Faculty of Science and
Bio-Engineering Sciences
Department of Computer Science

A Multi-Agent Reinforcement Learning Approach to Wind Farm Control

Dissertation submitted in fulfilment of the requirements for the degree of Doctor of Science: Computer science

Timothy Verstraeten

Brussels, March 2021

Promotors: Prof. Dr. Ann Nowé
Prof. Dr. Jan Helsen

List of Jury Members

Prof. Dr. Elisa Gonzalez Boix	Vrije Universiteit Brussel (Chair)
Prof. Dr. Bernard Manderick	Vrije Universiteit Brussel (Secretary)
Prof. Dr. Ann Nowé	Vrije Universiteit Brussel (Promotor)
Prof. Dr. Jan Helsen	Vrije Universiteit Brussel (Co-promotor)
Prof. Dr. Nikolaos Deligiannis	Vrije Universiteit Brussel
Prof. Dr. Jens Kober	Delft University of Technology
Prof. Dr. Daniel Kudenko	Leibniz University Hannover
Prof. Dr. Amir R. Nejad	Norwegian University of Science and Technology

Summary

Recently, we drastically shifted our energy production toward renewable energy sources, due to the pressing matter of climate change and the limited supply of fossil fuels. While offshore wind farms are an important driver toward renewable energy generation, their maintenance costs need to be significantly reduced to render them sustainable. Major causes of the elevated maintenance costs are failures due to unanticipated stress on the wind turbine components. Therefore, the health status of a wind turbine needs to be accurately quantified and incorporated in wind farm control schemes, such that unnecessary stress on high-risk turbines can be prevented. As this health status is determined by a complex multi-dimensional spectrum of stress factors, AI-driven control strategies are necessary. However, current wind farm controllers that use AI-driven optimization techniques are not scalable to the size of contemporary wind farms. In this dissertation, we focus on developing optimal and scalable AI-driven control methods. We adopt a multi-tiered methodology, in which we investigate several properties of wind farms independently, and then consolidate the acquired insights in a wind farm control solution. First, we develop a control algorithm for multi-device systems, in which the devices have similar technical specifications. The control method uses a similarity-based data exchange mechanism to increase the confidence of the environment model for a specific learning agent, based on relevant data of similar devices. We demonstrate that the use of such a mechanism increases learning accuracy by reducing uncertainty and bias due to negative transfer. Second, we propose a control method for generic multi-agent systems with a sparse dependency structure. Specifically, we exploit this sparse structure to factorize large multi-agent systems and learn optimal control decisions in the factored representation. We demonstrate, both theoretically and empirically, that our method significantly reduces the learning complexity when considering sparse dependency graphs, and thus can handle

the combinatorial explosion with respect to the joint action space when dealing with large multi-agent systems. The developed control methods are applicable to a variety of multi-agent systems that contain similar agents or have a loosely-coupled structure. Finally, we combine the obtained insights on device similarity and sparse dependency structures and extend our approaches to a AI-driven wind farm controller that is scalable and optimal with respect to the complex cost-functions inherent to contemporary wind farms. We show that our method is capable of closing the gap between power demand and the produced farm-wide power, while still considering the penalties induced on high-risk turbines, by preventing stressful control decisions.

Samenvatting

Recent hebben we onze energie productie drastisch verschoven naar hernieuwbare energiebronnen, vanwege het probleem van klimaatverandering en het beperkte aanbod van fossiele brandstoffen. Hoewel windparken een belangrijke factor zijn voor de opwekking van hernieuwbare energie, moeten hun onderhoudskosten aanzienlijk worden verlaagd om ze duurzaam te maken. Grote bijdragers aan de verhoogde onderhoudskosten zijn storingen ten gevolge van onverwachte belasting van de windturbinecomponenten. Daarom moet de gezondheidstoestand van een windturbine nauwkeurig worden bepaald en in rekening gebracht worden bij de besturing van windparken om onnodige belasting van risicovolle turbines te voorkomen. Aangezien deze gezondheidstoestand wordt bepaald door een complex multidimensionaal spectrum van stressfactoren, zijn AI-gestuurde controlestrategieën noodzakelijk. De huidige besturingssystemen van windparken die AI-gestuurde optimalisatietechnieken gebruiken, zijn echter niet schaalbaar naar windparken met een groot aantal turbines. In dit proefschrift richten we ons op het ontwikkelen van optimale en schaalbare AI-gestuurde controlemethoden. We hanteren een gelaagde methodologie, waarbij we verschillende eigenschappen van windparken onafhankelijk onderzoeken en vervolgens de verworven inzichten consolideren in een controlestrategie voor windparken. Ten eerste ontwikkelen we een besturingsalgoritme voor systemen met meerdere apparaten, waarbij de apparaten vergelijkbare technische specificaties hebben. De controlemethode maakt gebruik van gelijkenis-gebaseerde data-uitwisseling om de onzekerheid op het omgevingsmodel voor een bepaalde leer-agent te verlagen, op basis van relevante gegevens van vergelijkbare apparaten. We laten zien dat het gebruik van een dergelijk mechanisme de leernauwkeurigheid vergroot door onzekerheid en negatieve overdracht van data te verminderen. Ten tweede construeren we een controlemethode voor generieke multi-agentsystemen met een beperkte afhankelijkheidsstructuur. Concreet

maken we gebruik van deze structuur om grote multi-agentsystemen te ontleden en optimale controlebeslissingen te leren in de ontbonden representatie. We demonstreren, zowel theoretisch als experimenteel, dat onze methode de leercomplexiteit aanzienlijk vermindert bij het beschouwen van structuren met beperkte afhankelijkheden, en dus de combinatorische explosie met betrekking tot een groot aantal agenten. De ontwikkelde controle methodes zijn toepasbaar in verschillende multi-agent systemen die gelijkaardige agenten of een beperkte afhankelijkheidsstructuur hebben. Ten slotte combineren we de verkregen inzichten over apparaatgelijkenis en schaarse afhankelijkheidsstructuren en breiden we onze aanpak uit tot een volwaardig AI-gestuurd windparkbesturingsmethode die schaalbaar en optimaal is met betrekking tot de complexe kostenfuncties die inherent zijn aan hedendaagse windparken. We laten zien dat onze methode in staat is om een balans te vinden tussen de vraag naar energie en de geproduceerde stroom in het windpark, terwijl we toch rekening houden met de kosten die worden opgelegd aan risicovolle turbines, door beslissingen die potentieel schade veroorzaken te voorkomen.

Acknowledgments

While I truly enjoyed the five-year long journey I followed, it would not have been possible without a few people.

First and foremost, I would like to express my gratitude to my supervisors, prof. dr. Ann Nowé and prof. dr. Jan Helsen, for granting me the opportunity to be part of their labs and for their guidance during this journey. Thank you both for your close guidance and for allowing me the scientific freedom to pursue what interested me most. I look forward to continue to collaborate with both of you in the near future.

Next, I want to thank my jury, for reading this dissertation in great detail, and for providing constructive criticism. Thanks prof. dr. Daniel Kudenko (Leibniz University Hannover), prof. dr. Jens Kober (Delft University of Technology), prof. dr. Amir Nejad (Norwegian University of Science and Technology), prof. dr. Nikolaos Deligiannis (VUB), prof. dr. Bernard Manderick (VUB) and the president of the jury, prof. dr. Elisa Gonzalez Boix (VUB). Your suggestions and feedback were inspiring, and significantly improved this work.

In addition, I want to thank dr. Pieter Libin and dr. Diederik Roijers for the close collaboration and guidance. The research outcomes in this dissertation would not have been so solid if not for your insights and experience, for which I am grateful.

I would like to thank the Flemish research foundation (FWO) for funding my research the past four years. This personal research grant supported me to develop new AI algorithms and gave me the opportunity to connect with other researchers world-wide.

Thanks to all the colleagues from the AI Lab and AVRG, for the fruitful collaborations and for creating a pleasant working environment. In particular, I want to speak out to a few of you. Felipe, thanks for all the refreshing conversations at work and the fun climbing sessions we had. Eugenio, thanks for all your useful insights on RL, meaning both 'Reinforcement

Learning' and 'Rocket League'. And, Pieter, thanks for all the projects that we jointly investigated. Your valuable input always made my work better and your scientific rigour made me become a better scientist.

Thanks to my friends. Especially Krista and Sergiu, for inviting me to their place on Friday evenings after a long week of work. I want to thank my parents, grandparents and sister for their support, throughout all my life. Finally, I want to thank my lovely partner, Roxana. "Baby, you're my forever girl."

Contents

Nomenclature

$\mathcal{B}\text{er}\,(p_s)$	Bernoulli distribution with success probability p_s, page 34
$\mathcal{B}\text{eta}\,(\alpha, \beta)$	beta distribution with α successes and β failures, page 34
$\mathcal{I}\{x\}$	indicator function, which evaluates to 1 if event x is true and evaluates to 0 if event x is false, page 34
$\mathbb{R}$	real numbers, page 34
$x, \boldsymbol{x}$	scalar, vector, page 36
$\mathbb{E}\,[\cdot]$	expectation operator, page 36
$\mathcal{A}$	arm/action space, page 36
R	reward function, page 36
$\mu(a)$	unknown expected reward of arm a, page 36
a_*	optimal arm/action, page 37
a_t	chosen arm/action at time t, page 37
π	policy, decision strategy, page 37
$\mathcal{R}(T, \pi)$	cumulative regret at time T when executing policy π, page 37
$\Delta(a)$	difference between the expected rewards of the optimal arm and arm a, page 37

$\hat{\mu}_{t-1}(a)$	empirical mean reward of arm a at time t, page 37
$n_{t-1}(a)$	number of pulls of arm a at time t, page 37
$Q_a(\cdot)$	prior distribution on mean reward of arm a at time t, page 38
$\mathcal{H}_{t-1}$	history of observations at time t, page 38
$p(\cdot)$	probability measure, page 39
$O(\cdot)$	Big-O notation, page 39
$\mathcal{M}$	Markov decision process, page 40
$\mathcal{S}$	state space, page 40
τ	transition model, page 40
γ	discount factor, page 40
V	value function, page 41
Q	Q-value function, page 41
σ	standard deviation, page 45
$\mathcal{GP}(\mu, k)$	Gaussian process with mean function μ and covariance kernel k, page 45
$k(\boldsymbol{x}, \boldsymbol{x'})$	kernel describing the covariance between the two random variables associated with inputs $\boldsymbol{x}$ and $\boldsymbol{x'}$, page 45
$K_{X,X'}$	matrix obtained by applying the covariance kernel pairwise on the elements in matrix X and matrix X', page 45
k_θ^{SE}	squared exponential covariance kernel with hyperparameters θ, page 46
$\mathbb{V}[\cdot]$	variance operator, page 48
S_{supp}	support states in Gaussian process reinforcement learning, page 48
$\boldsymbol{v}_{\mathsf{supp}}$	support values in Gaussian process reinforcement learning, page 48
$\mathcal{M}_{\mathsf{PoD}}$	Markov decision process for pool of devices, page 49
$\mathcal{T}$	set of transition models, page 49
$\mathbb{C}\mathsf{ov}[\cdot, \cdot]$	covariance operator, page 51

ϕ	weight parameter, page 51
I	identity matrix, page 51
F	weight parameter, page 51
ρ	number of groups in a multi-agent multi-armed bandit, page 68
$\mathcal{D}$	set of agents in a multi-agent multi-armed bandit, page 68
$\mathcal{D}^e$	e^{th} group of agents in a multi-agent multi-armed bandit, page 68
$\mathcal{A}^e$	local joint action space of group $\mathcal{D}^e$ in a multi-agent multi-armed bandit, page 68
$\mu^e(\boldsymbol{a}^e)$	unknown expected local reward of local joint action $\boldsymbol{a}^e$, page 68
G	coordination/dependency graph, page 68
$\widetilde{A}$	number of local joint actions in a multi-agent multi-armed bandit, page 76
$\log$	natural logarithm, page 78
$\|\|\cdot\|\|_1$	absolute-value norm of a vector, page 81
$\|\|\cdot\|\|_2$	Euclidean norm of a vector, page 81
$\mathcal{P}(\lambda)$	Poisson distribution with rate parameter λ, page 87
$\mathcal{G}\text{amma}(\alpha, \beta)$	gamma distribution with parameters α and β, page 87
$\mathcal{Z}$	operational zones or regimes, page 97
$\mathcal{W}$	set of wind turbines, page 97
$G(w)$	group of agents on which agent w depends according to the dependency graph G, page 97
$\mathcal{A}^{G(w)}$	local joint action space of group $G(w)$, page 97
$\mathrm{P}^w(\boldsymbol{a})$	power production of wind turbine w associated with local joint action $\boldsymbol{a}$, page 97
$\mathrm{L}^w(\boldsymbol{a})$	load-based penalty for wind turbine w associated with local joint action $\boldsymbol{a}$, page 97
P_{dem}	power demand, page 97

f_z	fraction of the power demand assigned to regime z, page 97
$\mathcal{N}(\boldsymbol{\mu}, K)$	multivariate normal distribution with mean vector $\boldsymbol{\mu}$ and covariance matrix K, page 100
$\mathcal{NIW}(\theta)$	normal-inverse-Wishart distribution with parameters θ, page 100
$\mathcal{D}\mathrm{ir}(\alpha, K)$	symmetric Dirichlet distribution with concentration parameter α and K components, page 100
$\mathbb{N}$	natural numbers, page 100
τ	torque, page 100
ω	angular rotor speed, page 100
$\mathcal{N}(\mu, \sigma)$	normal distribution with mean μ and standard deviation σ, page 105
d	wind direction, page 109
n_{risk}	number of high-risk wind turbines, page 109

Glossary

agent (semi-)autonomous AI-driven entity that learns to perform a task. 26

arm action taken by an agent in a multi-armed bandit. 36

bearing mechanical component that reduces friction between moving parts. 23

dynamic load substantial stress applied to mechanical components during short-term dynamic events. 22

fatigue weakening of a structural component due to the rotations performed in normal operating conditions. 22

gearbox wind turbine component that increases rotational speed from a low-speed rotor into a high-speed electrical generator. 23

likelihood probability of observing an outcome during an experiment, given the statistical model. 34

load stress applied to mechanical components. 22

posterior probability distribution over statistical model parameters after an experiment takes place and the outcome is observed. 34

prior probability distribution over statistical model parameters before an experiment takes place. 34

regret loss in reward when executing a sub-optimal action. 36

set-point threshold on the power production of a wind turbine. 93

torque tendency of a force to rotate the body to which it is applied. 93

wake area behind an operating wind turbine where the wind speed is reduced. 26

1 Introduction

As a scientific community, we recognize the effects of climate change [Solomon et al., 2007; Mukherjee et al., 2018]. Moreover, many European countries are phasing out nuclear power generation [Glacer, 2012]. This reality presents an opportunity to strategically increase the generation of renewable energy. In 2017, the worldwide installed wind capacity increased by 10% [Global Wind Energy Council, 2018]. Offshore wind is expected to play a large role, as the European offshore installations doubled in 2017 compared to 2016 [Fraile et al., 2018], and is still increasing [REN21 Secretariat, 2020]. A major hurdle to the development and adoption of renewable electricity sources is ensuring that these are cost competitive with fossil fuel energy sources. More specifically, they need to meet or exceed the ratio between expected production and lifetime costs, commonly defined as Levelized Cost Of Energy (LCOE) [Short et al., 2005].

For long-term viability, offshore wind must significantly improve its cost efficiency [Irawan et al., 2017]. Today, land-based wind energy is cost competitive at sites with strong wind [Wiser and Bolinger, 2015]. However, in Europe ideal onshore sites are already becoming exhausted, so expansion of wind farms will require placement in more remote locations, resulting in higher logistics costs [Treviño Cantú, 2011; Huang et al., 2017]. To reduce LCOE for wind energy, reliability of turbines must rapidly be improved, even in harsh environments [Clark and DuPont, 2018].

We consider reliability to be the likelihood that a turbine and its components will meet or exceed their prescribed design life [Ansell and Phillips, 1994]. In that sense, a robust machine is optimized to perform its function in any condition that it may encounter during

its lifetime. Overdesigning the turbine is not a viable solution to improve its reliability, as the capital cost of the turbine would increase significantly.

Truly robust implementation at the lowest cost requires that the loads, i.e., the stress induced on the turbine, are intimately understood. The case of dynamically changing multidimensional loads of such great magnitude, as found in wind turbines, is uncommon with other industrial machinery, and insufficiently understood [Struggl et al., 2015; Junior et al., 2017]. The loading conditions that lead to failure are often a combination of environmental dynamics (i.e., rapidly fluctuating wind speeds) and electrical grid-based events [Keller et al., 2016]. Site-specific conditions and corresponding turbine responses play a significant role, because the most significant failures occur only on a subset of turbines within the wind farm [Schiermeier, 2016]. Sites differ in wind resource, terrain, seasonal events, and atmospheric changes. Additionally, the electricity grid covers different subregions, each with specific utility interconnection requirements, and even changes in behavior within the region [Muljadi et al., 2007]. This poses significantly fluctuating loading conditions on the turbine that are at the origin of failure.

While it is tempting to consider a wind turbine as a generic entity, and impose operational control from this perspective, in reality each individual mechanical unit is unique. Therefore, maximum reliability can best be achieved by considering the wind farm as a data-compiling collective, and capturing the similarities between turbines that exist on a statistical level, while acknowledging the uniqueness in their specific operational behavior.

In this dissertation, we develop multi-agent control techniques that are scalable to the size of contemporary wind farms, and can handle the complex cost-functions associated with the multi-dimensional load spectrum. Although wind farm control research mainly focuses on static loads and power production [Knudsen et al., 2015; Boersma et al., 2017], the reduction of dynamic loads (i.e., loads induced by dynamic events, such as storms and grid events) through operational measures has received less attention. However, dynamic loading conditions have a significant impact on the probability of failure. Failures directly affect the availability of the turbines and reduce lifetime costs [Clark and DuPont, 2018]. Therefore, it is crucial to investigate and to understand the link between dynamic loads and failures, such that these measures can be implemented through targeted wind farm control strategies.

1 Loads and Lifetime of Wind Turbines

Contemporary wind farm controllers only optimize loads due to fatigue, i.e., the weakening of a structural component due to the rotations performed in normal operating conditions. These loads are well-understood and are already considered during the design phase of the turbine [Ansell and Phillips, 1994]. Still, turbine components can fail prematurely at only

5% of their design life due to various loading conditions [Greco et al., 2013]. In this section, we provide evidence that loads induced by specific dynamic events significantly increase the probability of premature failure, which reduces the design life of turbine components.

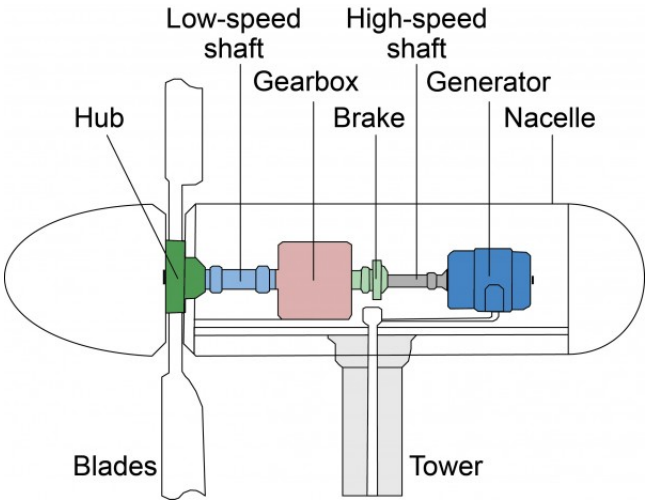


Figure 1.1: Diagram that depicts the inside of a turbine [U.S. Department of Energy's Office of Energy Efficiency and Renewable Energy, 2013].

A large majority of failures in the gearbox, i.e., the component that increases rotational speed from a low-speed rotor into a high-speed electrical generator (see Figure 1.1), are caused by microstructural cracks in the steel of the bearing [Link et al., 2011; National Renewable Energy Laboratory, 2016]. Bearings are mechanical elements that reduce friction between the moving parts in the gearbox. These elements are expected to fail due to fatigue at the end of their predicted lifetime, which is determined during the design phase. However, failure caused by microstructural cracks can result in a significant reduction of the bearing's design life (95%) [Greco et al., 2013]. Figure 1.2 shows an example of bearing damage. While unveiling the failure mechanism that causes such microstructural damage is still ongoing research [Pape et al., 2018], the associated loading conditions are not anticipated during the design phase.

Rolling bearings reduce friction between the attached moving rings through the means of rolling elements (e.g., balls), as shown in Figure 1.3. In a turbine's gearbox, such a bearing reduces friction between a rotating shaft and the stationary housing of the gearbox. Under normal operating conditions, the rollers' trajectories should be close to the optimal

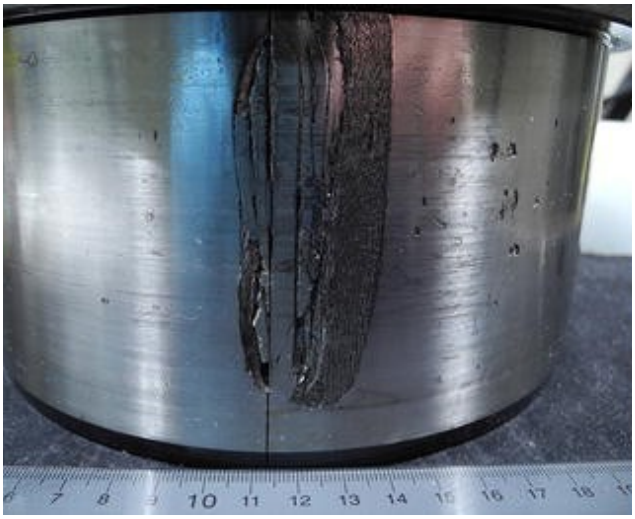
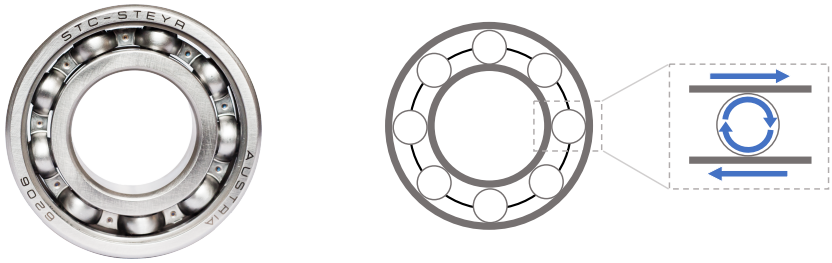


Figure 1.2: Example of bearing damage due to microstructural cracking caused by roller slipping [Gould and Aaron, 2015].

rolling path. For example, in Figure 1.3, if the inner ring is rotating and the outer ring is non-stationary, the balls should roll smoothly over the outer ring with little friction. Deviation from the optimal path induces slipping of the rollers, scratching the surface of the rings. Excessive slipping damages bearings because of sliding effects between the rollers and bearing rings, causing extensive wear and potentially crack initiation. This slipping can be caused by loads induced by certain control actions, such as pitching of the turbine's blades, start-up, emergency braking or rotating, during dynamic wind or grid events [Bruce et al., 2015; Greco et al., 2013].

To investigate the root causes of premature failure of wind turbine gearboxes, the USA's National Renewable Energy Lab (NREL) established the Gearbox Reliability Collaborative (GRC) project. In the context of this project, a series of experiments was conducted on a test setup that comprises a gearbox, mounted on a dynamometer that measures the force, power and speed of the gearbox [LaCava et al., 2011; Link et al., 2013]. The dynamometer has 4 degrees of freedom to control the load input to represent complex



(a) Ball bearing

(b) Diagram

Figure 1.3: Example of a small rolling bearing (a) with a diagram indicating the rolling elements and two bearing rings (b). The relative motion of the rings causes the balls to roll with little friction. (Photo of ball bearing by deel_de/Pixabay)

wind flow patterns. Additionally, this experimental setting allows to perform simulations of the electricity grid [Helsen et al., 2018a].

On the GRC setup, two fault scenarios were investigated, in which experimentally sampled loads of a real wind turbine were used as load inputs on the dynamometer. First, grid-loss events, i.e., a disconnection from the electricity grid during turbine operation, were simulated. As this causes a sudden stop of the turbine operation, slipping of the rolling bearings is investigated, which can lead to bearing damage [Guo et al., 2015; LaCava et al., 2013]. To test the effects of grid instability, the gearbox was subjected to several worst-case grid-loss events initiated at different power levels [Helsen et al., 2016b, 2018a]. During these events, the generator currents dropped to zero instantaneously. The system's response was consistent and resulted in multidimensional loading causing the rollers to slip excessively [Helsen et al., 2016a]. Because these slip conditions increase the risk of bearing damage, it follows that an electric event has the potential to cause unfavorable dynamic loading and possible degradation of the gearbox system. Second, bearing damage due to overloading caused by rotor moments was analyzed [Link et al., 2011]. While rotor moments significantly contribute to bearing failures, they are currently insufficiently understood. These moments are caused by overhung weight, excessive wind changes, orienting out of the wind direction and certain turbine control actions (e.g., emergency braking). These conditions lead to unfavorable loading on the turbine blades that, in many turbine designs, is subsequently passed to the gearbox [Helsen et al., 2014]. The inability to counteract bending of the turbine components, in combination with gear misalignment, causes overloading of the bearings and hence significantly accelerates bearing failure [LaCava et al., 2011; Keller et al., 2012].

These experiments demonstrate that a turbine's response to dynamic wind and grid events can initiate mechanical degradation of bearings, and ultimately lead to failure. Therefore, it is important to consider dynamic loads in wind farm control strategies.

2 Wind Farm Control and Reinforcement Learning

The design of wind farm controllers is typically grounded in domain knowledge about the physics of the system [Boersma et al., 2017; Siniscalchi-Minna et al., 2019]. For example, one can recognize that upstream turbines, with respect to the dominant wind direction, typically observe higher fatigue loads than downstream turbines [Jensen et al., 2016]. Therefore, control measures can be taken to reduce the power production of upstream turbines, while maximizing the power production of downstream turbines. While such heuristics drastically reduce the computational complexity of searching for optimal control strategies, they fail to capture the full complexity of the dynamically-changing multi-dimensional load spectrum. In order to develop advanced control strategies, it is necessary to consider the full load spectrum to increase reliability and sustainability of wind farms.

In contrast to physics-based heuristics, wind farm controllers based on artificial intelligence (AI) can learn strategies without requiring in-depth knowledge about the non-linear dependencies that exist between the wind turbines that make up the farm. An example of such a AI-driven framework is proposed by van Dijk et al. [2016], in which reinforcement learning techniques are used to search for the optimal rotor orientation to deflect wake away from downstream turbines. However, state-of-the-art control optimization algorithms scale poorly to larger wind farms, as the number of possible configurations grows exponentially with respect to the number of wind turbines. Therefore, we argue that a hybrid approach, combining both flexible AI-driven methods and physics-based domain knowledge, is key to guarantee both optimality and scalability of wind farm controllers.

In this dissertation, we focus on developing optimal and scalable AI-driven methods using reinforcement learning techniques [Sutton and Barto, 2018] and multi-armed bandit algorithms [Lattimore and Szepesvári, 2020]. Our techniques allow devices, or *agents*, to optimize their control strategy by interacting with the environment and assessing the quality of control decisions within this environment through various data sources. Such data-driven techniques do not require complete knowledge about the system dynamics, which render them useful in complex, non-linear settings. However, as state-of-the-art reinforcement learning and multi-armed bandit techniques typically require a large amount of data to learn good control strategies, a key challenge is to increase sample-efficiency, such that these techniques become viable in real-life applications [Yu, 2018]. In this regard,

Figure 1.4: Wake effect – Turbulent wind flow generated behind each turbine in the Horns Rev wind farm in Denmark [Christian Steiness/Vattenfall/Flickr, 2010].

we note that in many applications there is an abundance of domain knowledge available that can be used to guide the learning process.

In our case, wind farms exhibit properties that can be exploited to increase sample-efficiency. On the one hand, turbines are expected to be highly similar in terms of their operational behavior, as they typically have the same design specifications within a single wind farm [Feng and Shen, 2017]. This property allows wind farm controllers to perform data exchange between the turbines about suitable control strategies. However, there may be discrepancies in the turbines' conditions, e.g., due to production errors, their location within the wind farm or degradation [Staffell and Green, 2014]. Therefore, it is important to analyze the similarities between the turbines and perform data exchange between two turbines only when it is relevant and appropriate. On the other hand, due to the wind farm's topology, the dependency structure between turbines is expected to be sparse. As a wind farm is a large-scale multi-agent system, the wind farm controller has to consider all relevant combinations of turbine-specific actions. This space of combinations scales exponentially with the number of turbines. Therefore, it is important to exploit the sparsity of the wind farm structure to ensure for a scalable coordination mechanism. Indeed, within the wind farm's topology, it is possible to create a graph structure, describing the dependencies between turbines, and coordinate herein. For example, when upstream turbines extract energy from wind, the power production of neighboring downstream turbines is reduced. Figure 1.4 shows an example of this effect, in which we observe

that only a limited subset of downstream turbines is affected by the upstream turbines. Although considering these dependencies is essential to learn control strategies in a feasible manner, the optimization process is not trivial. Specifically, local optimization of the joint control decisions within the turbine groups is not possible, as a turbine may be part of multiple groups, and thus conflicting decisions in terms of local optimality may arise. Thus, global optimality of the wind farm controller must be ensured, while still allowing for local decision making.

3 Research Contributions

The ability to seamlessly include knowledge into the control mechanism is crucial to render AI-driven wind farm control tractable. In this dissertation, we focus on three properties that are inherent to wind farm technology. First, turbines are generally quasi-identical in design [Feng and Shen, 2017]. Exploiting the similarities that exist between turbines can significantly improve sample-efficiency. Second, due to the geographical structure of wind farms, the operating condition of a single turbine mainly influences its neighbors. This insight can prove to be useful to create a factored representation of the controller optimization problem. Third, general knowledge about operational data/statistics is often readily available. For example, the expected power production under specific wind conditions for a single turbine is part of the turbine's design specifications [Sohoni et al., 2016]. Leveraging this knowledge, the controller learning process can be guided toward viable solutions that are physically grounded.

We propose a multi-tiered approach to construct the wind farm controller (see Figure 1.5). First, we investigate each of the aforementioned properties separately and develop generic learning methods that focus on a subset of those properties. These generic methods will be useful for a wide variety of applications. Still, we demonstrate the practicality of every method on synthetic wind farm control tasks. Subsequently, we compile the insights gained during the development of these methods and construct a single wind farm control learning method that leverages the similarities between turbines, the sparse topology of the wind farm, and prior knowledge about the data. All methods will be developed within a Bayesian framework, which allows for the seamless integration of domain knowledge in the form of prior and likelihood distributions.

1. Data exchange over a pool of devices with similar dynamics to improve sample-efficiency of control learning methods

In many applications, multiple devices are instantiated to perform the same task. Typically, these devices have the same or similar design specifications, as is for example

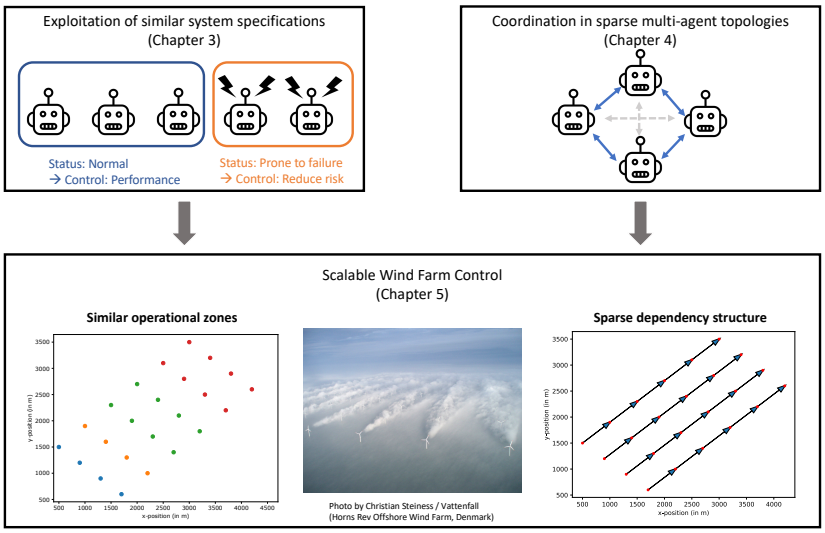


Figure 1.5: Multi-tiered approach to scalable AI-driven control.

the case in wind farms [Feng and Shen, 2017]. We refer to such systems as *pools of devices*. Due to the similarities that exist within a pool of devices[1] (PoD), a learned device-specific control strategy is expected to be reasonably representative for all devices that make up the pool. While learning, information about promising control strategies can be shared among members of the PoD. However, it is expected that not all devices have identical operational behavior, due to differences in health status or small design discrepancies. Therefore, it is important to identify similarities between devices, and implement a smart data exchange mechanism that only shares relevant control information between similar devices.

We propose a new reinforcement learning method, called policy iteration for pools of devices (PIPoD), in which knowledge transfer about the operational behavior of similar devices is possible without compromising the specificity of an individual's operational behavior. More specifically, we create a Bayesian reinforcement learning method that detects correlations between the transition models of the PoD members and uses them to transfer data between members. Additionally, the use of a

[1] In the literature, a pool of devices is sometimes referred to as a 'fleet'. However, to prevent any confusion with fleets in the context of vehicles (e.g., airplanes and ships), we opted for using the term 'pool of devices' to allow for a broader range of applications (e.g., wind farms and assembly lines).

Bayesian framework allows the inclusion of prior knowledge in terms of the operating conditions. We evaluate our method against two baselines. On the one hand, we compare against a learning agent that does not use a transfer mechanism. On the other hand, we compare against an agent that uses the data of the entire PoD, as if all PoD members are identical. As our results show, the latter leads to negative transfer, in which the transfer of unrepresentative information biases the learning process of the agent. Our method outperforms both baselines, which demonstrates its ability to identify similar PoD members and share data between them. Additionally, we demonstrate the applicability of our method on a synthetic wind farm control task, in which multiple rows of two turbines have to sequentially rotate the turbines with respect to the incoming wind vector in order to maximize the row's total power production. The efficiency of the electrical generator is varied over turbine rows to simulate discrepancies in operational behavior. We show that PIPoD successfully performs transfer between similar turbine rows and efficiently learns the optimal orientation of the turbines.

2. *Coordinated learning of joint control strategies in loosely-coupled multi-agent systems*

Learning control strategies in multi-agent systems is challenging. Without prior information about the structure of the problem, optimality can only be guaranteed by learning over the full joint action space. As the joint action space scales exponentially with the number of agents, it is infeasible to learn optimal control strategies. In this regard, we note that wind farms have a sparse topology, i.e., dependencies between turbines only exist locally. Specifically, it is expected that operational upstream turbines only affect a small subset of turbines downstream [González-Longatt et al., 2012]. When a sparse dependency structure is available, it is important to exploit it to ensure the scalability of control learning methods to larger multi-agent systems.

We propose a new sample-efficient multi-armed bandit approach for coordinating agents in multi-agent settings with a sparse topology, called multi-agent Thompson sampling (MATS). We prove theoretically that, for subgaussian data distributions, MATS can successfully factorize the joint action space, while still converging to the optimal policy over time. Moreover, MATS achieves state-of-the-art empirical performance on various benchmarks. Finally, we demonstrate the practical benefits of MATS on a synthetic wind farm control task, in which 11 turbines have to orient themselves with respect to the incoming wind vector to maximize the farm's total power production.

3. Scalable optimization for large-scale wind farm control

In the context of active power control, wind farm controllers need to assign thresholds on the power production to meet a power demand imposed by the grid operator. Such wind farm controllers are generally based on heuristics derived from physics-based knowledge. Although such heuristics are accurate and efficient in terms of reaching the demand, they fail to capture the complete load spectrum that defines a turbine's health status. As the load spectrum is highly complex, flexible AI-driven control approaches are necessary. However, learning control strategies in large wind farms is non-trivial and intractable when approached naively.

Therefore, we propose a new sample-efficient learning method for large-scale wind farm control, called set-point Thompson sampling (SPTS), that consolidates the previous contributions. Specifically, the learning method leverages similarities among the turbines' loading conditions and the sparse structure of the wind farm, to determine the optimal farm configuration that matches the demand as closely as possible, while considering the turbines' health status. Additionally, we adopt the Bayesian formalism to effectively include prior knowledge about statistics and the shape of the data distribution. We demonstrate that SPTS effectively and efficiently learns optimal set-point allocation strategies for contemporary wind farms.

We start by providing some background on multi-armed bandits and reinforcement learning, two frameworks that will be used to conceptualize the settings for which our methods are applicable, in Chapter 2. Next, we examine the possibility of data exchange among similar control agents in a pool of devices in Chapter 3. Then, we investigate how sparse coordination graphs can be incorporated in multi-agent control strategies in Chapter 4. In Chapter 5, we leverage all previously investigated properties to learn optimally advanced wind farm control strategies in a sample-efficient manner. Finally, we conclude with a general discussion, highlighting potential valorisation steps and future work, in Chapter 6.

The rationale of this dissertation and the survey on dynamic loads, described in this chapter, were published in Renewable & Sustainable Energy Reviews:

- *Verstraeten, T., Nowé, A., Keller, J., Guo, Y., Sheng, S. and Helsen, J. (2019), Fleetwide data-enabled reliability improvement of wind turbines, Renewable & Sustainable Energy Reviews, 109, 428–437*

2 Multi-Armed Bandits and Reinforcement Learning

We focus on learning control policies through AI-driven methods. Throughout this dissertation, we formalize our settings as both multi-armed bandits and Markov decision processes, commonly used in reinforcement learning. Reinforcement learning considers a set of techniques to solve decision making problems through trial-and-error learning [Sutton and Barto, 2018]. These techniques allow an agent to optimize its control strategy by interacting with its environment and assessing the quality of control decisions. Whenever a long-term reward over a sequence of state-transitions must be optimized, the environment is typically formalized as Markov decision process. In contrast, multi-armed bandits are used to formalize stateless settings, in which the reward of a control strategy as a whole must be maximized. In this chapter, we provide the necessary background on both multi-armed bandits and stateful reinforcement learning. As we heavily rely on Bayesian methodologies within this dissertation, we first introduce Bayesian inference.

1 Bayesian Inference

To provide an intuitive explanation of Bayesian inference, we start from a coin tossing example, in which we attempt to derive the probability of landing on heads by tossing a (loaded) coin repeatedly.

Consider a coin with an unknown probability p_H of landing on heads. The outcome of a coin toss is modeled as a sample from a Bernoulli distribution.

$$x \sim \mathcal{B}\text{er}\left(\cdot \mid p_H\right), \tag{2.1}$$

where $x = H$ is heads and $x = T$ is tails. This distribution is referred to as the likelihood distribution, as it describes how likely it is for a specific outcome to occur.

The distribution assumes that the parameter p_H is known. However, in the coin toss experiment, this is not the case. From the Bayesian perspective, parameters that are unknown to the person performing the experiment, i.e., the *user*, should be regarded as a random variable. Therefore, in the coin toss experiment, we model the parameter p_H as a random variable. As the uncertainty of this variable can be fully described in terms of the user's knowledge, it is the user's responsibility to formalize her/his prior uncertainty about the variable through a prior distribution. For example, the user might be inclined to construct a distribution centered around $p_H = 0.5$ (i.e., it is equally probable that the coin lands on heads or on tails), with a variance that represents the trust-worthiness of the entity that provides the coin. Although the user is allowed to describe any type of distribution, often priors that are conjugate to the likelihood are chosen, which allows for analytic inference. In the coin toss experiment, the conjugate prior to the binomial distribution is a Beta distribution, i.e.,

$$p_H \sim \mathcal{B}\text{eta}\left(\cdot \mid \alpha_0, \beta_0\right), \tag{2.2}$$

where α_0 and β_0 represent how many heads and tails, respectively, are believed to have been observed before the experiment is performed.

In practice, domain knowledge about the problem at hand is often available. However, in case such information is unavailable, an uninformative prior distribution can be chosen, such as the Jeffreys prior [Lunn et al., 2012], which is invariant under reparametrization. In the coin tossing example, the Jeffreys prior would be $\mathcal{B}\text{eta}\left(\alpha_0 = 0.5, \beta_0 = 0.5\right)$.

When tossing the coin, an outcome x (heads or tails) can be observed. We aim to derive the posterior distribution over the parameter p_H, which describes the information we have about the coin, taking into account the observed outcome. Formally, using Bayes' rule, the posterior distribution can be derived as follows:

$$\mathcal{B}\text{eta}\left(p_H \mid \alpha_1, \beta_1\right) = \frac{\mathcal{B}\text{er}\left(x \mid p_H\right)\mathcal{B}\text{eta}\left(p_H \mid \alpha_0, \beta_0\right)}{\int_{\mathbb{R}} \mathcal{B}\text{er}\left(x \mid p_H\right)\mathcal{B}\text{eta}\left(p_H \mid \alpha_0, \beta_0\right)\mathrm{d}p_H}, \tag{2.3}$$

where $\alpha_1 = \alpha_0 + \mathcal{I}\{x_i = H\}$ and $\beta_1 = \beta_0 + \mathcal{I}\{x_i = T\}$, representing the total number of times that heads and tails are observed, including the prior numbers α_0 and β_0. $\mathcal{I}\{x\}$ is the indicator function, which evaluates to 1 if event x is true and evaluates to 0 if event x is false.

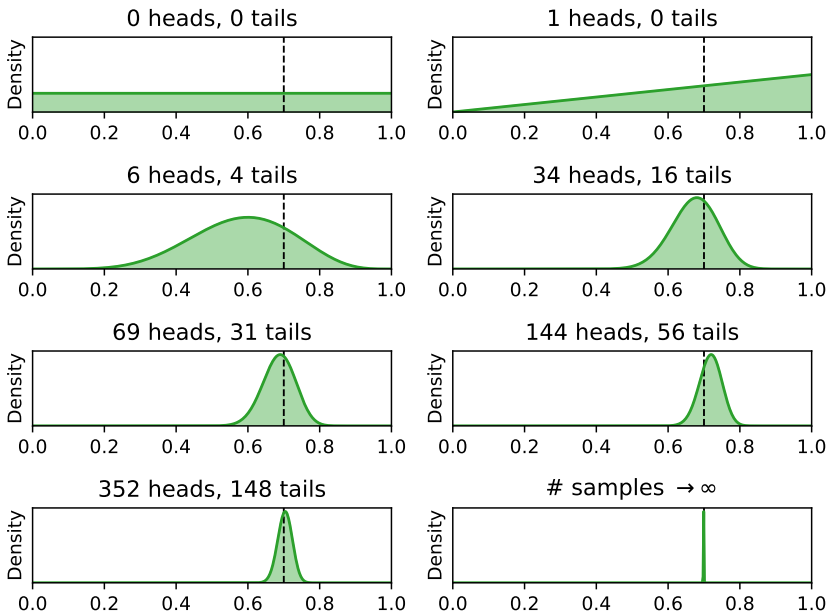


Figure 2.1: Evolution of the posterior distribution, depicted for 0, 1, 10, 50, 100, 200, 500 and an infinite amount of coin tosses. As the coin is tossed and the outcome is observed, the posterior's mean converges to the true probability of heads (dashed line), while its variance reduces.

While the likelihood distribution describes the data observed during the experiment, the prior and posterior distributions describe the beliefs of the user about unknown parameters. Therefore, we often refer to priors and posteriors as *belief distributions*.

Figure 2.1 shows an example of the evolution of the belief distribution over repeated coin tosses. The prior distribution is uniform, signifying maximum uncertainty about the possible values for p_H. When the coin is tossed multiple times, the belief distribution converges toward a peak centered around the true value.

2 Multi-Armed Bandits

The term "multi-armed bandits" refers to slot machines that are used in casinos. In a casino, the goal of a player is to pull the arm of the slot machine that has the highest expected gain. Naturally, which slot machine is the best to pull is unknown information beforehand, and must be found through trial-and-error without losing too much money in the process. This example demonstrates the trade-off between exploration, i.e., trying different arms to acquire information about which arm is best, and exploitation, i.e., focusing on the arm that is believed to be optimal. This formalism can be used to model various decision making problems, such as on-line advertisement [Chapelle and Li, 2011; Rhuggenaath et al., 2019], the mitigation of epidemics [Libin et al., 2019] and wind farm control [Bargiacchi et al., 2018].

Formally, a multi-armed bandit considers a discrete set of arms (i.e., actions) that when pulled (i.e., executed) return a stochastic reward [Auer et al., 2002].

> **Definition 1: Multi-armed bandit**
>
> The multi-armed bandit has a finite set of arms $\mathcal{A}$, where each arm $a \in \mathcal{A}$ returns a reward $R(a)$ when it is pulled. Each arm a has an associated reward distribution with an unknown expected reward, to which we refer as $\mu(a) = \mathbb{E}\left[R(a)\right]$.

The objective is to minimize the expected cumulative regret, which is the cost incurred over time when pulling a particular arm instead of the optimal one [Agrawal and Goyal, 2013a].

Definition 2: Cumulative regret

The expected cumulative regret of pulling a sequence of arms until time step T according to a policy π is

$$\mathbb{E}\left[\mathcal{R}(T,\pi)\right] = \mathbb{E}\left[\sum_{t=1}^{T} \Delta(a_t) \;\middle|\; \pi\right] \tag{2.4}$$

with

$$\Delta(a_t) = \mu(a_*) - \mu(a_t) \tag{2.5}$$

where a_* is the optimal arm and a_t is the arm pulled at time t. For the sake of brevity, we will omit π when the context is clear.

Note that the optimal arm a_* in Definition 2 is considered to be a known value. However, from the Bayesian perspective, it is often assumed that a_* is a random variable. When the regret is marginalised over the random variable a_*, and thus the regret becomes problem-independent, we often use the term *Bayesian regret*. Although a_* always refers to the optimal arm throughout the dissertation, we often use it as a fixed value or as a random variable when the context is clear.

To minimize the cumulative regret, a decision strategy π should be chosen that balances exploration (i.e., searching for the optimal arm) and exploitation (i.e., pulling the arm that is believed to be optimal given the acquired evidence).

2.1 Upper Confidence Bound

One of the most well-known decision strategies is the Upper Confidence Bound (UCB) algorithm [Auer et al., 2002]. UCB is a frequentist approach that establishes confidence bounds around the empirical mean of each arm, and balances exploration and exploitation by optimistically selecting arms for which the upper confidence bound is high. Specifically, at time t, it uses the following arm selection mechanism:

$$\arg\max_{a} \underbrace{\hat{\mu}_{t-1}(a)}_{\text{sample mean}} + \underbrace{c\sqrt{\frac{\ln t}{n_{t-1}(a)}}}_{\text{exploration term}}, \tag{2.6}$$

where c is a constant, $\hat{\mu}_{t-1}(a)$ is the sample mean, and $n_{t-1}(a)$ is the number of pulls of arm a at time t. Intuitively, this mechanism prefers promising arms that were played less (i.e., arms with a high sample mean and/or a high exploration term).

One of the drawbacks of frequentist approaches is the difficulty to include prior knowledge about the problem domain. For example, the exploration bound in Equation 2.6 is derived for a wide variety of reward distributions, and ensures that the algorithm eventually learns the optimal action [Auer et al., 2002; Lattimore and Szepesvári, 2020]. However, when more information about the reward distribution (e.g., the shape) is available, it is challenging to alter the exploration term to improve the learning speed. In many physical applications, expert knowledge is available that can guide the exploration-exploitation process toward arms that are believed to be good, and away from arms that are believed to be bad. In such cases, Bayesian methods are often favored.

2.2 Thompson Sampling

Thompson sampling is another popular approach to solve multi-armed bandits [Thompson, 1933]. It employs a Bayesian statistical framework, which provides a natural way to incorporate prior domain knowledge about available data and statistics. Thompson sampling has been shown to be highly competitive with other popular methods, e.g., UCB [Chapelle and Li, 2011]. Recently, theoretical guarantees on its regret have been established [Agrawal and Goyal, 2012], which renders the method increasingly popular in the literature. Additionally, due to its Bayesian nature, problem-specific priors can be specified. We argue that this has strong relevance in many practical fields, such as advertisement selection [Chapelle and Li, 2011] and influenza mitigation [Libin et al., 2018, 2019].

Thompson sampling adopts the Bayesian formalism, which means the user has to assert their beliefs over the unknown means $\mu(a)$ in the form of a prior distribution, Q_a. At each time step t, Thompson sampling draws a sample $\mu_t(a)$ from the posterior, which is the prior conditioned on the history, $\mathcal{H}_{t-1}$, consisting of previously pulled arms and observed rewards:

$$\begin{aligned} &\mu_t(a) \sim Q_a(\,\cdot \mid \mathcal{H}_{t-1}) \\ &\mathcal{H}_{t-1} = \cup_{i=1}^{t-1}\{\langle a_i, r_i(a_i)\rangle\}. \end{aligned} \tag{2.7}$$

Thompson sampling chooses the arm with the highest sampled mean, i.e.,

$$a_t = \arg\max_a \mu_t(a). \tag{2.8}$$

The chosen arm a_t is pulled, and a reward $r_t(a_t)$ is observed. The history is updated with the observed evidence using Bayesian inference and the cycle repeats.

Note that Thompson sampling draws directly from the posterior over the unknown means. This implies that the chosen action a_t and the unknown optimal action a_* are conditionally independent given the history of observations and identically distributed at time step t.

This feature is called probability matching [Lattimore and Szepesvári, 2020], which reflects that the posteriors maintained by the algorithm exactly matches the user's beliefs after observing the same evidence.

Definition 3

Probability matching is a decision strategy which chooses an arm with the same probability as that arm being optimal, given history $\mathcal{H}_{t-1}$, i.e.,

$$p\left(a_t = \cdot \mid \mathcal{H}_{t-1}\right) = p\left(a_* = \cdot \mid \mathcal{H}_{t-1}\right), \tag{2.9}$$

where a_* is the optimal arm and a_t is the pulled arm at time t.

It can be shown that, when using Thompson sampling for bounded reward distributions, the cumulative regret until time T has an asymptotic upper bound of $O\left(\sqrt{AT \log T}\right)$, where A is the number of arms [Russo and Van Roy, 2014]. This bound is also valid for subgaussian reward distributions, i.e., distributions of which the tails can be upper-bounded by a Gaussian [Vershynin, 2018]. The theoretical upper bound provides insights on how the performance of Thompson sampling evolves. First, the bound is sub-linear in terms of time. This entails that the performance gap converges to 0, which suggests that the chosen arm will improve over time. Second, the bound increases with the number of arms. In single-agent settings, this is typically not a problem. However, in multi-agent settings, the number of arms (and therefore, the regret bound) scales exponentially with respect to the number of agents. As we describe in Chapter 4, it is possible to alleviate this issue by leveraging the loose couplings that exist between agents.

3 Reinforcement Learning

Multi-armed bandits consider settings where the immediate rewards, gained when pulling a sequence of arms, are optimized. However, there are many problems where it is necessary to consider the long-term consequences of an action. Sequential decision making, in which the control policy maximizes the *cumulative* reward in the long term, is called reinforcement learning [Sutton and Barto, 2018].

Consider an agent playing chess, where a reward is given when the agent wins the game and a penalty is given when the agent loses the game. To win the game, the agent has to execute a sequence of moves based on the current state of the board. As it is often necessary to perform actions that in the short-term may lead to bad board states (e.g., sacrificing chess pieces), it is important that the agent considers the long-term reward of the game, which is only provided when it wins. Therefore, in this setting, the optimization

criterion should be the cumulative reward, rather than the immediate reward, gained when executing an action.

A reinforcement learning problem is typically formalized as a Markov decision process [Puterman, 1994].

Definition 4

A Markov decision process is a tuple $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \tau, \gamma, R \rangle$, where:

- $\mathcal{S}$ is the state space, which is a set of states containing contextual features about the environment.
- $\mathcal{A}$ is the action space, which is a set of actions that the agent can take in each state.
- $\tau : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ is a transition function returning the state s' when executing action a in state s,
- $R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is the immediate reward function, and
- $\gamma \in [0, 1)$ is the discount factor determining the importance of future rewards.

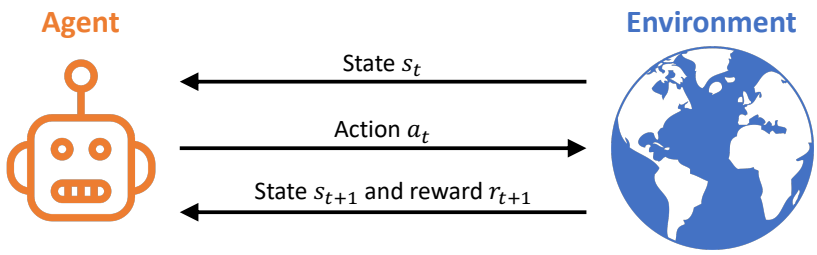


Figure 2.2: Reinforcement learning cycle – The agent first observes a state s_t at time t. Based on this information, it decides on an action a_t to execute in the environment. The environment returns a new state s_{t+1} and a reward r_{t+1} for performing the transition $\langle s_t, a_t, s_{t+1} \rangle$.

An agent acts according to a policy $\pi : \mathcal{S} \to \mathcal{A}$, which defines the action an agent should take in a particular state. The expected long-term reward, when following a policy π, is defined by a value function V^π. This function can be written recursively as the sum of

the expected immediate reward and future reward, i.e.,

$$V^{\pi}(\boldsymbol{s}) = \mathbb{E}\left[R(\boldsymbol{s}, \pi(\boldsymbol{s}), \boldsymbol{s}') + \gamma V^{\pi}(\boldsymbol{s}') \mid \boldsymbol{s}' = \tau(\boldsymbol{s}, \pi(\boldsymbol{s}))\right]. \tag{2.10}$$

This is the sum of all possible long-term rewards weighted by their probability of occurrence when executing a policy π. The expectation is taken over the possible trajectories that can manifest according to the stochasticity in the transition model and reward function. The goal of an agent is to learn the optimal policy,

$$\pi^* \ : \ \mathcal{S} \rightarrow \mathcal{A}, \tag{2.11}$$

which maximizes the value function over all states. The reinforcement learning cycle is depicted in Figure 2.2.

Many approaches exist to solve an MDP. One of the most popular techniques is Q-learning [Sutton and Barto, 2018], which is a value-based approach that aims to find the policy that greedily maximizes the total cumulative reward. It maintains Q-values, which describe the quality of actions taken at a particular state, i.e., the expected cumulative reward when first performing the action and then executing the current policy, and computes these values iteratively over time according to the following update-rule:

$$Q_{t+1}(\boldsymbol{s}_t, a_t) = Q_t(\boldsymbol{s}_t, a_t) + \alpha \delta_t(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1}), \tag{2.12}$$

where $\boldsymbol{s}_t$ and a_t are the state and action observed at time t, α is the learning rate and

$$\delta_t(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1}) = \underbrace{R(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1}) + \gamma \max_a Q_t(\boldsymbol{s}_{t+1}, a)}_{\text{new estimate}} - \underbrace{Q_t(\boldsymbol{s}_t, a_t)}_{\text{old estimate}} \tag{2.13}$$

is the temporal difference error at time t, i.e., the difference between the new and old estimates for the expected cumulative reward of action a_t in state $\boldsymbol{s}_t$. The full algorithm is provided in Algorithm 1.

Temporal difference learning is a type of model-free reinforcement learning. Although model-free methods are flexible toward many applications, they do not use (prior) information about the environment to guide the learning process. Using a model of the environment is often preferred, as these are readily available in many application domains, such as robotics [Polydoros and Nalpantidis, 2017; Kober et al., 2013; Rastogi et al., 2018], and significantly improve the sample-efficiency of reinforcement learning methods. In model-based reinforcement learning, the transition and reward functions are learned (or provided) to guide the optimization process of the control policy. In most cases, alternative trajectories are simulated and evaluated using an estimated model of the environment, in order to augment the real experiences of the agent [Sutton, 1990].

Algorithm 1: Q-Learning [Sutton and Barto, 2018]

Given: learning rate α
1 **for** $\boldsymbol{s} \in \mathcal{S}, a \in \mathcal{A}$ **do**
2 | initialize $Q_0(\boldsymbol{s}, a)$
3 **end**
4 Initialize $\boldsymbol{s}_0$
5 **for** *each time step:* $t \in [0, \ldots, +\infty[$ **do**
6 | Choose a_t in $\boldsymbol{s}_t$ using a policy derived from $Q_0(\boldsymbol{s}, a)$
7 | Take action a_t and observe reward r_{t+1} and the next state $\boldsymbol{s}_{t+1}$
8 | $Q_{t+1}(\boldsymbol{s}_t, a_t) \leftarrow Q_t(\boldsymbol{s}_t, a_t) + \alpha\left(r_{t+1} + \gamma \max_a Q_t(\boldsymbol{s}_{t+1}, a) - Q(\boldsymbol{s}_t, a_t)\right)$
9 **end**

Under certain conditions, it is possible to analytically derive the value function from the model [Deisenroth and Rasmussen, 2011].

In many situations, only partial information of the environment is available, and thus the complete environment needs to be estimated using state transitions. A commonly used estimator for modeling the environment is a Gaussian process [Deisenroth and Rasmussen, 2011; Rasmussen and Kuss, 2003]. Gaussian processes are Bayesian models that capture the uncertainty about the model parameters and has the ability to seamlessly incorporate prior knowledge about the data distribution, which renders them popular in many practical applications [Chen et al., 2013a; Vanhatalo et al., 2010]. As they rely on the Bayesian framework, Gaussian processes are updated through Bayesian inference.

In Chapter 3, we propose a new model-based reinforcement learning algorithm that uses Gaussian processes to jointly model the state-transitions performed by multiple similar learning agents. In Chapters 4 and 5, we introduce a variant of multi-armed bandits to formalize multi-agent systems with sparse interactions between the agents.

3 Policy Iteration for Pools of Devices

Many control applications are comprised of multiple devices performing the same task, such as autonomous vehicles [Gerla et al., 2014] and wind farms [Martin et al., 2016]. Often these devices have identical or highly similar design specifications [Feng and Shen, 2017], which leads to similar operational behaviors. We refer to such control applications as a *pool of devices* (PoD).

PoDs are prominent in industrial applications. For example, devices may be managed as a single system to reduce capital costs [Sarker and Faiz, 2017]. To this end, we present a method that aggregates the data of the distinct PoD members, in contrast to the data of a single device. The time is right for such a method, as the recent advances in the *Internet of Things* allow PoD members to share data from modern wireless sensors using a cloud-based architecture, rapidly providing a complete overview of the problem [Do et al., 2020; Helsen et al., 2018b].

PoDs are typically complex environments. In the context of wind farm control, the health status of a turbine is based on a multi-dimensional load spectrum and should be considered in control strategies, as argued in Chapter 1. Still, incorporating health information in a wind farm controller is challenging, as it requires optimizing a non-linear cost function. Moreover, the link between controller actions and a turbine's health is not fully known. Therefore, it is challenging to develop a controller based on solely prior knowledge of the system. The use of reinforcement learning is warranted, as this framework allows for

learning in complex environments with non-linear cost functions [Sutton and Barto, 2018]. However, as state-of-the-art reinforcement learning techniques typically require a large amount of data, a key challenge in reinforcement learning is to increase sample-efficiency, such that these techniques become viable in real-life applications [Yu, 2018].

As PoD members carry out the same task, they typically share the same design [Feng and Shen, 2017]. In reality, PoD members differ slightly in terms of operating conditions, for example due to production errors or degradation over time [Staffell and Green, 2014]. Thus, naively aggregating data over all members can be detrimental to the learning process. Therefore, information should only be shared between PoD members that are sufficiently similar.

In many physical applications, on-line reinforcement learning, in which data is collected through exploration of the environment while learning, is either challenging or impossible. For example, in the context of wind farm control, exploration of alternative control actions is costly, as they might lead to a lower productivity or even failure. However, a data set of previously observed controller actions and rewards/costs (e.g., power productions, loads and failures) is typically available. Therefore, we investigate an off-line reinforcement learning setting, which assumes the availability of a batch of training data prior to learning.

We propose policy iteration for pools of devices (PIPoD), a new off-line reinforcement learning method for PoDs where knowledge transfer about operational behavior of similar devices is possible without compromising the specificity of an individual's behavior. More specifically, we create a Bayesian reinforcement learning method that learns a Gaussian process to represent the transition model of a PoD member, based on a batch of transition samples over the entire PoD. Gaussian processes are Bayesian models known to successfully capture complex non-linear surfaces using only a limited amount of data points. They have previously been used in the context of reinforcement learning [Rasmussen and Kuss, 2003; Engel et al., 2005; Deisenroth and Rasmussen, 2011] and are popular when high sample-efficiency is required. In PIPoD, the transition model of a single PoD member is described by a Gaussian process that learns correlations between that member's model and the other members' models using coregionalization. Coregionalization was originally introduced in geostatistics to generate valid covariance matrices for modeling multivariate data sets [Goovaerts, 1997]. It has later been used in the context of multi-task learning to describe correlations between a set of tasks [Bonilla et al., 2008]. In this chapter, we propose a new sparse variant of coregionalization, which only considers correlations between a chosen target member and every other member. Our sparse variant significantly reduces the computational complexity of coregionalization in large PoDs (see Section 4).

We start by providing background information on Gaussian processes and the Gaussian Process Reinforcement Learning (GPRL) method in Section 1. Then, we position our research within existing literature in Section 2. Next, we construct our method in

Section 3 and analyse the computational complexity of the coregionalization step in Section 4. Afterwards, we provide results on two synthetic benchmarks and a wind farm control task in Section 5. Finally, we discuss our findings in Section 6.

The work presented in this chapter was published in the ECAI proceedings.

- *Verstraeten, T., Libin, P. J. K. and Nowé, A. (2020), Fleet Control using Coregionalized Gaussian Process Policy Iteration, Proceedings of the 24rd European Conference on Artificial Intelligence (ECAI)*

1 Background

1.1 Gaussian Process

Gaussian processes are an extension of multivariate normal distributions [Rasmussen and Williams, 2006]. Similar to a multivariate normal distribution, a Gaussian process describes a set of normally distributed random variables that are potentially correlated, i.e., knowledge about one variable gives information about another. However, the difference with multivariate normal distributions is that a Gaussian process is defined over arbitrary sets of annotated random variables. In a regression context, these random variables are the outputs of an unknown function and their annotations are the inputs to that function.

Formally, assuming a zero-mean Gaussian process prior, i.e.,

$$f(\boldsymbol{x}) \sim \mathcal{GP}\left(0, k(\boldsymbol{x}, \boldsymbol{x'})\right), \tag{3.1}$$

and any arbitrary set of inputs X, we can model the associated latent function values $\boldsymbol{f}$ as

$$\boldsymbol{f} \mid X \sim \mathcal{N}(\mathbf{0}, K) \tag{3.2}$$

where $K_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is the covariance between variables f_i and f_j. When regressing over a training set $(X_{\text{tr}}, \boldsymbol{y}_{\text{tr}})$, we can compute the posterior statistics of $(\boldsymbol{f} \mid X, X_{\text{tr}}, \boldsymbol{y}_{\text{tr}})$ to obtain the predictive outputs $\boldsymbol{f}$ for inputs X. For the zero-mean Gaussian process described in Equation 3.2, we have:

$$\begin{aligned}
\mathbb{E}\left[\boldsymbol{f} \mid X, X_{\text{tr}}, \boldsymbol{y}_{\text{tr}}\right] &= K_{X,X_{\text{tr}}} C^{-1}_{X_{\text{tr}},X_{\text{tr}}} \boldsymbol{y}_{\text{tr}} \\
\mathbb{V}\left[\boldsymbol{f} \mid X, X_{\text{tr}}, \boldsymbol{y}_{\text{tr}}\right] &= K_{X,X} - K_{X,X_{\text{tr}}} C^{-1}_{X_{\text{tr}},X_{\text{tr}}} K_{X_{\text{tr}},X} \\
C_{X_{\text{tr}},X_{\text{tr}}} &= K_{X_{\text{tr}},X_{\text{tr}}} + \sigma_n^2 I,
\end{aligned} \tag{3.3}$$

where $K_{X,X_{\text{tr}}}$ is a matrix containing the pair-wise covariances between sets $X \subset \mathbb{R}^D$ and $X_{\text{tr}} \subset \mathbb{R}^D$ (with D the dimensionality of the inputs) according to the covariance kernel and σ_n^2 is observational noise.

The choice of covariance kernel $k(\cdot,\cdot)$ is important, as it defines the various characteristics about how the model should generalize from the training set. A commonly used kernel is the squared exponential kernel, defined as:

$$k^{\mathsf{SE}}_{\theta^{\mathsf{SE}}}(\boldsymbol{x},\boldsymbol{x}') = \sigma_s \exp\left(-\sum_{d=1}^{D}\frac{(x_d - x'_d)^2}{2l_d^2}\right), \tag{3.4}$$

where θ^{SE} contains the hyperparameters σ_s, which denotes the output's scale, and l_d, which denotes the length scale along dimension d and characterizes the smoothness of the unknown function. These hyperparameters can be optimized using maximum likelihood optimization on the training set [Rasmussen and Williams, 2006]. The squared exponential kernel has several properties, including continuity, differentiability and stationarity, rendering it a popular choice for general modeling purposes.

1.2 Model-Based Reinforcement Learning using Gaussian Processes

Consider the Markov decision process $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \tau, \gamma, r\rangle$ (see Definition 4). In this chapter, we focus on learning similarities and differences in terms of transition models, rather than control tasks. Therefore, we focus on learning the transition models, and assume that the reward function is known.[1] Specifically, we assume a goal-based reward function that is described as a square-exponential function centered around a goal state $\boldsymbol{s}_{\mathsf{goal}}$ with width σ_R, i.e.,

$$R(\boldsymbol{s}, a, \boldsymbol{s}') = \frac{1}{\sqrt{2\pi\sigma_R^2}} \exp\left(-\frac{||\boldsymbol{s}' - \boldsymbol{s}_{\mathsf{goal}}||_2^2}{2\sigma_R^2}\right). \tag{3.5}$$

The transition function is unknown, and a Gaussian process is used to model the uncertainty about the function, prior to learning. In other words, we define the outputs of the transition function as samples from Gaussian processes with a squared exponential covariance kernel (see Equation 3.4), i.e.,

$$\tau_e(\boldsymbol{s}, a) \sim \mathcal{GP}\left(0, k^{\mathsf{SE}}_{\theta^{\mathsf{SE}}_e}\right), \tag{3.6}$$

for each output feature e.

The value function (see Equation 2.10) typically has no closed-form expression for arbitrary continuous reward and transition functions. Therefore, we approximate the value

[1] If the reward function is unknown, it can be learned using a Gaussian process without jeopardizing the analytical benefits of the GPRL method.

function using the Gaussian Process Reinforcement Learning (GPRL) method [Rasmussen and Kuss, 2003]. After fitting the transition model on a batch of transitions sampled from the environment, GPRL uses policy iteration to iteratively evaluate a policy π on a discrete set of states and improves it until convergence.

During the policy evaluation step, GPRL computes the values of a finite, but dense, vector of support points $S_{\text{supp}} = \langle \boldsymbol{s}^{(i)} \rangle_{i=1}^{N}$. We use Latin hypercube sampling [McKay et al., 1979] to generate this vector, such that the state space is sufficiently covered. Note that these support points do not have to match the training inputs of the transition model, as the Gaussian process generalizes the training data to the selected support points. The values of the support points can be computed analytically when the transition model and value function are described by a Gaussian process, and the reward function is bell-shaped. Formally, given a policy π, a reward function centered around $\boldsymbol{s}_{\text{goal}}$ with width σ_R^2, and an initial Gaussian process over the value function, the support values $\boldsymbol{v}_{\text{supp}}$ have the recursive form [Rasmussen and Kuss, 2003]:

$$\boldsymbol{v}_{\text{supp}} = \boldsymbol{r} + \gamma U \boldsymbol{v}_{\text{supp}} \tag{3.7}$$

with

$$\begin{aligned} r_i &= \frac{1}{\sqrt{|2\pi C^{(i)}|}} \exp\left(-\frac{1}{2}(\boldsymbol{s}_{\text{goal}} - \boldsymbol{\mu}^{(i)})^T {C^{(i)}}^{-1} (\boldsymbol{s}_{\text{goal}} - \boldsymbol{\mu}^{(i)})\right) \\ C^{(i)} &= \Sigma^{(i)} + \sigma_R^2 I \end{aligned} \tag{3.8}$$

with the statistics of the transition model,

$$\begin{aligned} \boldsymbol{\mu}^{(i)} &= \mathbb{E}\left[\boldsymbol{s}' \;\middle|\; \boldsymbol{s}' = \tau\left(\boldsymbol{s}^{(i)}, \pi\left(\boldsymbol{s}^{(i)}\right)\right)\right] \\ \Sigma^{(i)} &= \mathbb{V}\left[\boldsymbol{s}' \;\middle|\; \boldsymbol{s}' = \tau\left(\boldsymbol{s}^{(i)}, \pi\left(\boldsymbol{s}^{(i)}\right)\right)\right], \end{aligned} \tag{3.9}$$

and U a matrix that depends on the transition model and the value function:

$$\begin{aligned} U &= W\left(K_V + \sigma_n^2 I\right)^{-1} \\ W_{i,j} &= \sigma_V^2 \left|\Lambda_V^{-1}\Sigma^{(i)} + I\right|^{-0.5} \\ &\quad \exp\left(-0.5\left(\boldsymbol{s}^{(i)} - \boldsymbol{\mu}^{(i)}\right)^T \left(\Sigma^{(i)} + \Lambda_V\right)^{-1} \left(\boldsymbol{s}^{(i)} - \boldsymbol{\mu}^{(i)}\right)\right) \end{aligned} \tag{3.10}$$

where K_V is the covariance between the support values, σ_V^2 is the signal's variance and σ_n^2 is the noise parameter used by the value GP. The equation for the support values can be rewritten as a closed-form expression:

$$\boldsymbol{v}_{\text{supp}} = (I - \gamma U)^{-1} \boldsymbol{r}. \tag{3.11}$$

During the policy improvement step, a new Gaussian process is fitted over the value function $V^{\pi}(\cdot)$ using the support values to generalize over the state space. This function is used to optimize π:

$$\pi(\boldsymbol{s}) \leftarrow \arg\max_{a} \mathbb{E}\left[R(\boldsymbol{s}, a, \boldsymbol{s}') + \gamma V^{\pi}(\boldsymbol{s}') \mid \boldsymbol{s}' = \tau(\boldsymbol{s}, a)\right]. \tag{3.12}$$

An expression similar to the one presented in Equation 3.7 can be obtained for arbitrary actions using the vector of support states. The obtained closed-form expression of Equation 3.12 can then be maximized through enumeration for discrete action spaces or by using standard optimizers (such as gradient ascent) for continuous action spaces.

2 Related Work

The type of learning we consider is related to multi-task (or inductive transfer) reinforcement learning [Pan and Yang, 2009], where a set of control tasks is jointly learned, leveraging potential similarities between them. In contrast to our setting, most work on multi-task reinforcement learning considers different task parameters, while the system specifications remain the same [Ijspeert et al., 2003; Lazaric and Ghavamzadeh, 2010; Taylor and Stone, 2007; Wilson et al., 2007; Konidaris et al., 2012; Deisenroth et al., 2014]. Specifically, Lazaric and Ghavamzadeh [2010] and Wilson et al. [2007] construct a Bayesian (hierarchical) structure of tasks, where the task parameters are assumed to be drawn from a set of priors shared among similar tasks. Recent work has focused on Markov decision processes for which the system specifications are different, but the reward function remains the same [Doshi-Velez and Konidaris, 2016; Killian et al., 2017; Sæmundsson et al., 2018]. Typically, a latent embedding, or a clustering, of the system specifications is learned in order to share information among various devices.

Our focus is different as it concerns settings in which we assume that the system specifications are nearly identical except for degraded parts or small design discrepancies. This means that a more targeted approach is feasible. Rather than having a single latent embedding or cluster from which all members originate, a more directed transfer method between pairs of members can be obtained through correlations. Such a direct transfer mechanism is sample-efficient, as estimating the correlations within a PoD for a given target is limited to learning a set of parameters linear in the size of the PoD.

3 Coregionalization over Multiple Transition Models

To transfer knowledge between PoD members, we leverage the statistical properties of Gaussian processes. Specifically, we assert that PoD members should only share information when they are correlated. Intuitively, the Gaussian process' covariance kernel allows to generalize in the regression model by correlating unobserved outputs to observed ones. Coregionalization extends this concept to the outputs of different Gaussian processes, suggesting that information from one process can be generalized to another. The main contribution in this work is the introduction of coregionalization to capture similarities between multiple transition models in order to decide whether and how knowledge should be transferred.

Formally, we define a PoD Markov decision process as:

$$\begin{aligned} \mathcal{M}_{\text{PoD}} &= \langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \gamma, R \rangle, \text{with} \\ \mathcal{T} &= \{\tau_m\}_{m=1}^{M}. \end{aligned} \tag{3.13}$$

Compared to the definition of a standard Markov decision process (see Definition 4), all properties are the same, except that $\mathcal{T}$ is now a set of M transition models, one for each member m in the PoD.

Consider a single member from the PoD, which we refer to as the *target* and the rest of the PoD as the *sources*. We denote the target's index as $\mathfrak{t}$, and the index of a source as $\mathfrak{s}$. To achieve knowledge transfer from the sources to the target, we must define a model that allows for data sharing. Specifically, for inputs $\boldsymbol{x} = [\boldsymbol{s}, a]$, we consider this model for the transition functions:

$$\begin{aligned} &\tau_{\mathfrak{t}}(\boldsymbol{x}) = \sum_{\mathfrak{s} \neq \mathfrak{t}} \phi_{\mathfrak{t},\mathfrak{s}} g_{\mathfrak{s}}(\boldsymbol{x}) + \alpha_{\mathfrak{t}} l_{\mathfrak{t}}(\boldsymbol{x}) \\ &\forall \mathfrak{s} \neq \mathfrak{t}: \\ &\qquad \tau_{\mathfrak{s}}(\boldsymbol{x}) = \phi_{\mathfrak{s},\mathfrak{s}} g_{\mathfrak{s}}(\boldsymbol{x}) + \alpha_{\mathfrak{s}} l_{\mathfrak{s}}(\boldsymbol{x}). \end{aligned} \tag{3.14}$$

The transition function of the target is modelled as a linear combination of $M-1$ global functions $g_{\mathfrak{s}}$ shared with every source $\mathfrak{s}$, and member-specific local functions (i.e., $l_{\mathfrak{t}}$ for the target and $l_{\mathfrak{s}}$ for the sources) to model the member's specific behavior. This ensures that all sources can exchange information with the target without compromising the specifics of a member's transition model. The parameters $\alpha_{\mathfrak{t}}$ and $\alpha_{\mathfrak{s}}$ weigh the contribution of the local components $l_{\mathfrak{t}}$ and $l_{\mathfrak{s}}$ to the transition models of the target and source $\mathfrak{s}$, respectively. Additionally, the parameters $\phi_{\mathfrak{t},\mathfrak{s}}$ and $\phi_{\mathfrak{s},\mathfrak{s}}$ weigh the contribution of the global component $g_{\mathfrak{s}}$ to the transition models of the target and source $\mathfrak{s}$, respectively. For example, when source $\mathfrak{s}$ has relevant information for the target $\mathfrak{t}$, the parameter $\phi_{\mathfrak{t},\mathfrak{s}}$ should be high in order

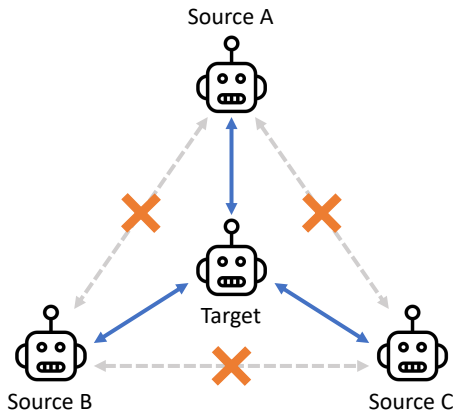


Figure 3.1: Data exchange mechanism between PoD members. The model only estimates correlations between the target and a source (blue arrows). The connections between two sources (grey arrows) are not considered in the model, and their correlations are thus assumed to be zero.

to transfer knowledge through function $g_{\mathfrak{s}}$. In contrast, when the source has no relevant information for the target, the parameter $\phi_{\mathfrak{t},\mathfrak{s}}$ should be zero to model independence between the source and the target. A schematic of the proposed data exchange framework is depicted in Figure 3.1.

We define the unknown components $g_{\mathfrak{s}}$, $l_{\mathfrak{s}}$ and $l_{\mathfrak{t}}$ as independent samples from a zero-mean Gaussian process with covariance kernel $k^{\mathsf{SE}}_{\theta_{\mathsf{SE}}}$ (see Equation 3.6). This entails that each of the transition functions is a linear combination of Gaussian process-distributed random variables, and is thus also a Gaussian process-distributed random variable [Rasmussen and Williams, 2006]. The mean functions of the transition models will be zero, due to the linearity of expectation property and the fact that all components have a mean of zero. Moreover, as the components are independently sampled, covariance can only exist within a single component, which is defined through the kernel $k^{\mathsf{SE}}_{\theta_{\mathsf{SE}}}$. The cross-covariance between the transition functions can be derived using the linearity properties (L) of the covariance operator and the fact that the covariance between independent components is zero. For example, the cross-covariance between the transition functions

of the target $\mathfrak{t}$ and a source $\mathfrak{s}$ is

$$\begin{aligned}
&\mathbb{C}\text{ov}\left[\tau_{\mathfrak{t}}(\boldsymbol{x}), \tau_{\mathfrak{s}}(\boldsymbol{x}')\right] \\
&= \mathbb{C}\text{ov}\left[\sum_{\mathfrak{s}' \neq \mathfrak{t}} \phi_{\mathfrak{t},\mathfrak{s}'} g_{\mathfrak{s}'}(\boldsymbol{x}) + \alpha_{\mathfrak{t}} l_{\mathfrak{t}}(\boldsymbol{x}), \phi_{\mathfrak{s},\mathfrak{s}} g_{\mathfrak{s}}(\boldsymbol{x}') + \alpha_{\mathfrak{s}} l_{\mathfrak{s}}(\boldsymbol{x}')\right] \\
&\overset{(\mathsf{L})}{=} \phi_{\mathfrak{t},\mathfrak{s}} \phi_{\mathfrak{s},\mathfrak{s}} \mathbb{C}\text{ov}\left[g_{\mathfrak{s}}(\boldsymbol{x}), g_{\mathfrak{s}}(\boldsymbol{x}')\right] \\
&= \phi_{\mathfrak{t},\mathfrak{s}} \phi_{\mathfrak{s},\mathfrak{s}} k^{\mathsf{SE}}_{\theta_{\mathsf{SE}}}(\boldsymbol{x}, \boldsymbol{x}').
\end{aligned} \tag{3.15}$$

In the last step, we use the fact that the components $g_{\mathfrak{s}}$ and $l_{\mathfrak{t}}$ are modelled using a Gaussian process with the squared exponential kernel $k^{\mathsf{SE}}_{\theta_{\mathsf{SE}}}$. Similar derivations can be performed for the inter-source and target's cross-covariances. The resulting cross-covariances for all possible cases are

$$\begin{aligned}
\mathbb{C}\text{ov}\left[\tau_{\mathfrak{t}}(\boldsymbol{x}), \tau_{\mathfrak{t}}(\boldsymbol{x}')\right] &= \left(\sum_{\mathfrak{s}' \neq \mathfrak{t}} \phi^2_{\mathfrak{t},\mathfrak{s}'} + \alpha^2_{\mathfrak{t}}\right) k^{\mathsf{SE}}_{\theta_{\mathsf{SE}}}(\boldsymbol{x}, \boldsymbol{x}') \\
\mathbb{C}\text{ov}\left[\tau_{\mathfrak{s}}(\boldsymbol{x}), \tau_{\mathfrak{s}}(\boldsymbol{x}')\right] &= \left(\phi^2_{\mathfrak{s},\mathfrak{s}} + \alpha^2_{\mathfrak{s}}\right) k^{\mathsf{SE}}_{\theta_{\mathsf{SE}}}(\boldsymbol{x}, \boldsymbol{x}') \\
\mathbb{C}\text{ov}\left[\tau_{\mathfrak{t}}(\boldsymbol{x}), \tau_{\mathfrak{s}}(\boldsymbol{x}')\right] &= \left(\phi_{\mathfrak{t},\mathfrak{s}} \phi_{\mathfrak{s},\mathfrak{s}}\right) k^{\mathsf{SE}}_{\theta_{\mathsf{SE}}}(\boldsymbol{x}, \boldsymbol{x}') \\
\mathbb{C}\text{ov}\left[\tau_{\mathfrak{s}}(\boldsymbol{x}), \tau_{\mathfrak{s}'}(\boldsymbol{x}')\right] &= 0,
\end{aligned} \tag{3.16}$$

where $\mathfrak{s}, \mathfrak{s}' \neq \mathfrak{t}$ and $\mathfrak{s} \neq \mathfrak{s}'$.

From these statistics, we can reformulate the target's transition function as a sample from a Gaussian process with the following kernel:

$$\begin{aligned}
k^{\mathsf{PoD}}_{\theta^{(\mathfrak{t})}}\left([\boldsymbol{x}, m], [\boldsymbol{x}', m']\right) &= k^{\mathsf{SE}}_{\theta_{\mathsf{SE}}}(\boldsymbol{x}, \boldsymbol{x}') F_{m,m'} \\
F &= \Phi + \left(\boldsymbol{\alpha}^2\right)^T I \\
\Phi &= \sum_{\mathfrak{s} \neq \mathfrak{t}} \boldsymbol{\phi}_{\mathfrak{s}} \boldsymbol{\phi}_{\mathfrak{s}}^T
\end{aligned} \tag{3.17}$$

where $\boldsymbol{\phi}_{\mathfrak{s}}$ only has non-zero elements at indices $\mathfrak{t}$ and $\mathfrak{s}$, and m, m' are the indices of two PoD members. The matrix Φ encodes relationships between the target and the sources, while $\boldsymbol{\alpha}$ contains independent terms for each member. The set $\boldsymbol{\theta}^{(\mathfrak{t})}$ contains now both the hyperparameters $\boldsymbol{\theta}_{\mathsf{SE}}$ of the SE kernel and of the matrix F, i.e., $\boldsymbol{\phi}$ and $\boldsymbol{\alpha}$. These parameters can be optimized using maximum likelihood optimization [Rasmussen and Williams, 2006] on the training set $(X^{\mathsf{PoD}}_{\mathsf{tr}}, \boldsymbol{y}^{\mathsf{PoD}}_{\mathsf{tr}})$ of the entire PoD, annotated with the indices of its members.

Using the new covariance kernel, we can describe a single Gaussian process jointly over the outputs of all PoD members (Equation 3.2) and compute the target's posterior statistics (Equation 3.3) for regression. Note that even though the new model uses the whole PoD's data set, the target can predict using its own transition function by computing the posterior statistics using index $\mathfrak{t}$, i.e.,

$$\boldsymbol{\tau}_{\mathfrak{t}} \mid X^{(\mathfrak{t})}, X_{\mathsf{tr}}^{\mathsf{PoD}}, \boldsymbol{y}_{\mathsf{tr}}^{\mathsf{PoD}}. \tag{3.18}$$

We can define such a model for each member in the PoD independently by setting that member as the target, and thus construct the set $\mathcal{T}$ by sampling the transition model of each member m from a Gaussian process:

$$\tau_m(\boldsymbol{x}) \sim \mathcal{GP}\left(0, k_{\theta^{(m)}}^{\mathsf{PoD}}\left([\boldsymbol{x}, m], [\boldsymbol{x'}, m']\right)\right). \tag{3.19}$$

GPRL can be used for policy iteration to learn the optimal value function and policy. A high-level description of the complete policy iteration method for a given target is provided in Algorithm 2.

4 Analysis of the Sparse Coregionalization Matrix

The proposed sparse coregionalization matrix F has several properties. First, the decomposition of F yields a valid covariance matrix, as it is symmetric positive semidefinite [Rasmussen and Williams, 2006], i.e.,

$$\forall \boldsymbol{z} \neq \boldsymbol{0} \ : \ \boldsymbol{z}^T F \boldsymbol{z} = \sum_{\mathfrak{s} \neq \mathfrak{t}} ||\boldsymbol{z}^T \boldsymbol{\phi}_{\mathfrak{s}}||_2^2 + ||\boldsymbol{z}^T \boldsymbol{\alpha}||_2^2 \geq 0. \tag{3.20}$$

Second, the matrix F contains $3M-2$ parameters per target (i.e., M in $\boldsymbol{\alpha}$ and $2(M-1)$ in the weight matrix Φ) with M the number of PoD members. Therefore, the number of parameter grows linearly per target with respect to the number members, and the method can be executed in a distributed manner over the PoD. This renders the method scalable to larger PoDs. Third, because of the sparsity of the defined covariances (see Equation 3.16), it is possible to significantly reduce the computational complexity of the matrix inversion in Equation 3.3. In the literature, inversion is achieved by performing backward and forward substitution on the Cholesky factors of the covariance matrix, as it faster and more numerically stable than direct inversion [Rasmussen and Williams, 2006]. This operation has a complexity of $O(N^3)$, where N signifies the number of rows. Given the sparsity of our coregionalization matrix F, we can derive a faster operation.

Consider a fleet of M members. Each member m has a data set of N_m samples, resulting in a total of $N = \sum_{m=1}^{M} N_m$ samples. Without loss of generality, assume that

Algorithm 2: Policy Iteration for Pools of Devices (PIPoD)

Input: Reward function R, set of support points S_{supp}, batch of transitions $(X_{\text{tr}}^{\text{PoD}}, \boldsymbol{y}_{\text{tr}}^{\text{PoD}})$ for all members of the PoD, target index $\mathfrak{t}$

Output: Learned policy $\pi(\boldsymbol{s})$

```
π(s) ← random policy;

Initialize support values and fit the Gaussian process on the values.
v_supp ← Apply reward function R on S_supp;
Define V ~ GP(0, k^SE_{θ_V});
Fit V using (S_supp, v_supp);

Given the batch of transitions, train the transition model using maximum
  likelihood estimation on the hyperparameters and coregionalization matrix F.
                                                                    (Section 4)
θ^(t) ← argmax_{θ_SE, F} p(y^PoD_tr | X^PoD_tr, θ_SE, F);
Define τ ~ GP(0, k^PoD_{θ^(t)});
Fit τ using (X^PoD_tr, y^PoD_tr);

Perform policy iteration (GPRL) using the pre-trained transition model and
  support points.                                                   (Section 2)
while v_supp not converged do
    v_supp ← Policy evaluation using S_supp, R, τ, V and π;
    Fit V using (S_supp, v_supp);
    π ← Policy improvement using S_supp, R, τ and V;
end
```

the target index is M and the sources' indices are in $[1, \ldots, (M-1)]$. When we use the PoD transition model, a covariance matrix needs to be inverted when fitting the GP. This covariance matrix has the following block form:

$$K = \begin{bmatrix} B & C \\ C^T & K_M^{\mathsf{SE}} \end{bmatrix}, \tag{3.21}$$

with block diagonal matrix

$$B = \begin{bmatrix} K_1^{\mathsf{SE}} & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & K_{M-1}^{\mathsf{SE}} \end{bmatrix}, \tag{3.22}$$

where K_m^{SE} is the covariance matrix of member m and C contains the cross-covariance matrices between the target and each of the sources.

The Cholesky decomposition [Rasmussen and Williams, 2006] of the full covariance matrix K is

$$K = L_K L_K^T = \begin{bmatrix} L_B & 0 \\ (L_B^{-1}C)^T & L_S \end{bmatrix} \begin{bmatrix} L_B^T & L_B^{-1}C \\ 0 & L_S^T \end{bmatrix} \tag{3.23}$$

where L_B and L_S are the Cholesky factors of block B and its Schur complement:

$$\begin{aligned} S &= K_M^{\mathsf{SE}} - C^T B^{-1} C \\ &= K_M^{\mathsf{SE}} - C^T (L_B L_B^T)^{-1} C \\ &= K_M^{\mathsf{SE}} - (L_B^{-1}C)^T L_B^{-1} C. \end{aligned} \tag{3.24}$$

Note that

$$L_B = \begin{bmatrix} L_{K_1^{\mathsf{SE}}} & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & L_{K_{M-1}^{\mathsf{SE}}} \end{bmatrix}. \tag{3.25}$$

The only matrices for which a Cholesky factor needs to be computed are matrices S and K_m^{SE}. The Cholesky decomposition has a cubic complexity in the number of rows. Thus, as each block in B can be processed separately, the computational complexity of the Cholesky decomposition on matrix B is $O\left(\sum_{m=1}^{M} N_m^3\right)$. The size of S is $N_M \times N_M$, and therefore computing its Cholesky factor is equal to $O\left(N_M^3\right)$. As the multiplication of $L_B C$ can also be decomposed per member (i.e., per block in the matrix L_B), the

complexity of this operation is $O\left(\sum_{m=1}^{M} N_M N_m^2\right)$. Thus, the combined complexity of our sparse matrix inversion is

$$O\left(\sum_{m=1}^{M} N_m^3 + N_M N_m^2\right), \tag{3.26}$$

which is significantly smaller than the cubic complexity on the full data set, i.e.,

$$O\left(\left(\sum_{m=1}^{M} N_m\right)^3\right). \tag{3.27}$$

The improved complexity of the matrix inversion renders the method scalable to larger PoD data sets.

5 Experiments

First, we experimentally analyze our method on the well-known mountain car [Moore, 1993] and cart-pole [Barto et al., 1983] benchmark problems. We consider a PoD of 3 members. The PoD consists of a target, a similar source member A and a significantly different source member B.

We adopt an off-line reinforcement learning setting, in which a batch of transition samples for the complete PoD is available to the method prior to learning. To construct this batch, we sample the environments of sources A and B sufficiently, such that the operational behaviors are well-represented by the transition model of the respective PoD members. We provide the target with a limited amount of data sampled from its own environment. The target cannot sufficiently estimate its transition model based on these samples, making it challenging to find the optimal policy. Therefore, transferring knowledge from member A will assist the target in finding the optimal policy. However, the environment in which member B operates is different from the target's environment, and sharing samples with member B would misinform the target's transition model. Therefore, the objective of the target is to estimate a sufficiently accurate transition model by estimating the correlations with all sources and use the sources' knowledge proportional to the estimated correlation.

Next, we apply our method on a state-of-the-art wind farm simulator [Gebraad et al., 2017] to demonstrate our method's real-world benefits on larger PoDs. We consider a PoD of 8 members. Again, we have a single target and sample the environments of the other PoD members. We assume that 3 sources are similar to the target, while the other 4 sources are different.

Once the transition model is learned, we compute the optimal value function and policy using the GPRL method presented in Section 1.2. We consider an off-line batch reinforcement learning setting and provide the learner with a random batch of transition samples. The benchmarks are deterministic environments, and we thus set the observational noise of the Gaussian process to 10^{-8} to ensure numerical stability. The discount factor γ is set to 0.99 and the observational noise of the Gaussian process fitted on the values is set to 0.1 to prevent overfitting.

In all experiments, we compare our method against two baselines, i.e., learning with a single target and learning with a joint target. The single target only uses its own samples to learn a transition model, while the joint target considers all PoD data jointly, assuming all members are identical. Specifically, we construct transition models that use the squared-exponential kernel described in Equation 3.4, fitted only on the target's own samples for the single target type, or using all PoD samples for the joint target type. For the PoD target type, we use our method to fit a transition model, using the PoD kernel described in Equation 3.17, based on all PoD samples.[2]

5.1 Mountain Car

To illustrate our method, we set up the continuous mountain car domain [Moore, 1990]. The car is positioned in a valley and its objective is to reach the top of the right-most hill. However, the slope is too steep for the car to simply accelerate to the top. Thus, it has to first drive up the opposite side of the valley and then accelerate from there to reach the top.

In this problem, a state consists of the position of the car (in $[-1.2, 0.6]$) and the velocity of the car (in $[-0.07, 0.07]$), while an action is a force applied to either side of the car (in $[-1, 1]$ multiplied by a power parameter). The start and goal states are, respectively, given by $\boldsymbol{s}_{\text{start}} = [-0.5, 0]$ and $\boldsymbol{s}_{\text{goal}} = [0.45, 0]$, i.e., the bottom and top of the hill when the car is at a standstill. The start state is depicted in Figure 3.2. The standard deviation of the reward function σ_R is set to 0.05. We use 200 support points, generated using Latin hypercube sampling [McKay et al., 1979], for which GPRL will estimate the values during the policy evaluation step.

We consider a PoD of three mountain cars: a target with a power of $1.5 \cdot 10^{-3}$ units, source A with power 10^{-3} and source B with power 10^{-4}. For each source, we provide a batch of 100 transitions sampled uniformly random from its environment. We do the same for the target, but only sample 20 times, resulting in a total of 220 samples. We run the experiment 50 times for the three target types: single, joint and PoD.

[2]The source code to reproduce the experiments is publicly available at: https://github.com/timo-verstraeten/fwpi-experiments

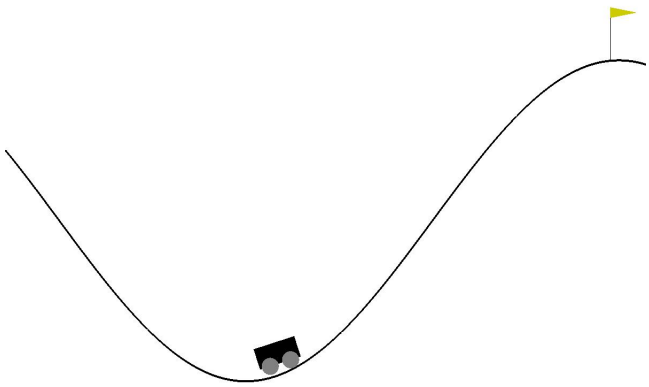
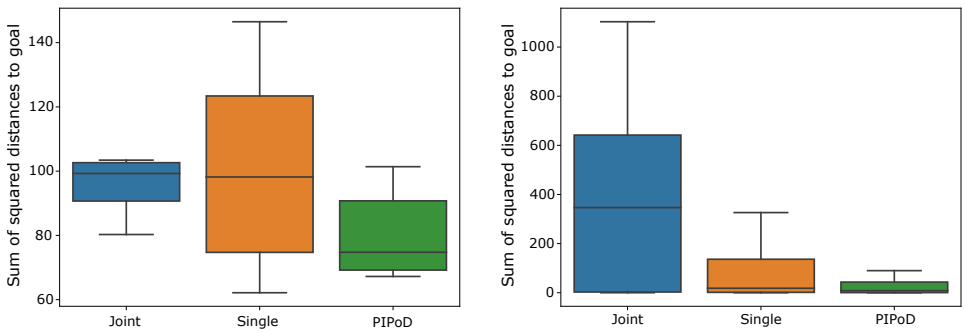


Figure 3.2: Mountain car environment from OpenAI gym [Brockman et al., 2016].



(a) Mountain car

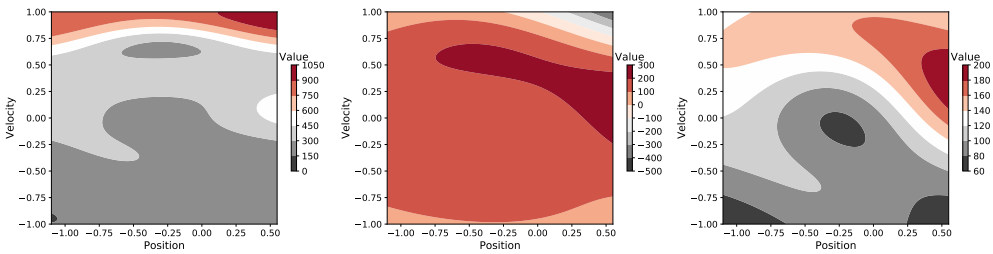
(b) Cart-Pole

Figure 3.3: Boxplot of the total sum of squared distances to the goal state for the mountain car (a) and cart-pole (b) benchmarks during 200 time steps. The experiment is repeated 50 times for each benchmark.

We measure the performance of the methods by reporting the total sum of squared distances to the goal state during 200 time steps.[3] The results are shown in Figure 3.3a. We observe that the joint target (i.e., learning from the full data set without the PoD

[3]The success rates of solving the task for each of the target types are reported in Appendix 1.

kernel) rarely reaches the goal. This is because the target uses the data of source B, which does not have enough power to reach the goal. Therefore, the target does not expect to reach the goal and is often remaining at the bottom of the hill during runs. The single target (i.e., learning from own experiences) can sometimes achieve good results, but is unable to accurately represent its operational behavior, due to the limited amount of data it can learn from. Because of the uncertainty in the transition model, the car is often incapable of finding a suitable policy. The PoD target consistently achieves good results, as the target is able to figure out which source is most useful to share data with through the PoD kernel.



(a) Single (b) Joint (c) PIPoD

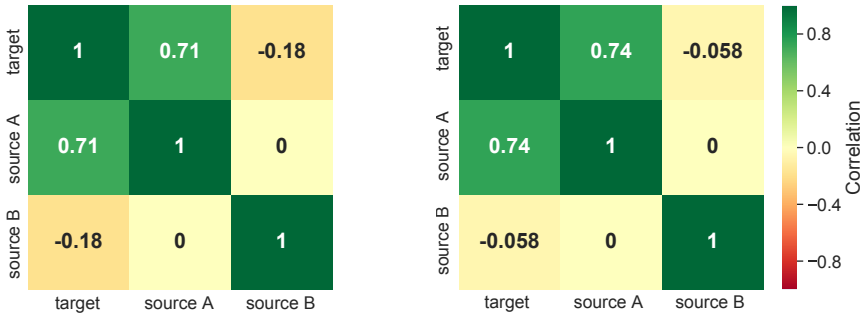
Figure 3.4: Mountain car – Contour plots of the learned value functions (means of the Gaussian processes) during the best runs of each target type. The value (color) is depicted for all states (axes) and for each type of learning, i.e., single learning (a), joint learning (b) and PIPoD (c).

In Figure 3.4, we show the resulting value function during the best performant run for each of the target types. We can see that that the region with highest value matches the goal state for the PoD target, while the single and joint targets misidentify this region. The average standard deviation over the surface of the Gaussian processes fitted on the values is 115 for the PoD target, 590 for the joint target and 3906 for the single target. This indicates that the single target is not confident about its value function, due to the lack of data.

Next, we plot the correlation matrices learned by the PoD target, averaged over all runs. Given the optimized cross-covariance matrix F from Equation 3.17, we can compute the correlation matrix

$$\text{corr}(F) = (\text{diag}(F))^{-0.5} F (\text{diag}(F))^{-0.5}, \tag{3.28}$$

where $\text{diag}(F)$ is the diagonal matrix constructed from the diagonal elements of F. The element-wise means of these matrices over all runs are given in Figure 3.5. The PoD target



(a) Position (b) Velocity

Figure 3.5: Mountain car – Optimized correlation matrix between the PoD members, i.e., target, source A and source B for both state dimensions.

successfully identifies source A to be similar, while assigning a notably lower correlation value to source B.

5.2 Cart-Pole

In the cart-pole domain, the goal is to keep a pole balanced on top of a controllable cart. Cart-pole is an underactuated system, which means that the system has more degrees of freedom than actuators. Balancing the pole is challenging, as its equilibrium is highly unstable.

In this problem, a state consists of the position of the cart (in $[-4.8, 4.8]$), the angular position of the pole (in $[-0.42, 0.42]$), the velocity of the cart (in $[-2, 2]$) and the angular velocity of the pole (in $[-2, 2]$). The start and goal state are the same, namely, at the equilibrium, i.e., $\boldsymbol{s}_{\text{start}} = \boldsymbol{s}_{\text{goal}} = [0, 0]$. The start state is depicted in Figure 3.6. The standard deviation of the reward function σ_R is set to 0.2. We set the number of support points to 300.

We consider a PoD of three carts: a target with a pole mass of 0.1 units, source A with mass 0.2 and source B with mass 0.5. For each source, we provide a batch of 50 transitions sampled uniformly random from its environment. We do the same for the target, but only sample 5 times, resulting in a total of 105 transitions. We run the experiment 50 times for the three target types: single, joint and PoD.

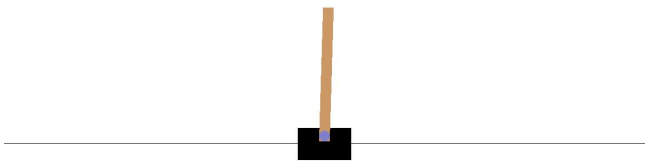


Figure 3.6: Cart-pole environment from OpenAI gym [Brockman et al., 2016].

We measure the performance of the methods by reporting the total sum of squared distances from the equilibrium during 200 time steps.[3] The results are shown in Figure 3.3b. Due to the instability of the equilibrium, it is necessary to accurately represent the transition model. The joint target fails to achieve this, as it aggregates samples over different transition models. The single target achieves better results, but is often uncertain about its transition model, leading to suboptimal behavior. The PoD target consistently manages to keep the pole around its equilibrium.

5.3 Wind Turbines

To demonstrate the need for our method in practical applications, we introduce a new setting in the context of wind energy. Current wind turbine controllers position the rotors toward the measured incoming wind vector to ensure high productivity [Boersma et al., 2017]. However, as wind passes through upstream turbines, the wind speed is reduced and the energy extracted by downstream turbines is significantly lower, which is referred to as the wake effect. When considering wind farms (i.e., groups of wind turbines), it is essential to take this effect in consideration [González-Longatt et al., 2012]. In recent work, steering wake through rotor misalignment is investigated [Gebraad et al., 2017; Bargiacchi et al., 2018]. For example, in a setting with two wind turbines, the upstream turbine slightly misaligns its rotor to deflect the wake effect and improve the productivity of the downstream turbine.

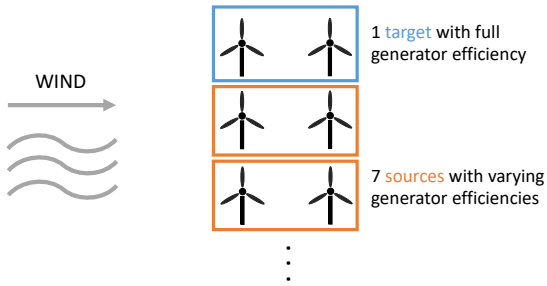


Figure 3.7: Wind turbine control benchmark with 1 target and 7 sources with varying generator efficiencies.

Due to the complexity of the wake effect and incomplete knowledge about a turbine's condition, it is necessary to gather data in the field about potential control policies, rendering it a reinforcement learning problem. As learning policies from scratch could result in potential revenue loss, PIPoD can improve the learning speed. Moreover, our batch reinforcement learning setting makes sense, as wind farm operators first need to thoroughly assess the performance of the acquired policy before implementing it [Boersma et al., 2017].

We demonstrate our method on a PoD of two-turbine rows in a wind farm that consists of 8 rows and show how information exchange between transition models can improve the learning speed and accuracy. We use the state-of-the-art open source FLORIS simulator to model the wind farm operating conditions and wake effect [National Renewable Energy Laboratory (NREL), 2019], and use the 5 MW reference turbine description from the National Renewable Energy Laboratory to model the individual wind turbines [Jonkman et al., 2009].

In this environment, the state of a member consists of the orientations of both turbines (values in $[-45, 45]$ degrees with respect to the wind vector) and the associated total power production (values in $[0.5, 1.05]$ MW). The actions are changes in orientation with values of either -1, 0 or 1 degrees. The start state is both turbines aligned with the wind vector, which is current practice in wind turbine control [Boersma et al., 2017]. The goal s_{goal} is centered around a power production of 1.07 MW with a scale σ_R of 0.05 to encourage high productivity. The number of support points is set to 300.

Each two-turbine row represents a PoD member. Again, we report the results for one target. This is considered to be a new row of which the generator efficiency is set to 1. However, we set the generator efficiencies to 0.9 for 3 source members, and to 0.8 for the

remaining 4 source members, which is a realistic configuration that could be the result of aging [Staffell and Green, 2014]. The wind speed is set to 6 m/s. We assume independence between turbine rows, which is a reasonable assumption given the wind vector that we use in our experiments, since wake generated by one row will not influence the other turbine rows. The setup is depicted in Figure 3.7.

To each turbine row, we provide a batch of 50 transitions randomly sampled from its environment. We measure the performance of the methods by reporting the power production (MW) achieved at the end of the run. We compare the targets to the performance achieved under the optimal policy and to the performance under the policy used in current practice, i.e., aligning all turbines with the incoming wind vector. The results are shown in Figure 3.8.

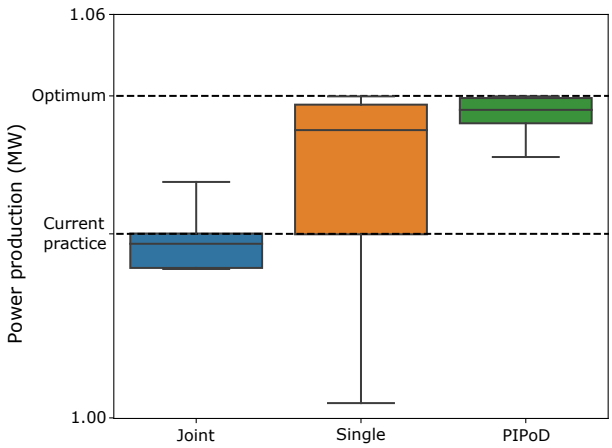


Figure 3.8: Wind Farm – Boxplot of the power productions for each target type over 50 runs. The optimal performance, as well as the performance achieved when using the control policy used in current practice, are given (dashed lines).

To each member, we provide a batch of 50 transitions sampled uniformly random from their environment, resulting in 400 PoD samples. We run the experiment 50 times for each of the target types: single, joint and PoD, and show the results in Figure 3.8.

We observe that the single target has a wide variance on its performance. The uncertainty about its transition model is high due to the limited amount of data it has access to. The joint target has lower variance, but has the worst performance, close to the performance of current practice policies. As all data is aggregated, many transition samples are not representative for the true operational behavior of the target. The PoD target consistently

achieves results that approach the optimal performance, as it has the ability to differentiate between relevant and irrelevant samples over the entire PoD data set.

6 Discussion

We introduced a new sample-efficient reinforcement learning method for PoDs, called policy iteration for pools of devices (PIPoD), based on Gaussian processes and coregionalization. Our method estimates cross-covariances between a pool of devices and transfers knowledge between them.

We provided experimental results on two benchmark problems: mountain car and cart-pole. PIPoD outperforms the two baselines on all of our experiments. The joint target regularly fails to reach the goal while the single target remains uncertain about its transition model. This exposes the need to balance between accuracy and confidence, where we have to decide to either use all data at the risk of misrepresenting the transition model (high estimation bias) or only use representative data while remaining uncertain about the model (low confidence). Our method successfully balances both by properly weighing each source with their correlation with the target. This is reflected in the learned value functions. The PoD target finds the region of highest reward, which is around the goal state, while the single and joint targets misrepresent their value function. We further validated the ability of our method to balance between accuracy and confidence through a sensitivity analysis on the mountain car setting. Specifically, we varied the power parameter of source A between $5 \cdot 10^{-3}$ and $15 \cdot 10^{-3}$ to simulate a range of similarities between source A and the target. The PoD target outperforms both baselines and exhibits similar performance to the single target when the target is significantly different from source A, and thus no information transfer is possible. More information on this analysis can be found in Appendix 2.

The successful exchange of data in PoDs has strong implications for real-world applications. The wind farm experiment shows that close-to-optimal performance can be achieved when using PIPoD, while the alternatives (i.e., single and joint learning) often yield performances close to current practice or worse.

The introduced transition model is a sparse variant of the intrinsic coregionalization model (ICM) [Goovaerts, 1997; Bonilla et al., 2008]. This model captures cross-covariances between multiple functions, and thus improves the accuracy of those functions jointly. However, as we consider multiple sources and a single target, the target's transition model will be tailored toward improving its own accuracy, rather than the joint accuracy over all PoD members. Additionally, the computational burden when using our sparse coregionalization model is significantly reduced. Our method can be executed in a distributed manner over the PoD which reduces the quadratic complexity of the covariance matrix inversion in the intrinsic coregionalization model to a linear complexity per target

member. Moreover, as the covariances in Equation 3.16 are sparse, the inversion operation can be made linear in the number of PoD members as well. This renders our method scalable to larger PoDs. We perform additional experiments to compare our sparse model against the intrinsic coregionalization model in Appendix 3.

In future work, we will further improve the scalability of our method by using sparse Gaussian process approximations [Snelson and Ghahramani, 2006; Wilson and Nickisch, 2015] to reduce the computational burden of the matrix inversion. As many of these methods are independent with respect to the covariance kernel or require a factorable kernel, our model is extensible to many of these approximations. By using a sparse Gaussian process approximation, our reinforcement learning method can handle even larger data sets and PoDs. Additionally, as we focused on settings with highly similar devices, we considered simple transformations of the data. Specifically, we assumed that the correlation values are the same over all states. In future work, we will consider more complex settings, in which the data transformations may be state-dependent. To this end, we will investigate the use of generalized Wishart processes to allow the correlation matrix to vary over states [Wilson and Ghahramani, 2011]. Finally, more complex settings may require non-linear transformations of the data. Therefore, in future work we will investigate the use of Gaussian Copula processes [Wilson and Ghahramani, 2010]. Gaussian Copula processes separate the dependency structure between random variables from their marginal distributions. This means that these variables can be transformed independently of the correlations that exist between them. In our case, such a mechanism maintains the transparency of the covariance kernel, which comprises the similarity structure of the devices, even when complex operations are performed on the transition models.

As demonstrated in this chapter, exploiting similarities between devices can significantly improve sample-efficiency. Still, in many PoD applications, the actions taken by multiple devices are inter-dependent. In these situations, devices must coordinate to jointly optimize a control strategy over the complete PoD [Busoniu et al., 2006]. However, for larger PoDs it becomes intractable to learn joint strategies, due to the exponential increase of the joint action space in terms of the number of devices. Therefore, in the next chapter, we propose a new control algorithm, called multi-agent Thompson sampling, that exploits sparse neighbourhood structures to effectively coordinate and learn joint strategies in large multi-agent systems.

4 Thompson Sampling for Multi-Agent Bandits

Multi-agent coordination is prevalent in many real-world applications, such as traffic light control [Wiering, 2000], warehouse commissioning [Claes et al., 2017], mitigation of epidemics [Libin et al., 2020] and wind farm control [Gebraad and van Wingerden, 2015]. Often, such settings can be formulated as coordination problems in which agents have to cooperate in order to optimize a shared team reward [Boutilier, 1996; Kapetanakis and Kudenko, 2002].

Handling multi-agent settings is challenging, as the size of the joint action space scales exponentially with the number of agents in the system. Therefore, an approach that directly considers all agents' actions jointly is computationally intractable. This has made such coordination problems the central focus in the planning literature [Koller and Parr, 2000; Guestrin et al., 2001a,b, 2002]. In this regard, we note that in real-world settings agents often only directly affect a limited set of neighboring agents. This means that the global reward received by all agents can be decomposed into local components that only depend on small subsets of agents. Exploiting such loose couplings is key in order to keep multi-agent decision problems tractable [Chapman et al., 2013; De Hauwere, 2011].

For example, consider a wind farm control task, which is comprised of a set of wind turbines, and we aim to maximize the farm's total productivity. When upstream turbines directly face the incoming wind stream, energy is extracted from the wind. This reduces the productivity of downstream turbines, potentially lowering the overall power production.

However, turbines have the option to rotate in order to deflect the turbulent flow away from turbines downwind [van Dijk et al., 2016]. Due to the complex nature of the aerodynamic interactions between the turbines, constructing a model of the environment and deriving a control policy using planning techniques is challenging [Marden et al., 2013]. Instead, a joint control policy among the turbines can be *learned* to effectively maximize the productivity of the wind farm. The system is loosely coupled, as redirection only directly affects adjacent turbines.

While the state of the art mainly focusses on empirical evaluation of approximate reinforcement learning methods in multi-agent systems, it has recently been shown [Bargiacchi et al., 2018] that it is possible to achieve theoretical bounds on the cumulative regret (i.e., how much reward is lost due to learning). Such bounds are necessary to confidently state guarantees toward end users. However, it remains challenging to incorporate prior domain knowledge about data and statistics of the problem using state-of-the-art methods. The inclusion of such knowledge can drastically improve learning speed, as it can effectively guide exploration toward high-potential control policies. We argue that this has strong relevance in many practical fields, such as on-line advertisement [Chapelle and Li, 2011; Rhuggenaath et al., 2019], influenza mitigation [Libin et al., 2018, 2019] and wind farm control [Verstraeten et al., 2019].

To this end, we propose a new multi-agent decision making algorithm, called multi-agent Thompson sampling (MATS), which exploits loosely-coupled interactions in multi-agent systems. Specifically, we target problem settings that can be formalized as a multi-agent multi-armed bandit. Our method leverages the exploration-exploitation mechanism of Thompson sampling, which allows it to quantify uncertainty about domain-speficic statistics and guide the learning process based on this uncertainty. We perform a finite-time Bayesian regret analysis and show that the upper regret bound of MATS is low-order polynomial in the number of actions of a single agent for sparse coordination graphs. This is a significant improvement over the exponential bound of classic Thompson sampling, which is obtained when the coordination graph is ignored [Agrawal and Goyal, 2012]. The full proof for the cumulative regret bound, together with a proof outline, is provided in Section 4. We show that MATS improves upon the state of the art in various synthetic settings. Finally, we demonstrate that MATS achieves high performance in a synthetic wind farm control task, in which multiple wind turbines have to be jointly aligned to maximize the total power production.

We start by providing background information on multi-agent multi-armed bandits and variable elimination in Section 1. Then, we position our research with respect to related work in Section 2. Next, we propose MATS, a new control method to factorize multi-agent systems using coordination graphs, in Section 2. We provide a problem-independent upper bound on the cumulative regret of MATS, as well as a proof outline and the full proof,

in Section 4. Afterwards, we evaluate our method on a variety of benchmark settings, as well as a synthetic wind farm control task, in Section 5.1. Finally, we discuss the results and insights in Section 6.

The work presented in this chapter was published in Nature's Scientific Reports and the AAMAS proceedings.

- *Verstraeten, T., Bargiacchi, E., Libin, P. J. K., Helsen, J., Roijers, D. M. and Nowé, A. (2020), Multi-Agent Thompson Sampling for Bandit Applications with Sparse Neighbourhood Structures, Scientific Reports, 10*
- *Verstraeten, T., Bargiacchi, E., Libin, P. J. K., Roijers, D. M. and Nowé, A. (2020), Thompson Sampling for Factored Multi-Agent Bandits, Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), pp. 2029–2031*

1 Background

1.1 Multi-Agent Multi-Armed Bandits

To formalize a multi-agent decision making process with loose couplings, we adopt the multi-agent multi-armed bandit framework [Bargiacchi et al., 2018; Stranders et al., 2012]. This framework is similar to the multi-armed bandit formalism [Thompson, 1933], but considers multiple agents factored into groups. When a joint action, i.e., a joint arm, has been executed, each group receives a reward. The goal shared by all agents is to maximize the total sum of rewards.

Definition 5

A multi-agent multi-armed bandit is a tuple $\langle \mathcal{D}, \mathcal{A}, R \rangle$ where

- $\mathcal{D}$ is the set of agents, factored into ρ, possibly overlapping, groups $\mathcal{D}^e$, where e is the group index.
- $\mathcal{A} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_{|\mathcal{D}|}$ is the set of joint actions, or joint arms, which is the Cartesian product of the sets of actions $\mathcal{A}_i$ for each agent i. We denote $\mathcal{A}^e$ as the set of local joint actions, or local arms, for the group $\mathcal{D}^e$.
- $R(\boldsymbol{a})$ is a stochastic function providing a global reward when a joint arm, $\boldsymbol{a} \in \mathcal{A}$, is pulled. The global reward function is decomposed into ρ noisy, observable and independent local reward functions, i.e., $R(\boldsymbol{a}) = \sum_{e=1}^{\rho} R^e(\boldsymbol{a}^e)$. A local reward function R^e only depends on the local joint arm $\boldsymbol{a}^e \in \mathcal{A}^e$ of the subset of agents in $\mathcal{D}^e$.

We denote the mean reward of a joint arm as

$$\mu(\boldsymbol{a}) = \sum_{e=1}^{\rho} \mu^e(\boldsymbol{a}^e),$$

with $\mu^e(\boldsymbol{a}^e) = \mathbb{E}\left[R^e(\boldsymbol{a}^e)\right]$. For simplicity, we refer to the ith agent by its index i.

The dependencies between the local reward functions and the agents are described as a coordination graph [Guestrin et al., 2001b].

Definition 6

A coordination graph is a bipartite graph $G = \langle \mathcal{D}, \{R^e\}_{e=1}^{\rho}, E \rangle$, whose nodes $\mathcal{D}$ are agents and components of a factored reward function $R = \sum_{e=1}^{\rho} R^e$, and an edge $\langle i, R^e \rangle \in E$ exists if and only if agent i influences component R^e.

The dependencies in a multi-agent multi-armed bandit can be described by setting $E = \{\langle i, R^e \rangle \mid i \in \mathcal{D}^e\}$. An example of a coordination graph is given in Figure 4.1.

Similar to a single-agent multi-armed bandit, the objective is to minimize the expected cumulative regret (see Definition 2) by means of exploration and exploitation of the joint arm that is believed to be optimal. However, in this case, the global means can be written as a sum of local means per group. Therefore, the expected cumulative regret is the total cost incurred when jointly pulling a set of *local* arms instead of the optimal ones over all groups.

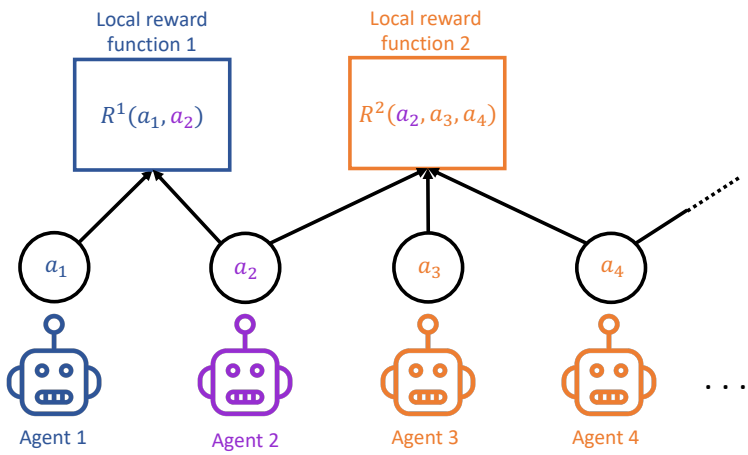


Figure 4.1: Example of a coordination graph – The reward function is factored into two local reward functions R^1 and R^2. R^1 is dependent on the actions of agents 1 and 2, while R^2 is dependent on the actions of agents 2, 3 and 4. Note that the action taken by agent 2 affects both functions.

Definition 7

The expected cumulative regret of pulling a sequence of joint arms until time step T according to policy π is

$$\mathbb{E}\left[\mathcal{R}(T,\pi)\right] = \mathbb{E}\left[\sum_{t=1}^{T} \Delta(\boldsymbol{a}_t) \,\middle|\, \pi\right] \tag{4.1}$$

with

$$\begin{aligned}\Delta(\boldsymbol{a}_t) &= \mu(\boldsymbol{a}_*) - \mu(\boldsymbol{a}_t) \\ &= \sum_{e=1}^{\rho} \mu^e(\boldsymbol{a}_*^e) - \mu^e(\boldsymbol{a}_t^e),\end{aligned} \tag{4.2}$$

where $\boldsymbol{a}_*$ is the optimal joint arm and $\boldsymbol{a}_t$ is the joint arm pulled at time t. For the sake of brevity, we will omit π when the context is clear.

Similar to Definition 2, it is typically assumed that $\boldsymbol{a}_*$ is a known value. When $\boldsymbol{a}_*$ is a random variable, and the regret is marginalized over this variable, we use the term *Bayesian regret*.

Cumulative regret can be minimized by using a policy that considers the full joint arm space, thereby ignoring loose couplings between agents. This leads to a combinatorial problem, as the joint arm space scales exponentially with the number of agents. Therefore, loose couplings should be considered whenever possible.

1.2 Variable Elimination in Coordination Graphs

To encourage exploitation in the learning process, we need to find the joint arm that is believed to be optimal. Specifically, we are interested in finding the joint arm that maximizes the global reward function, i.e.,

$$\arg\max_{\boldsymbol{a}} R(\boldsymbol{a}) = \arg\max_{\boldsymbol{a}} \sum_{e=1}^{\rho} R^e(\boldsymbol{a}^e) \tag{4.3}$$

However, as the joint arm comprises a combination of the decisions made by multiple agents, finding this arm is a combinatorial optimization problem, which is NP-hard [Wolsey and Nemhauser, 1999].

Efficient approaches for joint action optimization exist when a coordination graph is available, which significantly reduces the computational complexity of the optimization problem. Still, when using a factored representation of the joint action space, finding the best joint arm is not trivial, as conflicts may arise when an agent belongs to two groups. Figure 4.2 shows an example, in which three agents need to coordinate. The global reward function is decomposed into two factors, i.e., a local function that depends on agents 1 and 2 and a local function that depends on agents 2 and 3. In this case, optimizing the local reward for each group independently is impossible, as the locally-optimal action for agent 2 differs over the groups that it belongs to. Therefore, it is important to use an approach that ensures global optimization of the reward function, while deciding on the agents' actions on a local level.

One approach to perform global optimization using a factored representation is variable elimination [Zhang and Poole, 1994]. This technique originated in the context of Bayesian networks, which allowed for the fast computation of conditional or marginal distributions using the joint probability factors derived from a probabilistic graphical model. Later, variable elimination was used in the context of multi-agent coordination graphs to compute the joint action that maximizes the global reward without explicitly enumerating over the full joint action space [Guestrin et al., 2001b]. Instead, variable elimination consecutively

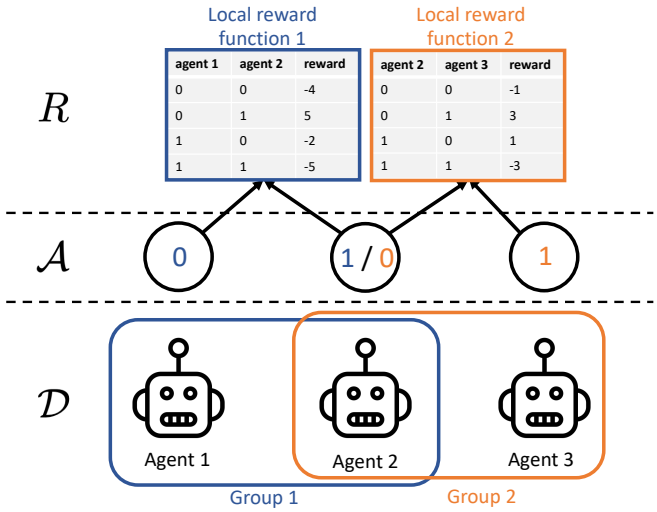


Figure 4.2: Binary optimization problem with three agents. The global reward function is factored into two local reward functions, i.e., one reward function dependent on agents 1 and 2 and one dependent on agents 2 and 3.

eliminates an agent from the coordination graph, while computing its best response with respect to its neighbors responses.

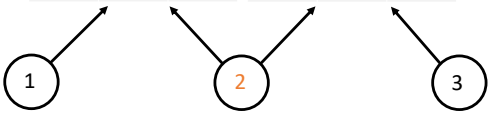
First, variable elimination picks an agent $i \in \mathcal{D}$, and decomposes the maximization problem into two terms, i.e., one with only the groups to which i belongs and one with groups to which it does not.

$$\begin{aligned}
&\max_{\boldsymbol{a}} \sum_{e=1}^{\rho} R^e(\boldsymbol{a}^e) \\
&= \max_{\boldsymbol{a}_{-i}} \max_{a_i} \sum_{e=1}^{\rho} R^e(\boldsymbol{a}^e) \\
&= \max_{\boldsymbol{a}_{-i}} \left(\max_{a_i} \sum_{i \in \mathcal{D}^e} R^e(\boldsymbol{a}^e_{-i}, a^e_i) \right) + \left(\sum_{i \notin \mathcal{D}^e} R^e(\boldsymbol{a}^e) \right),
\end{aligned} \tag{4.4}$$

where $\boldsymbol{a}_{-i}$ is the global joint action without agent i, and a_i is the action of agent i.

agent 1	agent 2	reward
0	0	-4
0	1	5
1	0	-2
1	1	-5

agent 2	agent 3	reward
0	0	-1
0	1	3
1	0	1
1	1	-3

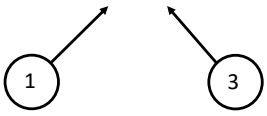


(a) Coordination graph

agent 1	agent 2	agent 3	total reward
0	0	0	-5
0	0	1	-1
0	1	0	6
0	1	1	2
1	0	0	-3
1	0	1	1
1	1	0	-4
1	1	1	-8

(b) Joint reward function

agent 1	agent 3	reward
0	0	6
0	1	2
1	0	-3
1	1	1



(c) New reward function without agent 2

agent 1	agent 3	agent 2
0	0	1
0	1	1
1	0	0
1	1	0

(d) Conditional policy of agent 2

Figure 4.3: Example of a variable elimination step – First, agent 2 is picked from the coordination graph (a). Next, the total reward is computed for all action combinations among agent 2 and its neighbors (b). For each joint action taken by the neighbors, the best response of agent 2 is recorded (green). Finally, these joint actions are used to construct a new reward function (c) and the conditional policy of agent 2 (d).

Next, it focuses on the smaller maximization problem involving agent i, and creates a new group containing all neighbors of agent i.

$$\mathcal{D}^{e'} = \cup_{e=1}^{\rho} \{j \mid i \in \mathcal{D}^e \wedge j \in \mathcal{D}^e\} \setminus \{i\} \tag{4.5}$$

It selects the best response of agent i for each possible combination of its neighbors' actions. This results in a conditional policy for agent i

$$\pi\left(\boldsymbol{a}_{-i}^{e'}\right) = \arg\max_{a_i} \sum_{i \in \mathcal{D}^e} R^e(\boldsymbol{a}_{-i}^e, a_i), \tag{4.6}$$

The second term in Equation 4.4 can then be replaced by a new function that only involves the neighbors of agent i, effectively eliminating agent i from the maximization problem:

$$R^{e'}(\boldsymbol{a}^{e'}_{-i}) = \sum_{i \in \mathcal{D}^e} R^e(\boldsymbol{a}^e_{-i}, \pi(\boldsymbol{a}^{e'}_{-i})), \tag{4.7}$$

The elimination process is repeated until every agent is eliminated from the graph. An example of an elimination step is depicted in Figure 4.3.

Variable elimination is guaranteed to return the optimal joint arm and has a computational complexity that is combinatorial in terms of the induced width of the graph, i.e., the number of neighbors of an agent at the time of its elimination. Specifically, variable elimination has a computational complexity of

$$O\left(\prod_{i \in \mathcal{D}^{\max}} |\mathcal{A}_i|\right), \tag{4.8}$$

where $\mathcal{D}^{\max}$ is the largest group of agents constructed during the elimination process (see Equation 4.5). As the method is typically applied to a loosely-coupled coordination graph, the induced width is much smaller than the size of the full joint action space, which renders the maximization problem tractable [Guestrin et al., 2001b, 2002].

2 Related work

Multi-agent reinforcement learning and planning with loose couplings has been investigated in sequential decision problems [Guestrin et al., 2002; Kok and Vlassis, 2006; De Hauwere et al., 2010; Scharpff et al., 2016; Brys et al., 2011]. In sequential settings, the value function cannot be factorized exactly. Therefore, it is challenging to provide convergence and optimality guarantees. While for planning some theoretical guarantees can be provided [Scharpff et al., 2016], in the learning literature the focus has been on empirical validation [Kok and Vlassis, 2006]. In this work, we focus on multi-agent multi-armed bandits, which are single-shot stateless problems. In such settings, the reward function is factored into components that only depend on a subset of agents.

The combinatorial bandit [Bubeck and Cesa-Bianchi, 2012; Cesa-Bianchi and Lugosi, 2012; Gai et al., 2012; Chen et al., 2013b] is a variant of the multi-armed bandit, in which, rather than one-dimensional arms, an arm vector has to be pulled. In our work, the arms' dimensionality corresponds to the number of agents in our system, and similarly to combinatorial bandits, the number of arms exponentially increases with this quantity. We consider a variant of this framework, called the semi-bandit problem [Audibert et al., 2011], in which local components of the global reward are observable when a joint arm

is evaluated. An application of this framework is in on-line advertisement, in which the advertiser can select a subset of web pages to maximize the total number of users that click on the advertisement. Chen et. al (2013) constructed an algorithm for this setting that assumes access to an (α, β)-oracle, which combines local arms into a joint arm that outputs a fraction α of the optimal expected reward with probability β. Instead, we assume the availability of a coordination graph, which we argue is a reasonable assumption in many multi-agent settings.

Sparse cooperative Q-learning is an algorithm that also assumes the availability of a coordination graph [Kok and Vlassis, 2004]. However, although strong experimental results are given, no theoretical guarantees were provided. Later, the UCB-like algorithm, HEIST, for exploration and exploitation in multi-agent multi-armed bandits was introduced [Stranders et al., 2012], which uses a message-passing scheme for resolving coordination graphs. They provide some theoretical guarantees on the regret for problems with acyclic coordination graphs. Multi-agent upper-confidence exploration (MAUCE) [Bargiacchi et al., 2018] is a more general method that uses variable elimination to resolve (potentially cyclic) coordination graphs. MAUCE demonstrates high performance on a variety of benchmarks and provides a tight theoretical upper bound on the regret. MATS provides a Bayesian alternative to MAUCE based on Thompson sampling.

Our problem definition is related to distributed constraint optimization (DCOP) problems [Yokoo et al., 1998]. In DCOPs, multiple agents control a set of variables in a distributed manner under a set of constraints. The objective is the same as for a multi-agent multi-armed bandit, i.e., optimize the sum over group rewards. However, in DCOPs, the rewards are assumed to be known beforehand. Distributed coordination of exploration and exploitation [Taylor et al., 2011] extends this setting to unknown rewards and considers the optimization of the cumulative reward achieved over a time span. Multi-agent multi-armed bandits, or multi-armed bandit DCOPs [Stranders et al., 2012], consider the optimization of a single-step expected reward over time.

Multi-agent multi-armed bandits can be used in the context of wind farm control, in which the farm's power output must be maximized by controlling the turbines jointly. In recent research on wind farm control, the impact of optimized rotor alignments on power production is investigated [van Dijk et al., 2016]. To search for the optimal alignments within the wind farm, data-driven methods are usually adopted, where the turbines' alignments are perturbed iteratively until they locally converge [Marden et al., 2013]. When optimizing the alignment of a wind turbine, only considering its neighbors can significantly boost the learning speed [Gebraad and van Wingerden, 2015]. MATS is also able to leverage neighborhood structures. In addition, rather than random perturbation of the alignments, MATS leverages an exploration-exploitation mechanism that is inspired by Thompson sampling and variable elimination, which allows for a global exploration

mechanism that targets the optimal alignment configuration, while retaining a small regret during the learning process itself.

3 Multi-Agent Thompson Sampling

We propose the multi-agent Thompson sampling (MATS) algorithm for decision making in loosely-coupled multi-agent multi-armed bandit problems. Consider a multi-agent multi-armed bandit with groups $\mathcal{D}^e$ (Definition 5). The local means $\mu^e(\boldsymbol{a}^e)$ are treated as unknown. According to the Bayesian formalism, we assert our beliefs over the local means $\mu^e(\boldsymbol{a}^e)$ in the form of a prior, $Q^e_{\boldsymbol{a}^e}(\cdot)$. At each time step t, MATS draws a sample $\mu^e_t(\boldsymbol{a}^e)$ from the posterior for each group and local arm given the history, $\mathcal{H}_{t-1}$, consisting of local actions and rewards associated with past pulls:

$$\begin{aligned} &\mu^e_t(\boldsymbol{a}^e) \sim Q^e_{\boldsymbol{a}^e}(\cdot \mid \mathcal{H}_{t-1}) \\ &\mathcal{H}_{t-1} \triangleq \cup_{i=1}^{t-1} \cup_{e=1}^{\rho} \{\langle \boldsymbol{a}^e_i, R^e_i(\boldsymbol{a}^e_i)\rangle\}. \end{aligned} \tag{4.9}$$

Note that during this step, MATS samples directly the posterior over the unknown local means, which implies that the sample $\mu^e_t(\boldsymbol{a}^e)$ and the unknown mean $\mu^e(\boldsymbol{a}^e)$ are independent and identically distributed at time step t, given the history $\mathcal{H}_{t-1}$.

Thompson sampling chooses the arm with the highest sample, i.e.,

$$\boldsymbol{a}_t = \arg\max_{\boldsymbol{a}} \mu_t(\boldsymbol{a}). \tag{4.10}$$

However, in our case, the expected reward is decomposed into several local means. As conflicts between overlapping groups will arise, the optimal local arms for an agent in two groups may differ. Therefore, we must define the argmax-operator to deal with the factored representation of a multi-agent multi-armed bandit, while still returning the full joint arm that maximizes the sum of samples, i.e.,

$$\boldsymbol{a}_t = \arg\max_{\boldsymbol{a}} \sum_{e=1}^{\rho} \mu^e_t(\boldsymbol{a}^e). \tag{4.11}$$

To this end, we use variable elimination, which computes the joint arm that maximizes the global reward without explicitly enumerating over the full joint arm space [Guestrin et al., 2001b]. Specifically, variable elimination consecutively eliminates an agent from the coordination graph, while computing its best response with respect to its neighbors actions. Variable elimination is guaranteed to return the optimal joint arm and has a computational complexity that is combinatorial in terms of the induced width of the graph. Since the method is typically applied to a loosely-coupled coordination graph, the induced width is

much smaller than the size of the full joint action space. Approximate efficient alternatives exist, such as max-plus [Vlassis et al., 2004], but using them will invalidate the proof for the Bayesian regret bound (Theorem 1).

Finally, the joint arm that maximizes Equation 4.11, $\boldsymbol{a}_t$, is pulled and a reward $R_t^e(\boldsymbol{a}_t^e)$ will be obtained for each group. MATS is formally described in Algorithm 3.[1]

Algorithm 3: Multi-Agent Thompson Sampling (MATS)

Input: Prior $Q^e_{\boldsymbol{a}^e}$ per group $\mathcal{D}^e$ and local action $\boldsymbol{a}^e$

$\mathcal{H}_0 \leftarrow \{\}$
for $t \in [1..T]$ **do**
  Sample means from prior for each local arm.
  $\forall e \in [1..\rho], \boldsymbol{a}^e \in \mathcal{A}^e$:
    $\mu_t^e(\boldsymbol{a}^e) \sim Q^e_{\boldsymbol{a}^e}(\ \cdot \mid \mathcal{H}_{t-1})$

  Compute best joint arm using local mean samples.
  $\boldsymbol{a}_t \leftarrow \arg\max_{\boldsymbol{a}} \sum_{e=1}^{\rho} \mu_t^e(\boldsymbol{a}^e)$ using variable elimination

  Pull best joint arm, receive local rewards and update history of observations.
  $\langle R_t^e(\boldsymbol{a}_t^e)\rangle_{e=1}^{\rho} \leftarrow$ Pull joint arm $\boldsymbol{a}_t$
  $\mathcal{H}_t \leftarrow \mathcal{H}_{t-1} \cup \{\langle \boldsymbol{a}_t^e, R_t^e(\boldsymbol{a}_t^e)\rangle\}_{e=1}^{\rho}$
end

MATS belongs to the class of probability matching methods (see Definition 3). Intuitively, this means that MATS samples the local mean rewards according to the beliefs of the user at each time step, and maximizes over those means to find the optimal joint arm according to Definition 5. This process is conceptually similar to traditional Thompson sampling (see Section 2.2 in Chapter 2).

4 Bayesian Regret Analysis

Many multi-agent systems are composed of locally connected agents. When formalized as a MAMAB (Definition 5), our method is able to exploit these local structures during the decision process. We provide a Bayesian regret bound for MATS that scales sublinearly with a factor $\widetilde{A}T$, where $\widetilde{A}$ is the number of local arms.

[1]Source code for MATS is publicly available in the AI-toolbox by Bargiacchi et al. [2020], which is implemented in C++ with Python bindings.

Consider a MAMAB $\langle \mathcal{D}, \mathcal{A}, R \rangle$ with ρ groups and the following assumption on the rewards:

Assumption 1

The global rewards have a mean between 0 and 1, i.e.,

$$\mu(\boldsymbol{a}) \in [0, 1], \forall \boldsymbol{a} \in \mathcal{A}.$$

Assumption 2

The local rewards shifted by their mean are σ-subgaussian distributed, i.e., $\forall e \in [1..\rho], \boldsymbol{a}^e \in \mathcal{A}^e$,

$$\mathbb{E}\left[\exp\left(t(R^e(\boldsymbol{a}^e) - \mu^e(\boldsymbol{a}^e))\right)\right] \leq \exp(0.5\sigma^2 t^2).$$

A σ-subgaussian distribution is a distribution of which the tails are bounded in the limit by a Gaussian with standard deviation σ. For example, any distribution with finite support (e.g., the uniform and Bernoulli distributions) are σ-subgaussian.

We maintain the pull counters $n^e_{t-1}(\boldsymbol{a}^e)$ and estimated means $\hat{\mu}^e_{t-1}(\boldsymbol{a}^e)$ for local arms $\boldsymbol{a}^e$. Note that, while $\mu^e_t(\cdot)$ is a random variable distributed according to the posterior at time t, reflecting the unknown mean parameter, $\hat{\mu}^e_{t-1}(\cdot)$ is the sample mean at time t, based on $t-1$ observed rewards.

Consider the event $\mathcal{E}_T$, which states that, until time step T, the differences between the local sample means and true means are bounded by a time-dependent threshold, i.e.,

$$\mathcal{E}_T \triangleq \left(\forall e, \boldsymbol{a}^e, t \ : \ |\hat{\mu}^e_{t-1}(\boldsymbol{a}^e) - \mu^e(\boldsymbol{a}^e)| \leq c^e_t(\boldsymbol{a}^e)\right) \tag{4.12}$$

with

$$c^e_t(\boldsymbol{a}^e) \triangleq \sqrt{\frac{2\sigma^2 \log(\delta^{-1})}{n^e_{t-1}(\boldsymbol{a}^e)}}. \tag{4.13}$$

where δ is a free parameter that will be chosen later. We denote the complement of the event by $\overline{\mathcal{E}}_T$.

Proof outline. The event $\mathcal{E}_T$ states that the sample means for all local arms and all time steps are sufficiently close to the true unknown means, i.e., they appear within a specified

parametrized interval. This allows us to decompose the regret bound into two terms, i.e., one for which the event is true and one for which it is false. If the event is true, it means that the errors on the sample means are bounded, and thus the cumulative regret can also be bounded in terms of these errors. Specifically, we can show that the local regret for each possible local arm is bounded by twice the specified interval. If the event is false, it means that the instantaneous regret cannot be reduced over time and will, according to Assumption 1, be equal to 1 in the worst case. Therefore, we show that the probability of this event being false reduces significantly over time. Specifically, we can use Hoeffding's inequality for subgaussian reward distributions, which states that the probability of a sample mean and the true mean not appearing in the specified interval is exponentially bounded in terms of this interval and the number of samples used to determine the estimated mean. Finally, by combining both cases and choosing the parameters of the aforementioned interval carefully, we obtain a regret bound that is sub-linear with respect to time and the size of the local joint action space.

We now provide the necessary lemmas and detailed proofs to establish an upper bound on the cumulative regret.

Lemma 1

(Bayesian regret bound under $\mathcal{E}_T$) Provided that the error bound on the local sample means is never exceeded until time T, the Bayesian regret bound, when using the MATS policy π, is of the order

$$\mathbb{E}\left[R(T,\pi) \mid \mathcal{E}_T\right] \leq \sqrt{32\sigma^2 \widetilde{A}\rho T \log(\delta^{-1})}. \tag{4.14}$$

Proof. Consider this upper bound on the sample means:

$$u_t(\boldsymbol{a}) \triangleq \sum_{e=1}^{\rho} \hat{\mu}_{t-1}^e(\boldsymbol{a}^e) + c_t^e(\boldsymbol{a}^e). \tag{4.15}$$

Given history $\mathcal{H}_{t-1}$, the statistics $\hat{\mu}_{t-1}^e(\boldsymbol{a}^e)$ and $n_{t-1}^e(\boldsymbol{a}^e)$ are known, rendering $u_t(\cdot)$ a deterministic function. Therefore, the probability matching property of MATS (Equation 2.9) can be applied as follows:

$$\mathbb{E}\left[u_t(\boldsymbol{a}_t) \mid \mathcal{H}_{t-1}\right] = \mathbb{E}\left[u_t(\boldsymbol{a}_*) \mid \mathcal{H}_{t-1}\right]. \tag{4.16}$$

Hence, using the tower-rule, i.e., for two random variables X and Y:

$$\mathbb{E}\left[X\right] = \mathbb{E}\left[\mathbb{E}\left[X \mid Y\right]\right], \tag{4.17}$$

the regret can be bounded as

$$
\begin{aligned}
\mathbb{E}\left[\sum_{t=1}^{T}\Delta(\boldsymbol{a}_t) \mid \mathcal{E}_T\right] &\overset{(4.17)}{=} \mathbb{E}\left[\sum_{t=1}^{T}\mathbb{E}\left[\mu(\boldsymbol{a}_*)-\mu(\boldsymbol{a}_t) \mid \mathcal{H}_{t-1},\mathcal{E}_T\right] \,\middle|\, \mathcal{E}_T\right] \\
&= \mathbb{E}\left[\sum_{t=1}^{T}\mathbb{E}\left[\mu(\boldsymbol{a}_*)-u_t(\boldsymbol{a}_t) \mid \mathcal{H}_{t-1},\mathcal{E}_T\right]\right. \\
&\qquad \left. + \sum_{t=1}^{T}\mathbb{E}\left[u_t(\boldsymbol{a}_t)-\mu(\boldsymbol{a}_t) \mid \mathcal{H}_{t-1},\mathcal{E}_T\right] \,\middle|\, \mathcal{E}_T\right] \\
&\overset{(4.16)}{=} \mathbb{E}\left[\sum_{t=1}^{T}\mathbb{E}\left[\mu(\boldsymbol{a}_*)-u_t(\boldsymbol{a}_*) \mid \mathcal{H}_{t-1},\mathcal{E}_T\right]\right. \\
&\qquad \left. + \sum_{t=1}^{T}\mathbb{E}\left[u_t(\boldsymbol{a}_t)-\mu(\boldsymbol{a}_t) \mid \mathcal{H}_{t-1},\mathcal{E}_T\right] \,\middle|\, \mathcal{E}_T\right].
\end{aligned}
\tag{4.18}
$$

Note that the expression $\mu(\boldsymbol{a}_*) - u_t(\boldsymbol{a}_*)$ is always negative under $\mathcal{E}_T$, i.e.,

$$
\begin{aligned}
\mu(\boldsymbol{a}_*) - u_t(\boldsymbol{a}_*) &\overset{(4.15)}{=} \sum_{e=1}^{\rho}\mu^e(\boldsymbol{a}_*^e) - \hat{\mu}_{t-1}^e(\boldsymbol{a}_*^e) - c_t^e(\boldsymbol{a}_*^e) \\
&\overset{(4.12)}{\leq} \sum_{e=1}^{\rho} c_t^e(\boldsymbol{a}_*^e) - c_t^e(\boldsymbol{a}_*^e) = 0,
\end{aligned}
\tag{4.19}
$$

while $u_t(\boldsymbol{a}_t) - \mu(\boldsymbol{a}_t)$ is bounded by twice the sum of the thresholds $c_t^e(\boldsymbol{a}^e)$, i.e.,

$$
\begin{aligned}
u_t(\boldsymbol{a}_t) - \mu(\boldsymbol{a}_t) &\overset{(4.15)}{=} \sum_{e=1}^{\rho}\hat{\mu}_{t-1}^e(\boldsymbol{a}_t^e) + c_t^e(\boldsymbol{a}_t^e) - \mu^e(\boldsymbol{a}_t^e) \\
&\overset{(4.12)}{\leq} \sum_{e=1}^{\rho} c_t^e(\boldsymbol{a}_t^e) + c_t^e(\boldsymbol{a}_t^e) = 2\sum_{e=1}^{\rho} c_t^e(\boldsymbol{a}_t^e).
\end{aligned}
\tag{4.20}
$$

Thus, Equation 4.18 can be bounded as

$$\begin{aligned}\mathbb{E}\left[\sum_{t=1}^{T}\Delta(\boldsymbol{a}_t) \mid \mathcal{E}_T\right] &\leq \mathbb{E}\left[2\sum_{t=1}^{T}\sum_{e=1}^{\rho} c_t^e(\boldsymbol{a}_t^e)\right] \\ &= \mathbb{E}\left[2\sum_{t=1}^{T}\sum_{e=1}^{\rho}\sqrt{\frac{2\sigma^2\log(\delta^{-1})}{n_{t-1}^e(\boldsymbol{a}_t^e)}}\right] \\ &= \mathbb{E}\left[2\sum_{e=1}^{\rho}\sum_{\boldsymbol{a}^e\in\mathcal{A}^e}\sum_{t=1}^{T}\mathcal{I}\{\boldsymbol{a}_t^e=\boldsymbol{a}^e\}\sqrt{\frac{2\sigma^2\log(\delta^{-1})}{n_{t-1}^e(\boldsymbol{a}^e)}}\right],\end{aligned} \tag{4.21}$$

where $\mathcal{I}\{\cdot\}$ is the indicator function. The terms in the summation are only non-zero at the time steps when the local action $\boldsymbol{a}^e$ is pulled, i.e., when $\mathcal{I}\{\boldsymbol{a}_t^e=\boldsymbol{a}^e\}=1$. Additionally, note that only at these time steps, the counter $n_t^e(\boldsymbol{a}^e)$ increases by exactly 1. Therefore, the following equality holds:

$$\sum_{t=1}^{T}\mathcal{I}\{\boldsymbol{a}_t^e=\boldsymbol{a}^e\}\sqrt{(n_{t-1}^e(\boldsymbol{a}^e))^{-1}} = \sum_{k=1}^{n_T^e(\boldsymbol{a}^e)}\sqrt{k^{-1}}. \tag{4.22}$$

The function $\sqrt{k^{-1}}$ is decreasing and integrable. Hence, using the right Riemann sum,

$$\sqrt{k^{-1}} \leq \int_{k-1}^{k}\sqrt{x^{-1}}dx. \tag{4.23}$$

Combining Equations 4.21–4.23 leads to a bound

$$\begin{aligned}\mathbb{E}\left[\sum_{t=1}^{T}\Delta(\boldsymbol{a}_t)\,\middle|\,\mathcal{E}_T\right] &\overset{(4.21)}{=} \mathbb{E}\left[2\sum_{e=1}^{\rho}\sum_{\boldsymbol{a}^e\in\mathcal{A}^e}\sum_{t=1}^{T}\mathcal{I}\{\boldsymbol{a}_t^e=\boldsymbol{a}^e\}\sqrt{\frac{2\sigma^2\log(\delta^{-1})}{n_{t-1}^e(\boldsymbol{a}^e)}}\right] \\ &\overset{(4.22)}{=} \mathbb{E}\left[\sqrt{8\sigma^2\log(\delta^{-1})}\sum_{e=1}^{\rho}\sum_{\boldsymbol{a}^e\in\mathcal{A}^e}\sum_{k=1}^{n_T^e(\boldsymbol{a}^e)}\sqrt{k^{-1}}\right] \\ &\overset{(4.23)}{\leq} \mathbb{E}\left[\sqrt{8\sigma^2\log(\delta^{-1})}\sum_{e=1}^{\rho}\sum_{\boldsymbol{a}^e\in\mathcal{A}^e}\int_0^{n_T^e(\boldsymbol{a}^e)}\sqrt{x^{-1}}dx\right] \\ &= \mathbb{E}\left[\sqrt{8\sigma^2\log(\delta^{-1})}\sum_{e=1}^{\rho}\sum_{\boldsymbol{a}^e\in\mathcal{A}^e}\sqrt{4n_T^e(\boldsymbol{a}^e)}\right].\end{aligned} \tag{4.24}$$

We use the relationship $||\mathbf{x}||_1 \leq \sqrt{n}||\mathbf{x}||_2$ between the 1- and 2-norm of a vector $\mathbf{x}$, where n is the number of elements in the vector, as follows:

$$\sum_{e=1}^{\rho} \sum_{\boldsymbol{a}^e \in \mathcal{A}^e} \left| \sqrt{n_T^e(\mathbf{a}^e)} \right| \leq \sqrt{\widetilde{A}} \sqrt{\sum_{e=1}^{\rho} \sum_{\boldsymbol{a}^e \in \mathcal{A}^e} \left(\sqrt{n_T^e(\mathbf{a}^e)} \right)^2}. \tag{4.25}$$

Finally, note that the sum of all counts $n_T^e(\mathbf{a}^e)$ is equal to the total number of local pulls done by MATS until time T, i.e.,

$$\sum_{e=1}^{\rho} \sum_{\boldsymbol{a}^e \in \mathcal{A}^e} n_T^e(\boldsymbol{a}^e) = \rho T. \tag{4.26}$$

Using the Equations 4.24–4.26, the complete regret bound under $\mathcal{E}_T$ is given by

$$\begin{aligned} \mathbb{E}\left[\sum_{t=1}^{T} \Delta(\boldsymbol{a}_t) \mid \mathcal{E}_T\right] &\overset{(4.24)}{\leq} \mathbb{E}\left[\sqrt{8\sigma^2 \log(\delta^{-1})} \sum_{e=1}^{\rho} \sum_{\boldsymbol{a}^e \in \mathcal{A}^e} \sqrt{4 n_T^e(\boldsymbol{a}^e)}\right] \\ &\overset{(4.25)}{\leq} \mathbb{E}\left[\sqrt{32\sigma^2 \log(\delta^{-1})} \sqrt{\widetilde{A}} \sqrt{\sum_{e=1}^{\rho} \sum_{\boldsymbol{a}^e \in \mathcal{A}^e} \left(\sqrt{n_T^e(\mathbf{a}^e)} \right)^2}\right] \\ &\overset{(4.26)}{=} \sqrt{32\sigma^2 \log(\delta^{-1})} \sqrt{\widetilde{A}} \sqrt{\rho T}. \end{aligned} \tag{4.27}$$

□

Lemma 2

(Concentration inequality) The probability of exceeding the error bound on the local sample means is linearly bounded by $\widetilde{A} T \delta$. Specifically,

$$p\left(\overline{\mathcal{E}}_T\right) \leq 2\widetilde{A} T \delta. \tag{4.28}$$

Proof. Using the union bound, i.e., for any finite or countably infinite set of events X_i:

$$p\left(\cup_i X_i\right) \leq \sum_i p\left(X_i\right), \tag{4.29}$$

we can bound the probability of observing event $\overline{\mathcal{E}}_T$ as

$$\begin{aligned} p\left(\overline{\mathcal{E}}_T\right) &\overset{(4.12)}{=} p\left(\exists t, e, \boldsymbol{a}^e \ : \ |\hat{\mu}_{t-1}^e(\boldsymbol{a}^e) - \mu^e(\boldsymbol{a}^e)| > c_t^e(\boldsymbol{a}^e)\right) \\ &\overset{(4.29)}{\leq} \sum_{t=1}^{T} \sum_{e=1}^{\rho} \sum_{\boldsymbol{a}^e \in \mathcal{A}^e} p\left(\left|\hat{\mu}_{t-1}^e(\boldsymbol{a}^e) - \mu^e(\boldsymbol{a}^e)\right| > c_t^e(\boldsymbol{a}^e)\right). \end{aligned} \tag{4.30}$$

The estimated mean $\hat{\mu}^e_{t-1}(\boldsymbol{a}^e)$ is a weighted sum of $n^e_{t-1}(\boldsymbol{a}^e)$ random variables distributed according to a σ-subgaussian with mean $\mu^e(\boldsymbol{a}^e)$. Hence, Hoeffding's inequality (H) can be used [Vershynin, 2018].

$$\begin{aligned} p\left(\left|\hat{\mu}^e_{t-1}(\boldsymbol{a}^e) - \mu^e(\boldsymbol{a}^e)\right| > c^e_t(\boldsymbol{a}^e)\right) &\overset{\text{(H)}}{\leq} 2\exp\left(-\frac{n^e_{t-1}(\boldsymbol{a}^e)}{2\sigma^2}(c^e_t(\boldsymbol{a}^e))^2\right) \\ &\overset{(4.13)}{=} 2\exp\left(-\frac{n^e_{t-1}(\boldsymbol{a}^e)}{2\sigma^2}\frac{2\sigma^2\log(\delta^{-1})}{n^e_{t-1}(\boldsymbol{a}^e)}\right) \\ &= 2\exp\left(-\log(\delta^{-1})\right). \\ &= 2\delta \end{aligned} \tag{4.31}$$

Therefore, the following concentration inequality on $\overline{\mathcal{E}}_T$ holds:

$$p\left(\overline{\mathcal{E}}_T\right) \leq \sum_{t=1}^{T}\sum_{e=1}^{\rho}\sum_{\boldsymbol{a}^e\in\mathcal{A}^e} 2\delta = 2\widetilde{A}T\delta. \tag{4.32}$$

□

Theorem 1

Let $\langle\mathcal{D},\mathcal{A},R\rangle$ be a MAMAB. If Assumptions 1 and 2 hold, then the MATS policy π satisfies a Bayesian regret bound of

$$\begin{aligned} \mathbb{E}\left[R(T,\pi)\right] &\leq \sqrt{64\sigma^2\widetilde{A}\rho T\log(\widetilde{A}T)} + \frac{2}{\widetilde{A}} \\ &\in O\left(\sqrt{\sigma^2\widetilde{A}\rho T\log(\widetilde{A}T)}\right). \end{aligned} \tag{4.33}$$

Proof. Using the law of excluded middle (M) and the fact that $\Delta(\boldsymbol{a}_t)$ and $p\left(\mathcal{E}_T \mid \mathcal{H}_{t-1}\right)$ are between 0 and 1 (B), the regret can be decomposed as

$$\begin{aligned} \mathbb{E}\left[\sum_{t=1}^{T}\Delta(\boldsymbol{a}_t)\right] &\overset{\text{(M)}}{=} \mathbb{E}\left[\sum_{t=1}^{T}\Delta(\boldsymbol{a}_t) \mid \mathcal{E}_T\right]p\left(\mathcal{E}_T\right) + \mathbb{E}\left[\sum_{t=1}^{T}\Delta(\boldsymbol{a}_t) \mid \overline{\mathcal{E}}_T\right]p\left(\overline{\mathcal{E}}_T\right) \\ &\overset{\text{(B)}}{\leq} \mathbb{E}\left[\sum_{t=1}^{T}\Delta(\boldsymbol{a}_t) \mid \mathcal{E}_T\right] + Tp\left(\overline{\mathcal{E}}_T\right). \end{aligned} \tag{4.34}$$

Then, according to Lemmas 1 and 2 (L), we have

$$\begin{aligned}\mathbb{E}\left[\sum_{t=1}^{T}\Delta(\boldsymbol{a}_t)\right] &\overset{(4.34)}{\leq} \mathbb{E}\left[\sum_{t=1}^{T}\Delta(\boldsymbol{a}_t) \mid \mathcal{E}_T\right] + Tp\left(\overline{\mathcal{E}}_T\right) \\ &\overset{(\mathsf{L})}{\leq} \sqrt{32\sigma^2\widetilde{A}\rho T\log(\delta^{-1})} + 2\widetilde{A}T^2\delta.\end{aligned} \tag{4.35}$$

Finally, choosing $\delta = (\widetilde{A}T)^{-2}$, we conclude that

$$\begin{aligned}\mathbb{E}\left[R(T,\pi)\right] &\overset{(4.35)}{\leq} \sqrt{32\sigma^2\widetilde{A}\rho T\log(\delta^{-1})} + 2\widetilde{A}T^2\delta \\ &\leq \sqrt{64\sigma^2\widetilde{A}\rho T\log\left(\widetilde{A}T\right)} + \frac{2}{\widetilde{A}} \\ &\in O\left(\sqrt{\sigma^2\widetilde{A}\rho T\log(\widetilde{A}T)}\right).\end{aligned} \tag{4.36}$$

□

Corollary 1

If $|\mathcal{A}_i| \leq k$ for all agents i, and if $|\mathcal{D}^e| \leq d$ for all groups $\mathcal{D}^e$, then

$$\mathbb{E}\left[R(T,\pi)\right] \in O\left(\rho\sqrt{\sigma^2 k^d T\log(\rho k^d T)}\right). \tag{4.37}$$

Proof. $\widetilde{A} = \sum_{e=1}^{\rho}|\mathcal{A}^e| = \sum_{e=1}^{\rho}\prod_{i\in\mathcal{D}^e}|\mathcal{A}_i| \leq \rho k^d$. □

Corollary 1 tells us that the regret is sub-linear in terms of time T and low-order polynomial in terms of the largest action space of a single agent when the number of groups and agents per group are small. This reflects the main contribution of this work. When agents are loosely coupled, MATS provides a mechanism that only considers small parts of the joint arm space at a time, rather than the full joint arm space directly. This is a significant improvement over the established classic regret bounds of vanilla Thompson sampling when the multi-agent multi-armed bandit is flattened and the factored structure is neglected [Russo and Van Roy, 2014; Lattimore and Szepesvári, 2020]. The classic bounds scale exponentially with the number of agents, which renders the use of vanilla Thompson sampling unfeasible in many multi-agent environments.

5 Experiments

We evaluate the performance of MATS on the benchmark problems proposed in the paper that introduced MAUCE [Bargiacchi et al., 2018], which is a state-of-the-art algorithm for multi-agent bandit problems, and one novel setting that falls outside the domain of the theoretical guarantees for both MAUCE and MATS.

Additionally, we execute MATS on a state-of-the-art wind farm simulator to address the practical benefits of MATS in wind farm control setting. Specifically, we use MATS to optimize joint rotor alignment of a group of wind turbines, such that the total power production maximized. This is a non-trivial problem, as upstream operating turbines affect the power output of downstream turbines, creating non-linear dependencies between the agents' actions. As we assume that the variance on the rewards is unknown, the established regret bounds for MAUCE and MATS do not apply.[2]

5.1 Synthetic Benchmarks

First, we evaluate the performance of MATS on two benchmarks that were introduced in the MAUCE paper, i.e., Bernoulli 0101-Chain and Gem Mining. We compare against a random policy (rnd), Sparse Cooperative Q-Learning (SCQL) [Kok and Vlassis, 2004], Learning with Linear Rewards (LLR) [Gai et al., 2012] and the state-of-the-art algorithm, Multi-Agent Upper Confidence Exploration (MAUCE) [Bargiacchi et al., 2018]. For SCQL and MAUCE, we use the same parameters as described by Bargiacchi et al. [2018]. For SCQL, we use an ϵ-greedy policy, which picks the current best action with a probability of $(1-\epsilon)$ and a random action with a probability of ϵ. We let ϵ decrease linearly over time, i.e., $\epsilon = 0.05 - 10^{-5}t$. For MAUCE, the exploration parameter reflects the maximum possible reward that can be returned. As both the Bernoulli 0101-Chain and Gem Mining experiments consider Bernoulli-distributed rewards, we set the exploration parameter to 1. For MATS, we always use non-informative Jeffreys priors to model the unknown means of the reward distributions, which are invariant toward reparametrization of the experimental settings [Robert, 2007]. For Bernoulli-distributed rewards, the Jeffreys prior is the Beta-distribution $\mathcal{B}\text{eta}\,(\alpha = 0.5, \beta = 0.5)$ [Lunn et al., 2012]. Although including additional prior domain knowledge could be useful in practice, we use well-known non-informative priors in our experiments to compare fairly with other state-of-the-art techniques.

Then, we introduce a novel variant of the 0101-Chain with Poisson-distributed local rewards. We expect this to be a particularly challenging task to solve [Libin et al., 2019], as a Poisson distribution is supergaussian, meaning that its tails tend slower toward zero

[2] The source code to reproduce the experiments is publicly available at: https://github.com/timo-verstraeten/mats-experiments

than the tails of any Gaussian. Therefore, both the assumptions made in Theorem 1 and in the established regret bound of MAUCE are violated. Additionally, as the rewards are highly skewed, we expect that the use of symmetric exploration bounds in MAUCE will often lead to either over- or underexploration of the local arms.

We assess the performance of LLR, SCQL, MAUCE and MATS on this benchmark. For SCQL, we use the same parametrization for the ϵ-greedy policy as used in the Bernoulli experiments. For MAUCE, an exploration parameter must be chosen, which denotes the range of the observed rewards. As a Poisson distribution has unbounded support, there is no maximum reward, and it is challenging to choose the exploration parameter. Therefore, we rely on a statistical approach, in which we consider the percentiles of the Poisson distribution. Specifically, as about 95$\%$ of the rewards when pulling the optimal arm falls below a value of 1, we choose 1 as the exploration parameter of MAUCE. For MATS we use non-informative Jeffreys priors on the unknown means, which for the Poisson likelihood is a Gamma prior, $\mathcal{G}$amma$(\alpha = 0.5, \beta = 0)$ [Lunn et al., 2012].

Bernoulli 0101-Chain

The Bernoulli 0101-Chain consists of n agents and $n-1$ local reward distributions. Each agent can choose between two actions: 0 and 1. In the coordination graph, agents i and $i+1$ are connected to a local reward $R^i(a_i, a_{i+1})$. Thus, each pair of agents should locally coordinate in order to find the best joint arm. The local rewards are drawn from a Bernoulli distribution with a different success probability per group. These success probabilities are given in Table 4.1. The optimal joint action is an alternating sequence of zeros and ones, starting with 0. In this work, we set the number of agents n to 10.

$R^i \sim \mathcal{B}$**er** $(\cdot)$	$a_{i+1} = 0$	$a_{i+1} = 1$
$a_i = 0$	0.75	1
$a_i = 1$	0.25	0.9

Table 4.1: Bernouilli 0101-Chain – The unscaled local reward distributions of agents i and $i+1$, where i is even. Each entry shows the success probability for each local arm of agents i and $i+1$, where i is even. The table is transposed for the case where i is odd.

To ensure that the assumptions made in the regret analyses of MAUCE and MATS hold, we divide the local rewards by the number of groups, such that the global rewards are between 0 and 1.

The results for the Bernoulli 0101-chains are shown in Figure 4.5a. These results demonstrate that MATS solves the problem in only a few time steps, while MAUCE still pulls many sub-optimal actions after 10000 time steps.

Gem Mining

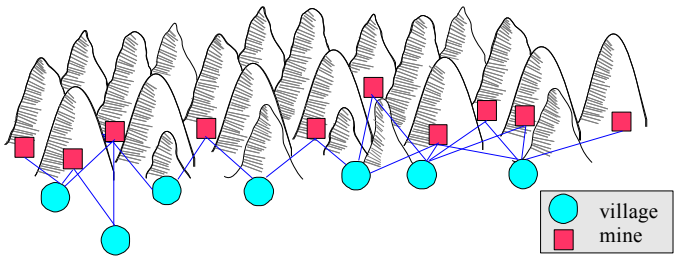


Figure 4.4: Example of a coordination graph in the Gem Mining problem. The red nodes are the mines (rewards), while the blue nodes are the villages (agents) [Roijers, 2016].

In the Gem Mining problem, a mining company wants to excavate a set of mines for gems (i.e., local rewards). The goal is to maximize the total number of gems found over all mines. However, the company's workers live in separate villages (i.e., agents), and only one van per village is available. Therefore, each village needs to decide to which mine it should send its workers (i.e., local action). Moreover, workers can only commute to nearby mines, which is described by the coordination graph. Hence, a group can be constructed per mine, consisting of all agents that can travel toward the mine. An example of a coordination graph is given in Figure 4.4.

To generate an instance of the problem, we randomly sample a number of villages n_v in $[5..15]$. Each village is randomly populated with 1 to 5 workers. The number of mines is equal to $n_v + 3$. To construct the coordination graph, we connect a village to the mines with indices i to $(i + m_i - 1)$, where m_i is a random number in $[2..4]$. The last village is always connected to 4 mines. This method ensures that there are no disconnected subgraphs in the coordination graph.

The reward is drawn from a Bernoulli distribution, where the probability of finding a gem at a mine is $1.03^{W-1}p_B$ with W the total number of workers assigned to the mine and p_B a randomly chosen base probability in $[0, 0.5]$ for each mine. When more workers are excavating a mine, the probability of finding a gem increases.

The results for the Gem Mining problem are shown in Figure 4.5b. We can observe that the cumulative regret of MAUCE is three times as high as the cumulative regret of MATS around 40000 time steps.

Poisson 0101-Chain

We introduce a novel benchmark with Poisson distributed local rewards. As a Poisson distribution is skewed, we expect this to be a challenging task to solve. Moreover, a Poisson distribution is supergaussian, which means that the established regret bounds of both MATS and MAUCE do not hold. Similar to the Bernoulli 0101-Chain, agents need to coordinate their actions in order to obtain an alternating sequence of zeroes and ones. However, as the rewards are highly skewed and supergaussian, this setting is much more challenging. The means of the Poisson distributions are given in Table 4.2. We also divide the rewards by the number of groups, similar to the Bernoulli 0101-Chain. Again, we set the number of agents n to 10.

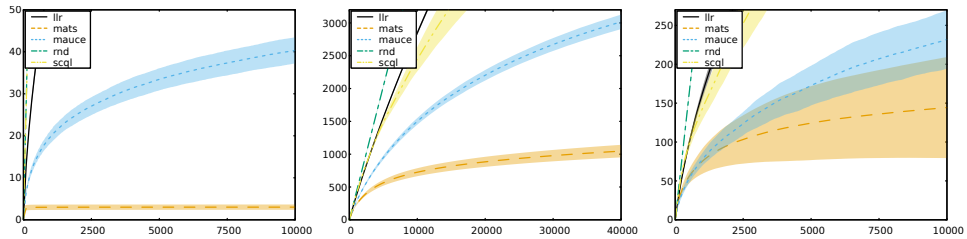
$R^i \sim \mathcal{P}(\cdot)$	$a_{i+1} = 0$	$a_{i+1} = 1$
$a_i = 0$	0.1	0.3
$a_i = 1$	0.2	0.1

Table 4.2: Poisson 0101-Chain – The unscaled local reward distributions of agents i and $i+1$. Each entry shows the mean for each local arm of agents i and $i+1$.

Figure 4.5c shows that the cumulative regret of MATS stagnates around 7500 time steps, while the cumulative regret of MAUCE continues to increase significantly. Moreover, for our chosen parametrization of the Poisson distributions, the mean falls well above 50% of all samples. Therefore, it is expected that for the initially observed rewards, the true mean will be higher than the sample mean. Naturally, this bias averages out in the limit, but may have a large impact during the early exploration stage. The high standard deviations in Figure 4.5c support this impact.

5.2 Wind Farm Control Application

We demonstrate the benefits of MATS on a state-of-the-art wind farm simulator and compare its performance to MAUCE, SCQL and LLR. A wind farm consists of a group of wind turbines, instantiated to extract energy from wind. From the perspective of a single turbine, aligning with the incoming wind vector usually ensures the highest productivity. However, translating this control policy directly toward an entire wind farm may be sub-optimal. As wind passes through the farm, downstream turbines observe a significantly



(a) Bernoulli 0101-Chain (b) Gem Mining (c) Poisson 0101-Chain

Figure 4.5: Cumulative normalized regret averaged over 100 runs for the (a) Bernoulli 0101-Chain, (b) Gem Mining and (c) Poisson 0101-Chain. Both the mean (line) and standard deviation (shaded area) are plotted.

lower wind speed. This is known as the *wake effect*, which is due to the turbulence generated behind operational turbines.

In recent work, the possibility of deflecting wake away from the farm through rotor misalignment is investigated [van Dijk et al., 2016]. While a misaligned turbine produces less energy on its own, the group's total productivity is increased. Physically, the wake effect reduces over long distances, and thus, turbines tend to only influence their neighbors. We can use this domain knowledge to define groups of agents and organize them in a graph structure. Note that the graph structure depends on the incoming wind vector. Nevertheless, atmospheric conditions are typically discretized when analyzing operational regimes [International Electrotechnical Commission, 2012], thus, a graph structure can be made for each possible incoming discretized wind vector independently. Here, we construct a graph structure for one possible wind vector.

We demonstrate our method on a virtual wind farm, consisting of 11 turbines, of which the layout is shown in Figure 4.6. We use the state-of-the-art 'FLOw Redirection and Induction in Steady-state' (FLORIS) wake simulator [National Renewable Energy Laboratory (NREL), 2019] and the 5 MW reference turbine description from the National Renewable Energy Laboratory to model the individual wind turbines [Jonkman et al., 2009]. Each turbine is an agent, and choosing an orientation with respect to the incoming wind vector corresponds to an action. The groups are constructed according to the graph depicted in Figure 4.6. The reward is described by the power production per agent, which we divide uniformly over the groups that the agent is part of. This entails that the reward received by an agent contributes equally to every group it is part of. The objective is to find the joint alignment of the wind farm that maximizes the total power production.

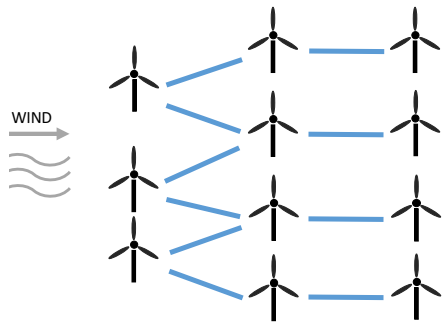


Figure 4.6: Wind farm layout – Dependency graph where the nodes are the turbines and the edges describe the dependencies between the turbines. The incoming wind is denoted by an arrow.

For SCQL, we use an ϵ-greedy policy and let ϵ decrease linearly over time, i.e., $\epsilon = 0.05 - 10^{-5}t$. For MAUCE, we use the same exploration parameter used by [Bargiacchi et al., 2018], which was calibrated to be 0.05. For MATS, we assume the local power productions are sampled from Gaussians with unknown mean and variance, which leads to a Student's t-distribution on the mean when using a Jeffreys prior [Honda and Takemura, 2014]. The results for the wind farm control setting are shown in Figure 4.7. We can see that MATS allowed for a five-fold increase of the normalized power productions with respect to MAUCE.

6 Discussion

We proposed multi-agent Thompson sampling (MATS), a novel Bayesian algorithm for multi-agent multi-armed bandits. The method exploits loose connections between agents to solve multi-agent coordination tasks efficiently. Theoretically, both MATS and MAUCE achieve sub-linear regret in terms of time and low-order polynomial regret in terms of the number of local arms for sparse coordination graphs. Establishing upper bounds on regret is important, as it provides guarantees about the learning performance for worst-case scenarios. Empirically, we showed a significant improvement over the state-of-the-art algorithms, MAUCE and SCQL, on several synthetic benchmarks.

Moreover, MATS is a Bayesian method that can seamlessly include domain knowledge about the shape of the reward distributions and treat the problem parameters as unknowns. To highlight the power of this property, we introduced the Poisson 0101-chain and provide

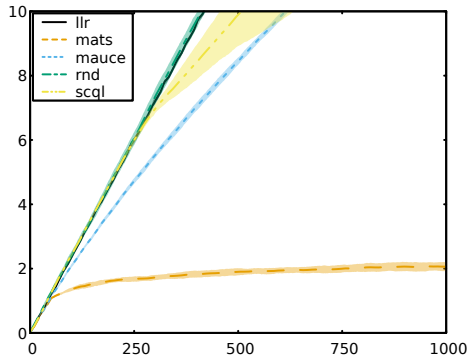


Figure 4.7: Cumulative normalized regret averaged over 10 runs for Wind Farm task. Both the mean (line) and standard deviation (shaded area) are plotted.

the Poisson likelihood function to MATS. In this setting, the reward distributions are highly skewed and supergaussian (i.e., the tails of the distribution are lower-bounded by a Gaussian in the limit). Although the established regret bounds of MATS and MAUCE do not apply to supergaussian reward distributions, we demonstrate that MATS exploits density information of the rewards to achieve more targeted exploration. As MAUCE only supports symmetric exploration bounds, it is challenging to correctly assess the amount of exploration needed to solve the task.

Throughout the experiments, exploration constants had to be specified for MAUCE, which were challenging to interpret and choose. In contrast, MATS uses either statistics about the data and, potentially non-informative, beliefs asserted by the user. For example, in the wind farm case, the spread of the measurements is unknown. MATS effectively maintains a posterior on the variance and uses it to balance exploration and exploitation, while still outperforming MAUCE with a manually optimized exploration range.

Currently, we established an upper bound on the cumulative regret to show that the MATS policy converges to a strategy that consistently plays the optimal joint arm (i.e., the cumulative regret reduces sub-linearly over time) and that the MATS policy effectively exploits the sparse structure of the joint action space (i.e., the regret bound is in terms of the number of local joint actions instead of the global joint actions). In future work, we aim to construct a lower bound for MATS, which will allow us to assess the tightness of the established upper bound.

We demonstrated that MATS achieves high performance on a synthetic wind farm control task, where the optimal rotor alignments of the wind turbines need to be jointly optimized to maximize the farm's power production. In many wind farm control settings, there exist

sparse neighborhood structures between turbines. Our results show that MATS is able to successfully exploit these structures, while leveraging prior knowledge about the data. Therefore, including MATS in data-driven wind farm control mechanisms will significantly improve their scalability toward contemporary wind farms.

Due to the increase in capacity of contemporary wind farms, it is important that AI-driven wind farm control methods are scalable with respect to the number of wind turbines, as well as flexible with respect to the used objective function. Therefore, in the next chapter, we further investigate the potential of factorization using the available domain knowledge in the context of wind farm control. Specifically, we start from the previous insights on device similarity (Chapter 3) and dependency graphs (Chapter 4), and propose a new wind farm control algorithm, called set-point Thompson sampling. This method exploits a factored representation of the wind farm to efficiently learn the optimal power configuration over wind turbines to match the power demand provided at farm-level, while considering load information.

5 Scalable Hybrid Optimization for Wind Farm Control

The rapid increase in the supply of renewable energy poses challenges with respect to the stability of the electrical grid. In contrast to conventional power plants (e.g., gas, hydro and oil), the power output of wind farms ultimately depends on environmental conditions. Due to the increase in capacity, the integration of wind energy in the electricity grid needs to comply with strict grid code requirements [Sourkounis and Tourou, 2013; Ahmed et al., 2020].

To ensure grid stability, wind farm controllers are developed to configure farm-wide power set-points, i.e., thresholds on the power production, in order to match the power demand [Aho et al., 2012]. This power demand is imposed by the transmission system operator, i.e., the entity responsible for balancing the energy supply and demand. The development of such controllers poses important challenges, as there exist complex non-linear dependencies between wind turbines. These dependencies originate from the wake effect [González-Longatt et al., 2012] in which upstream wind turbines reduce the available wind energy for downstream wind turbines. Additionally, when wind turbines are performing torque control, which regulates the power production by adapting the rotor speed, a higher power production typically results in increased loads on the mechanical components, which leads to a higher lifetime consumption [International Electrotechnical Commission, 2012;

Verstraeten et al., 2019]. Therefore, a careful balance between power and lifetime needs to be guaranteed [Boersma et al., 2017; van Binsbergen et al., 2020].

The design of wind farm controllers is typically grounded in domain knowledge about patterns in the turbines' behaviors when the wake effect is present [Boersma et al., 2017; Siniscalchi-Minna et al., 2019]. For example, one can recognize that upstream turbines, with respect to the dominant wind direction, typically observe higher fatigue loads than downstream turbines [Jensen et al., 2016]. Therefore, lower set-points should be chosen for upstream turbines to reduce damage accumulation through fatigue loads, in case the available power over the farm is larger than the desired power. While such heuristics simplify the computation of the optimal set-point allocation, they fail to capture the full complexity of the dynamically-changing multi-dimensional load spectrum (e.g., loads induced during storms). In order to develop advanced control strategies, it is necessary to consider the full load spectrum to reduce the probability of failure, which increases the reliability and sustainability of wind farms (see Section 1 of Chapter 1).

In contrast to physics-based heuristics, data-driven wind farm controllers can learn control strategies without requiring in-depth knowledge about the non-linear dependencies that exist between the turbines that make up the wind farm. An example of such a data-driven structure is proposed by van Dijk et al. [2016], in which reinforcement learning techniques are used to search for the optimal rotor orientation to deflect wake away from downstream turbines. However, state-of-the-art data-driven wind farm controllers scale poorly to larger wind farms, as the number of possible configurations grows exponentially with respect to the number of wind turbines.

Therefore, we argue that a hybrid approach, combining both flexible data-driven methods and physics-based domain knowledge, is key to guarantee both optimality and scalability of wind farm controllers. We propose a new method that learns farm-wide control strategies in simulation while leveraging knowledge about wake patterns, performance statistics and load profiles. Specifically, we formalize the farm-wide dependencies caused by wake as a dependency graph and cluster wind turbines with similar load profiles together to factorize the wind farm. This step effectively combines the previous insights gained on machine similarity (Chapter 3) and sparse topologies (Chapter 4), together with farm-wide load patterns, in the context of wind farm control. Using this factored representation, we propose a novel sampling method, called Set-Point Thompson Sampling (SPTS). This algorithm uses multi-agent Thompson sampling to evaluate promising control strategies using the factored representation of the wind farm, with the objective to match the power demand as well as possible, while minimizing stress on wind turbines with a low remaining useful life. In this work, we assume that a penalty function is defined by the wind farm operator, which punishes high-risk wind turbines with damage-inducing responses (e.g., turbines that historically have observed abnormal trends in vibration signals [Peeters et al.,

2019]). Additionally, we use a Bayesian formalism, which allows the inclusion of available knowledge about the data in the form of prior belief distributions. Specifically, as the expected power production of a single turbine is readily available in the turbine's design specifications [Lydia et al., 2014], we construct a prior distribution for every set-point, centered around the expected power production of the turbine that is associated with this set-point. This guides the learning process toward sensible power productions and allows for sufficient exploration to find the optimal set-point under wake conditions.

We start by positioning our research within related work and argue that, given the complexities inherent to dynamic loads, data-driven controller optimization in large-scale wind farms is necessary in Section 1. Next, we formalize the set-point configuration problem and the optimization objective in Section 2. Then, we describe a methodology to factorize the wind farm using physics-based knowledge in Sections 3 and 4. Afterwards, we construct SPTS, an efficient method to explore possible set-point configurations using the proposed factorization, in Section 5. We evaluate our method on wind farm settings *in silico* using an extensive set of parametrizations in Section 6. Finally, we discuss the results, highlighting the benefits and limitations of the method, in Section 7.

The methodology for analyzing the loading zones within the wind farm was published in Renewable & Sustainable Energy Reviews. The wind farm control method was accepted for publication at AAMAS 2021.

- *Verstraeten, T., Nowé, A., Keller, J., Guo, Y., Sheng, S. and Helsen, J. (2019), Fleetwide data-enabled reliability improvement of wind turbines, Renewable & Sustainable Energy Reviews, 109, 428–437*
- *Verstraeten, T., Daems P.-J., Bargiacchi, E., Roijers, D. M., Libin, P. J. K. and Helsen, J. (2021), Scalable Optimization for Wind Farm Control using Coordination Graphs, Proceedings of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS). (In press, accepted on Dec. 17th 2020)*

1 Related Work

Wind farm control strategies have mainly focused on power-load optimization (i.e., strategies that maximize power production and minimize fatigue load) or active power control (i.e., strategies that determine power set-points to meet the demand set by the electricity grid or an operator) [Boersma et al., 2017]. In both cases, the wake effect is an important factor to consider when selecting power set-points. For power-load optimization, data-driven optimization approaches typically focus on reducing the wake effect, such as wake redirection control and axial induction control [van Binsbergen et al., 2020]. Wake

redirection control is concerned with finding a joint rotor orientation of the wind turbines to redirect wake from downstream turbines [van Dijk et al., 2016; Wagenaar et al., 2012]. Axial induction control is an approach to reduce the wake effect, by lowering the power set-points of upstream turbines to reduce the energy extraction from the wind, and thus maintain a steady wind speed behind the turbines [Soleimanzadeh et al., 2012; Gebraad and van Wingerden, 2015].

In this work, we focus on active power control to match the wind farm's total power output to the power demand [Aho et al., 2012]. Many heuristic approaches based on physical knowledge about the turbines and environmental conditions exist [Aho et al., 2012; Spudic et al., 2010; Siniscalchi-Minna et al., 2019; Jensen et al., 2016]. For example, one can notice that due to the wake effect, a higher power production for upstream turbines results in lower wind speeds for downstream turbines. Therefore, the power demand can be reached using a heuristic approach, where the power contributions of downstream turbines are maximized while the power contributions of upstream turbines are minimized [Siniscalchi-Minna et al., 2019].

We argue that, similar to the power-load optimization case, data-driven optimization approaches can complement existing physics-based knowledge to improve the flexibility of active power control. Such flexibility is important, as health-aware wind farm control decisions must consider the complex multi-dimensional load profiles to improve reliability (see Section 1 of Chapter 1). Nevertheless, it remains challenging to scale data-driven optimization methods to larger wind farms, where the optimal joint configuration exists in a high-dimensional solution space.

2 Problem Statement

When the transmission system operator imposes a power demand, the wind farm controller needs to configure each wind turbine to a power set-point such that the total actual power production matches the demand as closely as possible. These set-points need to be chosen such that no high-load set-points are assigned to wind turbines with a low remaining lifetime. The lifetime of a turbine is dependent on many load factors that can lead to failure. However, the link between specific loading conditions and failure is currently not sufficiently understood [Keller et al., 2016; Junior et al., 2017]. Therefore, in this work, we assume that the wind farm operator constructs a cost-function that heavily penalizes high loads on high-risk turbines based on expert knowledge (see Section 7).

We formalize the setting as a tuple $\langle \mathcal{W}, G, \mathcal{Z}, \mathcal{A}, \langle \mathrm{P}, \mathrm{L} \rangle, \mathrm{P}_{\mathsf{dem}} \rangle$, which can be regarded as an extension of a multi-agent multi-armed bandit (see Definition 5), where

- $\mathcal{W}$ is a set of wind turbines.

- G is a directed dependency graph that describes which turbines influence each other. We refer to the dependencies of a wind turbine $w \in \mathcal{W}$ as its *parents*, and denote the set that contains turbine w and its parents as $G(w)$.
- $\mathcal{Z}$ is a set of operational zones, or *regimes*, within the wind farm, in which turbines observe similar loads under normal operating conditions. A fraction of the power demand will be allocated to each of these regimes.
- $\mathcal{A} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_{|\mathcal{W}|}$ is the set of joint set-point configurations (or actions), which is the Cartesian product of the turbine-specific set-points $a^w \in \mathcal{A}_w$ for each turbine $w \in \mathcal{W}$. We denote $\mathcal{A}^{G(w)}$ as the set of local joint set-points for the set $G(w)$.
- $\mathrm{P}(\boldsymbol{a})$ is a stochastic function providing the farm-wide power production when a joint set-point configuration, $\boldsymbol{a} \in \mathcal{A}$, is evaluated. The global power production can be decomposed into $|\mathcal{W}|$ observable and independent local functions, i.e., $\mathrm{P}(\boldsymbol{a}) = \sum_{w \in \mathcal{W}} \mathrm{P}^w \left(\boldsymbol{a}^{G(w)}\right)$. The local function $\mathrm{P}^w \left(\boldsymbol{a}^{G(w)}\right)$ represents the power production achieved by wind turbine w and only depends on the local joint set-point $\boldsymbol{a}^{G(w)}$ of the subset of wind turbines in $G(w)$.
- $\mathrm{L}^w \left(\boldsymbol{a}^{G(w)}\right)$ assigns a penalty for performing high-load actions by wind turbine w. We assume that the wind farm operator heavily penalizes high-risk turbines (e.g., machines that are expected to have a low remaining life) based on available domain knowledge.
- $\mathrm{P}_{\mathsf{dem}}$ is the power demand imposed by the transmission system operator.

Note that we formalize the dependencies among agents as a directed dependency graph, rather than a group of agents, as described in Chapter 4. As a turbine can only affect downstream turbines in one direction, a directed dependency graph is more appropriate for the wind farm control setting. This graph structure allows us to decompose the reward function into local functions that relate to the power production achieved by one turbine, rather than a whole group. An example of a dependency graph is given in Figure 5.1.

We aim to find the joint set-point configuration that matches the power demand as well as possible, while penalizing high-load actions on wind turbines with low remaining useful life:

$$\min_{\boldsymbol{a}} \left(\sum_{z \in \mathcal{Z}} \left| f_z \mathrm{P}_{\mathsf{dem}} - \sum_{w \in z} \mathrm{P}^w \left(\boldsymbol{a}^{G(w)}\right) \right| + \sum_{w \in \mathcal{W}} \mathrm{L}^w \left(\boldsymbol{a}^{G(w)}\right) \right), \tag{5.1}$$

where f_z is a parameter that assigns a fraction of the demand to operational regime z.

The regimes are constructed based on the operational parameters observed under steady-state conditions, i.e., situations in which the wind and operational conditions at the wind farm site are stable and no transient events are present (e.g., storms and grid-loss events). Based on the fatigue loads observed in the past, i.e., stress induced by rotations performed under normal operating conditions, each regime will be assigned a fraction of the demand.

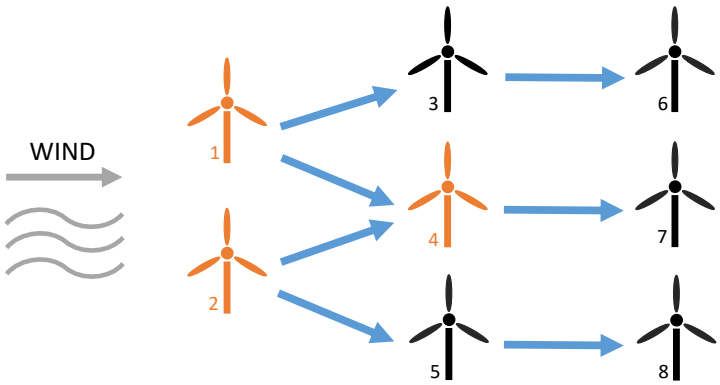


Figure 5.1: Example of a dependency graph – Upstream turbines affect downstream turbines in one direction (arrows) with respect to the wind from the west. The set of turbines on which turbine 4 depends is highlighted in orange, i.e., $G(4) = \{1, 2, 4\}$.

Specifically, if a regime has observed high fatigue loads in the past, a smaller fraction of the demand is assigned to prevent further potential damage. To allow for the inclusion of additional costs related to other types of loads, we introduce the penalty functions $\mathrm{L}^w(\boldsymbol{a}^{G(w)})$. This penalty should be formalized by the wind farm operator.

3 Operational Regimes

To ensure scalability of the control optimization process toward large wind farms, we need a learning algorithm that can leverage the similarities that exist between wind turbines. To this end, we provide a methodology to cluster wind turbines into operational regimes, i.e., a group of turbines with similar operational parameters (e.g., rotor speed and power production) in steady-state conditions. We qualitatively validate the obtained regimes against their load profiles.

To establish which turbines are similar, we analyze operational 1-second field data, provided by a supervisory control and data acquisition system (SCADA) [Boyer, 2009]. By identifying patterns in operational wind farm data and connecting them with fundamental research on loads, the control expert gains insights about the loading conditions over the wind farm for various environmental contexts. This knowledge can be incorporated in the learning process of the controller.

In order to define these similarities, we model operational farm data using Bayesian Gaussian mixture models (GMMs) [Blei et al., 2006]. These models describe the data using a finite set of (multivariate) Gaussian distributions. By modeling the data using GMMs, a soft clustering is constructed, where each data point belongs to a cluster with a certain probability. As each cluster is described using a multivariate Gaussian, the parameters that should be learned are the mean vectors and covariance matrices. The Bayesian variant introduces a Dirichlet prior [Blei et al., 2006] to infer the number of Gaussians used to model the data. This reduces model complexity to maintain interpretability and to prevent overfitting, compared to the standard Gaussian mixture models, for which the number of Gaussians needs to be known beforehand.

Restricting the data to a certain distributional form gives GMMs several desirable properties: the automatic inference of the number of clusters, the robust differentiation between noise and outliers, and the ability to perform well on small data sets. This provides a suitable mechanism to detect the frequently changing patterns in operational data and to detect similarities among the turbines' behaviors for specific environmental loading conditions.

Formally, the Bayesian GMM for the operational regimes is as follows:

$$\boldsymbol{\phi} \sim \mathcal{D}\text{ir}\left(1, \left|\widetilde{\mathcal{Z}}\right|\right) \tag{5.2a}$$

$$p\left(w \in \tilde{z}\right) = \phi_{\tilde{z}}, \forall \tilde{z} \in \widetilde{\mathcal{Z}} \tag{5.2b}$$

$$\langle \boldsymbol{\mu}^{\tilde{z}}, K^{\tilde{z}} \rangle \sim \mathcal{NIW}\left(\theta\right) \tag{5.2c}$$

$$\langle \omega^{w}, \mathrm{P}^{w} \rangle \mid w \in \tilde{z} \sim \mathcal{N}\left(\boldsymbol{\mu}^{\tilde{z}}, K^{\tilde{z}}\right), \forall w \in \mathcal{W}. \tag{5.2d}$$

First, we introduce a finite set $\widetilde{\mathcal{Z}}$ of potential regimes $\tilde{z}$ to which a turbine w can be assigned (5.2a). The size of this set reflects the maximum number of regimes that can be present in the final clustering. For the wind farms we consider in our analyses, we choose $|\widetilde{\mathcal{Z}}| = 10$ to be a sufficient number of potential regimes. The probability of turbine w belonging to a regime $\tilde{z}$ before observing any data is equal to $\phi_{\tilde{z}}$ (5.2b), which is drawn from a symmetric Dirichlet distribution with $|\widetilde{\mathcal{Z}}|$ components (i.e., the number of potential regimes) and concentration parameter 1. Intuitively, using this type of Dirichlet distribution leads to a uniform distribution over possible assignments of turbines to regimes. Thus, this distribution is often used when no assignment of turbines to regimes is favored before observing any data. Then, we associate a multivariate normal distribution to each regime $\tilde{z}$ with parameters $\boldsymbol{\mu}^{\tilde{z}}$ and $K^{\tilde{z}}$ (5.2c), which describe the means of, and covariance between, the rotor speed ω^{w} and power production P^{w} of turbine w, respectively. These means and covariance parameters are sampled from a normal-inverse-Wishart distribution $\mathcal{NIW}\left(\theta\right)$ with hyperparameters θ. This prior distribution is commonly used due to its conjugacy to the multivariate normal likelihood, which means that the posterior can be

analytically derived [Sun and Berger, 2007]. We choose θ such that we obtain a non-informative Jeffreys prior [Sun and Berger, 2007]. Finally, given the information that turbine w belongs to $\tilde{z}$, the operational parameters of w are distributed according to a multivariate normal distribution $\mathcal{N}\left(\boldsymbol{\mu}^{\tilde{z}}, K^{\tilde{z}}\right)$ (5.2d).

Using the described GMM, it is possible to derive the probability of a particular turbine belonging to any cluster $\tilde{z}$. Although it is often beneficial to use probability measures to track the uncertainty about the assignment of turbines to regimes, we focus on the established maximum likelihood means and covariances per regime, and assign each turbine to the regime it most likely belongs to. This procedures allows us to construct the final set of regimes $\mathcal{Z}$ defined in Section 2:

$$\mathcal{Z} = \cup_{\tilde{z} \in \widetilde{\mathcal{Z}}} \{w \in \mathcal{W} \mid \forall \tilde{z}' \neq \tilde{z} : p\left(w \in \tilde{z}\right) > p\left(w \in \tilde{z}'\right)\}. \tag{5.3}$$

Note that $\mathcal{Z} \subset \widetilde{\mathcal{Z}}$, and therefore not every regime in $\widetilde{\mathcal{Z}}$ has to be used in the clustering.

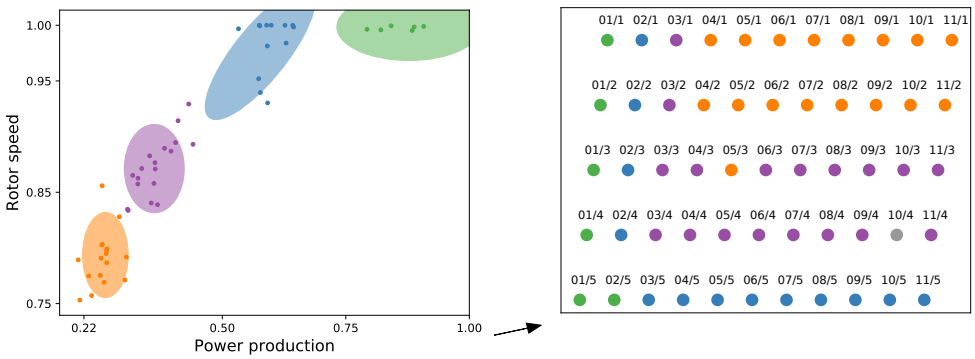
To validate the found operational zones (i.e., regimes) in the data, we perform qualitative studies of various statistics over the regimes. Specifically, we investigate the load profiles, which contain statistical properties about the loads of the turbines within a particular operational zone. For example, the produced power and fatigue loads of each turbine within the farm can be aggregated and compared per regime. Such an approach generates insightful results, as our model, combined with load statistics, describes which turbines should observe similar load spectra for specific environmental conditions.

We demonstrate the proposed method on a real-world wind farm with 55 turbines and a steady-state time window for which the environmental conditions are stable. Specifically, we investigate a 2-minute window of 1-second SCADA data and average the wind speed, wind direction, power production and rotor speed per turbine. The minimum and maximum of the wind vector is 7.9 m/s at $235.0°$ and 8.7 m/s at $235.7°$, respectively, which verifies the stability of the time window. The average incoming wind vector at the farm site is 8.2 m/s at $235.4°$. At this wind speed, the turbines are operating solely based on generator torque, in contrast to blade pitching, which is fully described by rotor speed and power production [Johnson, 2004]. Formally, the torque of turbine w at time t is defined as

$$\tau_t^w = \frac{\mathrm{P}_t^w}{\omega_t^w}, \tag{5.4}$$

where P_t^w and ω_t^w are, respectively, the produced power (in W) and angular rotor speed (in rad/s) of turbine w at time t. Per turbine, we average the rotor speed, power production and torque. The Bayesian GMMs are fitted on two operational parameters, i.e., the rotor speeds and power productions, using variational inference [Blei et al., 2006] until convergence (error $< 10^{-5}$).

Figure 5.2 shows that the GMM extracted four regimes.[1] As the majority of the downstream turbines belonged to the same regime, we apply the method a second time on this subset of turbines to extract the orange and purple clusters. Therefore, it is expected that these regimes exhibit similar operational behavior compared to the other regimes. Such a multi-step clustering approach may be necessary to further decompose the farm into sub-regimes to render the wind farm control problem tractable. Due to the wake effect [Troldborg et al., 2011], it is expected that the upstream turbines are clustered together higher on the power curve, while downstream turbines are located significantly lower, which is verified in Figure 5.2b.



(a) Operational zones (b) Wind farm view

Figure 5.2: The operational zones identified through GMM clustering: (a) Each measurement (point) is assigned to one 2D Gaussian (ellipse). Each axis is normalized between 0 and 1, respectively, denoting the minimum and maximum value possible. In the wind farm, a turbine is associated with exactly one data point, and is colored according to the most likely regime it belongs to (b). The average incoming wind vector is denoted by an arrow.

We link the established regimes to fatigue load by computing the load duration distribution (LDD) and load revolution distribution (LRD), which are two metrics that are often used in design standards and fatigue load analyses [International Electrotechnical Commission, 2012; Nejad et al., 2014a]. The LDD/LRD are frequency distributions that describe the number of seconds/revolutions performed at a particular torque level. By aggregating these distributions over all turbines per regime, we obtain the regime-specific load profiles.

[1] The data for turbine 10/4 is not available.

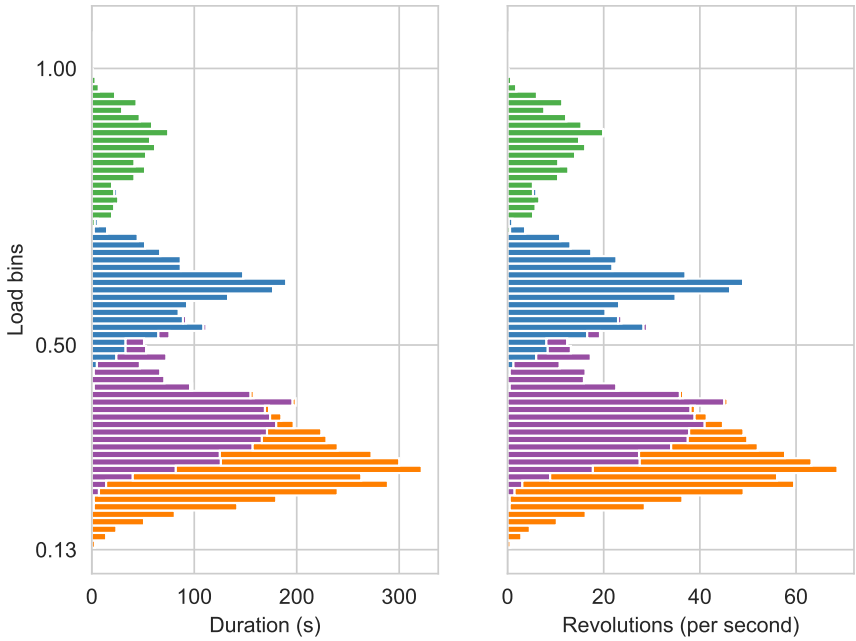


Figure 5.3: Load duration (left) and revolution (right) distribution over all turbines in the wind farm. The measurements are colored according to the corresponding operational zones in Figure 5.2. The loads are normalized between 0 and 1, respectively, denoting the minimum and maximum load possible.

Figure 5.3 shows the load duration distribution and load revolution distribution, aggregated over all turbines. We can see that each regime can be related to a Gaussian in the load distributions. As expected, regimes that mainly operate at a higher power output induce more fatigue loads. As mentioned before, the purple and orange regimes comprise turbines that are more similar compared to the other regimes, which is reflected in the large overlap of the two respective load distributions.

These results demonstrate that compiling 1-second SCADA data between similarly behaving turbines provides accurate context descriptions on a short timescale. More specifically, the load profiles produce valuable insights into the lifetime consumption of

the wind farm. Therefore, we use these profiles as guidance in the context of wind farm control, in which we assign larger fractions of the power demand to the regimes (obtained under the dominant wind direction) that have observed higher loads in the past.

4 Factorization

To accurately decompose the problem, we rely on two aspects. First, wind turbines depend on each other due to the wake effect. The decisions made by upstream turbines affect downstream turbines. Therefore, local coordination between a reference turbine and its affected neighbors is necessary to guarantee optimality of the solution. Second, due to the wake effect, upstream wind turbines produce more power than downstream turbines. As power and torque loading are highly correlated, wind turbines with similar power productions observe similar fatigue loads (see Section 3). Since turbines with a lower observed fatigue load should be responsible for the majority of the demand, a group of similar turbines can be assigned a fraction of the demand inversely proportional to their observed loads.

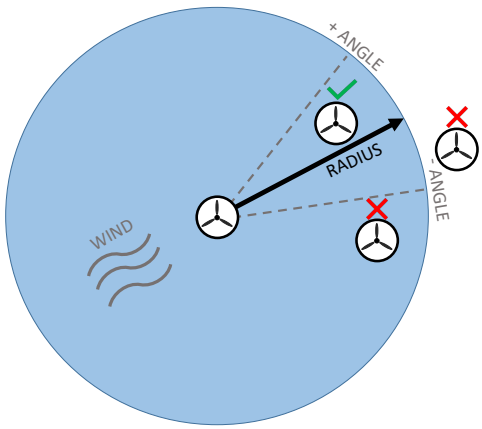


Figure 5.4: An example with one upstream reference turbine and three downstream turbines. Only one downstream turbine is considered to be dependent on the reference turbine (green check mark), as it is both within the specified radius and within the specified angle with respect to the incoming wind vector from the reference turbine.

To construct the dependency graph G, we analyze the wake field generated at a given wind direction. Using a geometric approach, we derive, for a particular reference turbine,

which downstream turbines are in its wake. Specifically, we consider a downstream turbine w to be dependent on an upstream reference turbine w_{ref} if and only if w is geographically positioned within a specified radius from w_{ref} and is within a specified angle from the incoming wind vector. Naturally, a larger angle and radius would lead to a more dense dependency graph. For our purpose, we found that downstream turbines located at an angle of $22.5°$ and a radius of 1 km sufficiently lie outside of the wake effect induced by the reference turbine, according to the Jensen wake model [Katic et al., 1986]. A visualisation of the geometric approach is shown in Figure 5.4. Examples of a wake field and corresponding dependency graph are shown in Figure 5.5.

To construct the operational regimes $\mathcal{Z}$, we group wind turbines based on their observed fatigue loads. As mentioned before, turbines with similar overall power productions have similar load profiles. Since the majority of turbine operations are performed in steady states under the dominant wind direction, we cluster the turbines based on their rotor speeds and power productions under active wake conditions *in silico* using the GMM approach described in Section 3. Once the regimes have been defined, we assign a fraction f_z of the demand to each regime z. This fraction should be proportional to the remaining useful life of the regime. We construct the load revolution distribution [International Electrotechnical Commission, 2012], i.e., the number of rotations performed by the turbines within a regime operating at a particular torque level, for each operational regime, based on real wind farm data of 24 turbines. We define the remaining lifetime in terms of the accumulated damage, which is the number of rotations operated under high-load conditions. We consider an operation to be high-load when the expected (main shaft) torque exceeds a certain threshold [Alvarez and Ribaric, 2018]. Thus, given the torque computed using Equation 5.4, we can derive the number of rotations performed at a torque higher than this threshold from the load revolution distribution of each turbine and aggregate the results per regime. After normalization of the remaining rotations over the entire wind farm, the fraction f_z is set to the inverse of the sum over all turbines within regime z. An example of regimes, associated with the normalized number of life-consuming rotations per turbine, is shown in Figure 5.6.

5 Set-Point Thompson Sampling

Consider the set-point allocation problem (as described in Section 2). The expected power productions $\mathrm{P}^w\left(\boldsymbol{a}^{G(w)}\right)$ for each possible local set-point configuration $\boldsymbol{a}^{G(w)}$ are unknown. Similar to multi-agent Thompson sampling (see Section 3 of Chapter 4), SPTS uses a Bayesian formalism, which means users can exert their beliefs over $\mathrm{P}^w\left(\boldsymbol{a}^{G(w)}\right)$ in the form of a prior. If wind turbines were not affected by wake, the incoming wind speed can be used to predict the expected power production of the wind turbine, which is provided

with the turbine design specifications [Lydia et al., 2014]. Therefore, for a given set-point, wind speed and turbine, we model the achieved power using a Gaussian prior, where the mean is the expected power production given the set-point under no wake conditions, and the standard deviation σ represents prior uncertainty about the achieved power.

$$\begin{aligned} \mathrm{P}^w(\boldsymbol{a}) &\sim \mathcal{N}(\cdot \mid \mu^w_{a^w}, \sigma), \\ \mu^w_{a^w} &= \min(a^w, \mathrm{P}^w_{\mathsf{av}}), \end{aligned} \tag{5.5}$$

where the mean is the minimum between the power set-point a^w and the available power $\mathrm{P}^w_{\mathsf{av}}$. The standard deviation σ balances exploration of alternative power production outcomes (high σ), and exploitation of the provided domain knowledge (low σ).

At each time step t, SPTS draws a sample $\mathrm{P}^w_t\left(\boldsymbol{a}^{G(w)}\right)$ from the posterior for each wind turbine and possible local set-point configuration, given the history $\mathcal{H}_{t-1}$, consisting of previously evaluated set-points and associated power productions:

$$\begin{aligned} &\mathrm{P}^w_t\left(\boldsymbol{a}^{G(w)}\right) \sim \mathcal{N}(\cdot \mid \mu^w_{a^w}, \sigma, \boldsymbol{a}^{G(w)}, \mathcal{H}_{t-1}), \text{ with} \\ &\mathcal{H}_{t-1} = \bigcup_{i=1}^{t-1} \bigcup_{w \in \mathcal{W}} \left\{ \left\langle \boldsymbol{a}^{G(w)}_i, \mathrm{P}^w\left(\boldsymbol{a}^{G(w)}_i\right) \right\rangle \right\}. \end{aligned} \tag{5.6}$$

Note that during this step, SPTS samples directly the posterior over the unknown local means, which implies that the sample $\mathrm{P}^w_t\left(\boldsymbol{a}^{G(w)}\right)$ and the unknown mean $\mathrm{P}^w\left(\boldsymbol{a}^{G(w)}\right)$ are independent and identically distributed at time step t, given history $\mathcal{H}_{t-1}$.

SPTS takes the set-point that minimizes the objective function (see Equation 5.1). In traditional Thompson sampling [Thompson, 1933], the optimal solution is found by maximizing over the full joint action space. However, this is intractable for larger multi-agent settings, as the joint action space scales exponentially with the number of agents. For example, a wind farm comprised of 20 wind turbines, where each turbine can choose from 3 possible set-points, would have 3^{20} (approximately 3.5 billion) possible configurations. In this regard, we note that due to the structure of wind farms, the optimal set-point configuration exists in the sparse factored representation of the joint action space. Therefore, the minimization problem defined in Equation 5.1 can be solved exactly and effectively using variable elimination [Guestrin et al., 2001b] or linear programming [Lougee-Heimer, 2003].

Finally, the joint set-point configuration that minimizes the objective function, $\boldsymbol{a}_t$, is executed in simulation and the associated power productions $\mathrm{P}^w_t\left(\boldsymbol{a}^{G(w)}_t\right)$ will be recorded for each wind turbine w. SPTS is formally described in Algorithm 4.

Algorithm 4: Set-Point Thompson Sampling (SPTS)

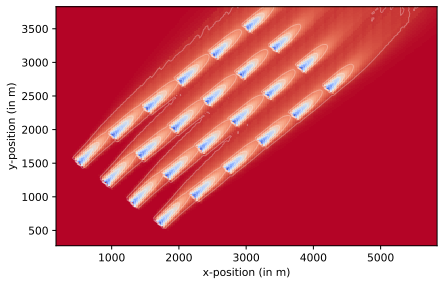
$G, \mathcal{Z} \leftarrow$ Construct dependency graph and regimes
$\mathcal{H}_0 \leftarrow \{\}$
for $t \in [1..T]$ **do**
  Sample expected performance for every possible local set-point configuration.
  for $w \in \mathcal{W}, \boldsymbol{a} \in \mathcal{A}^{G(w)}$ **do**
    $\mathrm{P}^w(\boldsymbol{s}) \sim \mathcal{N}(\,\cdot \mid \mu^w_{a^w}, \sigma, \mathcal{H}_{t-1})$
  end

  Select best joint configuration.
  $\boldsymbol{a}_t \leftarrow \arg\min_{\boldsymbol{a}} \sum_{z \in \mathcal{Z}} \left| f_z \mathrm{P}_{\mathsf{dem}} - \sum_{w \in z} \mathrm{P}^w\left(\boldsymbol{a}^{G(w)}\right) \right|$
  $+ \sum_{w \in \mathcal{W}} \mathrm{L}^w\left(\boldsymbol{a}^{G(w)}\right)$

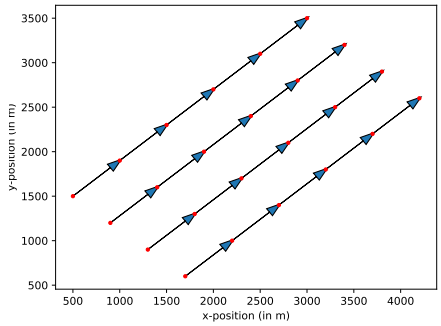
  Simulate chosen set-point configuration.
  $\left\langle \mathrm{P}^w_t\left(\boldsymbol{a}_t^{G(w)}\right)\right\rangle_{w \in \mathcal{W}} \leftarrow$ Simulate configuration $\boldsymbol{a}_t$

  Update belief distributions using observed performance.
  $\mathcal{H}_t \leftarrow \mathcal{H}_{t-1} \cup \left\{ \left\langle \boldsymbol{a}_t^{G(w)}, \mathrm{P}^w_t\left(\boldsymbol{a}_t^{G(w)}\right)\right\rangle_{w \in \mathcal{W}} \right\}$
end

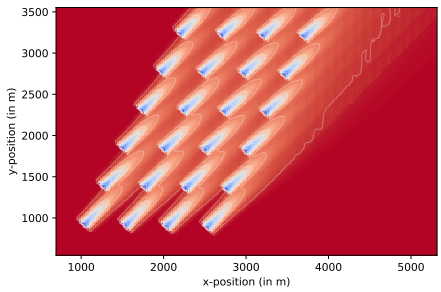
6 Experiments



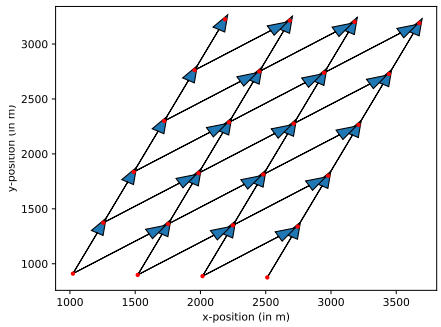
(a) Wake field – 0°



(b) Dependency graph – 0°



(c) Wake field – 30°



(d) Dependency graph – 30°

Figure 5.5: Wind farm – Based on the wake field generated for a particular wind direction, wind speed of 11 m/s, and maximal power set-points, we create a dependency graph. We show the results for wind directions of 0° (a & b) and 30° (c & d). For a wind direction of 30°, the wind farm is rotated by -30°, such that the global wind vector always starts at $(0, 0)$.

As the majority of offshore wind farms have a symmetric grid-like topology [Tao et al., 2020], we conduct our experiments in a wind farm that has the shape of a parallelogram. Grid-like layouts are often beneficial toward the planning and construction of the farms. However, such layouts cause wake due to the proximity of the turbines, reducing the overall

power production of the wind farm [Tao et al., 2020]. We place 24 turbines in a 4-by-6 grid, 500 m apart along the x-axis and 400 m apart along the y-axis, as shown in Figure 5.5.

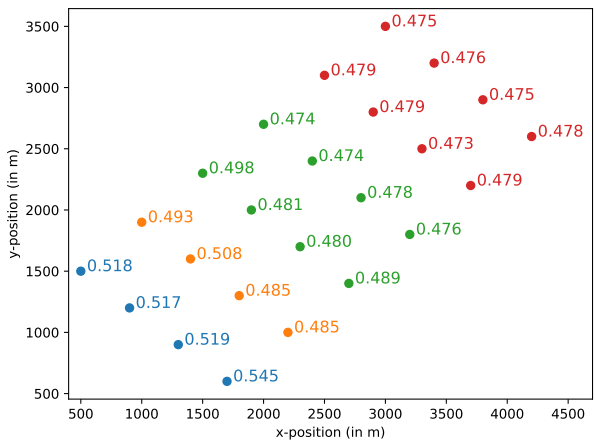


Figure 5.6: Operational regimes in the wind farm, obtained under the dominant wind vector coming from the origin $(0, 0)$. The turbines are annotated with their observed normalized damage-inducing load.

We investigate our method under the same global wind conditions used by [Siniscalchi-Minna et al., 2019], in which we assume that $0°$ is the dominant wind direction and orthogonal to the wind farm grid. Specifically, we investigate the incoming wind vectors at $0°$ and $30°$, with a speed of 11 m/s. Under these conditions, the wake effect is strong. The wake fields and dependency graphs for both wind vectors are shown in Figure 5.5.

To model the wake effect and steady-state conditions at the farm, we use the state-of-the-art 'FLOw Redirection and Induction in Steady-state' (FLORIS) simulator [National Renewable Energy Laboratory (NREL), 2019]. This simulator has been used extensively in the context of wind farm layout optimization [Gebraad et al., 2017; Tingey and Ning, 2017], as well as power maximization [Boersma et al., 2017]. In addition, we use the LW-8MW reference turbine [Desmond et al., 2016] to model the turbines' operational behavior. Both the wake simulator and the turbine model accurately represent the scale of, and conditions at, contemporary offshore wind farms. From real wind farm data, we compute damage-inducing fatigue loads for every turbine, according to [Alvarez and Ribaric, 2018], and derive the fractions f_z as discussed in Section 4. The normalized damage-inducing fatigue loads are reported in Figure 5.6. We provide 3 possible set-points to a wind turbine w, i.e., $a_w \in \{1490 \text{ kW}, 6420 \text{ kW}, 8000 \text{ kW}\}$ (equivalent to measured wind speeds of 6.5,

10.0 or 13.5 m/s at the turbine's location), which translates into a low, medium and high power production.

We perform experiments for all combinations of the following sets of parameters:

- Wind direction: $d \in \{0°, 30°\}$
- Demand: $\mathrm{P}_{\text{dem}} \in \{60 \text{ MW}, 70 \text{ MW}, 80 \text{ MW}, 90 \text{ MW}, 100 \text{ MW}\}$
- Number of high-risk turbines: $n_{\text{risk}} \in \{1, 2, 3, 4\}$
- Penalty: $\mathrm{L}^w\left(\boldsymbol{a}^{G(w)}\right) = \begin{cases} +\infty \text{ if } w \text{ is high-risk and} \\ \qquad \mathrm{P}_t^w\left(\boldsymbol{a}^{G(w)}\right) \geq 5.2 \text{ MW} \\ 0 \text{ otherwise} \end{cases}$

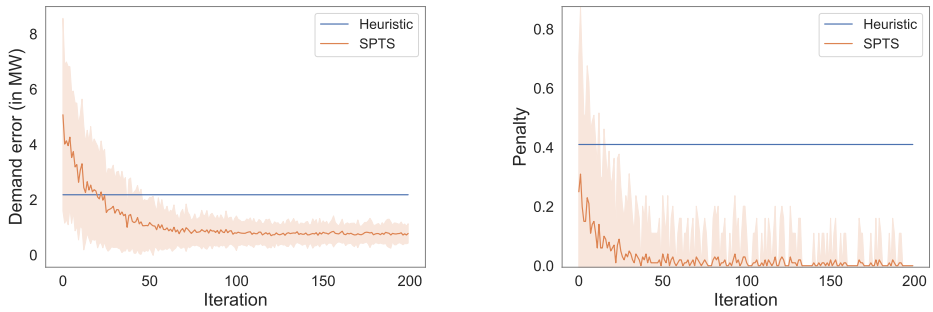
An infinite penalty is provided to a high-risk turbine when the used set-point leads to damage accumulation. We assume that damage occurs when the torque exceeds the one achieved at 65% of the turbine's maximum power level [Alvarez and Ribaric, 2018]. For 8 MW wind turbines, this is equal to a power production of 5.2 MW. Although any non-linear penalty function can be used, this infinite penalty allows the wind farm operator to identify high-risk turbines and ensure that no high-load set-points are assigned to them (see Section 7). The n_{risk} high-risk turbines are randomly chosen. We set the standard deviation σ in the prior distribution (Equation 5.5) to 1 MW, which allows for a sufficient amount of exploration over the complete power range of $[0 \text{ MW}, 8 \text{ MW}]$. Each experiment is repeated 100 times.[2]

We compare SPTS with a set-point allocation strategy, based on the heuristic proposed by Siniscalchi-Minna et al. [2019]. This heuristic approach first assigns higher set-points to turbines which are further back in the farm, with respect to the incoming wind vector. This process is repeated toward the front of the farm until the required demand is reached. The solution with the power production that is closest to the demand is recorded. For both the heuristic and SPTS, the best performing control strategies of each run are compared. As the main focus is to prevent high-load actions on high-risk turbines, we define the performance of a set-point configuration in terms of the total penalty first, and in case of draws, the configuration that matches the demand the closest is chosen.

Figure 5.7 shows the learning curve of SPTS for the setting with a wind direction of $0°$, a demand of 80 MW and 3 high-risk turbines. The trend indicates that 200 iterations are sufficient to ensure convergence. The learning curves for all settings are reported in Appendix 4.1. Note that the heuristic is a deterministic approach, and thus the variance on the outcomes over multiple repetitions of the experiment is zero.

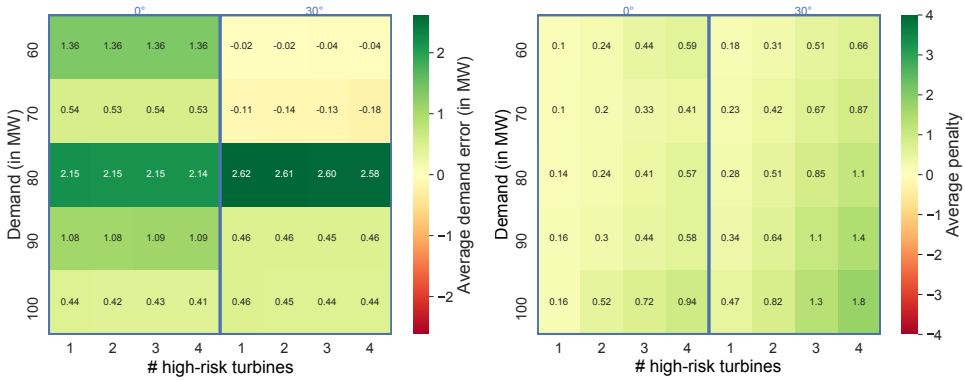
Figure 5.8 shows the average absolute difference between the best performances of the heuristic and of SPTS for all parameter combinations, both in terms of penalty and demand

[2]The source code to reproduce the experiments is publicly available at:
https://github.com/timo-verstraeten/spts-experiments



(a) Demand error (b) Penalty

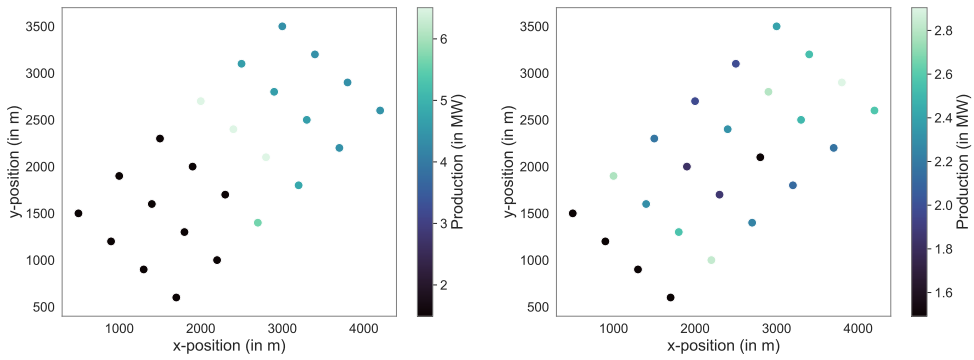
Figure 5.7: Learning curves of the demand error (a) and total penalty (b) obtained at each iteration, for a wind direction of $0°$, a demand of 80 MW and 3 high-risk turbines. The average trend (line) and standard deviation (shaded area) are shown. The experiment is repeated 100 times.



(a) Demand error (b) Penalty

Figure 5.8: Heatmaps of the average absolute difference between the performance of the heuristic and of SPTS. The performance in terms of demand error (left) and total penalty (right) are plotted. A positive value (green) indicates a better average performance obtained by SPTS, while a negative value (red) indicates a worse performance compared to the heuristic. The performances are averaged over 100 samples.

error. SPTS reaches comparable or better results than the heuristic in terms of demand error. However, the heuristic approach receives more penalties when the number of high-risk turbines is increased, or when the required demand is increased. This is expected, as the heuristic approach will allocate higher set-points to high-risk machines to reach a higher demand. Moreover, SPTS significantly outperforms the heuristic with respect to the demand error when the demand is 80 MW. This is due to the fact that many possible set-point configurations exist to meet this demand, and can thus be easily found by SPTS. In contrast, when the demand is 60 MW or 100 MW, only a few configurations are viable, in which most of the set-points are low or high, respectively. It is important to note that, over all runs, the best configurations achieved by SPTS *never* contained a damage-inducing set-point assigned to a high-risk turbine (i.e., the total penalty of the solution is always 0 for all settings). Box plots of the best performing control strategies obtained by SPTS and the heuristic for all parameter combinations are reported in Appendix 4.2.



(a) Heuristic

(b) SPTS

Figure 5.9: Farm-wide view of the power productions for the wind direction of $0°$, a demand of 80 MW and 3 high-risk turbines. The power productions for the best set-point configurations found by the heuristic approach (a) and SPTS (b) are shown. The power productions are averaged over 100 runs.

Figure 5.9 provides a farm-wide view of the power productions obtained, given the best set-point configuration found by the heuristic approach and SPTS. This is shown for the setting with a wind direction of $0°$, a demand of 80 MW and 3 high-risk turbines. The farm-wide power productions for all settings are reported in Appendix 4.3. By the heuristic's definition, set-points are first maximized for downstream turbines. This process is repeated toward the front of the farm. Therefore, it is expected to see a pattern in

which the downstream turbines exhibit a higher actual power production, compared to the upstream turbines. While the heuristic generates an intuitive pattern, the solution of SPTS indicates that this is not necessarily an optimal solution. Therefore, the results shown in Figure 5.8a and 5.9 demonstrate that a data-driven approach is essential in order to come up with context-aware solutions. Nevertheless, due to the allocation of demand over the different regimes, SPTS also favors higher set-points for downstream turbines. The reduced power generation of turbines with higher observed loads, combined with the ability to include arbitrary penalty functions (see Figure 5.8b), demonstrates that SPTS is suitable to match the power demand in a manner that reduces overall lifetime consumption.

7 Discussion

We propose a new wind farm control algorithm that allocates set-points taking into account load information. The results demonstrate that SPTS can successfully match the power demand without assigning high-load set-points to turbines.

We focus on a wind farm with a symmetric grid-like structure, which comprises the majority of contemporary wind farm designs. However, our method is not biased toward symmetric topologies. Instead, SPTS guides the learning process by using information about the environmental and operational conditions at the farm site, acquired from wake analyses and knowledge regarding the turbines' health. Therefore, SPTS is applicable to irregularly shaped wind farms, which can be constructed due to landscape constraints [Gu and Wang, 2013].

To model the turbine's dynamics and wind flow, we use the FLORIS wake simulator, in which set-point configurations can be evaluated quickly. SPTS has the ability to include expert knowledge without being dependent on the simulator used. Therefore, it is easy to switch between different types of simulators, such as computational fluid dynamics simulators [Castellani et al., 2013].

In our experiments, we focus on optimizing control strategies under steady wind conditions. Therefore, it is sufficient to analyze the expected performance of control strategies in a noise-free setting. Nevertheless, in the case of transient wind conditions (e.g., wind gusts or storms), it may be important to investigate the variance on the performance measure as well. Since SPTS uses a Bayesian sampling approach, it is straightforward to introduce a likelihood distribution with a (possibly unknown) noise parameter and update the posterior distributions according to Bayes' rule [Russo et al., 2017].

Our approach uses multi-agent Thompson sampling (see Section 3 of Chapter 4) for sampling the set-point configuration space. For multi-agent Thompson sampling, an asymptotic upper bound was established on the cumulative regret, i.e., the total

performance loss obtained by executing sub-optimal actions during the learning phase. As we investigate a noise-free setting, and every local set-point will likely be sampled once eventually, the optimal set-point configuration will almost surely be found. Therefore, the asymptotic upper bound has no practical use here. Nevertheless, the composition of the bound suggests that the performance loss of SPTS is in terms of the number of local joint set-points, rather than the number of global joint set-points, as is the case in traditional Thompson sampling. Therefore, it is expected that SPTS can learn efficiently using noisy set-point evaluations as well.

SPTS is a sampling technique inspired by Thompson sampling to select promising joint actions. Recently, theoretical guarantees have been established for Thompson sampling with respect to cumulative regret, i.e., the total difference between the expected reward of the optimal (unknown) action and chosen actions obtained *during* the learning process [Agrawal and Goyal, 2012]. However, as the set-point optimization is performed *in silico*, the performance of the evaluated set-points during the learning phase is only used to guide the learning process. In our setting, we focus on the performance of the end result, i.e., the best set-point configuration obtained *after* learning. This highlights the distinction between exploration-exploitation and best arm identification [Audibert and Bubeck, 2010]. Note that this is a difference in terms of convergence speed, rather than optimality, as the action chosen by Thompson sampling still converges to the optimal one (under certain conditions) [Lattimore and Szepesvári, 2020]. To our knowledge, there are no Bayesian best arm identification algorithms available for dealing with factored multi-agent systems. Although the learning curves converge quickly (see Appendix 4.1), further research into best arm identification algorithms for loosely-coupled multi-agents is warranted to improve sample efficiency.

In our experiments, we use objective functions that heavily penalize a set of turbines, which allows wind farm operators to mark high-risk turbines and reduce their loading conditions. Still, SPTS finds the optimal joint set-point configuration under arbitrary penalties (see Section 2). Such flexibility is required for capturing the multi-dimensional load spectrum that is present in wind turbine technology. However, further research needs to be conducted to establish the links between turbine responses during dynamic events and potential failure modes [Keller et al., 2016; Junior et al., 2017; Verstraeten et al., 2019]. Through fundamental research, maintenance costs can be formalized as a penalty function within SPTS, which is important toward the further development of advanced wind farm controllers.

SPTS finds the optimal combination of set-points taken from a discrete space. To further reduce the demand error, continuous set-points should be considered. Optimizing over a factored representation in continuous space is challenging, as the choice of a turbine is possibly dependent on an infinite amount of configurations chosen by its parents. To our

knowledge, no optimization algorithms currently exist that fully operate within a factored continuous action space. Therefore, research should focus on continuous optimization techniques for loosely-coupled multi-agent systems, such that accurate solutions can be provided in a feasible manner. Nevertheless, one can use the optimal discrete set-point configuration provided by SPTS as a starting point and further optimize over the continuous set-point space using an iterative approach [Siniscalchi-Minna et al., 2019]. Naturally, such a solution may not be provably optimal and convergence may be challenging to guarantee.

Finally, our approach considers each scenario independently and does not generalize over environmental conditions. In data-driven wind farm control research, control strategies are often learned without considering the dependencies between environmental parameters (e.g., wind speed and wind direction) [van Dijk et al., 2016; Verstraeten et al., 2019]. However, generalizing over environmental conditions, rather than learning for each condition independently, would improve the sample-efficiency of SPTS over multiple settings. This could be achieved, for instance, through the use of contextual bandits [Agrawal and Goyal, 2013b].

6 Discussion

In this chapter, we summarize the contributions that we presented throughout this dissertation. Next, we discuss the valorisation potential of our research, both in the context of wind farm control, as well as other applications. Finally, we discuss different opportunities for future work.

1 Contributions

Our first contribution is a data-driven control method for pools of devices, i.e., systems that comprise multiple similar devices. Reinforcement learning techniques typically require a large amount of samples to accurately and effectively learn the optimal control policy. Therefore, in Chapter 3, **we propose a new reinforcement learning technique, called policy iteration for pools of devices (PIPoD), that leverages the similarities between devices to inform the learning process of the devices' transition models**. Specifically, the transition models are jointly defined as a coregionalized Gaussian process, which establishes correlations between the different devices. We evaluate PIPoD on two well-known reinforcement learning benchmark settings, i.e., mountain car and cart-pole, as well as a synthetic wind farm control task. Our results show that PIPoD successfully reduces uncertainty in the learning process by using relevant data from similar devices, and prevents negative transfer by using irrelevant data from other devices.

Our second contribution is a control method for multi-agent systems with topology information. Find the optimal control policy is challenging in large-scale multi-agent systems, as the joint action space scales exponentially with the number of agents. Still,

when the dependency structure of the agents is sparse, the system can be factored into subgroups of agents, significantly reducing the complexity. To this end, in Chapter 4, **we propose a new control method, called multi-agent Thompson sampling (MATS), that exploits the sparse neighborhood structure of the agents to improve the scalability of the learning process toward large-scale multi-agent systems**. Specifically, MATS constructs groups of agents based on a coordination graph, and allows decision making on a local level, while focusing on the action that is globally optimal. We provide theoretical guarantees showing that, for subgaussian rewards, MATS always improves upon the current solution and can effectively use the neighborhood structure of the multi-agent system to reduce the complexity of the joint action space. We demonstrate that MATS achieves state-of-the-art performance on several benchmark problems, i.e., the Bernoulli 0101-Chain, the Gem Mining problem and the Poisson 0101-Chain, as well as a synthetic wind farm control task.

Our third contribution is a scalable data-driven wind farm control method in the context of active power control, i.e., assigning power set-points to each turbine in order to match the farm-wide power demand. Control optimization methods are necessary, as physics-based heuristics fail to capture the complex high-dimensional load spectrum that is inherent to wind farm technology. However, current data-driven methods scale poorly to the large size of contemporary wind farms. Therefore, in Chapter 5, **we propose a new scalable wind farm control method, called set-point Thompson sampling (SPTS), that uses similarity and topology information to factorize the wind farm and learns a control strategy within this factored space**. SPTS employs a hybrid approach that balances the flexibility of AI-based learning techniques with physics-based expertise to improve the tractability of the learning process. We evaluate our method on a realistic wind farm control task *in silico*, covering an exhaustive set of parametrizations. We compare to a commonly used physics-based heuristic as a baseline, and show that our method achieves competitive performance in terms of demand error, while incorporating complex load-based penalty functions.

2 Valorisation Potential

The research in this dissertation was funded by an FWO[1] grant for strategic basic research. The FWO assigns this grant to challenging and innovative research, which may in the long term lead to technological advancements with economic and/or societal added value. As we start from investigating AI-driven control for generic multi-agent systems, our methods are useful in a wide variety of applications, such as the monitoring and dynamic

[1]Fonds voor Wetenschappelijk Onderzoek - Vlaanderen, Research Foundation – Flanders

optimization of multiple manufacturing processes [Shen, 2019]. For example, using the methodology described in this dissertation, one could develop a health-aware controller for a group of similar manufacturing machines. Nevertheless, in this section, we mainly focus on the potential of valorisation in the wind farm industry.

In this dissertation, we investigate several properties to render learning in large multi-agent systems tractable. In the wind farm industry there is a steep trend to increase capacity to maximize the power output with respect to the used space and the associated capital costs [Fraile et al., 2018]. To cope with the increase in size of real-world multi-agent systems, the complexity of control learning methods needs to be reduced significantly. Our proposed methods approach this challenge leveraging three properties that are present in many applications. First, when learning tasks are complex, sharing data among similar agents can significantly improve sample-efficiency. Especially in settings where evaluations are costly [Brochu et al., 2010], the number of samples can be significantly reduced by reusing data within the same pool of devices, and therefore lower the costs associated with the learning process. PIPoD can be used to handle data exchange within the control learning process of multiple independent agents. Second, we show that leveraging topological information can significantly increase the learning speed in multi-agent systems. Learning in large-scale multi-agent systems has been challenging in the real-world. When the number of agents increase in a system, the learning complexity of control tasks increases exponentially, which directly affects operation and maintenance costs. Using MATS in the industry would significantly reduce these costs. Third, we adopt the Bayesian framework to allow operators to easily introduce domain knowledge, as such information is often readily available. Additionally, Bayesian methods explicitly quantify uncertainty over the unknown parameters, given the provided expert knowledge. Such statistics are useful for assessing the reliability of the found solutions.

In the context of wind farm control, we propose SPTS to allow for AI-driven control of contemporary wind farms in a scalable manner. The ability to include arbitrary cost functions is important, as the load conditions in wind farms are multi-dimensional. Wind farm operators can penalize high-risk turbines to reduce maintenance costs, while still leveraging maximal power production. Therefore, the use of SPTS can significantly improve the reliability and sustainability of wind farm technology, which is important to reduce operations and maintenance costs.

Several aspects need to be investigated to render our control methods suitable for real-world applications. First, privacy is a concern that exist in many industries. To fully leverage the data sources at hand, it must be guaranteed that they are handled in a privacy-preserving manner. We have taken initial steps to render PIPoD privacy-preserving. Specifically, we have shown that it is possible to extend PIPoD with secure

multi-party computation and homomorphic encryptions [Loeb et al., 2019].[2] Although such mechanisms introduce communicational overhead between the devices, it guarantees a privacy-preserving encoding of each device-specific data set, while still yielding high-performing policies. Second, data-driven control policies must guarantee safe operation. To this end, we limitedly explored the use of control barrier functions to regulate the safety measures imposed on the control strategies learned by PIPoD [Hennion, 2020].[3] We demonstrated that safe reinforcement learning in PoD settings is feasible, while achieving a minimal performance gap with respect to the unconstrained controller.

3 Future Work

In this dissertation, we focus on steady-state control, in which the environmental and operational conditions are stable. Set-point Thompson sampling (SPTS) is a data-driven method for steady-state control that has the ability to consider non-linear health indicators of the turbines, while learning optimal farm-wide controller actions. Nevertheless, as described in Chapter 1, the loads induced during dynamic events (e.g., storms and grid-loss events) have a significant impact on failure. Therefore, preventive measures must be implemented that minimize the impact of loads induced during transient events. We argue that the topology and similarities between turbines in terms of environmental conditions are important to successfully develop transient-state wind farm controllers to prevent high-load situations that may lead to failure.

To model transient wind conditions, such as storms, we will investigate the use of SPTS in a simulator that uses computational fluid dynamics to model wind flow [Richmond et al., 2019]. A sample-efficient method, such as SPTS will be necessary to find the best control strategy, as executing a computational fluid dynamics simulator is time-consuming, and thus evaluating alternative control strategies in such a simulator is costly.

To construct control strategies that can reduce the probability of failure, it is necessary to conduct fundamental research to understand the high-dimensional load spectrum inherent to wind farm technology, and establish the links between turbine responses and failure modes. To bridge this gap, we will rely on state-of-the-art condition monitoring systems that report the health status of a turbine through proxies, e.g., temperature and vibration signals [Peeters et al., 2019]. In future work, we aim to consider these signals as health indicator, and incorporate penalties in SPTS through the use of failure prognosis

[2] This research was conducted by Regis Loeb in the context of a Master thesis under the joint supervision of prof. dr. Ann Dooms and prof. dr. Ann Nowé.

[3] This research was conducted by Domien Hennion in the context of a Master thesis under the supervision of prof. dr. Ann Nowé.

methodologies (e.g., [Nejad et al., 2014b]) that prevent high-load actions on turbines with low health.

High-frequency data sources are necessary to swiftly initiate preventive control measures during dynamic events [Verstraeten et al., 2019]. However, several environmental and operational parameters are extremely noisy when measured at high frequency (e.g., wind speed) [Gonzalez et al., 2017]. To accurately represent the condition of the wind farm during a short-term transient events, a mechanism is necessary to reduce measurement noise, while retaining the fine granularity of the data sources. To this end, we will investigate the use of the coregionalized Gaussian process, described in Chapter 3, to perform similarity-based data exchange between the turbines to reduce noise, rather than averaging over time windows, as is commonly used [Gonzalez et al., 2017]. Such an approach can be used to process real wind farm data, and provide a reliable parametrization of a wake simulator to represent the wind and turbine conditions at the wind farm site. Combined with the SPTS control algorithm, scenario optimization can be performed to provide set-point configurations in real time, allowing our proposed approaches to be used in the real world.

A Appendices

1 Success Rates of PIPoD

In Table A.1, we show the percentages of success of policy iteration for pools of devices (PIPoD) for both the mountain car and cart-pole benchmarks, compared to the joint and single target types, over 50 runs. For mountain car, a successful run is one where the car reaches the goal within 200 time steps, while for cart-pole, a successful run is one where the cart manages to keep the pole above the threshold for 200 time steps. Although the percentage of success for both the single and PoD targets are high, we show in the main manuscript that the performance (measured by the sum of the squared distances from the goal) of a policy learned by the single target type is significantly higher than the performance of a policy learned by PIPoD. This means that the single target type infers a sub-optimal policy for reaching the goal, while PIPoD often exhibits optimal behavior.

	joint	single	PIPoD
mountain car	64%	80%	86%
cart-pole	38%	74%	86%

Table A.1: Success rates

2 Sensitivity Analysis of PIPoD

We perform a sensitivity analysis on the PoD variant of the continuous mountain car domain to investigate the robustness of policy iteration for pools of devices (PIPoD), proposed in Chapter 3. In this problem, an action is a force applied to either side of the car. The magnitude of this force (i.e., the power) can be configured. We consider a fleet of three mountain cars: a target, source A and source B. We use the same parameters for the setting as described in the main paper, except that we vary the power parameter of source A. Specifically, the target has a power of $15 \cdot 10^{-4}$, source B has a power of 10^{-4} units, and source A has a power that varies according to $(5 + i) \cdot 10^{-4}$, for $i \in [0..10]$. This ranges captures various grades of similarity between the target and source A.

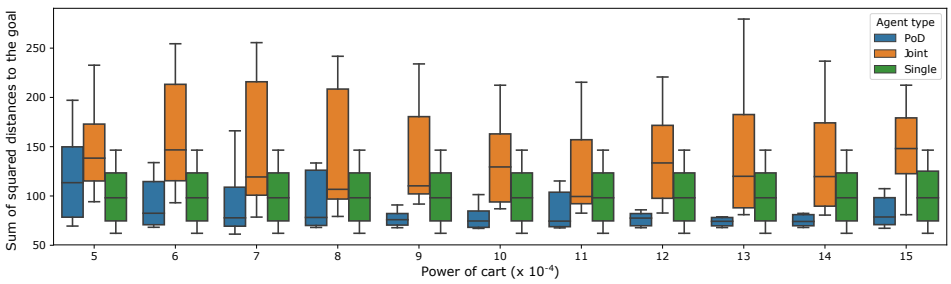


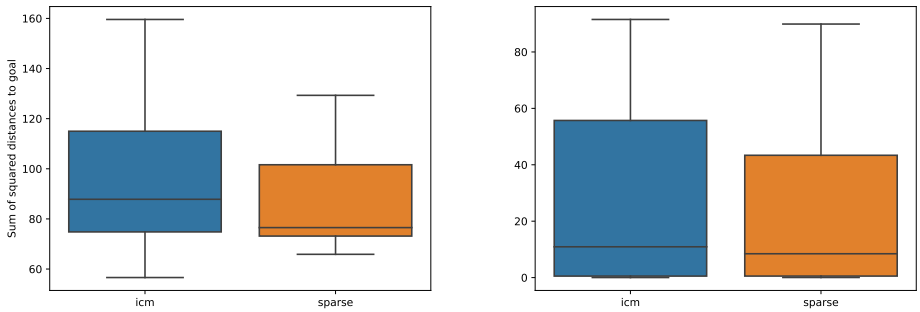
Figure A.1: Sensitivity analysis – Boxplot of the sum of squared distances to the goal over 200 time steps for 50 runs.

We run the experiment 50 times for the three target types: single, joint and PoD. We measure performance in terms of the sum of squared distances to the goal. We repeat each experiment 50 times. The results are shown in Figure A.1. As expected, the PoD target performs better when source A has a power closer to the target's power. When the power of source A decreases, there is less relevant information for source A to share with the target. This results in similar outcomes as for the single target type, which does not perform any transfer. Still, we can see that the PoD target significantly outperforms both the joint and single targets for the higher power levels, and exhibits a performance similar to the single target for the lower power levels. The discrepancy between the performance of the single and fleet targets at a power level of $5 \cdot 10^{-4}$ is expected, as the single target already assumes there is no correlation between the target and the sources, while our fleet target still needs to learn this fact. As additional accurate domain knowledge is available to the single target, the problems becomes strictly easier for the single target to solve.

Naturally, such domain knowledge will not be available in the real world, highlighting the need for a flexible method as PIPoD.

3 Comparison of Intrinsic Coregionalization Model with Sparse Variant

The transition model described in Section 3 of Chapter 3 is a sparse variant of the intrinsic coregionalization model [Bonilla et al., 2008]. The computational complexity is significantly decreased when using our sparse model, compared to a fully-connected model such as the intrinsic coregionalization model. To show that using our sparse transition model does not incur a significant decrease in performance, we compare our model (sparse) against the intrinsic coregionalization model (icm) on the same synthetic benchmarks as in the main manuscript (see Figure A.2).



(a) Mountain car

(b) Cart-pole

Figure A.2: Boxplot of the total sum of squared distances to the goal state for the mountain car (a) and cart-pole (b) benchmarks during 200 time steps for 50 runs.

4 Exhaustive List of Empirical Results for SPTS

We perform experiments for the all combinations of the following sets of parameters:

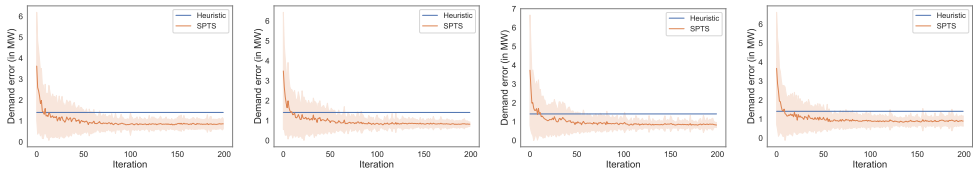
- Wind direction: $d \in \{0°, 30°\}$
- Demand: $\mathrm{P}_{\text{dem}} \in \{60 \text{ MW}, 70 \text{ MW}, 80 \text{ MW}, 90 \text{ MW}, 100 \text{ MW}\}$
- Number of high-risk turbines: $n_{\text{risk}} \in \{1, 2, 3, 4\}$
- Penalty: $\mathrm{L}^w\left(\boldsymbol{a}^{G(w)}\right) = \begin{cases} +\infty \text{ if } w \text{ is high-risk and} \\ \qquad \mathrm{P}_t^w\left(\boldsymbol{a}^{G(w)}\right) \geq 5.2 \text{ MW} \\ 0 \text{ otherwise} \end{cases}$

Each experiment is repeated 100 times.

Overall, the results discussed in the main manuscript are representative for all experiments, and sufficiently demonstrate the performance of SPTS compared to the heuristic approach. Nevertheless, we disclose all obtained results in this document.

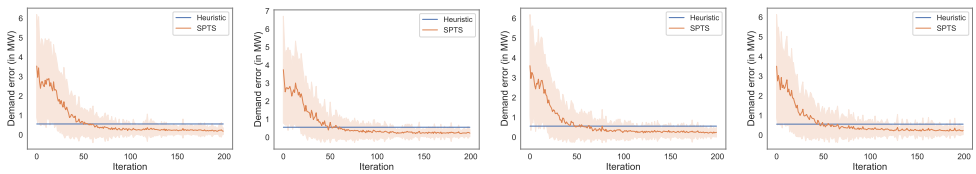
4.1 Learning Curves

Demand Error



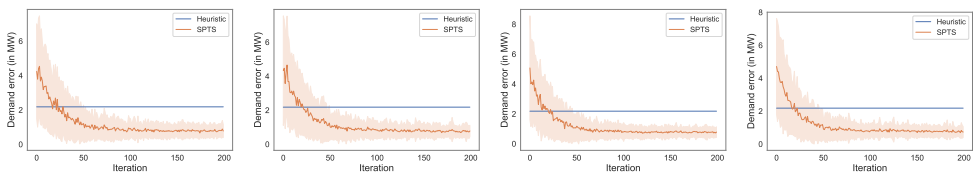
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.3: Demand error for $d = 0°$ and $\mathrm{P}_{\text{dem}} = 60$ MW. The average trend and standard deviation are plotted.



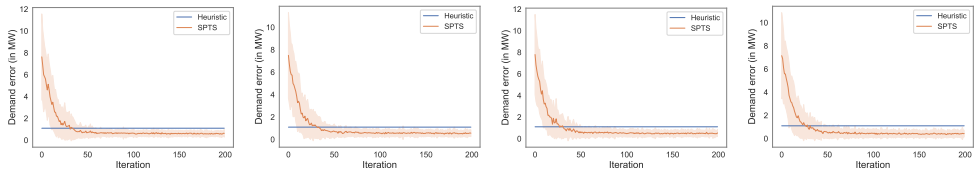
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.4: Demand error for $d = 0°$ and $\mathrm{P}_{\text{dem}} = 70$ MW. The average trend and standard deviation are plotted.



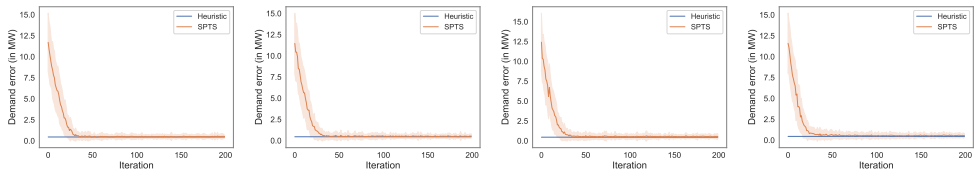
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.5: Demand error for $d = 0°$ and $\mathrm{P}_{\text{dem}} = 80$ MW. The average trend and standard deviation are plotted.



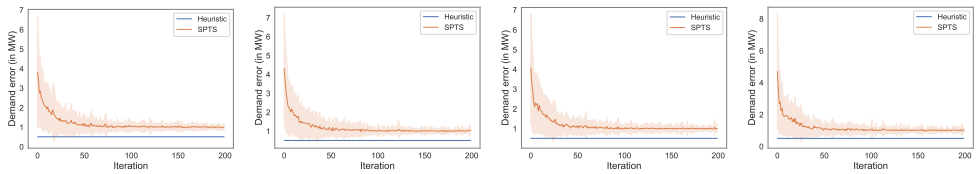
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.6: Demand error for $d = 0°$ and $\mathrm{P}_{\text{dem}} = 90$ MW. The average trend and standard deviation are plotted.



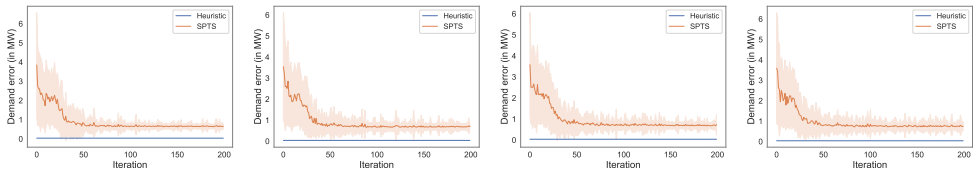
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.7: Demand error for $d = 0°$ and $\mathrm{P}_{\text{dem}} = 100$ MW. The average trend and standard deviation are plotted.



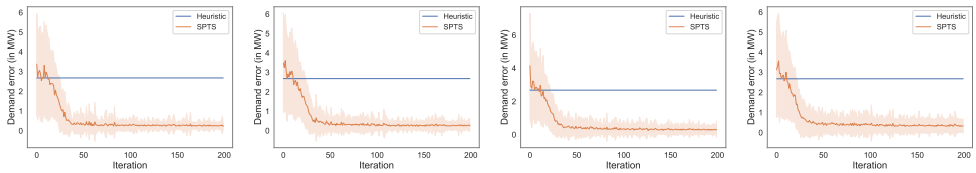
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.8: Demand error for $d = 30°$ and $\mathrm{P}_{\text{dem}} = 60$ MW. The average trend and standard deviation are plotted.



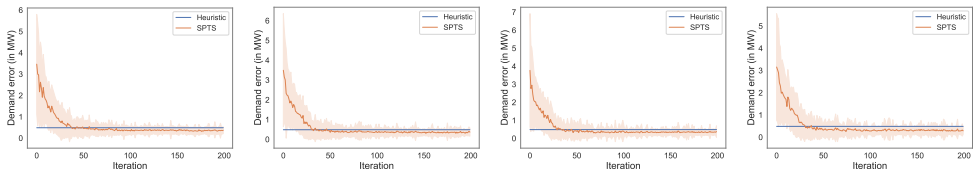
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.9: Demand error for $d = 30°$ and $\mathrm{P}_{\text{dem}} = 70$ MW. The average trend and standard deviation are plotted.



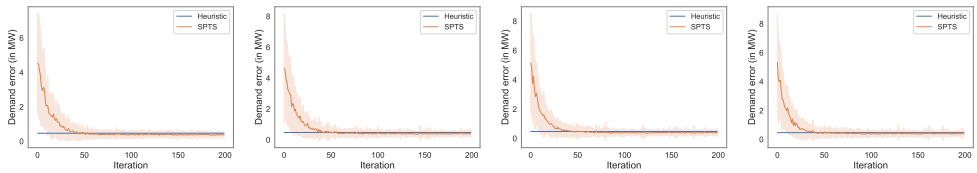
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.10: Demand error for $d = 30°$ and $\mathrm{P}_{\text{dem}} = 80$ MW. The average trend and standard deviation are plotted.



(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

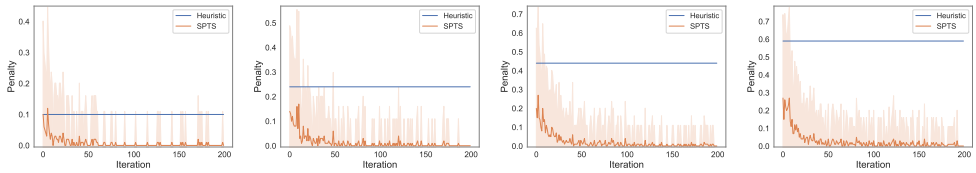
Figure A.11: Demand error for $d = 30°$ and $\mathrm{P}_{\text{dem}} = 90$ MW. The average trend and standard deviation are plotted.



(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

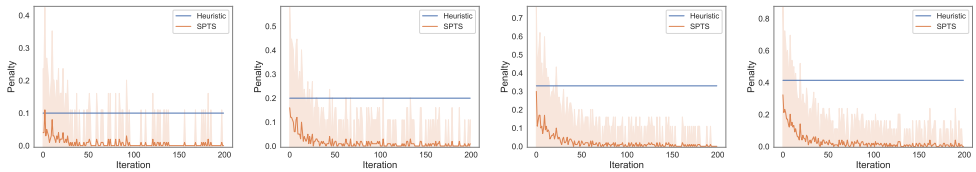
Figure A.12: Demand error for $d = 30°$ and $\mathrm{P}_{\text{dem}} = 100$ MW. The average trend and standard deviation are plotted.

Penalty



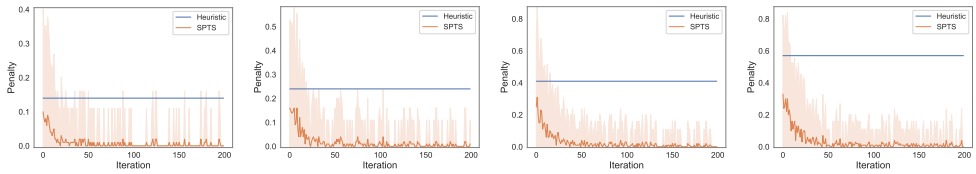
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.13: Penalty for $d = 0°$ and $\mathrm{P}_{\text{dem}} = 60$ MW. The average trend and standard deviation are plotted.



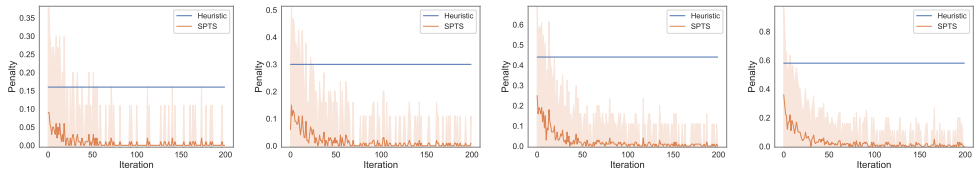
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.14: Penalty for $d = 0°$ and $\mathrm{P}_{\text{dem}} = 70$ MW. The average trend and standard deviation are plotted.



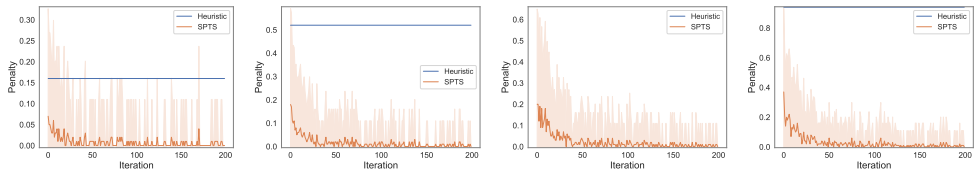
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.15: Penalty for $d = 0°$ and $\mathrm{P}_{\text{dem}} = 80$ MW. The average trend and standard deviation are plotted.



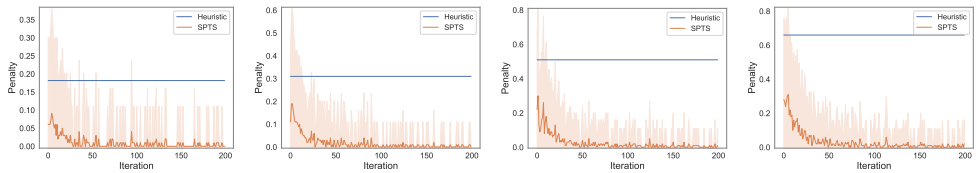
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.16: Penalty for $d = 0°$ and $\mathrm{P}_{\text{dem}} = 90$ MW. The average trend and standard deviation are plotted.



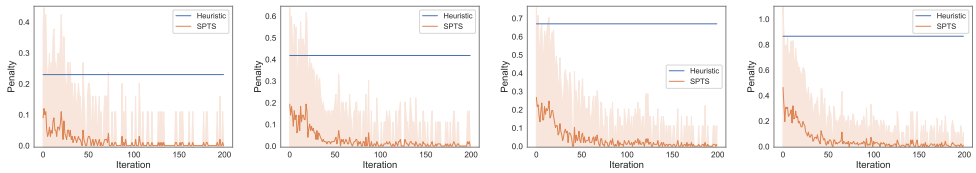
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.17: Penalty for $d = 0°$ and $\mathrm{P}_{\text{dem}} = 100$ MW. The average trend and standard deviation are plotted.



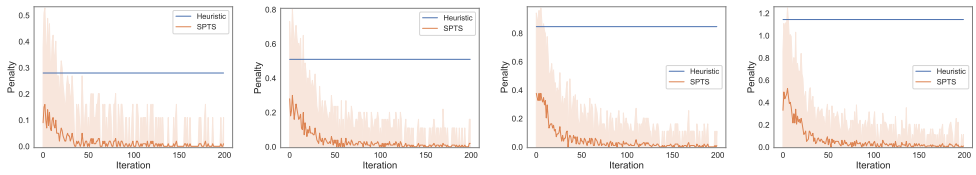
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.18: Penalty for $d = 30°$ and $\mathrm{P}_{\text{dem}} = 60$ MW. The average trend and standard deviation are plotted.



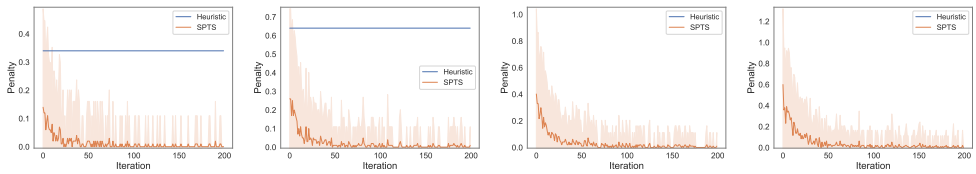
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.19: Penalty for $d = 30°$ and $\mathrm{P}_{\text{dem}} = 70$ MW. The average trend and standard deviation are plotted.



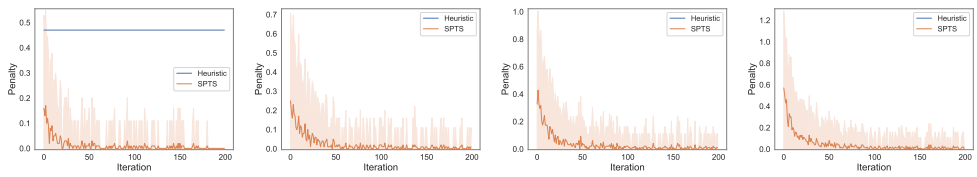
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.20: Penalty for $d = 30°$ and $\mathrm{P}_{\text{dem}} = 80$ MW. The average trend and standard deviation are plotted.



(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.21: Penalty for $d = 30°$ and $\mathrm{P}_{\text{dem}} = 90$ MW. The average trend and standard deviation are plotted.

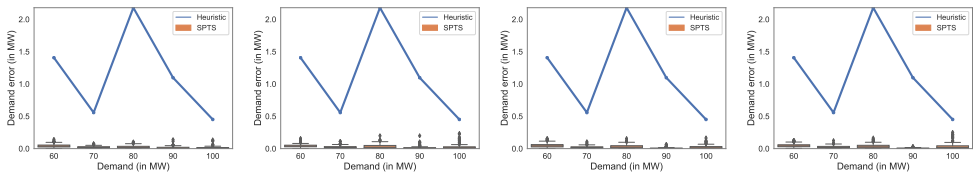


(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.22: Penalty for $d = 30°$ and $\mathrm{P}_{\text{dem}} = 100$ MW. The average trend and standard deviation are plotted.

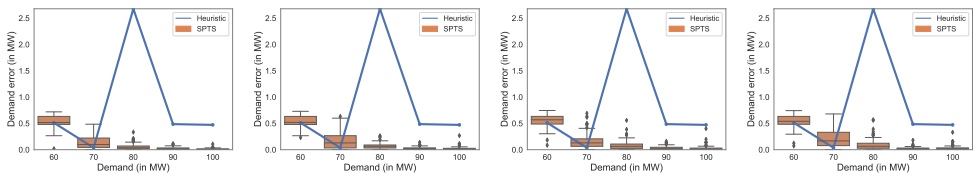
4.2 Best Set-Point Configurations

The best solutions found by SPTS have no penalty. Therefore, we only show the total penalty for the best solutions obtained by the heuristic approach.



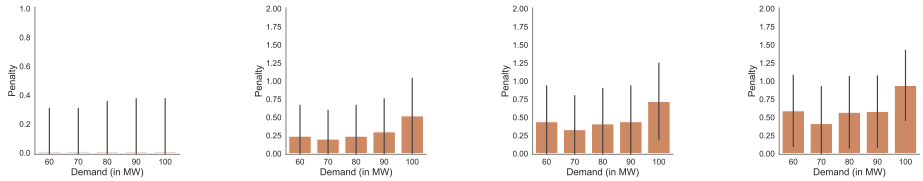
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.23: Demand error of best solution for $d = 0°$.



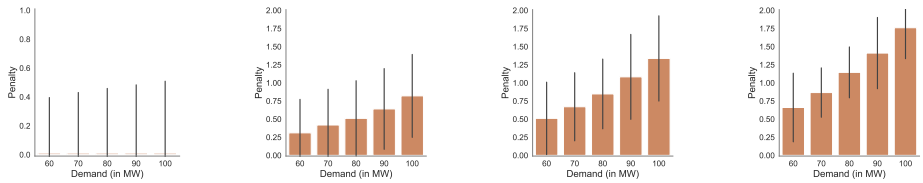
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.24: Demand error of best solution for $d = 30°$.



(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.25: Penalty of best solution found by the heuristic approach for $d = 0°$.



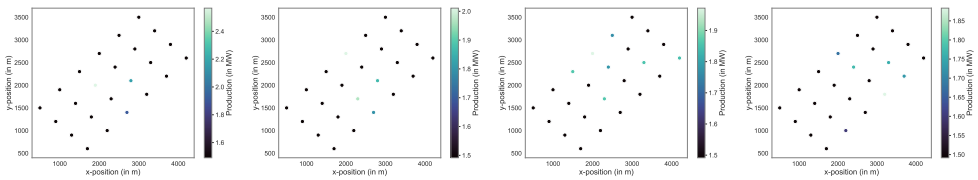
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.26: Penalty of best solution found by the heuristic approach for $d = 30°$.

4.3 Farm-Wide Power Productions

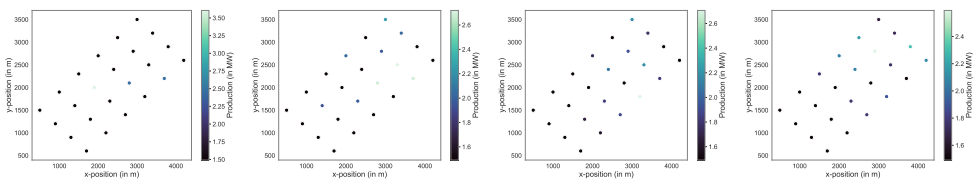
As the heuristic approach does not consider penalties, the farm-wide power productions are identical over a different number of penalized turbines for the same wind direction and power demand.

Set-Point Thompson Sampling



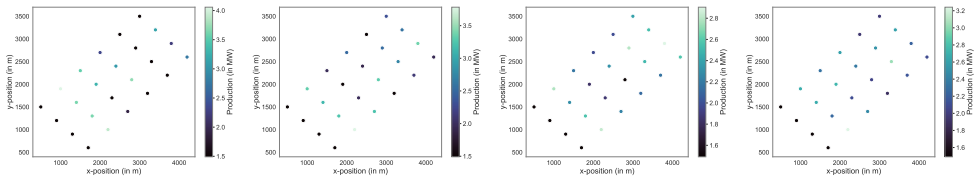
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.27: Average farm-wide power production of the best solution achieved by SPTS for $d = 0°$ and demand $\mathrm{P}_{\text{dem}} = 60$ MW.



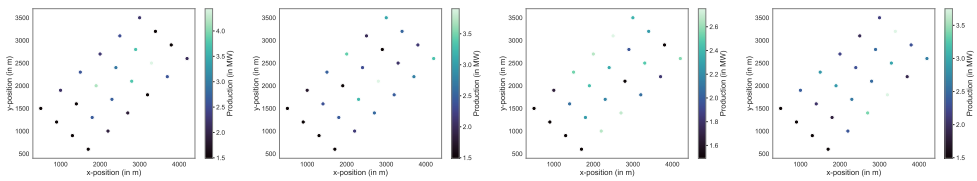
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.28: Average farm-wide power production of the best solution achieved by SPTS for $d = 0°$ and demand $\mathrm{P}_{\text{dem}} = 70$ MW.



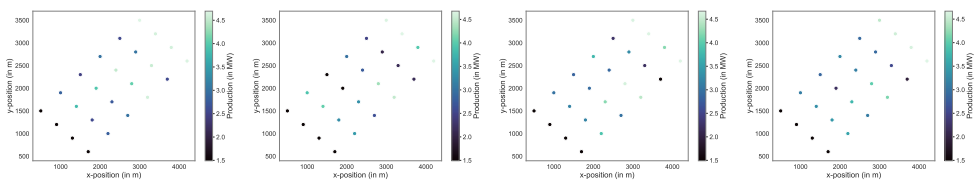
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.29: Average farm-wide power production of the best solution achieved by SPTS for $d = 0°$ and demand $\mathrm{P}_{\text{dem}} = 80$ MW.



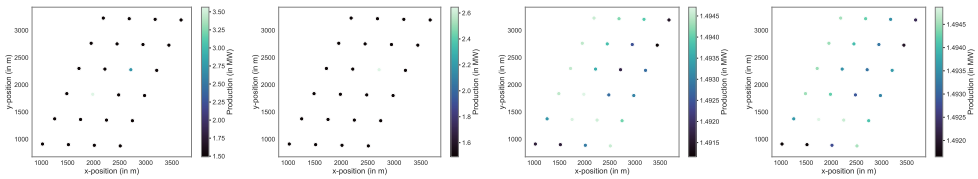
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.30: Average farm-wide power production of the best solution achieved by SPTS for $d = 0°$ and demand $\mathrm{P}_{\text{dem}} = 90$ MW.



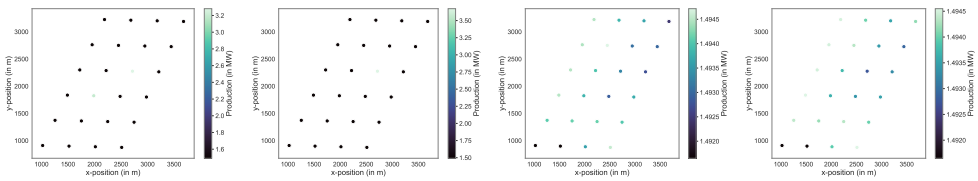
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.31: Average farm-wide power production of the best solution achieved by SPTS for $d = 0°$ and demand $\mathrm{P}_{\text{dem}} = 100$ MW.



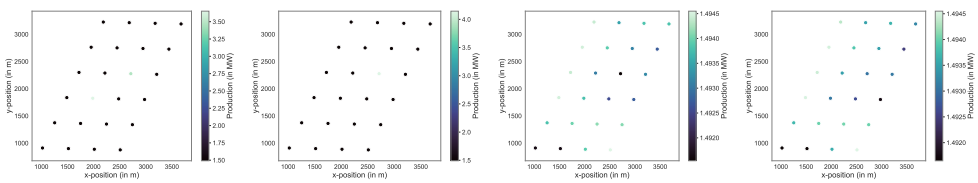
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.32: Average farm-wide power production of the best solution achieved by SPTS for $d = 30°$ and demand $\mathrm{P}_{\text{dem}} = 60$ MW.



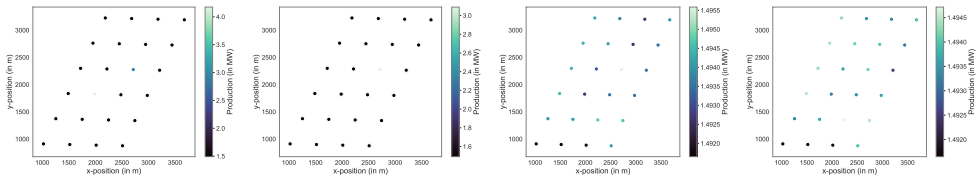
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.33: Average farm-wide power production of the best solution achieved by SPTS for $d = 30°$ and demand $\mathrm{P}_{\text{dem}} = 70$ MW.



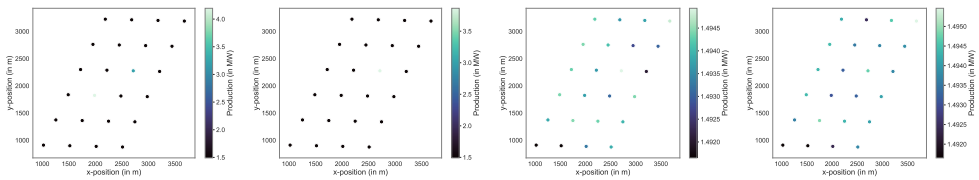
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.34: Average farm-wide power production of the best solution achieved by SPTS for $d = 30°$ and demand $\mathrm{P}_{\text{dem}} = 80$ MW.



(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

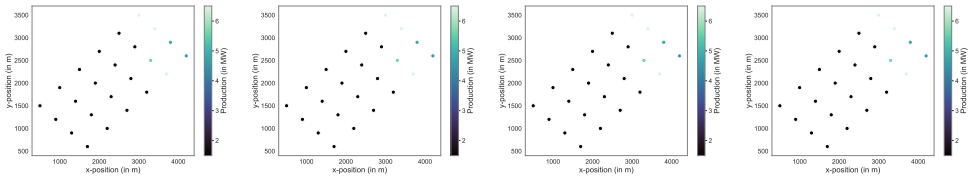
Figure A.35: Average farm-wide power production of the best solution achieved by SPTS for $d = 30°$ and demand $\mathrm{P}_{\text{dem}} = 90$ MW.



(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

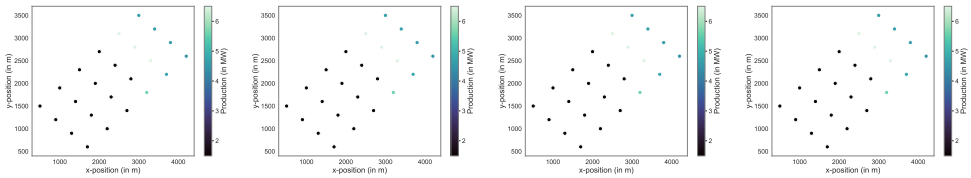
Figure A.36: Average farm-wide power production of the best solution achieved by SPTS for $d = 30°$ and demand $\mathrm{P}_{\text{dem}} = 100$ MW.

Heuristic Approach



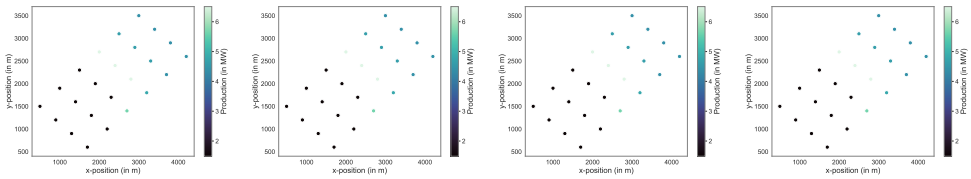
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.37: Average farm-wide power production of the best solution achieved by the heuristic approach for $d = 0°$ and demand $\mathrm{P}_{\text{dem}} = 60$ MW.



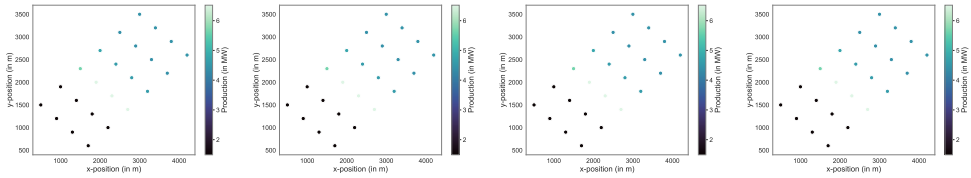
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.38: Average farm-wide power production of the best solution achieved by the heuristic approach for $d = 0°$ and demand $\mathrm{P}_{\text{dem}} = 70$ MW.



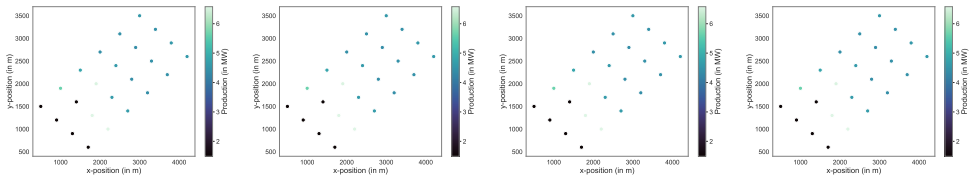
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.39: Average farm-wide power production of the best solution achieved by the heuristic approach for $d = 0°$ and demand $\mathrm{P}_{\text{dem}} = 80$ MW.



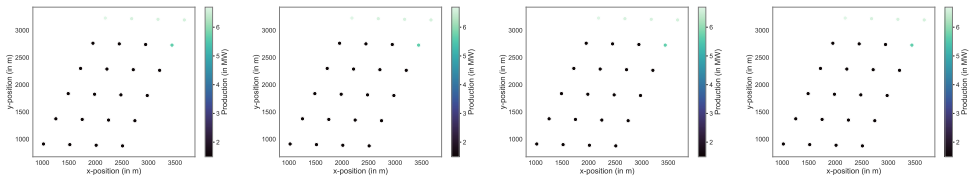
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.40: Average farm-wide power production of the best solution achieved by the heuristic approach for $d = 0°$ and demand $\mathrm{P}_{\text{dem}} = 90$ MW.



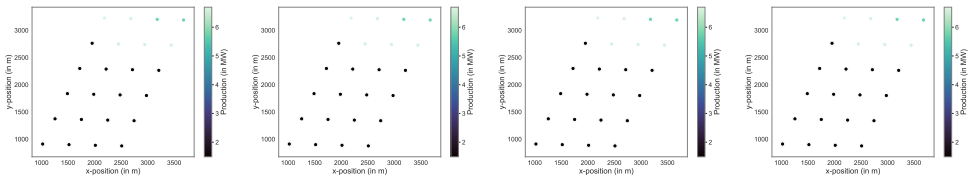
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.41: Average farm-wide power production of the best solution achieved by the heuristic approach for $d = 0°$ and demand $\mathrm{P}_{\text{dem}} = 100$ MW.



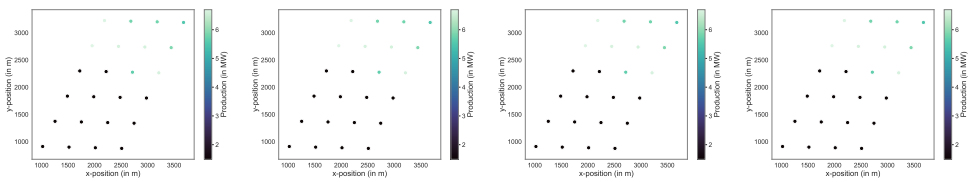
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.42: Average farm-wide power production of the best solution achieved by the heuristic approach for $d = 30°$ and demand $\mathrm{P}_{\text{dem}} = 60$ MW.



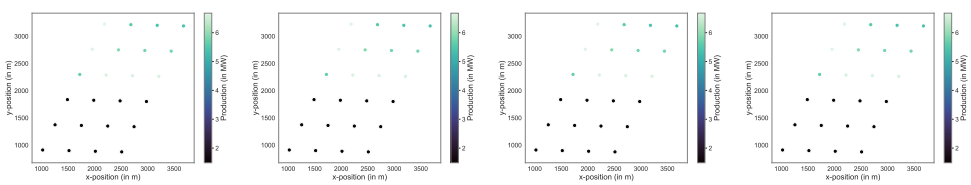
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.43: Average farm-wide power production of the best solution achieved by the heuristic approach for $d = 30°$ and demand $\mathrm{P}_{\text{dem}} = 70$ MW.



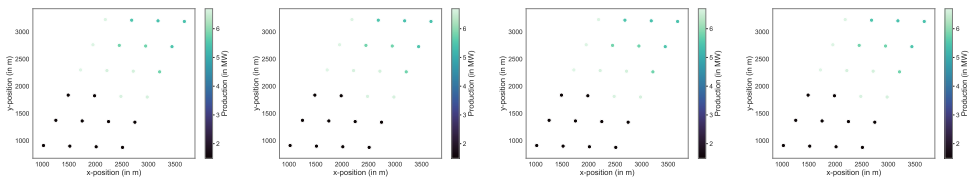
(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.44: Average farm-wide power production of the best solution achieved by the heuristic approach for $d = 30°$ and demand $\mathrm{P}_{\text{dem}} = 80$ MW.



(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.45: Average farm-wide power production of the best solution achieved by the heuristic approach for $d = 30°$ and demand $\mathrm{P}_{\text{dem}} = 90$ MW.



(a) $n_{\text{risk}} = 1$ (b) $n_{\text{risk}} = 2$ (c) $n_{\text{risk}} = 3$ (d) $n_{\text{risk}} = 4$

Figure A.46: Average farm-wide power production of the best solution achieved by the heuristic approach for $d = 30°$ and demand $\mathrm{P}_{\text{dem}} = 100$ MW.

Curriculum Vitae

Personal information

Timothy VERSTRAETEN, male, born in Lier, Belgium (07/01/1992)

Education

Master in Computer Science

> Department of computer science, Vrije Universiteit Brussel
> Graduated *summa cum laude* in 2015
> Master thesis: "Modeling Exoskeleton-Assisted Human Motion with Gaussian Processes"

Academic Bachelor in Informatics

> Department of computer science, University of Antwerp
> Graduated *magna cum laude* in 2013
> Bachelor thesis: Developing a graphical interface and cell division algorithm for a plant-cell growth simulator (VirtualLeaf)

Professional history

2017–2021:

PhD student at the Artificial Intelligence Lab
Doctoral strategic basic research grant awarded by the national science foundation (FWO)
Vrije Universiteit Brussel, Department of computer science

2016–2017:

PhD student at the Artificial Intelligence Lab
Doctoral research grant awarded by the VUB research council
Vrije Universiteit Brussel, Department of computer science

2015–2016:

Teaching assistant at the Artificial Intelligence Lab
Vrije Universiteit Brussel, Department of computer science

Summers of 2013 and 2014:

Job student at the Computational Modeling and Programming research group
University of Antwerp, Department of computer science

Awards & honors

- Visionary paper award, for the paper titled "Efficient evaluation of influenza mitigation strategies using preventive bandits", by the Adaptive Learning Agents workshop, 29/03/2017
- Best student paper award, for the paper titled "IPC-Net: 3D Point-Cloud Segmentation Using Deep Inter-Point Convolutional Layers", International Conference on Tools with Artificial Intelligence, 07/11/2018
- Top reviewer certificate of appreciation at ICML 2020 (top 33%), 15/09/2020

Grants

- Doctoral strategic basic research grant, by the national science foundation (FWO), 01/01/2017–31/12/2020
- Doctoral research grant, by the VUB research council, 01/01/2016–31/12/2017
- Grant for participation in a conference abroad (AAMAS conference, May, 2020), by the National science foundation (FWO) [conditionally accepted, but canceled due to the COVID-19 pandemic]

Teaching experience

- Teaching assistant for the course "Artificial Intelligence" (Prof. Dr. Bernard Manderick), during two academic years (2015–2017)
- Teaching assistant for the course "Theory of Computation" (Prof. Dr. Bernard Manderick), during one academic year (2015–2016)
- Teaching assistant for the course "Wetenschappelijk Rekenen" (Prof. Dr. Ann Nowé, Prof. Dr. Bernard Manderick), during five academic years (2015–2020)

Master students

- Faras Jamil, Vrije Universiteit Brussel, 2019–2020, with a project titled "Fault detection from sensor data using machine learning techniques" (promotors: Ann Nowé & Jan Helsen)
- Domien Hennion, Vrije Universiteit Brussel, 2019–2020, with a project titled "Safe fleet-wide policy iteration for fleet applications." (promotor: Ann Nowé)
- Regis Loeb, Vrije Universiteit Brussel, 2018–2019, with a project titled "Privacy Preserving Reinforcement Learning over Distributed Datasets" (promotors: Ann Dooms & Ann Nowé)
- Thomas Cloostermans, Vrije Universiteit Brussel, 2018–2020, with a project titled "Bayesian optimization for conflict resolution in air traffic control" (joint advisorship with Pieter Libin, promotors: Ann Nowé)

Bachelor students

- Senne Deproost, Vrije Universiteit Brussel, 2017–2018, with a project titled "Yaw optimization using reinforcement learning" (promotor: Ann Nowé)
- Tom Lauwers, Vrije Universiteit Brussel, 2017–2019, with a project titled "Explainable random forests" (joint advisorship with Isel Grau, promotor: Ann Nowé)

Academic service

Invited speaker at the Gaussian process seminar, with a presentation titled "Fleet-Wide Policy Iteration using Gaussian Processes", 26/02/2020, University of Antwerp, Belgium

Lecturer at the ACAI summer school on reinforcement learning, with a presentation titled "The Basics of Reinforcement Learning", 06/10/2017, Nieuwpoort, Belgium

Assistant at the ACAI summer school on reinforcement learning, 06/10/2017–13/10/2017, Nieuwpoort, Belgium
Program committee member: ICML 2020, ALA workshop at AAMAS 2017–2021
Reviewer: ICML 2019–2020, NeurIPS 2019, The Knowledge Engineering Review, ALA workshop at AAMAS 2017–2021

Journal publications (peer-reviewed)

1. Pieter J.K. Libin, Lander Willem, **Timothy Verstraeten**, Andrea Torneri, Joris Vanderlocht, Niel Hens. "Assessing the feasibility and effectiveness of household-pooled universal testing to control COVID-19 epidemics". PLOS Computational Biology, 2021. (In press) [Peer reviewed, 2-yearly JCR impact factor 2019: 4.38]

2. Pieter-Jan Daems, **Timothy Verstraeten**, Cédric Peeters, Jan Helsen. "Effects of wake on gearbox design load cases identified from fleet-wide operational data", Forschung im Ingenieurwesen, Springer Nature, 2021. (In press) [Peer reviewed, 2-yearly JCR impact factor 2019: 0.766]

3. **Timothy Verstraeten**, Eugenio Bargiacchi, Pieter J.K. Libin, Jan Helsen, Diederik M. Roijers, Ann Nowé. "Multi-agent Thompson sampling for bandit applications with sparse neighbourhood structures". *Nature Scientific Reports*, 10(1):1–13, 2020. [Peer reviewed, 2-yearly JCR impact factor 2019: 3.998]

4. **Timothy Verstraeten**, Ann Nowé, Jonathan Keller, Yi Guo, Shuangwen Sheng, Jan Helsen. "Fleetwide data-enabled reliability improvement of wind turbines". *Renewable and Sustainable Energy Reviews*, 109:428–437, 2019. [Peer reviewed, 2-yearly JCR impact factor 2019: 12.110]

Conference proceedings (peer-reviewed)

1. **Timothy Verstraeten**, Pieter-Jan Daems, Eugenio Bargiacchi, Diederik M. Roijers, Pieter J.K. Libin, Jan Helsen. "Scalable Optimization for Wind Farm Control using Coordination Graphs". Proceedings of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2021. (In press)

2. Eugenio Bargiacchi, **Timothy Verstraeten**, Diederik M. Roijers. "Model-based Multi-Agent Reinforcement Learning with Cooperative Prioritized Sweeping". Proceedings of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2021. (In press)

3. **Timothy Verstraeten**, Pieter J.K. Libin, Ann Nowé. "Fleet control using coregionalized Gaussian process policy iteration". *Proc. of the 24th European Conference on Artificial Intelligence*, 2020.

4. **Timothy Verstraeten**, Eugenio Bargiacchi, Pieter J.K. Libin, Diederik M. Roijers, Ann Nowé, "Thompson sampling for factored multi-agent bandits". *Proc. of the International Conference on Autonomous Agents and Multiagent Systems*, pp. 2029–2031. 2020.

5. Pieter-Jan Daems, Len Feremans, **Timothy Verstraeten**, Boris Cule, Bart Goethals, Jan Helsen. "Fleet-oriented pattern mining combined with time series signature extraction for understanding of wind farm response to storm conditions". *Proc. of World Conference for Condition Monitoring*, pp. 275–287. 2020.

6. Nicoletta Gioia, Pieter-Jan Daems, **Timothy Verstraeten**, Patrick Guillaume, Jan Helsen. "Combining Machine Learning and Operational Modal Analysis Approaches to Gain Insights in Wind Turbine Drivetrain Dynamics". *Proc. of the Society for Experimental Mechanics Series*. 2020.

7. Pieter J.K. Libin, Arno Moonens, **Timothy Verstraeten**, Fabian Perez-Sanjines, Niel Hens, Philippe Lemey, Ann Nowé. "Deep reinforcement learning for large-scale epidemic control". *Proc. of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2020

8. Pieter Libin, **Timothy Verstraeten**, Diederik M. Roijers, Wenjia Wang, Kristof Theys, Ann Nowé. "Bayesian anytime m-top exploration". *Proc. of the 31st IEEE International Conference on Tools with Artificial Intelligence*, pp. 1414–1420, 2019.

9. Cédric Peeters, **Timothy Verstraeten**, Ann Nowé, Jan Helsen. "Wind Turbine Planetary Gear Fault Identification Using Statistical Condition Indicators and Machine Learning". *Proc. of the ASME 2019 38th International Conference on Offshore Mechanics and Arctic Engineering*. 2019.

10. Cédric Peeters, Nicoletta Gioia, Pieter-Jan Daems, Jonas Verbeke, **Timothy Verstraeten**, Ann Nowé and Jan Helsen. "Drivetrain reliability improvements from long-term field data processed in the cloud". *Journal of Physics: Conference Series*, 1222, WindEurope Conference and Exhibition, 2019.

11. Cédric Peeters, **Timothy Verstraeten**, Ann Nowé, Pieter-Jan Daems, Jan Helsen. "Advanced Vibration Signal Processing Using Edge Computing to Monitor Wind Turbine Drivetrains". *Proc. of the ASME 2019 2nd International Offshore Wind Technical Conference*. 2019.

12. Cédric Peeters, Pieter-Jan Daems, **Timothy Verstraeten**, Ann Nowé, Jan Helsen. "Combining Edge and Cloud Computing for Monitoring a Fleet of Wind Turbine Drivetrains Using Combined Machine Learning Signal Processing Approaches". *Proc. of the 12th International Workshop on Structural Health Monitoring*. 2019. [workshop paper]

13. Felipe Gomez Marulanda, Pieter J.K. Libin, **Timothy Verstraeten**, Ann Nowé. "Deep hybrid approach for 3D plane segmentation". *Proc. of the 27th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, vol. 27, pp. 529–534, 2019.

14. Eugenio Bargiacchi, **Timothy Verstraeten**, Diederik M. Roijers, Ann Nowé, Hado van Hasselt. "Learning to coordinate with coordination graphs in repeated single-stage multi-agent decision problems". *Proc. of the 35th International Conference on Machine Learning*, pp. 482–490, 2018

15. Pieter J.K. Libin, **Timothy Verstraeten**, Diederik M. Roijers, Jelena Grujic, Kristof Theys, Philippe Lemey, Ann Nowé. "Bayesian best-arm identification for selecting influenza mitigation strategies". *Proc. of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 456–471, 2018.

16. Felipe Gomez Marulanda, Pieter Libin*, **Timothy Verstraeten***, Ann Nowé. "IPC-Net: 3D point-cloud segmentation using deep inter-point convolutional layer". *Proc. of the 30th IEEE International Conference on Tools with Artificial Intelligence*, pp. 293–301, 2018. (* denotes equal contribution)

17. Pieter J.K. Libin, **Timothy Verstraeten**, Kristof Theys, Diederik M. Roijers, Peter Vrancx, Ann Nowé, "Efficient evaluation of influenza mitigation strategies using preventive bandits". *Autonomous Agents and Multiagent Systems*, pp. 67–85. 2017.

Conference presentations

1. **Timothy Verstraeten**, Eugenio Bargiacchi, Pieter J.K. Libin, Jan Helsen, Diederik M. Roijers, Ann Nowé. "Multi-agent Thompson sampling for bandits with sparse neighborhood structures". Benelux Conference on Artificial Intelligence and the Belgian Dutch Conference on Machine Learning, 19/11/2020–20/11/2020, virtual event. Oral presentation.

2. Pieter J.K. Libin, Arno Moonens, **Timothy Verstraeten**, Fabian Perez-Sanjines, Niel Hens, Philippe Lemey, Ann Nowé. "Deep reinforcement learning for large-scale

epidemic control". Benelux Conference on Artificial Intelligence and the Belgian Dutch Conference on Machine Learning, 19/11/2020–20/11/2020, virtual event. Oral presentation.

3. **Timothy Verstraeten**, Eugenio Bargiacchi, Pieter J.K. Libin, Jan Helsen, Diederik M. Roijers, Ann Nowé. "Thompson sampling for loosely-coupled multi-agent systems: An application to wind farm control". Adaptive Learning Agents (ALA) workshop @ AAMAS, 09/05/2020–10/05/2020, Auckland, New-Zealand. Oral presentation. [workshop paper, peer reviewed]

4. Pieter J.K. Libin, Arno Moonens, **Timothy Verstraeten**, Fabian Perez-Sanjines, Niel Hens, Philippe Lemey, Ann Nowé. "Deep reinforcement learning for large-scale epidemic control". Adaptive Learning Agents (ALA) workshop @ AAMAS, 09/05/2020–10/05/2020, Auckland, New-Zealand. Oral presentation. [workshop paper, peer reviewed]

5. Timothy Verstraeten, Ann Nowé, Jan Helsen. "Failure avoidance for wind turbines through fleetwide control". Benelux Conference on Artificial Intelligence and the Belgian Dutch Conference on Machine Learning, 06/11/2019–08/11/2019, Brussels, Belgium. Poster presentation.

6. **Timothy Verstraeten**, Eugenio Bargiacchi, Pieter J.K. Libin, Diederik M. Roijers and Ann Nowé. "Multi-Agent Thompson Sampling". Multi Armed Bandit Workshop @ Imperial College London, 25/09/2019–26/09/2019, London, United Kingdom. Poster presentation.

7. Regis Loeb, **Timothy Verstraeten**, Ann Nowé, Ann Dooms. "Privacy Preserving Reinforcement Learning over Distributed Datasets". Benelux Artificial Intelligence Conference, 06/11/2019–08/11/2019, Brussels, Belgium. Oral presentation. [conference abstract]

8. Pieter J.K. Libin, **Timothy Verstraeten**, Diederik M. Roijers, Wenjia Wang, Kristof Theys, Ann Nowé. "Thompson sampling for m-top exploration". Benelux Conference on Artificial Intelligence, 06/11/2019–08/11/2019, Brussels, Belgium. Oral presentation.

9. **Timothy Verstraeten**, Felipe Gomez Marulanda, Cédric Peeters, Pieter-Jan Daems, Ann Nowé, Jan Helsen. "Edge computing for advanced vibration signal processing". Surveillance, Vishno and AVE conferences, 08/07/2019–10/07/2019, Lyon, France. Oral presentation.

10. Pieter J.K. Libin, **Timothy Verstraeten**, Diederik M. Roijers, Wenjia Wang, Kristof Theys, Ann Nowé. "Boundary Focused Thompson Sampling". Adaptive Learning Agents (ALA) workshop @ AAMAS, 13/05/2019–14/05/2019, Montreal, Canada. Oral presentation.

11. **Timothy Verstraeten**, Ann Nowé. "Reinforcement learning for fleet applications using coregionalized Gaussian processes". Adaptive Learning Agents (ALA) workshop @ AAMAS, 14/07/2018–15/07/2018, Stockholm, Sweden. Oral presentation.

12. Pieter J.K. Libin, **Timothy Verstraeten**, Kristof Theys, Diederik M. Roijers, Peter Vrancx, Ann Nowé. "Efficient evaluation of influenza mitigation strategies using preventive bandits". Adaptive Learning Agents (ALA) workshop @ AAMAS, 08/05/2017–09/05/2017, São Paolo, Brazil. Oral presentation.

13. Jan Helsen, Cédric Peeters, **Timothy Verstraeten**, Nicoletta Gioia, Ann Nowé. "Fleet-wide condition monitoring combining vibration signal processing and machine learning rolled out in a cloud-computing environment". International Conference on Noise and Vibration Engineering, 17/09/2018–19/09/2018, 6 pages, Leuven, Belgium. Oral presentation.

14. **Timothy Verstraeten**, Peter Vrancx, Ann Nowé. "Fleet reinforcement learning using dependent Gaussian processes". Belgium-Netherlands Reinforcement Learning (BENERL) workshop, 26/01/2017, Nijmegen, the Netherlands. Oral presentation. [workshop paper, peer reviewed]

15. **Timothy Verstraeten**, Peter Vrancx, Ann Nowé. "Fleet reinforcement learning using dependent Gaussian processes". Learning, Inference and Control of Multi-Agent Systems (MALIC) workshop @ NIPS, 08/12/2016, Barcelona, Spain. Poster presentation. [workshop paper, peer reviewed]

16. **Timothy Verstraeten**, Roxana Rădulescu, Yannick Jadoul, Tom Jaspers, Robrecht Conjaerts, Tim Brys, Anna Harutyunyan, Peter Vrancx, Ann Nowé. "Human guided ensemble learning in StarCraft". Adaptive Learning Agents (ALA) workshop @ AAMAS, 09/05/2016–10/05/2016, Singapore. Oral presentation. [workshop paper, peer reviewed]

17. **Timothy Verstraeten**. "Modeling Exoskeleton-Assisted Human Motion with Gaussian Processes". Benelux Conference on Artificial Intelligence, 05/11/2015–06/11/2015, Hasselt, Belgium. Poster presentation.

Bibliography

Agrawal, S. and N. Goyal
2012. Analysis of Thompson sampling for the multi-armed bandit problem. In *Proceedings of the 25th Annual Conference on Learning Theory (COLT)*, volume 23, Pp. 39.1–39.26.

Agrawal, S. and N. Goyal
2013a. Further optimal regret bounds for Thompson sampling. In *Proceedings of the 16th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 31, Pp. 99–107.

Agrawal, S. and N. Goyal
2013b. Thompson sampling for contextual bandits with linear payoffs. In *International Conference on Machine Learning (ICML)*, Pp. 127–135.

Ahmed, S. D., F. S. Al-Ismail, M. Shafiullah, F. A. Al-Sulaiman, and I. M. El-Amin
2020. Grid integration challenges of wind energy: A review. *IEEE Access*, 8:10857–10878.

Aho, J., A. Buckspan, J. Laks, P. Fleming, Y. Jeong, F. Dunne, M. Churchfield, L. Pao, and K. Johnson
2012. A tutorial of wind turbine control for supporting grid frequency through active power control. In *American Control Conference (ACC)*, Pp. 3120–3131.

Alvarez, E. J. and A. P. Ribaric
2018. An improved-accuracy method for fatigue load analysis of wind turbine gearbox based on scada. *Renewable Energy*, 115:391–399.

Ansell, J. I. and M. J. Phillips
1994. *Practical methods for reliability data analysis*, 1 edition. Oxford, United Kingdom: Clarendon Press.

Audibert, J.-Y. and S. Bubeck
2010. Best arm identification in multi-armed bandits. In *Proc. of the 23rd Annual Conference on Learning Theory (COLT)*.

Audibert, J.-Y., S. B. Bubeck, and G. Lugosi
2011. Minimax policies for combinatorial prediction games. In *Proceedings of the 24th Annual Conference on Learning Theory (COLT)*, volume 19, Pp. 107–132.

Auer, P., N. Cesa-Bianchi, and P. Fischer
2002. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47:235–256.

Bargiacchi, E., D. M. Roijers, and A. Nowé
2020. AI-Toolbox: A C++ library for Reinforcement Learning and Planning (with Python Bindings). *Journal of Machine Learning Research*, 21(102):1–12.

Bargiacchi, E., T. Verstraeten, D. Roijers, A. Nowé, and H. van Hasselt
2018. Learning to coordinate with coordination graphs in repeated single-stage multi-agent decision problems. In *International Conference on Machine Learning (ICML)*, Pp. 491–499.

Barto, A. G., R. S. Sutton, and C. W. Anderson
1983. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE transactions on systems, man, and cybernetics*, 5:834–846.

Blei, D. M., M. I. Jordan, et al.
2006. Variational inference for Dirichlet process mixtures. *Bayesian analysis*, 1(1):121–143.

Boersma, S., B. Doekemeijer, P. M. Gebraad, P. A. Fleming, J. Annoni, A. K. Scholbrock, J. Frederik, and J.-W. van Wingerden
2017. A tutorial on control-oriented modeling and control of wind farms. In *2017 American Control Conference (ACC)*, Pp. 1–18. IEEE.

Bonilla, E. V., K. M. A. Chai, and C. K. I. Williams
2008. Multi-Task Gaussian Process Prediction. *Advances in Neural Information Processing Systems*, 20:153–160.

Boutilier, C.
1996. Planning, learning and coordination in multiagent decision processes. In *TARK 1996: Proceedings of the 6th conference on Theoretical aspects of rationality and knowledge*, Pp. 195–210.

Boyer, S. A.
2009. *SCADA: Supervisory control and data acquisition*. International Society of Automation.

Brochu, E., V. M. Cora, and N. De Freitas
2010. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599*.

Brockman, G., V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba
2016. Openai gym. *arXiv preprint arXiv:1606.01540*.

Bruce, T., H. Long, and R. S. Dwyer-Joyce
2015. Dynamic modelling of wind turbine gearbox bearing loading during transient events. *IET Renewable Power Generation*, 9(7):821–830.

Brys, T., Y.-M. De Hauwere, A. Nowé, and P. Vrancx
2011. Local coordination in online distributed constraint optimization problems. In *European Workshop on Multi-Agent Systems*, Pp. 31–47. Springer.

Bubeck, S. and N. Cesa-Bianchi
2012. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends in Machine Learning*, 5(1):1–122.

Busoniu, L., R. Babuska, and B. De Schutter
2006. Multi-agent reinforcement learning: A survey. In *The 9th International Conference on Control, Automation, Robotics and Vision*, Pp. 1–6.

Castellani, F., A. Gravdahl, G. Crasto, E. Piccioni, and A. Vignaroli
2013. A practical approach in the cfd simulation of off-shore wind farms through the actuator disc technique. *Energy Procedia*, 35:274–284.

Cesa-Bianchi, N. and G. Lugosi
2012. Combinatorial bandits. *Journal of Computer and System Sciences*, 78(5):1404–1422.

Chapelle, O. and L. Li
2011. An empirical evaluation of Thompson sampling. In *Advances in Neural Information Processing Systems (NIPS)*, volume 24, Pp. 2249–2257.

Chapman, A. C., D. S. Leslie, A. Rogers, and N. R. Jennings
2013. Convergent learning algorithms for unknown reward games. *SIAM Journal on Control and Optimization*, 51(4):3154–3180.

Chen, N., Z. Qian, I. T. Nabney, and X. Meng
2013a. Wind power forecasts using Gaussian processes and numerical weather prediction. *IEEE Transactions on Power Systems*, 29(2):656–665.

Chen, W., Y. Wang, and Y. Yuan
2013b. Combinatorial multi-armed bandit: General framework, results and applications. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, volume 28, Pp. 151–159.

Christian Steiness/Vattenfall/Flickr
2010. Horns rev offshore wind farm. [Online; accessed November 15, 2020; license CC BY-ND 2.0].

Claes, D., F. Oliehoek, H. Baier, and K. Tuyls
2017. Decentralised online planning for multi-robot warehouse commissioning. In *Proceedings of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, Pp. 492–500. International Foundation for Autonomous Agents and Multiagent Systems.

Clark, C. E. and B. DuPont
2018. Reliability-based design optimization in offshore renewable energy systems. *Renewable and Sustainable Energy Reviews*, 97:390–400.

De Hauwere, Y.-M.
2011. *Sparse interactions in multi-agent reinforcement learning*. PhD thesis, Vrije Universiteit Brussel.

De Hauwere, Y.-M., P. Vrancx, and A. Nowé
2010. Learning multi-agent state space representations. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, Pp. 715–722. International Foundation for Autonomous Agents and Multiagent Systems.

Deisenroth, M. P., P. Englert, J. Peters, and D. Fox
2014. Multi-task policy search for robotics. In *IEEE International Conference on Robotics and Automation (ICRA)*, Pp. 3876–3881. IEEE.

Deisenroth, M. P. and C. E. Rasmussen
2011. PILCO: A model-based and data-efficient approach to policy search. In *Proc. of the 28th International Conference on Machine Learning (ICML)*, Pp. 465–472.

Desmond, C., J. Murphy, L. Blonk, and W. Haans
2016. Description of an 8 MW reference wind turbine. *Journal of Physics: Conference Series*, 753(9).

Do, T. H., E. Tsiligianni, X. Qin, J. Hofman, V. P. La Manna, W. Philips, and N. Deligiannis
2020. Graph-deep-learning-based inference of fine-grained air quality from mobile IoT sensors. *IEEE Internet of Things Journal*, 7(9):8943–8955.

Doshi-Velez, F. and G. Konidaris
2016. Hidden parameter markov decision processes: A semiparametric regression approach for discovering latent task parametrizations. In *Proc. of the 25th International Joint Conference on Artificial Intelligence (IJCAI)*, Pp. 1432–1440.

Engel, Y., S. Mannor, and R. Meir
2005. Reinforcement Learning with Gaussian Processes. In *Proc. of the 22nd International Conference on Machine Learning*, Pp. 201–208. ACM Press.

Feng, J. and W. Z. Shen
2017. Design optimization of offshore wind farms with multiple types of wind turbines. *Applied Energy*, 205:1283–1297.

Fraile, D., A. Mbistrova, I. Pineda, P. Tardieu, and L. Miró
2018. Wind in power 2017: Annual combined onshore and offshore wind energy statistics. Technical report, Wind Europe. accessed 6 March 2019.

Gai, Y., B. Krishnamachari, and R. Jain
2012. Combinatorial network optimization with unknown variables: Multi-armed bandits with linear rewards and individual observations. *IEEE/ACM Transactions on Networking (TON)*, 20(5):1466–1478.

Gebraad, P., J. J. Thomas, A. Ning, P. Fleming, and K. Dykes
2017. Maximization of the annual energy production of wind power plants by optimization of layout and yaw-based wake control. *Wind Energy*, 20(1):97–107.

Gebraad, P. M. and J.-W. van Wingerden
2015. Maximum power-point tracking control for wind farms. *Wind Energy*, 18(3):429–447.

Gerla, M., E.-K. Lee, G. Pau, and U. Lee
2014. Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds. In *2014 IEEE world forum on internet of things (WF-IoT)*, Pp. 241–246. IEEE.

Glacer, A.
2012. From Brokdorf to Fukushima: The long journey to nuclear phase-out. *Bulletin of the Atomic Scientists*, 68(6):10–21.

Global Wind Energy Council
2018. Global wind statistics 2017. accessed 6 March 2019.

Gonzalez, E., B. Stephen, D. Infield, and J. J. Melero
2017. On the use of high-frequency SCADA data for improved wind turbine performance monitoring. *Journal of Physics: Conference Series*, 926.

González-Longatt, F., P. Wall, and V. Terzija
2012. Wake effect in wind farm performance: Steady-state and dynamic behavior. *Renewable Energy*, 39(1):329–338.

Goovaerts, P.
1997. *Geostatistics for Natural Resource Evaluation*. New York, USA: Oxford University Press.

Gould, B. and G. Aaron
2015. The influence of sliding and contact severity on the generation of white etching cracks. *Tribology Letters*, 60(2):1–13.

Greco, A., S. Sheng, J. Keller, and A. Erdemir
2013. Material wear and fatigue in wind turbine systems. *Wear*, 302(1-2):1583–1591.

Gu, H. and J. Wang
2013. Irregular-shape wind farm micro-siting optimization. *Energy*, 57:535–544.

Guestrin, C., D. Koller, and R. Parr
2001a. Max-norm projections for factored MDPs. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI)*, Pp. 673–682.

Guestrin, C., D. Koller, and R. Parr
2001b. Multiagent planning with factored MDPs. In *Advances in Neural Information Processing Systems (NIPS)*, volume 14, Pp. 1523–1530.

Guestrin, C., S. Venkataraman, and D. Koller
2002. Context-specific multiagent coordination and planning with factored mdps. In *Proceedings of the 18th National Conference on Artificial Intelligence (AAAI)*, Pp. 253–259.

Guo, Y., J. Keller, and W. LaCava
2015. Planetary gear load sharing of wind turbine drivetrains subjected to non-torque loads. *Wind Energy*, 18(4):757–768.

Helsen, J., C. Devriendt, W. Weijtjens, and P. Guillaume
2016a. Experimental dynamic identification of modeshape driving wind turbine grid loss event on nacelle testrig. *Renewable Energy*, 85:259–272.

Helsen, J., Y. Guo, and J. Keller
2018a. Gearbox high-speed-stage bearing slip induced by electric excitation in a test facility. *Wind Energy*, 21(11):1191–1201.

Helsen, J., Y. Guo, J. Keller, and P. Guillaume
2016b. Experimental investigation of bearing slip in a wind turbine gearbox during a transient grid loss event. *Wind Energy*, 19(12):2255–2269.

Helsen, J., C. Peeters, T. Verstraeten, J. Verbeke, N. Gioia, and A. Nowé
2018b. Fleet-wide condition monitoring combining vibration signal processing and machine learning rolled out in a cloud-computing environment. In *International Conference on Noise and Vibration Engineering (ISMA)*.

Helsen, J., P. Peeters, K. Vanslambrouck, F. Vanhollebeke, and W. Desmet
2014. The dynamic behavior induced by different wind turbine gearbox suspension methods assessed by means of the flexible multibody technique. *Renewable Energy*, 69:336–346.

Hennion, D.
2020. Safe fleet-wide policy iteration for fleet applications. Master's thesis, Vrije Universiteit Brussel.

Honda, J. and A. Takemura
2014. Optimality of Thompson sampling for Gaussian bandits depends on priors. In *Artificial Intelligence and Statistics*, Pp. 375–383.

Huang, Y.-F., X.-J. Gan, and P.-T. Chiueh
2017. Life cycle assessment and net energy analysis of offshore wind power systems. *Renewable Energy*, 102:98–106.

Ijspeert, A. J., J. Nakanishi, and S. Schaal
2003. Learning attractor landscapes for learning motor primitives. In *Advances in neural information processing systems*, Pp. 1547–1554.

International Electrotechnical Commission
2012. Wind turbines – Part 4: Design requirements for wind turbine gearboxes (No. IEC 61400-4). accessed 6 March 2019.

Irawan, C. A., D. Ouelhadj, D. Jones, M. Stålhane, and I. B. Sperstad
2017. Optimisation of maintenance routing and scheduling for offshore wind farms. *European Journal of Operational Research*, 256(1):76–89.

Jensen, T., T. Knudsen, and T. Bak
2016. Fatigue minimising power reference control of a de-rated wind farm. *Journal of Physics: Conference Series*, 753.

Johnson, K. E.
2004. Adaptive torque control of variable speed wind turbines. Technical Report NREL/TP-500-36265, National Renewable Energy Laboratory (NREL), Golden, CO, United States. accessed 6 March 2019.

Jonkman, J., S. Butterfield, W. Musial, and G. Scott
2009. Definition of a 5-MW reference wind turbine for offshore system development. Technical report, Golden, CO, USA.

Junior, V. J., J. Zhou, S. Roshanmanesh, F. Hayati, S. Hajiabady, X. Y. Li, H. Dong, and M. Papaelias
2017. Evaluation of damage mechanics of industrial wind turbine gearboxes. *Insight – Non-Destructive Testing and Condition Monitoring*, 59(8):410–414.

Kapetanakis, S. and D. Kudenko
2002. Reinforcement learning of coordination in cooperative multi-agent systems. In *Proceedings of the 18th National Conference on Artificial Intelligence (AAAI)*, volume 2002, Pp. 326–331.

Katic, I., J. Højstrup, and N. O. Jensen
1986. A simple model for cluster efficiency. In *European wind energy association conference and exhibition*, volume 1, Pp. 407–410.

Keller, J., Y. Guo, W. LaCava, H. Link, and B. McNiff
2012. NREL gearbox reliability collaborative phase 1 and 2: Testing and modeling results. In *Proc. of the International Conference on Noise and Vibration Engineering*, Pp. 4371–4379, Leuven, Belgium.

Keller, J., S. Sheng, J. Cotrell, and A. Greco
2016. Wind turbine drivetrain reliability collaborative workshop: A recap. Technical report, National Renewable Energy Laboratory (NREL), Golden, CO, United States. accessed 6 March 2019.

Killian, T. W., S. Daulton, G. Konidaris, and F. Doshi-Velez
2017. Robust and efficient transfer learning with hidden parameter Markov decision processes. In *Advances in Neural Information Processing Systems (NIPS)*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds., volume 30, Pp. 6250–6261. Curran Associates, Inc.

Knudsen, T., T. Bak, and M. Svenstrup
2015. Survey of wind farm control – power and fatigue optimization. *Wind Energy*, 18(8):1333–1351.

Kober, J., J. A. Bagnell, and J. Peters
2013. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274.

Kok, J. R. and N. Vlassis
2004. Sparse cooperative Q-learning. In *Proceedings of the 21st International Conference on Machine Learning (ICML)*, New York, NY, USA.

Kok, J. R. and N. Vlassis
2006. Using the max-plus algorithm for multiagent decision making in coordination graphs. In *RoboCup 2005: Robot Soccer World Cup IX*, A. Bredenfeld, A. Jacoff, I. Noda, and Y. Takahashi, eds., volume 4020 of *Lecture Notes in Computer Science*, Pp. 1–12. Springer.

Koller, D. and R. Parr
2000. Policy iteration for factored MDPs. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence (UAI)*, Pp. 326–334, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Konidaris, G., I. Scheidwasser, and A. Barto
2012. Transfer in reinforcement learning via shared features. *Journal of Machine Learning Research (JMLR)*, 13:1333–1371.

LaCava, W., J. van Dam, and R. Wallen
2011. NREL Gearbox Reliability Collaborative: Comparing in-field gearbox response to different dynamometer test conditions. Technical Report NREL/CP-5000-51690,

National Renewable Energy Laboratory (NREL), Golden, CO, United States. accessed 6 March 2019.

LaCava, W., Y. Xing, C. Marks, Y. Guo, and T. Moan
2013. Three-dimensional bearing load share behaviour in the planetary stage of a wind turbine gearbox. *IET Renewable Power Generation*, 7(4):359–369.

Lattimore, T. and C. Szepesvári
2020. Bandit algorithms. *Cambridge University Press*.

Lazaric, A. and M. Ghavamzadeh
2010. Bayesian multi-task reinforcement learning. In *Proc. of the 27th International Conference on Machine Learning (ICML)*, Pp. 599–606. Omnipress.

Libin, P., A. Moonens, T. Verstraeten, F. Perez-Sanjines, N. Hens, P. Lemey, and A. Nowé
2020. Deep reinforcement learning for large-scale epidemic control. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*. Springer.

Libin, P. J. K., T. Verstraeten, D. M. Roijers, J. Grujic, K. Theys, P. Lemey, and A. Nowé
2018. Bayesian best-arm identification for selecting influenza mitigation strategies. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*, Pp. 456–471. Springer.

Libin, P. J. K., T. Verstraeten, D. M. Roijers, W. Wang, K. Theys, and A. Nowé
2019. Thompson sampling for m-top exploration. In *Proceedings of the IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, Pp. 1414–1420.

Link, H., J. Keller, Y. Guo, and M. B.
2013. Gearbox Reliability Collaborative Phase 3 Gearbox 2 Test Plan. Technical Report NREL/TP-5000-58190, National Renewable Energy Laboratory (NREL), Golden, CO, United States. accessed 6 March 2019.

Link, H., W. LaCava, J. van Dam, B. McNiff, S. Sheng, R. Wallen, M. McDade, S. Lambert, S. Butterfield, and F. Oyague
2011. Gearbox Reliability Collaborative project report: findings from phase 1 and phase 2 testing. Technical Report NREL/TP-5000-51885, National Renewable Energy Laboratory (NREL), Golden, CO, United States. accessed 6 March 2019.

Loeb, R., T. Verstraeten, A. Nowé, and A. Dooms
2019. Privacy preserving reinforcement learning over distributed datasets. In *Proceedings of the 31st Benelux Conference on Artificial Intelligence and the 28th Belgian Dutch Conference on Machine Learning (BNAIC/BENELEARN)*.

Lougee-Heimer, R.
2003. The common optimization interface for operations research. *IBM Journal of Research and Development*, 47(1):57–66.

Lunn, D., C. Jackson, N. Best, D. Spiegelhalter, and A. Thomas
2012. *The BUGS book: A practical introduction to Bayesian analysis*. Chapman and Hall/CRC.

Lydia, M., S. S. Kumar, A. I. Selvakumar, and G. E. P. Kumar
2014. A comprehensive review on wind turbine power curve modeling techniques. *Renewable and Sustainable Energy Reviews*, 30:452–460.

Marden, J. R., S. D. Ruben, and L. Y. Pao
2013. A model-free approach to wind farm control using game theoretic methods. *IEEE Transactions on Control Systems Technology*, 21(4):1207–1214.

Martin, R., I. Lazakis, S. Barbouchi, and L. Johanning
2016. Sensitivity analysis of offshore wind farm operation and maintenance cost and availability. *Renewable Energy*, 85:1226–1236.

McKay, M. D., R. J. Beckman, and W. J. Conover
1979. Comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):239–245.

Moore, A. W.
1990. *Efficient Memory-Based Learning for Robot Control*. PhD thesis, University of Cambridge.

Moore, A. W.
1993. The parti-game algorithm for variable resolution reinforcement learning in multidimensional state-spaces. In *Proc. of the 6th International Conference on Neural Information Processing Systems (NIPS)*, Pp. 711–718.

Mukherjee, S., A. Mishra, and K. E. Trenberth
2018. Climate change and drought: a perspective on drought indices. *Current Climate Change Reports*, 4(2):145–163.

Muljadi, E., C. P. Butterfield, and B. Parsons
2007. Effect of variable speed wind turbine generator on stability of a weak grid. *IEEE Transactions on Energy Conversion*, 22(1):29–36.

National Renewable Energy Laboratory
2016. Gearbox Reliability Database. accessed 6 March 2019.

National Renewable Energy Laboratory (NREL)
2019. FLORIS. Version 1.0.0.

Nejad, A. R., Z. Gao, and T. Moan
2014a. On long-term fatigue damage and reliability analysis of gears under wind loads in offshore wind turbine drivetrains. *International Journal of Fatigue*, 61:116–128.

Nejad, A. R., P. F. Odgaard, Z. Gao, and T. Moan
2014b. A prognostic method for fault detection in wind turbine drivetrains. *Engineering Failure Analysis*, 42:324–336.

Pan, S. J. and Q. Yang
2009. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359.

Pape, F., J. T. Terwey, S. Wiesker, S. Averbeck, C. Muhmann, D. Lipinsky, H. F. Arlinghaus, E. Kerscher, B. Sauer, and G. Poll
2018. Tribological research on the development of white etching cracks (WECs). *Forschung im Ingenieurwesen*, 82(4):341–352.

Peeters, C., T. Verstraeten, A. Nowé, and J. Helsen
2019. Wind turbine planetary gear fault identification using statistical condition indicators and machine learning. In *International Conference on Offshore Mechanics and Arctic Engineering*. American Society of Mechanical Engineers.

Polydoros, A. S. and L. Nalpantidis
2017. Survey of model-based reinforcement learning: Applications on robotics. *Journal of Intelligent & Robotic Systems*, 86(2):153–173.

Puterman, M. L.
1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, 1st edition. John Wiley & Sons, Inc.

Rasmussen, C. E. and M. Kuss
2003. Gaussian processes in reinforcement learning. *Advances in Neural Information Processing Systems*, 16:751–758.

Rasmussen, C. E. and C. K. I. Williams
2006. *Gaussian Processes for Machine Learning*. Cambridge, MA, USA: The MIT Press.

Rastogi, D., I. Koryakovskiy, and J. Kober
2018. Sample-efficient reinforcement learning via difference models. In *Machine Learning in Planning and Control of Robot Motion Workshop at ICRA*.

REN21 Secretariat
2020. Renewables 2020: Global status report. Technical report, REN12 Secretariat: Paris, France.

Rhuggenaath, J., A. Akcay, Y. Zhang, and U. Kaymak
2019. Optimizing reserve prices for publishers in online ad auctions. In *2019 IEEE Conference on Computational Intelligence for Financial Engineering Economics (CIFEr)*, Pp. 1–8.

Richmond, M., A. Antoniadis, L. Wang, A. Kolios, S. Al-Sanad, and J. Parol
2019. Evaluation of an offshore wind farm computational fluid dynamics model against operational site data. *Ocean Engineering*, 193.

Robert, C.
2007. *The Bayesian choice: from decision-theoretic foundations to computational implementation*. Springer Science & Business Media.

Roijers, D. M.
2016. *Multi-Objective Decision-Theoretic Planning*. PhD thesis, University of Amsterdam.

Russo, D. and B. Van Roy
2014. Learning to optimize via posterior sampling. *Mathematics of Operations Research*, 39(4):1221–1243.

Russo, D., B. Van Roy, A. Kazerouni, I. Osband, and Z. Wen
2017. A tutorial on thompson sampling. *arXiv preprint arXiv:1707.02038*.

Sæmundsson, S., K. Hofmann, and M. P. Deisenroth
2018. Meta reinforcement learning with latent variable gaussian processes. In *Proc. of the 34th Conference on Uncertainty in Artificial Intelligence (UAI)*, Pp. 642–652.

Sarker, B. R. and T. I. Faiz
2017. Minimizing transportation and installation costs for turbines in offshore wind farms. *Renewable Energy*, 101:667–679.

Scharpff, J., D. M. Roijers, F. A. Oliehoek, M. T. J. Spaan, and M. M. de Weerdt
2016. Solving transition-independent multi-agent MDPs with sparse interactions. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI)*.

Schiermeier, Q.
2016. And now for the energy forecast: Germany works to predict wind and solar power generation. *Nature*, 535(7611):212–213.

Shen, W.
2019. *Multi-agent systems for concurrent intelligent design and manufacturing*. CRC press.

Short, W., D. J. Packey, and T. Holt
2005. *A manual for the economic evaluation of energy efficiency and renewable energy technologies*. University Press of the Pacific.

Siniscalchi-Minna, S., F. D. Bianchi, M. De-Prada-Gil, and C. Ocampo-Martinez
2019. A wind farm control strategy for power reserve maximization. *Renewable energy*, 131:37–44.

Snelson, E. and Z. Ghahramani
2006. Sparse Gaussian processes using pseudo-inputs. In *Advances in neural information processing systems*, Pp. 1257–1264.

Sohoni, V., S. Gupta, and R. Nema
2016. A critical review on wind turbine power curve modelling techniques and their applications in wind based energy systems. *Journal of Energy*, 2016.

Soleimanzadeh, M., R. Wisniewski, and S. Kanev
2012. An optimization framework for load and power distribution in wind farms. *Journal of Wind Engineering and Industrial Aerodynamics*, 107:256–262.

Solomon, S., D. Qin, M. Manning, Z. Chen, M. Marquis, K. B. Averyt, M. Tignor, and H. L. Miller, eds.
2007. *Climate Change 2007: The Physical Science Basis - Contribution of Working Group I to the Fourth Assessment Report of the Intergovernmental Panel on Climate Change*. New York, NY, United States: Cambridge University Press.

Sourkounis, C. and P. Tourou
2013. Grid code requirements for wind power integration in Europe. In *Conference Papers in Energy*, volume 2013. Hindawi.

Spudic, V., M. Jelavic, M. Baotic, and N. Peric
2010. Hierarchical wind farm control for power/load optimization. *The science of making torque from wind (TORQUE 2010)*.

Staffell, I. and R. Green
2014. How does wind farm performance decline with age? *Renewable energy*, 66:775–786.

Stranders, R., L. Tran-Thanh, F. M. D. Fave, A. Rogers, and N. R. Jennings
2012. DCOPs and bandits: Exploration and exploitation in decentralised coordination. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, Pp. 289–296. International Foundation for Autonomous Agents and Multiagent Systems.

Struggl, S., V. Berbyuk, and H. Johansson
2015. Review on wind turbines with focus on drive train system dynamics. *Wind Energy*, 18(4):567–590.

Sun, D. and J. O. Berger
2007. Objective Bayesian analysis for the multivariate normal model. *Bayesian Statistics*, 8:525–562.

Sutton, R. S.
1990. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Machine Learning Proceedings 1990*, Pp. 216–224. Elsevier.

Sutton, R. S. and A. G. Barto
2018. *Reinforcement Learning: An Introduction*, 2 edition. Cambridge, MA, United States: MIT Press.

Tao, S., A. Feijóo, J. Zhou, and G. Zheng
2020. Topology design of an offshore wind farm with multiple types of wind turbines in a circular layout. *Energies*, 13(3):556.

Taylor, M. E., M. Jain, P. Tandon, M. Yokoo, and M. Tambe
2011. Distributed on-line multi-agent optimization under uncertainty: Balancing exploration and exploitation. *Advances in Complex Systems*, 14(03):471–528.

Taylor, M. E. and P. Stone
2007. Cross-domain transfer for reinforcement learning. In *Proc. of the 24th International Conference on Machine Learning (ICML)*, Pp. 879–886. ACM.

Thompson, W. R.
1933. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294.

Tingey, E. B. and A. Ning
2017. Trading off sound pressure level and average power production for wind farm layout optimization. *Renewable Energy*, 114:547–555.

Treviño Cantú, H.
2011. Life-cycle cost analysis for offshore wind farms: Reliability and maintenance. Master's thesis, Gotland University.

Troldborg, N., G. C. Larsen, H. A. Madsen, K. S. Hansen, J. N. Sørensen, and R. Mikkelsen
2011. Numerical simulations of wake interaction between two wind turbines at various inflow conditions. *Wind Energy*, 14(7):859–876.

U.S. Department of Energy's Office of Energy Efficiency and Renewable Energy
2013. The inside of a wind turbine.

van Binsbergen, D. W., S. Wang, and A. R. Nejad
2020. Effects of induction and wake steering control on power and drivetrain responses for 10 mw floating wind turbines in a wind farm. *Journal of Physics: Conference Series*, 1618(2).

van Dijk, M. T., J.-W. Wingerden, T. Ashuri, Y. Li, and M. A. Rotea
2016. Yaw-misalignment and its impact on wind turbine loads and wind farm power output. *Journal of Physics: Conference Series*, 753(6).

Vanhatalo, J., V. Pietiläinen, and A. Vehtari
2010. Approximate inference for disease mapping with sparse Gaussian processes. *Statistics in medicine*, 29(15):1580–1607.

Vershynin, R.
2018. *High-dimensional probability: An introduction with applications in data science*, volume 47. Cambridge University Press.

Verstraeten, T., A. Nowé, J. Keller, Y. Guo, S. Sheng, and J. Helsen
2019. Fleetwide data-enabled reliability improvement of wind turbines. *Renewable and Sustainable Energy Reviews*, 109:428–437.

Vlassis, N., R. Elhorst, and J. R. Kok
2004. Anytime algorithms for multiagent decision making using coordination graphs. In *IEEE International Conference on Systems, Man and Cybernetics*, volume 1, Pp. 953–957.

Wagenaar, J., L. Machielse, and J. Schepers
2012. Controlling wind in ECN's scaled wind farm. *Proc. of the Europe Premier Wind Energy Event*, Pp. 685–694.

Wiering, M.
2000. Multi-agent reinforcement learning for traffic light control. In *Proceedings of the 17th International Conference on Machine Learning (ICML)*, Pp. 1151–1158.

Wilson, A., A. Fern, S. Ray, and P. Tadepalli
2007. Multi-task reinforcement learning: A hierarchical Bayesian approach. In *Proc. of the 24th International Conference on Machine Learning (ICML)*, Pp. 1015–1022.

Wilson, A. and Z. Ghahramani
2011. Generalised Wishart processes. In *Proc. of the Conference on Uncertainty in Artificial Intelligence (UAI)*, Pp. 736–744.

Wilson, A. and H. Nickisch
2015. Kernel interpolation for scalable structured Gaussian processes (KISS-GP). In *International Conference on Machine Learning (ICML)*, Pp. 1775–1784.

Wilson, A. G. and Z. Ghahramani
2010. Copula processes. In *Proc. of the 24th Annual Conference on Neural Information Processing Systems 2010 (NIPS)*.

Wiser, R. and M. Bolinger
2015. Wind technologies market report. Technical report, National Renewable Energy Laboratory (NREL), Golden, CO, United States. accessed 6 March 2019.

Wolsey, L. A. and G. L. Nemhauser
1999. *Integer and combinatorial optimization*, volume 55. John Wiley & Sons.

Yokoo, M., E. H. Durfee, T. Ishida, and K. Kuwabara
1998. The distributed constraint satisfaction problem: Formalization and algorithms. *IEEE Transactions on knowledge and data engineering*, 10(5):673–685.

Yu, Y.
2018. Towards sample efficient reinforcement learning. In *Proc. of the 27th International Joint Conference on Artificial Intelligence (IJCAI)*, Pp. 5739–5743. International Joint Conferences on Artificial Intelligence Organization.

Zhang, N. L. and D. Poole
1994. A simple approach to Bayesian network computations. In *Proc. of the 10th Canadian Conference on Artificial Intelligence*.