \tag{4-8}$$

The state space $\mathbb{X}$ is finite, allowing for an exhaustive consideration of all options for x_k. The standard approach to solving this problem is via dynamic programming, either using value iteration or policy iteration. Using the Bellman equation [38], value functions $V(x_k)$ or policies $\pi(x_k)$ are updated until the algorithm converges to an optimal policy π^*. This policy essentially consists of a lookup table with the best action for every x_k. A value iteration approach is explained below.

4-4-1 Value iteration

The algorithm begins by assigning an arbitrary value, typically zero, to every possible state $x \in \mathbb{X}$. This assigns a value $V(x)$ for every state and every n-sized wind sequence, making this a very large space. Then, the iteration starts: for every x, the value is updated as in Equation 4-9. This represents an adaptation of the Bellman equation, using an increasing variable α that starts at 1 to get the average cost:

$$V_{new}(x) \leftarrow \min_u \frac{\ell(x,u) + \alpha \mathbb{E}[V_{old}(\mathcal{F}(\cdot|x,u))]}{1+\alpha} \tag{4-9}$$

Here, the expectation is calculated over the distribution of the random variable gained from $\mathcal{F}$. If the difference between the old value function V_{old} and the new value function $V_{new}(x)$ becomes minimal, the iteration has converged. This is measured using the 2-norm of the difference of a vector containing the value functions:

$$\delta = ||V_{new}(x) - V_{old}(x)||_2 \tag{4-10}$$

When δ is smaller than the convergence criterion, the iteration stops. Otherwise, both the value function and α are updated:

$$V_{old}(x) \leftarrow V_{new}(x) \qquad \forall x \in \mathbb{X} \tag{4-11}$$

$$\alpha \leftarrow \alpha + 1 \tag{4-12}$$

After this update, the process is repeated. After convergence, there is a final step: the minimisation of Equation 4-9 is done once more, to assign the optimal action u to each $x \in \mathbb{X}$, essentially creating a lookup table that provides the policy π. Pseudo code of this algorithm can be found in Appendix C.

4-4-2 Limitations

While SDP is guaranteed to find the optimal policy for the problem in Equation 4-8, this process also comes with flaws. Due to the brute-forcing nature of the algorithm, exploring all possible states, the computation cost can explode for larger state spaces. As the state space increases, for example, when taking a higher order for the Markov process, the number of situations to consider can explode, making solving infeasible. This is also known as the *Curse of dimensionality* and is a common problem in optimisation.

4-5 Approximate Dynamic Programming

Approximate Dynamic Programming (ADP) is designed to overcome the curse of dimensionality that comes with SDP in larger state spaces. Although ADP still tries to solve the same problem in Equation 4-8, it does not update a separate value function $V(x)$ for every state in $\mathbb{X}$. Instead, the value function is estimated by a function, typically a weighted sum.

$$\hat{V}(x) \approx \theta^{\mathsf{T}} \phi(x) \tag{4-13}$$

Here $\phi(x)$ is a set of features aimed to capture the structure of the state, and θ consists of a vector of weights that are parameters to be learned. The ADP algorithm typically iterates several trajectories, during which a learning algorithm is applied to update the weights. This update can be seen as a similar process of minimising δ in Equation 4-10. When learning is successful, $\hat{V}(x) \approx V(x)$ with $V(x)$ the value function gained from SDP. When controlling, the controller then compares the values $\hat{V}(x_k + 1)$ to determine the best action. This can be done online or beforehand. Other ADP techniques, such as Q-learning, reinforced learning, exploration vs exploitation, and advanced basis functions are kept outside the scope of this research.

4-5-1 Limitations

Despite its practical advantages, using ADP introduces its own challenges:

- To begin, the approximation results in the loss of the optimality guarantee. Because the algorithm does not consider every possible state, it cannot guarantee to work for all states.
- Second, the performance of ADP is very dependent on the features selected in $\phi(x)$. The features need to be selected in such a way that every sort of important behaviour (wind trends, combinations of variables, et cetera) is included. Choosing the right features represents a whole field of study itself.
- The weights are adjusted with the goal of converging to an optimal vector θ. Convergence is not guaranteed, however. Neither is the point, if converged, a global optimum necessarily. All the common challenges regarding local optima are present in ADP as well.

Chapter 5

Simulation Framework

This chapter provides a setup for the simulations. First, the objectives of the tests are outlined, followed by a short recap of the wind model and the controllers. Afterwards, the parameter values are given, and finally, the different test scenarios are given.

5-1 Simulation Objectives

The simulation objectives are closely linked with the sub-questions that were introduced in Chapter 1. Nevertheless, there are some added goals that are meant to further increase the understanding of the performance of the model.

1. **MPC limit:** In the previous chapter, the claim was made that Model Predictive Control (MPC) results in a high on-line computation cost. Nevertheless, the feasibility of MPC is explored, comparing the runtime at different order n and horizon N.

2. **SDP limit:** Similarily, the limit of Stochastic Dynamic Programming (SDP) is investigated. By increasing the order, the trade-off between performance and complexity is explored.

3. **Controller comparisons:** Using the data where the model is trained on, fair comparisons can be done between the controllers, including a higher-order Approximate Dynamic Programming (ADP) control.

4. **Test on other data:** Arguably, the most important test is to simulate the controller on real-world data, outside of the training set. This can provide concrete results, showing the benefit of pro-active control of the DOT 500kW Pilot Reverse Osmosis (DOT500PRO). These tests are done in separate conditions:

    - **Set A:** On the same period as the training data, but using a previous year.
    - **Set B:** On a period of data with different seasonal conditions.

5-2 Simulation components

In this section an overview is given of the wind model and the controllers that are being simulated.

5-2-1 Wind Model

For the wind model, the Markov Process explained in Chapter 3 is used. This consists of a transition matrix of order n that provides a probabilistic forecast. The wind has been discretised into 41 intervals of different lengths. The average value of this interval is used to calculate the loss function in training. In out-of-model simulations, the actual continuous wind space is used, reflecting real-life deployment.

5-2-2 Controller overview

In the simulations, 5 types of controllers are being tested:

1. **Threshold Control:** A simple rule-based controller to serve as a baseline. The controller changes states reactively, using just the current wind speed. The threshold values are determined by looking at the policy of the first-order SDP solution.
2. **MPC:** Solves a finite-horizon version of the problem on-line. This controller finds the best trajectory over a finite-horizon based on a wind prediction.
3. **SDP:** This controller has solved the infinite-horizon problem offline, exhausting all possible policies. In usage, the controller simply follows the action as stated in the policy.
4. **ADP:** Has approximated the SDP solution by iterating the value function. Does not suffer from the curse of dimensionality and can therefore handle a higher order n.
5. **Omnipotent Control:** All-knowing controller that makes the perfect decision, as it knows the future wind speeds. The Omnipotent controller provides the best possible performance for the given wind.

5-3 Parameter values

The parameters for the different models are shown in the overview below, with a more detailed explanation provided in Appendix F.

Category	Parameter / Description
Wind Model	
Order	To be determined (Markov order n)
Bins	41 bins (see Figure 3-6)
Turbine Model	
Electricity price	0.09 €/kWh
Water price	1.13 €/m^3
Hibernation overhead	0.462 kWh
Standby overhead	11.72 kWh
Contamination penalty	0.01 €/hr
Rinse cost	17.6 kWh
Start Electricity cost	78 kWh
Other transition cost	0
Low wind production penalty	-1000
Hibernation constraint	20 minutes
Contamination constraint	30 minutes
Controller Parameters	
Threshold Control	
Stop Hibernation	4.9 m/s
Start Electricity	4.4 m/s
Stop Electricity	2.9 m/s
Long Start Water	6.1 m/s
Start Water	6.6 m/s
Stop Water	5.8 m/s
Rinse	3.5 m/s
MPC	
Horizon	At least 100 (e.g., 30 min); 300 for order 3
SDP	
Convergence criterion	$\delta < 10^{-4}$
ADP	
Feature design	Custom function using turbine state, wind sequence, trackers
Optimiser	Adam
Learning approach	TD(0), value iteration, ϵ-greedy
Discount factor γ	0.99

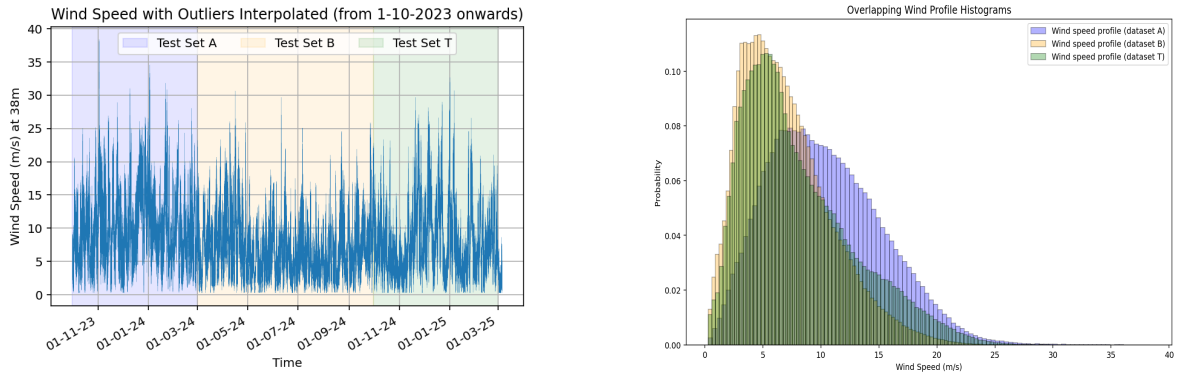
Table 5-1: Overview of Parameters Used in the Simulation

5-4 Scenarios

The controllers are evaluated in different settings. To begin, tests are done using in-model simulations on data that is generated by the corresponding Markov process. This removes the approximation step from the wind model and allows for a check whether the controller is properly implemented. Out-of model simulation begins by simulating on the training set. Here, the goal is to find the limits of the controllers and make a fair comparison. Afterwards, the best controllers are tested on other wind speed data, either in a different season or in the same season of the previous year.

5-4-1 Data split

The data is split in sets A, B and T as shown in Figure 5-1a.



(a) The split of the three test sets

(b) The wind profile of the test sets

Figure 5-1: The data used for the three test sets

The goal was to select one dataset with a similar wind profile, from the same months in a different year (set A) and one dataset from a different season (set B). This can reveal how the controller performs with seasonal or yearly differences in wind data.

Interestingly, the wind profile appears to vary more when compared to the previous year than it does when compared to a different season, as shown in Figure 5-1b.

5-4-2 Set T: Training set scenarios

In the training set scenarios, all controllers are considered. From the dataset, starting points are selected randomly. The 6-hourly (or 1200 time steps) wind sequence from that point is used as the input for the simulation. All controllers are tested against the threshold control, which provides a baseline.

1. **Test T1: Finding the MPC limit.** The test is done on both different orders and horizon lengths. The computation time is also measured to see if it limits the use of MPC control.

2. **Test T2: Finding the SDP limit.** Simulations are performed for different orders for the SDP control. The goal is to compare the performance of different orders and determine at what order the curse of dimensionality results in a limit.
3. **Test T3: Comparison of controllers.** This test includes all controllers. The objective is to find the best control method for the training set. The resulting control method can then be tested on the different data sets.

5-4-3 Other Test sets

In order to test the dependency of the control on the dataset and the learned wind model, the best controllers from the previous tests are simulated on different wind speed data.

1. **Test A: Different year.** The controllers are tested on the same months in the previous year, aiming to reduce the effect of seasonality.
2. **Test B: Different season.** Simulation is done on the rest of the year. This should indicate how sensitive the control is on the different data.

5-4-4 Evaluation Metrics

The controllers are evaluated both on performance and feasibility. This is measured using several metrics:

- **Total Revenue:** The main metric is the total loss of the system over the simulation time (6 hours). The loss of the threshold control is subtracted from the loss of the other controllers to indicate the difference. Focus is put on the average improvement.
- **Computation time & Memory usage:** For both the MPC and SDP controller, feasibility issues are foreseen as the order and horizon grows. Simulations are run up to the point where the controller becomes unfeasible for usage.
- **Number of transitions:** Even though this is not the intended optimisation of the control algorithms, the number of transitions is an interesting thing to measure. The threshold controller has a big drawback in that it makes a lot of transitions. If the other controllers lower this amount, it could indicate an improvement in the durability of the system. The number of times the system enters and leaves water production mode can be measured, as the RO module is the most sensitive part of the system.

Using these metrics, the results of the simulation can be analysed to draw conclusions about the performance of the controllers.

Chapter 6

Results

The results of the simulations are presented in this chapter. Based on these results, conclusions are drawn.

6-1 In-model simulations

For the in-model simulations, 100 2-hour sequences of wind speeds were generated based on the second-order Markov model. For every sequence, the order 2 Stochastic Dynamic Programming (SDP) and Model Predictive Control (MPC) controls are executed. The threshold and omnipotent control were also tested on the same sequences. The results of the controller performance is shown as a boxplot in Figure 6-1. Here the dependency on wind data, rather than control, is highlighted as all controllers (apart from the omnipotent control) share the same minimum and maximum loss. If there is no wind, the type of controller is not going to change that.

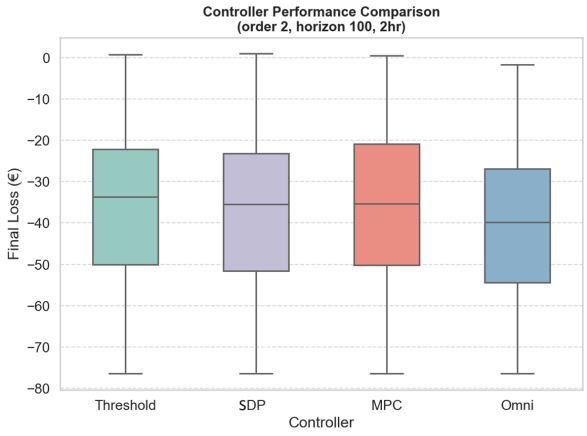
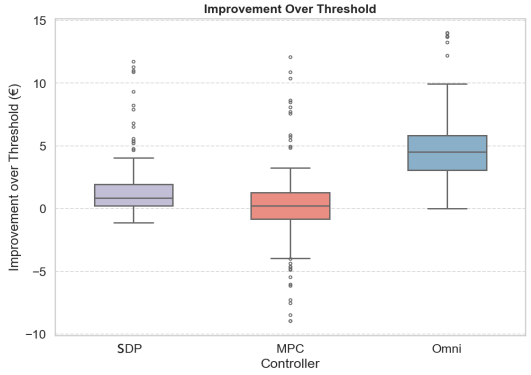


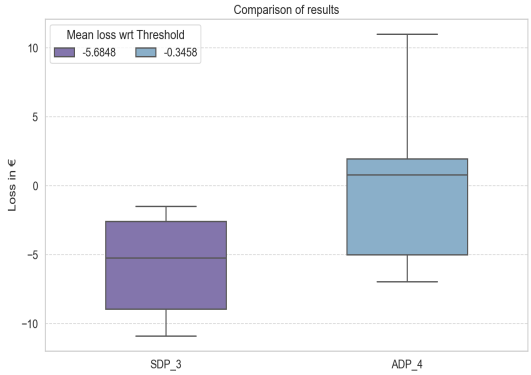
Figure 6-1: In-model performances of the controllers

If the threshold results are subtracted from the data of the other controllers, the improvement in Figure 6-2a can be seen more clearly. The omnipotent control is, as expected, always performing

better than or equal to the threshold control. The SDP and MPC controllers, however, do not always perform better. This result shows that sometimes the proactive controllers make a mistake and pre-emptively go into the wrong state. This can be attributed to the controller assuming that the wind data follows the prediction when in reality, it does not. Nevertheless, both controllers make an improvement over the heuristic on average.SDP performs the best, with an average improvement of approximately 3 euros on a 2-hour wind sequence. This would come down to an improvement of around 3 percent, which would result in a big difference if applied on a large scale.
The Approximate Dynamic Programming (ADP) controller of order 4 does not perform better than the threshold control, as can be seen in Figure 6-2b.



(a) In-model improvement over the threshold



(b) In-model comparison of SDP(3) and ADP(4)

Figure 6-2: In-model comparison of the controllers w.r.t threshold control

6-2 Training set tests

The training set tests are done on dataset T in Figure 5-1a. The wind model of the controllers are based on this dataset as well. These simulations are intended to help determine the best controller without interference from different data. Figure 6-3 shows the training data.

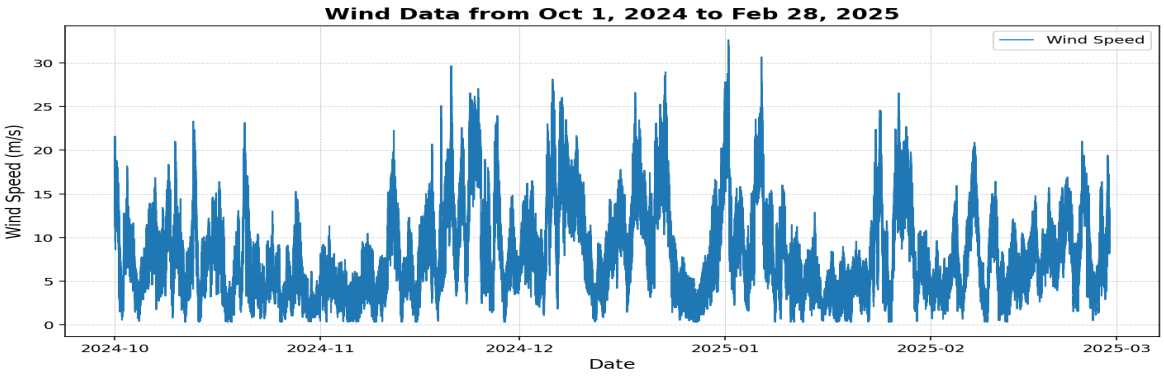
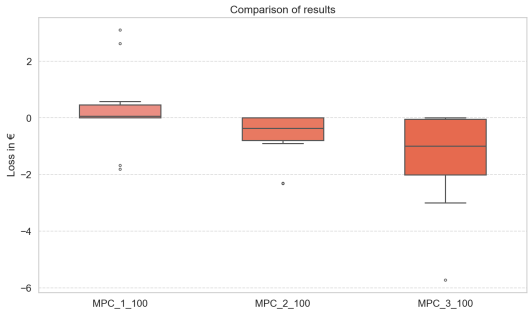


Figure 6-3: Training set T

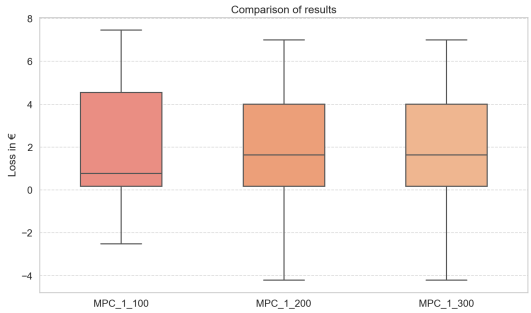
6-2-1 Test T1: MPC Computation time

The MPC control is tested for orders 1, 2 and 3. The result is compared with the baseline of the threshold control. A lower loss indicates better performance. The result is shown in Figure 6-4a. As expected, the higher-order model is performing better, which can be seen as the loss is more negative. An order 1 MPC controller performs even worse than the heuristic.

A similar improvement is observed as the horizon increases, as seen in Figure 6-4b. This shows three MPC controllers of order 1 with different horizon lengths of 100, 200, and 300. The difference in the results is smaller compared to the previous test of different orders. This indicates the importance of order over horizon. The change of a horizon of 200 to 300 timesteps does not results in any change. This can be attributed to the steady-state behaviour of the first-order wind model being reached before the 200th timestep.



(a) Comparison of different orders of MPC



(b) Comparison of different horizons of MPC

Figure 6-4: Overall comparison of MPC parameters

However, this increased performance of a higher order or a longer horizon comes at a cost: the MPC computation time grows rapidly, as shown in Figure 6-5. The threshold given here is already very generous, as the next wind update arrives after 18 seconds, meaning that with a 9-second computation time, the controller is half a timestep too late. For orders of 4 or higher, the MPC is deemed infeasible. The order 3 controller is bounded to a horizon of 300 timesteps.

Figure 6-5: MPC computation time

6-2-2 Test T2: Stochastic Dynamic Programming

In this test, the order n of the SDP controller is increased. As the order increases, the controller becomes more elaborate and considers more measurements before making a decision. This addition of information seems to increase the performance for orders 1 to 3, as indicated in Figure 6-6. Solving the MDP using SDP for orders of 4 was unsuccessful because of the curse of dimensionality. The limit of SDP is reached at order 3.

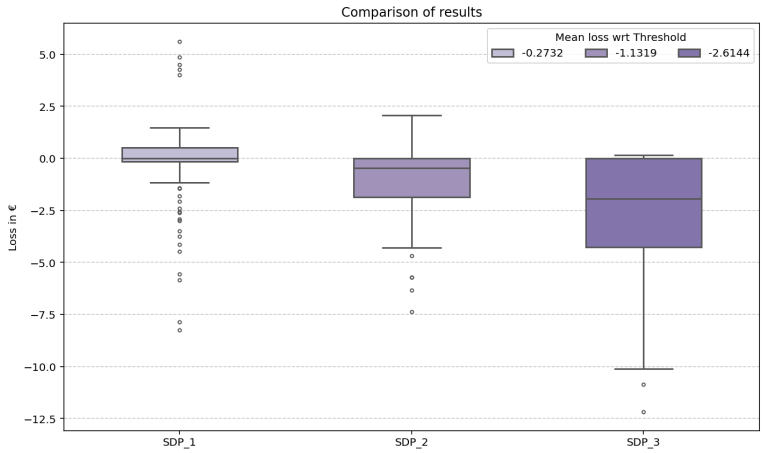


Figure 6-6: Comparison of SDP of orders 1 to 3

6-2-3 Test T3: Comparison of the controllers

In Table 6-1 the results of test T3 are shown. Here, the best variants of MPC (order 3, horizon 300) and SDP (order 3) are compared with the threshold and a fourth-order SDP controller. On average, the SDP controller performs best, followed by ADP and MPC. All controllers perform better than the threshold control.

Controller	Threshold	SDP_3	ADP_4	MPC_3_300
Average loss	-155.4	**-157.2**	-156.7	-156.9

Table 6-1: Results of test T3

Whilst higher orders were indications of better results in the previous tests, the ADP controller of order 4 is not performing better than the order 3 SDP control. This can be explained by the approximation nature of the algorithm. The features might not behave optimal, and the optimizer might not have found the best weights in time, or got stuck in a local minimum.

When looking at a single week-long relative loss graph that compares SDP(3) and ADP(4) in Figure 6-7, both controllers can be seen making mistakes on the way, resulting in a very close end result. The SDP(3) controller makes the right decision at 10000 timesteps, while the ADP(4) controller performs correctly around 20000 timesteps, where the SDP(4) makes a comeback afterwards. This might reveal that the ADP controller has the ability to outperform SDP(3), but it does not manage to do so here.

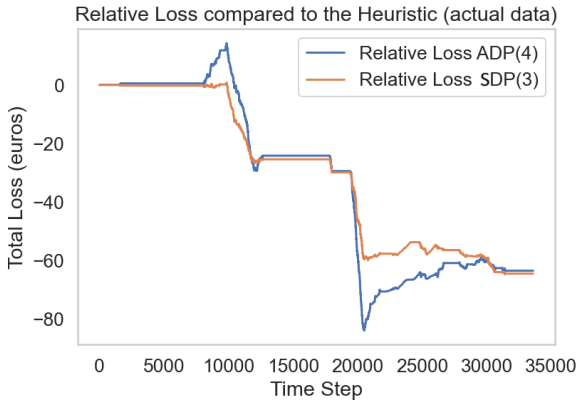


Figure 6-7: Example of cost graph comparing SDP and ADP over a week

6-3 Other test sets

Seeing as SDP(3) performed the best on the training set, this controller is tested on the other test sets. For the sake of curiosity, it is also check whether ADP(4) is acting better here.

6-3-1 Test A: Different year

As seen in Figure 5-1b, dataset A is not quite as similar as the training set T. Looking at the wind data in Figure 6-8, it can be seen that it was a stormy winter, including one of the worst storms of the 21st century. [1] This compromises the model its forecasting ability, as seen in Figure 6-9a

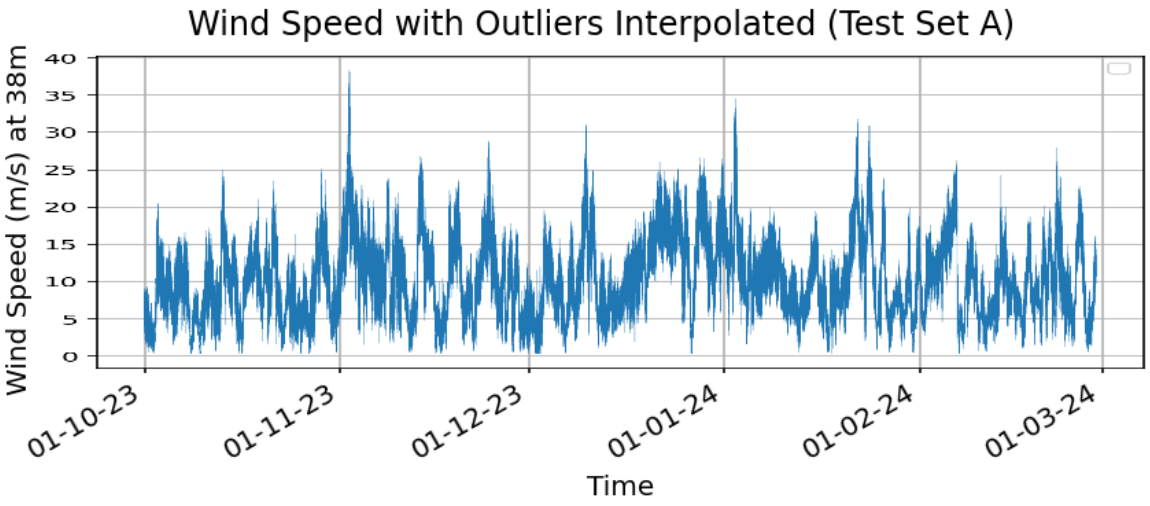
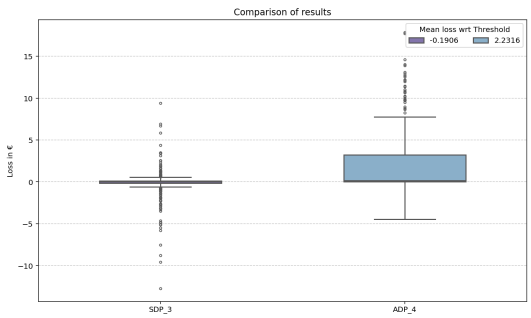


Figure 6-8: Test Set A

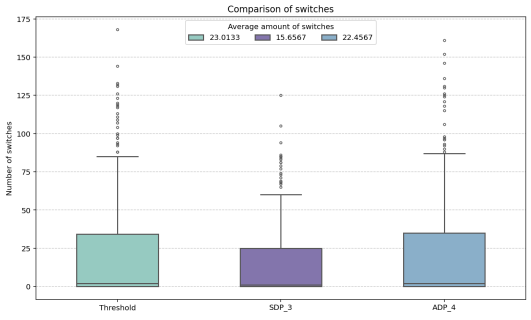
Whilst there is little to no difference in the amount of revenue generated by the SDP controller

[1] According to BBC: https://www.bbc.com/weather/articles/cvglvnyxpx0o

compared to the threshold, the controllers do not make the same decisions. The SDP controller is acting much more conservative, as it is making less transitions, as is shown in Figure 6-9b.



(a) Results of test A in terms of revenue



(b) Results of test A in terms of number of transitions into water production

Controller	Threshold	SDP_3	ADP_4
Average loss	-164.0	**-164.2**	-161.6
Number of transitions	23.0	**15.7**	22.5

Table 6-2: Results of test A

6-3-2 Test B: Different season

In Figure 5-1b, test data set B seems more similar to the training data compared to test set A. However, when looking at the results in Figure 6-11a, no significant improvement can be stated. With an average of 3 eurocents over 6 hours, this change in revenue seems insignificant. Again, the ADP controller is not performing better than the threshold control, but both proactive controllers make less switches to water production, as is shown in Figure 6-11b.

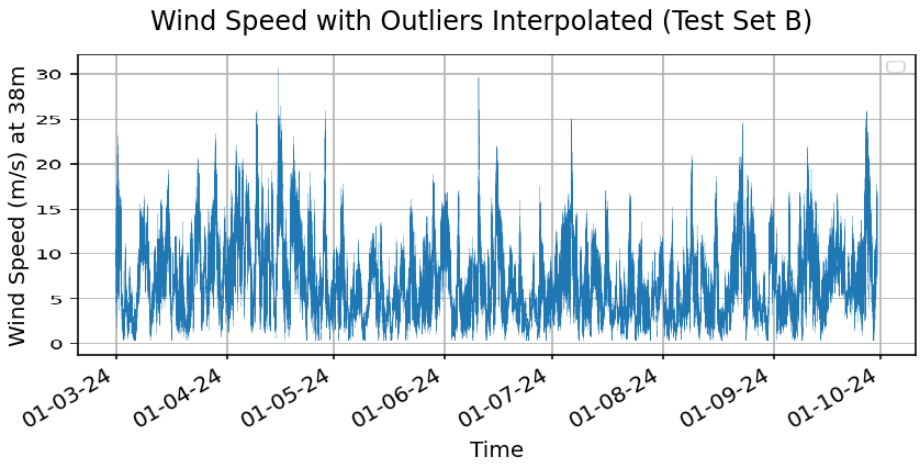
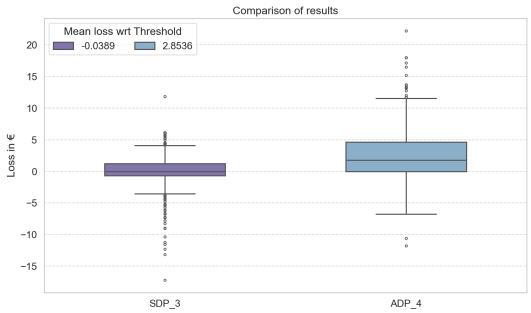
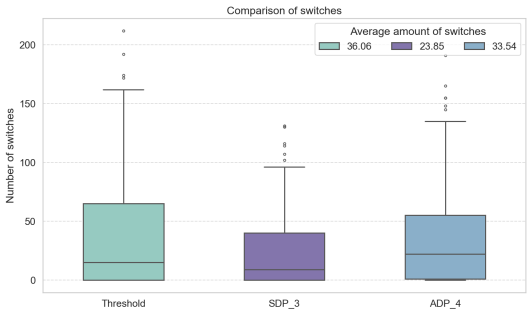


Figure 6-10: Test Set B



(a) Results of test B in terms of revenue



(b) Results of test B in terms of number of transitions into water production

Controller	Threshold	SDP_3	ADP_4
Average loss	-106.4	**-106.5**	-103.6
Number of transitions	36.1	**23.9**	33.5

Table 6-3: Results of test B

6-4 Conclusion of the results

Based on the results from the simulations, several conclusions can be drawn:

1. From testing using MPC, the bottleneck of the computation time is quickly found. The conclusion is drawn that this technique is too slow for the fast dynamics of the system. Using MPC in this application is therefore not a feasible approach.

2. SDP is limited by the curse of dimensionality, resulting in a maximum order of three with the current setup. This technique performs better than the threshold control on the training set, revealing the controller's potential.

3. For the higher-order dynamic programming, ADP can be used. The results show that the model needs more tuning, but it has the potential to overtake the third-order SDP as the best-performing controller.

4. The results from tests A and B show the limitations of predictive control. When the wind behaves differently from what the model predicts, the SDP controller is no longer able to make the best decisions. The predictive controller is too sensitive to different wind patterns, for instance different seasons and years. This results in the reactive threshold controller getting similar results.

Chapter 7

Conclusions and Discussion

7-1 Conclusion

The goal of this thesis is to find an optimal control strategy for the DOT 500kW Pilot Reverse Osmosis (DOT500PRO) statemachine. In this approach, the optimal control results in the most revenue generated. The question this thesis tries to answer is therefore:

How can the state machine of the DOT500PRO turbine be controlled to operate for maximum revenue?

Analysis of the system reveals an opportunity for improvement by the use of proactive control, switching states based on a prediction of the wind. Several statistical models were considered to model the wind, and a Markov Process was found to be the most appropriate for this application.

Using the Markov model, the problem was formulated as an Markov Decision Process (MDP). Several control techniques were applied to this framework. These control methods were tested in a Python simulation with a heuristic threshold controller as a benchmark for comparisons.

Some concerns about Model Predictive Control (MPC) were confirmed, as for higher orders, the computation time exceeds the limit. The Stochastic Dynamic Programming (SDP) performed the best overall, improving average performance as the order of the SDP method increases.

Above Markov order 3, the curse of dimensionality prevents the usage of SDP due to the state space growing too large. Higher orders of control are possible using Approximate Dynamic Programming (ADP) to approximate a SDP solution. The use of ADP might result in better performance, but requires careful tuning. Unfortunately, this is currently not achieved due to a lack of time.

Sensitivity of SDP and ADP to different wind patterns is tested by using data from both another year and from another season. Both tests led to a decrease in performance, resulting in a negligible improvement over the threshold control. Nevertheless, the number of transitions decreases compared to the threshold control. Where the controller did not increase revenue directly, the implementation should decrease the amount of wear and tear, leading to other benefits.

In conclusion, the state machine of the DOT500PRO can be controlled using several methods, with an SDP control performing slightly better than a heuristic threshold control. For optimal revenue in the training data, this approach is seriously worth considering for implementation. For other uses, this might not be recommended when compared with the ease of implementation of a threshold control.

7-2 Discussion

In order to improve on the work done in this thesis, several key points can be addressed and improved upon. In this section, several topics for further research are mentioned.

7-2-1 Turbine model

Transition time. For the sake of simplicity, the assumption was made that the transitions between states are instantaneous. This assumption does not reflect real life. Adding this factor to the model might give the pro-active controllers a better performance, considering the higher amount of transitions of the threshold control.

Momentum in the system. In the model, an assumption is made on the speed of the response in the system. It is assumed that when the wind speed increases, the turbine's production increases immediately with it. However, in practice, there is a momentum in the system where the rotor speeds up or slows down.

Simplification of the revenue function. The revenue function that is being used is based on an indoor test setup, where the electricity and water production were measured for different rotor speeds. This is multiplied by current prices for water and electricity in the Netherlands. In a real-world business evaluation, the prices of the target location need to be accounted for, as well as the difference between the buy and sale prices. In addition, the wear and tear is resulting in eventual repairs and other operational costs. These are not accounted for in the function right now.

Constraints. Constraints such as contamination were considered very simple and very strict. In real-world applications, this might be much more complex than the model representation. Additionally, rinsing is more complex than illustrated in this thesis, but it is beyond the scope of this thesis. Cut-out of the wind is also not addressed here, but can be implemented in the future.

Control timestep. The speed of the control, together with the speed of wind updates are important parameters to consider. In the future, more research is needed on the choice of these timesteps, where a separation might be interesting, updating the wind more often than the control.

7-2-2 Wind model

Advanced wind models. The use of a more complicated wind model or higher orders Markov models might capture the dynamics of the wind better. In real life, wind speed depends on many external physical factors, which could also be incorporated as additional measurements or (filtered) noise.

Seasonality. More research can go into seasonality and other effects on the wind data. It might be worth considering to create a different model per season, or an adaptive model that updates the dynamics in real-time.

Drawbacks of Markov process prediction. Using a Markov process to do predictions gives some approximations and complexities to the forecast. A discretisation has to be done on the wind speed space, and an order of the Markov chain has to be chosen. This approach is limited by the computational power of the system and the available data to create a transition matrix.

Data availability. The method used for wind prediction relies on accurate and credible data. For the actual application of the turbine, different wind conditions might be applicable. Different seasons, locations, and climate change can all cause different weather patterns, making predictions difficult. In this thesis, data on a short timescale is used, but this might not show the full picture. For the Markov table to be as accurate as possible, it may seem that incorporating more data would improve the accuracy. However, the relevance of historical data from previous years remains uncertain. This is a difficult trade-off that requires more research.

7-2-3 Control methods

Feature functions. In ADP, the feature functions can be experimented with. Using more or different feature functions, combined with non-linear functions or weights can be worth exploring.

Order. More research can go into the optimal order of the control system, figuring out a limit where overfitting leads to diminishing results.

Multiple objectives. In this thesis, the objective was brought back to one variable: the total revenue in euros. However, the ideal control would balance water and electricity production with wear and tear, and maybe even more factors. Using a multi-objective based control method could address this.

Approximations. MPC and ADP require an approximation of an exact model. This is done either by limiting to a finite time or by approximating a policy using basis functions. Both of these approaches result in approximations, possibly leading to shortcomings in the control policy.

Fault tolerance. In the controller, it is assumed that a wind speed is always measured, and it is reasonable. If a sensor fails or no wind speed data is transmitted, the controller is incapable of dealing with such situations. Creating fault-tolerant control is something that would need more attention before deployment of the controller, especially considering the usage offshore.

7-2-4 Experimentation

Real-life experimentation. Performing real-life tests and field experiments is a necessary next step to validate the control strategies and models. Out of all options for future work, this will probably give the most insights.

Appendix A

Optimal control

An optimal control is proposed by Greco et al. in [3]. In Figure A-1 The steady state operation is shown for this control. The different control points are noted from 0 to D.

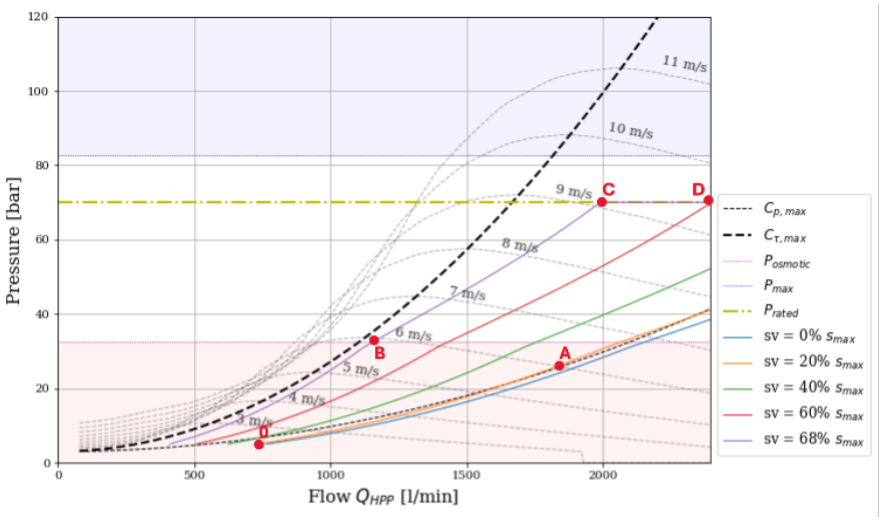


Figure A-1: Pressure-flow graph

For wind speeds below 6 m/s, the system follows the orange line (0-A in Figure A-1), as water production is not possible since the pressure is not high enough. The system is restricting flow to the membranes (FCV is closed), and the SV towards the Pelton is set to close just twenty percent. This value is chosen as it coincides best with the $C_{p,max}$, which means that the power transmission is optimal. Between 0 and A, the system relies on *passive control* as explored in [39]. Due to the nature of the SV, no active control is necessary.

As the wind reaches a speed of 6 m/s, the pressure is not high enough for reverse osmosis with this flow. On the other hand, if the SV is closed more, the pressure builds up and is able to just reach above osmotic pressure (Point B in Figure A-1). This means that Reverse osmosis can

start from a wind speed of 6 m/s and we are now in water production mode.

From point B to C, the FCV is controlled to keep the system from moving left of the $C_{\tau,max}$ line, to keep the system stable. Meanwhile, the SV is kept at a constant 68 percent closure. This prevents the system from having 2 active controllers interfering with each other. As the wind speeds rise, the system follows the purple line.

Although the maximum pressure of the HPP is not reached at point C, the membranes of the RO unit have reached rated pressure. To prevent the membranes from being damaged, the pressure is controlled by opening the SV. The FCV is kept constant during this time (C-D in Figure A-1).

The HPP is limited to a rated flow of 2394 l/min. Therefore, at point D, the blades start pitching out to decrease the amount of hydraulic power. Up until cut-out wind speed at 20 m/s, the system keeps operating at the flow and pressure of point D.

Although extensive, there are a few gaps with this current approach. This is an idealized control and wind speed does not usually increase or decrease linearly or predictably. The wind might oscillate around 6 m/s, where this control would repeatedly turn the water production on and off. Also, the startup and shutdown of the system (around point 0 in Figure A-1) are not included. During the last test setup at the Maasvlakte, these decisions were carefully made by hand to test the control at the valves. As this has proved itself, the control between the states is now investigated here.

Appendix B

Table of actions

State s_k	**Available actions** $\mathbb{U}(s_k)$
Hibernation (hibernation timer $<$ 20 min)	Stay
Hibernation (hibernation timer $>$ 20 min)	Stay, Stop Hibernation
Standby (clean)	Stay, Start Hibernation, Start Electricity
Electricity Production (clean)	Stay, Stop Electricity, Long Start Water
Water Production	Stay, Stop Water
Standby (contaminated) (contamination timer $<$ 30 min)	Stay, Start Electricity, Rinse
Electricity Production (contaminated) (contamination timer $<$ 30 min)	Stay, Stop Electricity, Start Water, Rinse
Standby/Electricity Production (contaminated) contamination timer = 30 min	Rinse

Table B-1: Table of actions

Appendix C

Pseudo Code

Algorithm 1 Construct Higher-Order Markov Transition Table for Wind

Require: Wind series $w_1, w_2, \ldots, w_T$, bin edges B, order k
Ensure: Transition tensor T of shape $(N, \ldots, N)$ with $k+1$ dimensions

Discretize w_t into bins using edges B
Initialize count tensor $C[i_1, \ldots, i_k, j] \leftarrow 0$ for all bins
for $t \leftarrow 0$ to $T - k - 1$ **do**
    $(i_1, \ldots, i_k) \leftarrow$ bin indices of $(w_t, \ldots, w_{t+k-1})$
    $j \leftarrow$ bin index of w_{t+k}
    $C[i_1, \ldots, i_k, j] \leftarrow C[i_1, \ldots, i_k, j] + 1$
end for
for all $(i_1, \ldots, i_k)$ **do**
    Normalize $C[i_1, \ldots, i_k, :]$ to get probabilities $T[i_1, \ldots, i_k, :]$
end for
return T

Algorithm 2 Define MDP Components for Turbine Control

Require: Discretized wind model T, system state descriptors
Ensure: Definition of (S, A, P, R) for DP

Define state: $s = (\text{turbine\_state}, \text{wind}_{1:k}, \text{hib}, \text{cont})$
Define actions: $a \in \text{ValidActions}(s)$
Define transitions: $P(s' \mid s, a)$ using T and environment rules
Define reward: $R(s, a) = -\text{get\_loss}(s, a)$

Algorithm 3 Stochastic Dynamic Programming for Turbine MDP

Require: Horizon H, MDP (S, A, P, R)
Ensure: Value function V, policy π

Initialize $V_{\text{next}}(s) \leftarrow 0$ for all terminal states
Initialize action cache $\mathcal{C} \leftarrow \emptyset$
for $t \leftarrow H-1$ to 0 **do**
    Initialize $V_{\text{new}} \leftarrow \{\}$
    for all $s \in V_{\text{next}}$ **do**
        $v^* \leftarrow \infty$, $a^* \leftarrow$ None
        for all $a \in \text{ValidActions}(s)$ **do**
            if $t = H-1$ **then**
                next_states $\leftarrow \text{GetNextStates}(s, a)$
                $\mathcal{C}[s, a] \leftarrow$ next_states
            else
                next_states $\leftarrow \mathcal{C}[s, a]$
            end if
            $\mathbb{E} \leftarrow \sum_{(p,s')} p \cdot V_{\text{next}}(s')$
            $v \leftarrow \frac{R(s,a)+(H-t)\cdot\mathbb{E}}{H-t+1}$
            if $v < v^*$ **then**
                $v^* \leftarrow v$, $a^* \leftarrow a$
            end if
        end for
        $V_{\text{new}}(s) \leftarrow v^*$
        $\pi(s) \leftarrow a^*$
    end for
    if $\max_s |V_{\text{new}}(s) - V_{\text{next}}(s)| < \epsilon$ **then**
        break ▷ Converged
    end if
    $V_{\text{next}} \leftarrow V_{\text{new}}$
end for
return V_{next}, π

Appendix D

Wind performances

We test ARIMA, Kalman Filtering and Markov processes on their ability to predict wind speeds.

D-1 Test setup

Every method follows the same pipeline: The parameters are fitted on the training data, using hyperparameters which are compared on the validation data. The best hyperparameters are chosen and used to train the parameters on the combined dataset of training and validation data before being tested on the unseen data in the testing dataset. Here the three methods are compared.
For the Markov process, the parameters are the $p_{i,j}$ in the transition table, and the hyperparameter is the order m of the chain.
The ARIMA model has θ_i and σ_j as parameters determined in training, and (p, d, q) as hyperparameters.
Lastly, the Kalman filtering performs system identification to get matrices A and C as parameters,

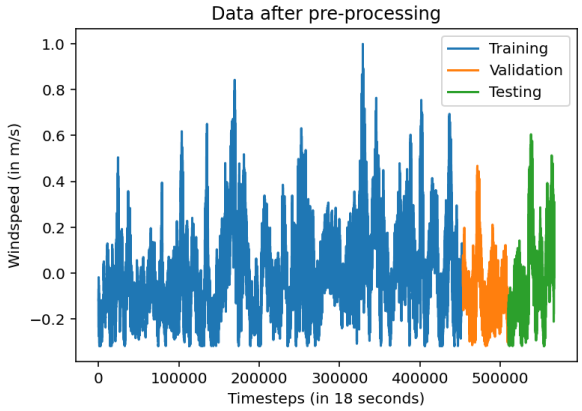


Figure D-1: Data used for the wind predictions, totalling to 1 month of wind data

and has the size of the state L as hyperparameter.

D-2 Performance metric

We test our methods by comparing Root Mean Squared Error (RMSE)[40]. This is given as

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(\tilde{y}_t - y_t)^2} \tag{D-1}$$

Here the prediction is denoted by $\tilde{y}$ and the actual observation is y. We use a horizon of n datapoints to compare the prediction. In our case this is 200, which is equal to one hour of data. Another measure we use is the Mean Interval Score (MIS).following [40], This is defined as

$$MIS(\alpha) := (l_b - u_b) + \frac{2}{\alpha}\begin{cases} l_b - y & y < l_b \\ 0, & l_b < y < u_b \\ y - u_b & u_b < y \end{cases} \tag{D-2}$$

Here α denotes the confidence level of the prediction interval. For our application we look at a 95% and 68% prediction interval, coinciding with one and two variances. This results in an α of 0.05 and 0.32.
The values of l_b and u_b denote the lower and upper bound of the interval. When the observation is within this interval, no score is added. However, if it is outside of the interval, the score is increased proportional to the distance to the closest bound. This gives an intuitive appeal to measure both accuracy and precision. The lower the MIS score, the better the performance. For our case of a longer horizon, we simply take the average MIS over time.
The RMSE score is a standard metric giving insight into the precision of the prediction. However, this does not allow for measuring the variability of a probabilistic measure. For this, the MIS is a better fit, giving more weight to the size of the confidence region.

D-3 Results

The validation step finds us that a Markov order of 5, ARIMA(1,1,3), ARIMA(2,1,3) and a state size of 1 or 2 gives the best results for the Kalman Filter. All hyperparameter results from the validation step, these results can be found in Appendix E.
Re-training these models on both training and validation data allows us to test the three models against each other on the testing data. We also compared our prediction to Persistence, that is, using the latest value as a prediction for all values. The RMSE and MIS of all methods are shown in Table D-1.

D-4 Conclusion

Although the best results seem to come from ARIMA(2,1,3) according to this table, the difference between methods is very small. This means that more research is probably needed. Note how

Method	RMSE	MIS68	MIS95
Markov(5)	0.0584	0.16	0.29
ARIMA(2,1,3)	0.0486	0.14	0.24
ARIMA(1,1,3)	0.05223	0.15	0.27
Kalman(1)	0.05901	-	-
Kalman(2)	0.09871	0.15	0.27
Persistence(naive method)	0.06513	-	-

Table D-1: Comparison of the methods

almost all methods manage to beat Persistence though. Also, remember that the Markov method gives a better probabilistic prediction, instead of a mean and variance. This makes Markov a more desirable method in terms of control and risk management.

Appendix E

Table of Performances

Markov Order	RMSE	MIS_68	MIS_95
1	0.125760	0.373594	0.855555
2	0.097483	0.310535	0.714383
3	0.077824	0.267865	0.601183
4	0.065724	0.237610	0.520504
5	**0.060807**	**0.218061**	**0.476738**

Table E-1: Performance of the Markov Model for Different Orders

p, d, q	RMSE	MIS_68	MIS_95
ARIMA(1,0,0)	0.119947	0.382267	0.653018
ARIMA(1,0,1)	0.051063	0.214932	0.409821
ARIMA(1,0,2)	0.051776	0.180717	0.324812
ARIMA(1,0,3)	0.046189	0.166709	0.301716
ARIMA(1,1,0)	0.044587	0.488213	0.961452
ARIMA(1,1,1)	0.042547	0.169415	0.316843
ARIMA(1,1,2)	0.042780	0.152771	0.275900
ARIMA(1,1,3)	0.042739	**0.150023**	**0.268055**
ARIMA(2,0,0)	0.095838	0.328808	0.601925
ARIMA(2,0,1)	0.061762	0.190449	0.305426
ARIMA(2,0,2)	0.068597	0.208304	0.328008
ARIMA(2,0,3)	0.047705	0.174244	0.318369
ARIMA(2,1,0)	0.042687	0.389167	0.765188
ARIMA(2,1,1)	0.042791	0.157713	0.288362
ARIMA(2,1,2)	0.042770	0.160894	0.295123
ARIMA(2,1,3)	**0.042258**	0.151183	0.272687
ARIMA(3,0,0)	0.078833	0.292391	0.553262
ARIMA(3,0,1)	0.048268	0.158015	0.272088
ARIMA(3,0,2)	0.050641	0.168456	0.293724
ARIMA(3,0,3)	0.054224	0.171452	0.287428
ARIMA(3,1,0)	0.043191	0.332218	0.652079
ARIMA(3,1,1)	0.042750	0.153871	0.277941
ARIMA(3,1,2)	0.042260	0.155964	0.284596
ARIMA(3,1,3)	0.042274	0.167921	0.313105

Table E-2: Performance of the ARIMA Model for Different Hyperparameters (p, d, q)

L (State Size)	RMSE	MIS_68	MIS_95
Kalman(1)	**0.045721**	0.543805	1.087609
Kalman(2)	0.108232	**0.351905**	**1.0275176**
Kalman(3)	0.120872	0.417792	1.295632
Kalman(4)	0.126459	0.441921	1.301793
Kalman(5)	0.130008	0.461649	1.403410
Kalman(6)	0.131779	0.468146	1.367391
Kalman(10)	0.136381	0.490489	1.418039
Kalman(25)	0.140097	0.506550	1.421857

Table E-3: Performance of the Kalman Model for Different State Sizes (L)

Appendix F

Simulation Parameters

F-1 Wind Model Parameters

- **Markov order:** This is yet to be determined dynamically during simulations. The order n of the markov model is varied to compare performance.
- **Discretisation :** The wind speed data is discretised in 41 wind speed bins, distributed using an equal-frequency approach. The distribution is as shown in Figure 3-6.

F-2 Turbine Model Parameters

- **Electricity price:** For the Electricity price, the current price of €0.09 per kWh is taken. This is used for both the electricity that is used and the electricity generated by the turbine.
- **Water price:** The water price is taken as €1.13 per m^3. This is the current price of drinking water in the Netherlands.
- **Energy overheads:** Electrical costs of both being in a state, and performing transitions like rinsing and starting electricity are assumed to be known and constant. This is still a rough estimation, but stems from earlier experimentations of the DOT 500kW Pilot Reverse Osmosis (DOT500PRO).
  - Hibernation: 0.462 kWh
  - Standby: 11.72 kWh
  - Rinsing: 17.6 kWh
  - Start Electricity: 78 kWh

  The start of electricity production involves generating an initial speed of the Pelton turbine, as blasting water on the Pelton turbine is inefficient and can damage the Pelton wheel if it is at a standstill. This does mean that a substantial electricity price is coupled for this

transition. Rinsing is also a costly operation, but the other transitions are more complex to evaluate. They are modelled without a cost for simplicity.

- **Contamination penalty:** A small penalty of €0.01 per hour is subtracted from the reward function to discourage being in the contaminated state.
- **Penalty for water production at low wind:** As the Reverse Osmosis (RO) system can suffer greatly from being in water production at insufficient wind speeds, a penalty of €–1000 per hour is implemented.
- **Operational constraints:** Two operational constraints were identified in the DOT500PRO system. These are:
  - Hibernation minimum duration: 20 minutes.
  - Maximum contamination time: 30 minutes

Using the turbine model parameters, a revenue function can be created, as is shown in Figure F-1. Here, the 6 states and their revenue are graphed against the wind speed. Note that this assumes that the current action is 'Stay'.

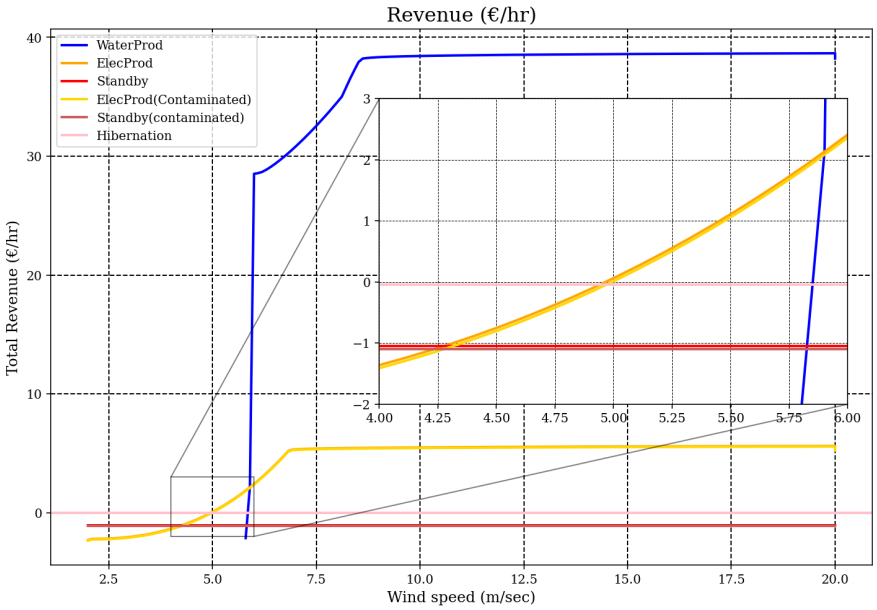


Figure F-1: Revenue Function, assuming $u_k = Stay$

F-3 Controller Parameters

F-3-1 Threshold Control

- **Wind speed thresholds (m/s):** Thresholds for the heuristic controllers were based on the results of Stochastic Dynamic Programming (SDP) with Markov order 1. This performs best in the in-model tests.

- Stop Hibernation 4.9 m/s
- Start Electricity 4.4 m/s
- Stop Electricity 2.9 m/s
- Long Start Water 6.1 m/s
- Start Water 6.6 m/s
- Stop Water 5.8 m/s
- Rinse 3.5 m/s

F-3-2 Model Predictive Control (MPC)

- **Horizon:** As the longest interaction in the model, the contamination time, is already 100 time steps, the horizon needs to be ≥ 100 time steps (30 minutes). The maximum length of the horizon is still to be determined in simulation, but keep in mind that the forecast converges to some steady state, from where more predictions are no longer necessarily beneficial, for instance the horizon for $n = 3$ reaches steady state in 300 steps[1].
- **Terminal cost:** The terminal cost is taken as zero for the sake of simplicity, for now.

F-3-3 Stochastic Dynamic Programming (SDP)

- **Convergence criterion:** For the value iteration algorithm used in SDP, convergence is assumed when the difference $\delta < 10^{-4}$.

F-3-4 Approximate Dynamic Programming (ADP)

- **Features:** Feature selection is an essential part of Approximate Dynamic Programming (ADP). The number of features is dependent on the order n of the Markov process. All of these features are normalised. The chosen features are:
  - One-hot turbine state (6 types)
  - Normalized wind sequence (n values)
  - Wind differences, clipped to [-3, 3] and normalized ($n - 1$ values)
  - Hibernation tracker / 66
  - Contamination tracker / 100
  - Wind prediction expectation and median
  - State-wind interaction terms (element-wise product) (6 values)
- **Optimiser:** Adam is chosen for its stability abilities.
- **epsilon-adam** 1e-8
- **adam beta values** beta1 = 0.9, beta2 = 0.999

[1] see Chapter 3

- **Learning methods:** For the learning, a Value iteration approach was taken, using TD(0)
- **Policy strategy:** The action in training is a trade-off between exploration and exploitation. We use an ϵ-greedy algorithm with $\epsilon = 0.05$
- **Initial learning rate:** 0.05
- **Num episodes:** 1000
- **steps per episode:** 500
- **discount factor** γ: 0.99

Bibliography

[1] P. H. Gleick, Pacific Institute for Studies in Development, Environment, and Security, Stockholm Environment Institute, and Pacific Institute for Studies in Development, Environment, and Security, Eds., *Water in crisis: a guide to the world's fresh water resources*. New York, NY: Oxford Univ. Press, 1993.

[2] "Reverse Osmosis (RO) - Definition, Working Principle, Process, Experiment, Advantages, Disadvantages of Reverse Osmosis." [Online]. Available: https://byjus.com/chemistry/reverse-osmosis/

[3] F. Greco, D. De Bruycker, A. Velez-Isaza, N. Diepeveen, and A. Jarquin-Laguna, "Preliminary design of a hydraulic wind turbine drive train for integrated electricity production and seawater desalination," *Journal of Physics: Conference Series*, vol. 1618, no. 3, p. 032015, Sep. 2020. [Online]. Available: https://iopscience.iop.org/article/10.1088/1742-6596/1618/3/032015

[4] F. Greco, S. Heijman, and A. Jarquin-Laguna, "Integration of Wind Energy and Desalination Systems: A Review Study," *Processes*, vol. 9, no. 12, p. 2181, Dec. 2021. [Online]. Available: https://www.mdpi.com/2227-9717/9/12/2181

[5] Y. Xie, C. Li, M. Li, F. Liu, and M. Taukenova, "An overview of deterministic and probabilistic forecasting methods of wind energy," *iScience*, vol. 26, no. 1, p. 105804, Jan. 2023. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2589004222020776

[6] UN Water, Ed., *Water for prosperity and peace*, ser. The United Nations World Water Development Report. Paris: UNESCO, 2024, no. 2024.

[7] C. He, Z. Liu, J. Wu, X. Pan, Z. Fang, J. Li, and B. A. Bryan, "Future global urban water scarcity and potential solutions," *Nature Communications*, vol. 12, no. 1, p. 4667, Aug. 2021. [Online]. Available: https://www.nature.com/articles/s41467-021-25026-3

[8] A. K. Biswas and C. Tortajada, "United Nations water conferences: reflections and expectations," *International Journal of Water Resources Development*, vol. 39, no. 2, pp. 177–183, Mar. 2023. [Online]. Available: https://www.tandfonline.com/doi/full/10.1080/07900627.2023.2176655

[9] M. Rodell, A. Barnoud, F. R. Robertson, R. P. Allan, A. Bellas-Manley, M. G. Bosilovich, D. Chambers, F. Landerer, B. Loomis, R. S. Nerem, M. M. O'Neill, D. Wiese, and S. I. Seneviratne, "An Abrupt Decline in Global Terrestrial Water Storage and Its Relationship with Sea Level Change," *Surveys in Geophysics*, Nov. 2024. [Online]. Available: https://link.springer.com/10.1007/s10712-024-09860-w

[10] M. Qasim, M. Badrelzaman, N. N. Darwish, N. A. Darwish, and N. Hilal, "Reverse osmosis desalination: A state-of-the-art review," *Desalination*, vol. 459, pp. 59–104, Jun. 2019. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0011916418325037

[11] A.-C. Pytharouliou, "Numerical simulation and operating strategy of a wind-powered electricity and freshwater production system," Thesis, Delft University of Technology, 2023.

[12] P. Cabrera, M. Folley, and J. A. Carta, "Design and performance simulation comparison of a wave energy-powered and wind-powered modular desalination system," *Desalination*, vol. 514, p. 115173, Oct. 2021. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0011916421002447

[13] M. A. Abdelkareem, M. Al Radi, M. Mahmoud, E. T. Sayed, T. Salameh, R. Alqadi, E.-C. A. Kais, and A. Olabi, "Recent progress in wind energy-powered desalination," *Thermal Science and Engineering Progress*, vol. 47, p. 102286, Jan. 2024. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S245190492300639X

[14] C. Matos, P. Cabrera, J. A. Carta, and N. Melián-Martel, "Wind-Powered Desalination on Islands: A Review of Energy–Water Pathways," *Journal of Marine Science and Engineering*, vol. 12, no. 3, p. 464, Mar. 2024. [Online]. Available: https://www.mdpi.com/2077-1312/12/3/464

[15] J. Carta, J. González, and V. Subiela, "The SDAWES project: an ambitious R&D prototype for wind-powered desalination," *Desalination*, vol. 161, no. 1, pp. 33–48, Feb. 2004. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0011916404900380

[16] H. Frisinger, "meteorology before Aristotle," *Bulletin of the American Meteorological Society*, vol. 52, no. 11, pp. 1078–1080, 1971, publisher: American Meteorological Society. [Online]. Available: http://www.jstor.org.tudelft.idm.oclc.org/stable/26253850

[17] J. Duan, H. Zuo, Y. Bai, J. Duan, M. Chang, and B. Chen, "Short-term wind speed forecasting using recurrent neural networks with error correction," *Energy*, vol. 217, p. 119397, Feb. 2021. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0360544220325044

[18] X. Li, K. Li, S. Shen, and Y. Tian, "Exploring Time Series Models for Wind Speed Forecasting: A Comparative Analysis," *Energies*, vol. 16, no. 23, p. 7785, Nov. 2023. [Online]. Available: https://www.mdpi.com/1996-1073/16/23/7785

[19] W.-C. Tsai, C.-M. Hong, C.-S. Tu, W.-M. Lin, and C.-H. Chen, "A Review of Modern Wind Power Generation Forecasting Technologies," *Sustainability*, vol. 15, no. 14, p. 10757, Jul. 2023. [Online]. Available: https://www.mdpi.com/2071-1050/15/14/10757

[20] Y. Wang, R. Zou, F. Liu, L. Zhang, and Q. Liu, "A review of wind speed and wind power forecasting with deep neural networks," *Applied Energy*, vol. 304, p. 117766, Dec. 2021. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0306261921011053

[21] X. Liu, Z. Lin, and Z. Feng, "Short-term offshore wind speed forecast by seasonal ARIMA - A comparison against GRU and LSTM," *Energy*, vol. 227, p. 120492, Jul. 2021. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0360544221007416

[22] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, Mar. 1960. [Online]. Available: https://asmedigitalcollection.asme.org/fluidsengineering/article/82/1/35/397706/A-New-Approach-to-Linear-Filtering-and-Prediction

[23] E. Bossanyi, "Short-Term Wind Prediction Using Kalman Filters," *Wind Engineering*, vol. 9, no. 1, pp. 1–8, 1985, publisher: Sage Publications, Ltd. [Online]. Available: http://www.jstor.org.tudelft.idm.oclc.org/stable/43749025

[24] Z. Huang and Z. Chalabi, "Use of time-series analysis to model and forecast wind speed," *Journal of Wind Engineering and Industrial Aerodynamics*, vol. 56, no. 2-3, pp. 311–322, May 1995. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/016761059400093S

[25] P. Louka, G. Galanis, N. Siebert, G. Kariniotakis, P. Katsafados, I. Pytharoulis, and G. Kallos, "Improvements in wind speed forecasts for wind power prediction purposes using Kalman filtering," *Journal of Wind Engineering and Industrial Aerodynamics*, vol. 96, no. 12, pp. 2348–2362, Dec. 2008. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0167610508001074

[26] H. Liu, H.-q. Tian, and Y.-f. Li, "Comparison of two new ARIMA-ANN and ARIMA-Kalman hybrid methods for wind speed prediction," *Applied Energy*, vol. 98, pp. 415–424, Oct. 2012. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0306261912002875

[27] K. Chen and J. Yu, "Short-term wind speed prediction using an unscented Kalman filter based state-space support vector regression approach," *Applied Energy*, vol. 113, pp. 690–705, Jan. 2014. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0306261913006600

[28] O. B. Shukur and M. H. Lee, "Daily Wind Speed Forecasting Through Hybrid AR-ANN and AR-KF Models," *Jurnal Teknologi*, vol. 72, no. 5, Jan. 2015. [Online]. Available: https://journals.utm.my/index.php/jurnalteknologi/article/view/3946

[29] S.-h. Hur, "Short-term wind speed prediction using Extended Kalman filter and machine learning," *Energy Reports*, vol. 7, pp. 1046–1054, Nov. 2021. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2352484720317212

[30] E. Arslan Tuncar, S. Saglam, and B. Oral, "A review of short-term wind power generation forecasting methods in recent technological trends," *Energy Reports*, vol. 12, pp. 197–209, Dec. 2024. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2352484724003603

[31] A. Shamshad, M. Bawadi, W. Wanhussin, T. Majid, and S. Sanusi, "First and second order Markov chain models for synthetic generation of wind speed time series," *Energy*, vol. 30, no. 5, pp. 693–708, Apr. 2005. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0360544204002609

[32] A. Carpinone, M. Giorgio, R. Langella, and A. Testa, "Markov chain modeling for very-short-term wind power forecasting," *Electric Power Systems Research*, vol. 122, pp. 152–158, May 2015. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0378779614004714

[33] M. Jacobsen, "Short-term wind power prediction based on Markov chain and numerical weather prediction models: A case study of Fakken wind farm," Master's thesis, The Arctic University of Norway, Jun. 2014.

[34] H. Fan, X. Zhang, S. Mei, and J. Zhang, "A Markov Regime Switching Model for Ultra-Short-Term Wind Power Prediction Based on Toeplitz Inverse Covariance Clustering," *Frontiers in Energy Research*, vol. 9, p. 638797, Mar. 2021. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fenrg.2021.638797/full

[35] L. Zhou, X. Zhou, H. Liang, M. Huang, and Y. Li, "Hybrid Short-Term Wind Power Prediction Based on Markov Chain," *Frontiers in Energy Research*, vol. 10, p. 899692, May 2022. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fenrg.2022.899692/full

[36] Y. Zhao, L. Ye, Z. Wang, L. Wu, B. Zhai, H. Lan, and S. Yang, "Spatio-temporal Markov chain model for very-short-term wind power forecasting," *The Journal of Engineering*, vol. 2019, no. 18, pp. 5018–5022, Jul. 2019. [Online]. Available: https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/joe.2018.9294

[37] "Wind - lidar wind profiles measured at North Sea wind farm TenneT platforms 1 second raw data - KNMI Data Platform." [Online]. Available: https://dataplatform.knmi.nl/dataset/windlidar-nz-wp-platform-1s-1

[38] R. Bellman, *Dynamic programming*. Princeton, NJ: Princeton Univ. Pr, 1984.

[39] N. F. B. Diepeveen and A. Jarquin-Laguna, "Wind tunnel experiments to prove a hydraulic passive torque control concept for variable speed wind turbines," *Journal of Physics: Conference Series*, vol. 555, p. 012028, Dec. 2014. [Online]. Available: https://iopscience.iop.org/article/10.1088/1742-6596/555/1/012028

[40] T. Gneiting and A. E. Raftery, "Strictly Proper Scoring Rules, Prediction, and Estimation," *Journal of the American Statistical Association*, vol. 102, no. 477, pp. 359–378, Mar. 2007. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1198/016214506000001437

Glossary

List of Acronyms

DOT500PRO	DOT 500kW Pilot Reverse Osmosis
HPP	High-pressure pump
RO	Reverse Osmosis
ERD	Energy Recovery Device
SV	Spear Valve
FCV	Flow Control Valve
ERD	Energy Recovery Device
NWP	Numerical Weather Prediction
MDP	Markov Decision Process
ADP	Approximate Dynamic Programming
SDP	Stochastic Dynamic Programming
MPC	Model Predictive Control
RMSE	Root Mean Squared Error
MIS	Mean Interval Score
ACF	AutoCorrelation Function
PACF	Partial AutoCorrelation Function

List of Symbols

$R(s_k, u_k, w_k)$	Revenue function
$\ell(s_k, u_k, w_k)$	Loss function, equal to $-R(s_k, u_k, w_k)$
$\mathbb{S}$	Turbine state space
$\mathbb{U}$	Action space

$\mathbb{W}$	Wind space
$\mathbb{X}$	State space combining $\mathbb{S}$ and $\mathbb{W}^n$
Π	Policy function space
$\pi(s_k, w_k, \ldots, w_{k-n+1})$	Policy function
$f(s_k, u_k)$	Turbine state update function
s_k	Turbine state at time k
u_k	Action at time k
w_k	Wind speed at time k
x_k	State vector combining s_k and w_k