UNIVERSITY OF LIÈGE

DOCTORAL THESIS

Deep Reinforcement Learning for the Control of Energy Storage in Grid-Scale and Microgrid Applications

Author:
Ioannis BOUKAS

Supervisors:
Prof. Bertrand CORNÉLUSSE
Prof. Damien ERNST

A thesis submitted in fulfillment of the requirements
for the degree of Doctor of Philosophy

in the

Smart-micro Grids
Montefiore Institute of Electrical Engineering and Computer Science

May 25, 2021

“ἕν οἶδα ὅτι οὐδὲν οἶδα *(I know that I know nothing)* ”

Σωκράτης (Socrates)

UNIVERSITY OF LIÈGE

Abstract

Faculty of Applied Sciences

Montefiore Institute of Electrical Engineering and Computer Science

Doctor of Philosophy

Deep Reinforcement Learning for the Control of Energy Storage in Grid-Scale and Microgrid Applications

by Ioannis BOUKAS

The European and worldwide directives and targets for renewable energy integration, motivated by the imminent need to decarbonize the electricity sector, are imposing severe changes to the conventional electrical power system. The inherent unpredictability of the instantaneous energy production from variable renewable energy sources (VRES) is expected to make the reliable and secure operation of the system, a challenging task. Flexibility, and in particular, energy storage is expected to assume a key role in the integration of large shares of VRES in the power system, and thus, in the transition towards a carbon-free electricity sector. One of the main storage mechanisms that can facilitate the integration of VRES is energy arbitrage, i.e. the transfer of electrical energy from a period of low demand to another period of high demand. In this thesis, we investigate and develop novel operating strategies for maximizing the value of energy arbitrage from storage units at different scales (i.e. grid-scale or distributed) and in different settings (i.e. interconnected or off-grid). The decision-making process of an operator optimizing the energy arbitrage value of storage is an inherently complex problem, mainly due to uncertainties induced by: i) the stochasticity of market prices and ii) the variability of renewable generation. In view of the great successes of deep reinforcement learning (DRL) in solving challenging tasks, the goal of this thesis is to investigate its potential in solving problems related to the control of storage in modern energy systems.

Firstly, we address the energy arbitrage problem of a storage unit that participates in the European Continuous Intraday (CID) market. We develop an operational strategy in order to maximize its arbitrage value. A novel modeling framework for the strategic participation of energy storage in the European CID market is proposed, where exchanges occur through a process similar to the stock market. A detailed description of the market mechanism and the storage system management is provided. A set of necessary simplifications that constitutes the problem tractable are described. The resulting problem is solved using a state-of-the-art DRL algorithm. The outcome of the proposed method is compared with the state-of-the-art industrial practices and the resulting policy is found able to outperform this benchmark.

Secondly, we address the energy arbitrage problem faced by an off-grid microgrid operator in the context of rural electrification. In particular, we propose a novel model-based reinforcement learning algorithm that is able to control the storage device in order to accommodate the different changes that might occur over the microgrid lifetime. The algorithm demonstrates generalisation properties, transfer capabilities and better robustness in case of fast-changing system dynamics. The proposed algorithm is compared against two benchmarks, namely a rule-based and a model predictive controller (MPC). The results show that the trained agent is

able to outperform both benchmarks in the lifelong setting where the system dynamics are changing over time.

In the context of an off grid-microgrid, the optimal size of the components (i.e. the capacity of photovoltaic (PV) panels, storage) depends heavily on the control policy applied. In this thesis, we propose a new methodology for jointly sizing a system and designing its control law that is based on reinforcement learning. The objective of the optimization problem is to jointly find a control policy and an environment over the joint hypothesis space of parameters such that the sum of the initial investment and the operational cost are minimized. The optimization problem is then addressed by generalizing the direct policy search algorithms to an algorithm we call Direct Environment Search with (projected stochastic) Gradient Ascent (DESGA). We illustrate the performance of DESGA on two benchmarks. First, we consider a parametrized space of Mass-Spring-Damper (MSD) environments and control policies. Then, we use our algorithm for optimizing the size of the components and the operation of a small-scale autonomous energy system, i.e. a solar off-grid microgrid, composed of photovoltaic panels, batteries. On both benchmarks, we show that DESGA results in a set of parameters for which the expected return is nearly equal to its theoretical upper-bound.

Finally, in Chapter 6, we provide the general conclusions and remarks of this thesis and we propose a list of future research directions that emerge as an outcome of this work.

Acknowledgements

Throughout the writing of this dissertation I have received a great deal of support and assistance.

I would first like to thank my supervisors, Professors Damien Ernst and Bertrand Cornélusse, for their support and guidance during the past four years. You provided me with the tools that I needed to choose the right direction and successfully complete my dissertation. In particular, I have Damien to thank for his tough but always insightful feedback that pushed past my limits towards analysing and solving a given problem. I would also like to thank Damien for introducing me into and transmitting to me his passion for the field of reinforcement learning. I would like to thank Bertrand specifically for his help in enriching my knowledge on energy markets and microgrids as well as extending my understanding of optimization techniques.

Secondly, I would like to thank Prof. Louis Wehenkel and Raphaël Fonteneau for all the enlightening discussions we had on topics related to machine learning, reinforcement learning and power systems. Additionally, I would like to thank all my colleagues from Montefiore, and in particular, my officemate Mathias, David, Daniele, Eythimis, Labros, Adrien, Thibaut, Selmane, Selim, Jonathan, Sebastien, Quentin, Julien, Vageesh, who provided stimulating discussions as well as happy distractions that made it possible to endure the tough moments. I would like to thank Prof. Sylvain Quoilin for inviting me to study in Liège and for his guidance throughout my studies.

I would like to thank Prof. Anthony Papavasiliou, Gilles Bertrand and Gauthier de Maere d'Aertrycke for the fruitfull discussions and their wonderful collaboration. I would also like to thank Prof. Jalal Kazempour and Dimitrios Papadaskalopoulos for their guidance throughout the past years.

I would like to thank Alexandre Huynen, Martin Buchwald and Christelle Wynants from Engie Brussels for the invaluable opportunity they gave me to collaborate with them and to learn so much from them during my internship. Special thanks to Kris Vandekeybus, Musa Dinc and Erwin Simons for the insightful discussions we had regarding practical issues around trading in the intraday market.

I would like to thank Simone Totaro and Prof. Anders Jonsson for welcoming me in the Universitat Pompeu Fabra. Our collaboration has been a stepping stone towards enriching my knowledge of reinforcement learning, both fundamental and practical.

I would like to thank all my friends for all the great times we had. In particular, I would like to thank Miguel and Sergio who have been side by side with me in this journey all this time. I would like to thank Queralt for her support, her understanding and her patience. Finally, I would like to thank my family for their unconditional love and support and in particular, my grandpa Ioannis, whose passion for knowledge led me here today.

Contents

Abstract **vi**

Acknowledgements **ix**

1 Introduction **1**

1.1 Energy Arbitrage 5

1.1.1 Storage participation in the electricity markets 6

1.1.2 Storage in the context of microgrids 9

1.1.3 Reinforcement learning 10

Background 11

Recent advancements 13

1.2 Contributions and outline of this thesis 15

1.3 Publications 17

2 A Deep Reinforcement Learning Framework for Continuous Intraday Market Bidding **21**

2.1 Introduction 21

2.1.1 Intraday markets in Europe 22

2.1.2 Bidding strategies in literature 24

2.1.3 Contributions of the chapter 27

2.1.4 Outline of the chapter 28

2.2 Continuous Intraday Bidding process 28

2.2.1 Continuous Intraday market design 29

2.2.2 Continuous Intraday market environment 30

2.2.3 Asset trading 32

2.2.4 Trading rewards 38

2.2.5 Trading policy 39

2.2.6 Trading objective 39

2.3 Markov Decision Process Formulation 40

2.3.1 Assumptions on the decision process 40

2.3.2 Decision process 41

2.3.3 Assumptions on the trading actions 42

2.3.4 Restrictions on the storage operation 42

2.4 Methodology 43

2.4.1 Collection of trajectories 44

2.4.2 Batch-mode reinforcement learning 44

2.4.3 Limitations 47

2.4.4 Action space reduction: High-level actions 48

"Trade" 48

"Idle" 50

2.4.5 State space reduction 51

2.4.6 Generation of artificial trajectories 54

2.4.7 Neural Network architecture 58

2.4.8 Asynchronous Distributed Fitted Q iteration 59

2.5 Case study 60

2.5.1 Parameters specification 61

2.5.2 Exogenous variable 62

2.5.3 Benchmark strategy 62

2.5.4 Validation process 62

2.5.5 Results 64

2.6 Discussion 65

2.6.1 Behaviour of the rest of the agents 67

2.6.2 Partial observability of the process 68

2.6.3 Action space reduction 68

2.6.4 Exploration 69

2.7 Conclusions and future work 69

2.8 Nomenclature 70

3 Expanding the scope of the CID agent 77

3.1 New state space representation 77

3.1.1 Compact state space representation 78

3.1.2 Making the state stationary 79

3.2 New reward function 80

3.2.1 Making the reward function stationary 81
3.3 Case study 81
3.3.1 Parameters specification 81
3.3.2 Exogenous variable 82
3.3.3 Anticipative benchmark strategy 82
3.3.4 Results 83
3.3.5 Policy analysis 84
3.4 Conclusions 87

4 Lifelong Control of Off-grid Microgrid with Model Based Reinforcement Learning 89

4.1 Introduction 89
4.2 Related Work 92
4.3 Microgrid Description 94
4.3.1 Components 95
Consumption 95
Storage model 96
Steerable generator model 96
Non-steerable generators model 97
Power balance 97
4.3.2 Characterizing changes in the environment 98
Gradual changes 98
Sudden or abrupt changes 98
4.4 Reinforcement Learning Background 99
4.4.1 Markov Decision Process 99
4.4.2 Dyna 100
4.4.3 Proximal Policy Optimization 101
4.4.4 Quantile Regression 102
4.5 Problem Statement 103
4.5.1 Microgrid Simulator 104
4.6 Methodology 104
4.7 Benchmark strategies 105
4.7.1 Rule-based controller 106
4.7.2 Model-predictive controller 106

4.8 Case study . . . 107

4.8.1 System configuration . . . 107

4.8.2 Partial Observability . . . 109

4.8.3 Action Space and Meta-Actions . . . 109

4.8.4 Comparison with the benchmarks . . . 111

4.8.5 Generalization . . . 111

4.8.6 Robustness . . . 112

4.8.7 Transfer . . . 113

4.9 Conclusions . . . 114

4.10 Appendix . . . 116

4.10.1 Algorithms . . . 116

4.10.2 Notation . . . 117

5 Learning optimal environments using projected stochastic gradient ascent 121

5.1 Introduction . . . 122

5.2 Theoretical background and problem statement . . . 126

5.2.1 Discrete-time dynamical systems . . . 126

5.2.2 Problem statement: optimizing over a set of environments . . . 127

5.3 Direct environment search with gradient ascent . . . 128

5.3.1 Gradient for learning optimal environments . . . 128

5.3.2 Parameter optimization with projected stochastic gradient ascent . . . 131

5.4 Experiments . . . 132

5.4.1 Methodology . . . 132

5.4.2 Mass-Spring-Damper environment . . . 133

5.4.3 Sizing and operation of a solar off-grid microgrid . . . 134

5.4.4 Discussion on the alternative to the DESGA algorithm . . . 136

5.5 Conclusions and Future Work . . . 136

5.6 Analytical derivation of the gradient for learning optimal environments . . . 137

5.7 Direct environment search with (projected stochastic) gradient ascent . . . 146

5.8 Mass-Spring-Damper environment . . . 147

5.9 Optimal design of a solar off-grid microgrid . . . 150

6 Concluding remarks and future work 159

6.1 Conclusions . . . 159

6.2 Future work . . . 161

Bibliography **163**

List of Figures

1.1 Schematic of the interactions between an agent with its environment in reinforcement learning. 11

2.1 Trading (continuous and discrete) and delivery timelines for products Q_1 to Q_4 29

2.2 54

2.3 55

2.4 Schematic of the decision process. The original MDP is highlighted in a gray background. The state of the original MDP $h_{i,t}$ is encoded in pseudo-state $z_{i,t}$. Based on $z_{i,t}$, agent i takes an high-level action $a'_{i,t}$, according to its policy π_i. This action $a'_{i,t}$ is mapped to an original action $a_{i,t}$ and submitted to the CID market. The CID market makes a transition based on the action of agent i and the actions of the other agents $a_{-i,t}$. After this transition, the market position of agent i is defined and the control actions for storage device are derived according to the “default" strategy. Each transition yields a reward $r_{i,t}$ and a new state $h_{i,t}$. 56

2.5 Schematic of the neural network architecture. 56

2.6 Schematic of the asynchronous distributed architecture. Each actor runs on a different thread and contains a copy of the environment, an individual ε-greedy policy based on the latest version of the network parameters and a local buffer. The actors generate trajectories that are stored in their local buffers. When the local buffer of each actor is filled, it is appended to the global buffer and the agent collects the latest network parameters from the learner. A single learner runs on a separate thread and is continuously training using experiences from the global buffer. 60

2.7 Profitability ratio. 66

2.8 Progressive evaluation in the train set. 66

2.9 Progressive evaluation in the test set. 67

3.1 States from two different days. 79

3.2 Normalized states from two different days. 80

3.3 Profitability ratio. 85

3.4 Progressive evaluation in the train set. 85

3.5 Progressive evaluation in the test set. 86

3.6 Probability of trading for each decision step according to π^{FQ}. 86

4.1 Schematic of the considered microgrid. 95

4.2 PV production and its daily, weekly and monthly rolling average. 108

4.3 Electrical load and its daily, weekly and monthly rolling average. 109

4.4 Cumulative returns (cost) on the test set as a function of the training progress of the RL algorithms. 112

4.5 Cumulative returns (costs) when the battery is excluded as a function of the training progress of the RL algorithms. 113

4.6 Cumulative returns (cost) during February as a function of the training progress of the RL algorithms. 115

4.7 Cumulative returns (cost) during August as a function of the training progress of the RL algorithms. 115

5.1 Assessment of DESGA. Left: the average value of $V(\theta_k, \psi_k)$ and its standard deviation over 20 executions of the DESGA algorithm as a function of the number of iterations k of the algorithm. Right: the average expected return of five policies identified with the REINFORCE algorithm for every element $\psi_d \in \Psi_d$. 134

5.2 Assessment of DESGA on the solar off-grid microgrid environment. Left: the average value of $V(\theta_k, \psi_k)$ and its standard deviation over 20 executions of the DESGA algorithm as a function of the number of iterations k of the algorithm. Right: the average expected return of five policies identified with the REINFORCE algorithm for every element $\psi_d \in \Psi_d$. A magnified area of the original graph is presented, where the maximum values are located. . . . 136

5.3 Mass-Spring-Damper system. 147

5.4 Microgrid configuration 150

List of Tables

2.1 Order Book for Q_1 and time slot 00:00-00:15 31

2.2 Order book features used for the state reduction. 53

2.3 Descriptive statistics of the returns obtained on the days of the training set for policies π^{FQ} and π^{RI}. The last column also provides the corresponding profitability ratios. 64

2.4 Descriptive statistics of the returns obtained on the days of the test set for policies π^{FQ} and π^{RI}. The last column also provides the corresponding profitability ratios. 65

3.1 Descriptive statistics of the returns obtained on the days of the training set for policies π^{FQ}, π^{RI} and π^{LA}. 84

3.2 Descriptive statistics of the returns obtained on the days of the test set for policies π^{FQ}, π^{RI} and π^{LA}. 84

4.1 Input parameters. 110

5.1 Parameters for the MSD. 150

5.2 Parameters for the solar off-grid microgrid. 156

5.3 Electrical load consumption and PV production power factor data. 157

To my family

Chapter 1

Introduction

Climate change, as a result of the excessive anthropogenic emissions of greenhouse gases taking place from the mid-20th century until today, is a major contemporary challenge. This change in the climate of the planet is translated among other effects, into a global temperature rise. It is estimated that Earth's average temperature has risen more than 1.2 °C since the late 19th century, with years 2016 and 2020 being the warmest years ever recorded [1]. The Intergovernmental Panel on Climate Change (IPCC), which includes more than 1,300 scientists from countries around the world, forecasted a temperature rise of 1.4°C to 5.5°C over the next century depending on the assumptions [1]. The effects of this temperature rise are manifold and the net damage costs[1] are likely to be significant and to increase over time. Some of these effects have the potential to be long-lasting and even irreversible, such as the loss of ecosystems [2]. In an attempt to combat climate change, one that for the first time brings nearly all nations together into a common cause, the *Paris Agreement* [2] [3] establishes a clear goal to limit the global temperature increase to well below 2°C and, ideally to 1.5°C, as compared to pre-industrial levels. A deep transformation of the global energy landscape is necessary to achieve this climate target. The main goal of this transformation is to considerably limit any energy-related CO_2 emissions and to reach carbon neutrality[3] by 2050.

The European Union (EU) has well aligned its energy policy with the target established by the Paris Agreement. In 2019, the EU agreed on the *Clean Energy for all Europeans package* (so-called Clean Energy package) [4], a new energy rulebook that facilitates the energy transition and the implementation of the energy union strategy. The Clean Energy package contains directives that aim at accomplishing targets related to improving the energy

[1] The IPCC predicts that increases in global mean temperature of 1 to 3 degrees Celsius above 1990 levels will produce beneficial impacts in some regions and harmful ones in others. Net annual economic costs will increase over time as global temperatures increase.

[2] The Paris Agreement was adopted by 196 Parties at COP 21 in Paris, on 12 December 2015 and entered into force on 4 November 2016

[3] Carbon neutrality (or net zero) means having a balance between emitting carbon and absorbing carbon from the atmosphere into carbon sinks.

performance in buildings, enhancing the overall efficiency of energy use to 32.5% by 2030 and increasing the share of renewable energy sources (RES) in the energy mix of the EU up to 32% by 2030. Additionally, it aims at establishing a modern harmonized electricity market design that would facilitate energy exchanges across regions, which plays a crucial role in the integration of RES and thus, in a successful energy transition strategy by 2050. Finally, it contains a robust governance and regulation system to ensure that each member state adopts and fulfils a National Energy and Climate Plan (NECP). Every two years, the most recent NECPs submitted by each Member State to the European Commission are used to define a central policy scenario named *National Trends* in the Ten Year National Development Plans (TYNDPs) 2020 scenario report [5] published by the European Network of Transmission System Operators for electricity and gas (ENTSO-E and ENTSOG respectively). In addition to that, the ENTSOs provide two more scenarios called *Distributed Energy* and *Global Ambition* in an attempt to capture the different pathways (centralized vs decentralized) towards achieving the COP 21 targets (see footnote 2). All three scenarios represent projections of how the demand and supply of energy, as well as the CO_2 emissions, will evolve by 2050. A key common aspect in all three scenarios is the fact that by 2030, more than 40% of the European electricity demand will be covered by variable renewable energy sources (VRES). At a global level, the plans for renewable electrification and, in particular, for VRES installations follow similar trends to reach the EU goals. The Global Renewables Outlook report published by the International Renewable Energy Association (IRENA) in 2020 presents two scenarios describing the evolution of the energy sector by 2050, namely the *Planned Energy Scenario*[4] and the *Transforming Energy Scenario*[5]. According to the former, VRES installations are expected to reach globally, 38% of the total generation capacity by 2030 and 55% by 2050, whereas the more ambitious *Transforming Energy Scenario* projects a 57% share of VRES by 2030 and 86% by 2050 [6].

The inherent unpredictability of the instantaneous energy production from VRES will inevitably lead to situations when the originally forecasted supply of electricity will not match the demand in real-time. This effect will become critical when large shares of VRES are integrated in the power system. High levels of variability in the power systems are expected to make the reliable and secure operation of the system, a challenging task [7]. To this end, flexibility is a key factor to enable the large-scale integration of VRES and has a vital role in the

[4] The *Planned Energy Scenario* provides a perspective on energy system developments based on governments' current energy plans and other planned targets and policies (as of 2019), including Nationally Determined Contributions under the Paris Agreement unless the country has more recent climate and energy targets or plans.

[5] The *Transforming Energy Scenario* describes an ambitious, yet realistic, energy transformation pathway based largely on RES and steadily improved energy efficiency

future power system. The flexibility of a power system refers to "the extent to which a power system can modify electricity production or consumption in response to variability, expected or otherwise" [8]. Flexibility resources can be actively used to offset any discrepancies[6] between demand and supply and they constitute one of the main mechanisms that ensures the reliable operation of the power system. There exist various sources of flexibility that originate from both the supply and the demand side. Due to the abrupt and sharp changes in the residual load curve[7], all flexibility sources need to share a common feature, that is their fast/agile response time [9]. At this stage of technological development, flexibility sources include:

- **Fast-ramping power plants**, which are able to regulate (increase or decrease) on-demand their generation output rapidly and have short start-up/shut-down times. In this category, gas-fired power plants are the dominant technology due to their inherent fast ramping capabilities [10]. Coal power plants are less suitable to provide this type of flexibility due to thermal and other operational contraints [11]. In addition to conventional fossil-fueled units, fully-controllable renewable-based power plants, such as biogas power plants, hydroelectric power plants and geothermal power plants can be used to provide flexibility.

- **Transmission capacity**, which can be used to transfer power accross the grid between neighbouring regions/grids in order to cover for power deficits. In this direction, the International Grid Control Cooperation (IGCC) was launched by ENTSO-E in 2016 to implement the imbalance netting process[8] [12].

- **Demand-side flexibility**, where consumers adjust (reduce, increase or shift in time) their consumption in order to facilitate the stable and/or economical operation of the system. Demand-side flexibility is usually offered by large industrial consumers or aggregators that manage portfolios of consumers (that may be industrial or residential). Smart metering and information and communications technology (ICT) infrastructure enable the monitoring and control of electricity consumption with high granularity. Consequently, and in contrast with the conventional doctrine, the demand can adapt its consumption behavior to match the volatility of VRES and facilitate the large integration of these generation technologies. In addition to that, demand-side flexibility

[6] These discrepancies are commonly known as imbalances.

[7] The difference between demand and VRES production

[8] Transmission system operators (TSOs) coordinate in order to avoid the simultaneous activation of frequency restoration reserves (FRR) in opposing signs.

can facilitate the total system cost reduction by shifting demand from peak to off-peak [9].

- **Sector coupling**, where the power grid is part of an integrated (larger) system that contains other carriers. For instance, the gas network can be closely operated with the electrical network and serve as a buffer for storing energy in the form of biogas by making use of technologies like power-to-gas and methanation. Moreover, the electrification of the transportation sector can serve as an additional energy integration lever that could provide flexibility to the power grid. Additionally, the thermal or water networks can be used to store energy and thus, provide flexibility.

- **Storage**, where electricity energy is transformed and stored so that it can be used at a later moment. There exist three main forms of energy storage that are currently used at large. First, kinetic energy-based technologies, such as pumped hydro energy storage (PHES) plants, compressed air energy storage (CAES) and flywheels, have been traditionally used in grid-scale applications. Among these technologies, PHES units harness the potential energy of water at height by consuming the excess of electricity from the network and using it to pump water from a lower to a higher reservoir. Inversely, water passes through hydraulic turbines on its way to the lower reservoir thus producing electricity when needed. In a similar process, CAES units compress air that is directed and stored to underground caverns. The compressed air then flows through turbines to produce electricity when needed. Second, electrochemical technologies, where energy is stored in different types of batteries such as Li-ion, lead-acid or flow batteries. These batteries are used in both grid-scale and small-scale applications. Finally, in thermal storage technologies, energy is stored by heating or cooling a liquid or solid storage medium (e.g. water, salts).

All the aforementioned flexibility sources are essential for the efficient integration of VRES. However, as the costs of storage technologies and, in particular, of lithium-ion (Li-ion) battery storage, are declining and will continue in this trend for the next 30 years, storage has emerged as a potentially attractive, carbon-free solution to the problems posed by increased VRE penetration [13]. According to [6], the amount of stationary storage (excluding electric vehicles) is expected to increase from around 30 GWh today to over 9,000 GWh by 2050. When considering storage capacity from the electric vehicles (EV) fleet, this value is expected

[9] Peak (off-peak) periods are considered to be parts of the day with high (low) electricity consumption. Indicatively, peak periods can be considered from 7 am to 10 pm on weekdays (Monday to Friday) while off-peak periods are from 10 pm to 7 am on weekdays (Monday to Friday) and the weekend

to reach globally a level of 23,000 GWh. In the context of this thesis, motivated by the increasing capacity and the importance that it is expected to assume in the years to come, we focus on the role of storage in the future energy systems.

There is a wide range of flexibility services that storage systems can provide depending on their capacity, their underlying technology, their point of connection to the grid (e.g. transmission, distribution level) and their use case [14]. For instance, large-scale energy storage devices can be used for energy services such as arbitrage (the transfer of electrical energy from a period of low demand to another period of high demand), ancillary services (e.g. primary, secondary and tertiary reserves[10], black-start capability etc.) or frequency regulation[11]. Additionally, energy storage can be used in the transmission or the distribution level to guarantee the reliability of the system (e.g. congestion relief, transmission/distribution deferral[12]) and to ensure power quality by dampening variations in voltage magnitude [14]. Medium-scale and small-scale storage devices can be used at the end-consumer level for reducing the peak power that is drawn from the grid, thus ensuring uninterrupted power supply, and for improving self-sufficiency rates[13] for prosumers (i.e. consumers that can also inject power to the grid, usually coming from renewable sources such as solar photovoltaic panels). In particular, when coupled with VRES (in the context of virtual power plants, grid-connected microgrids) such as wind or solar power, energy storage can provide a nearly constant power output by absorbing peaks and by reducing the rate of change of the RES generation. On the other hand, energy storage is an essential component for the operation of off-grid microgrids, especially when large shares of VRES are used to supply the load, as in the case of rural electrification.

1.1 Energy Arbitrage

Out of the various value proposition mechanisms of storage, in this thesis, we study the energy arbitrage that storage can achieve as a way to transform intermittent renewable energy production to electricity that can be used on-demand, during periods when it is needed. In particular, we investigate different ways in which existing storage capacity can be operated in order to optimize the value of energy arbitrage and, consequently, to maximize the VRES

[10] Capacity available to the system operator within a short interval of time to meet demand in case a generator goes down or there is another disruption to the supply.

[11] Frequency regulation is the process of injecting or withdrawing electricity from the power grid for purposes of maintaining system frequency in between the safe operational bandwidth.

[12] Installing storage capacity at certain points of the network can relieve congested parts of the grid and result in delay or avoidance of costly equipment upgrades.

[13] Percentage of energy consumed by the users that is produced locally by the distributed VRES.

utilization in an indirect or a direct way. The indirect way (price arbitrage) refers to settings where storage is considered to participate in electricity markets and maximizing VRES utilization is achieved implicitly by optimizing its market returns. The direct way usually refers to settings where the main goal is to operate energy storage with an objective that directly translates to VRES utilization. For instance, the operational strategy for energy storage in a microgrid setting usually achieves that objective by minimizing grid imports or fossil-fuel generated energy. In the following, we elaborate on the value of price arbitrage in electricity markets and on the value of energy arbitrage in the decentralized context of microgrids. We provide an overview of the existing methodologies for optimizing the arbitrage value of storage devices. The complex nature of the storage control problem under uncertainty as well as the computational limitations of the existing methods constitute a bottleneck to the optimization of the energy arbitrage value. Alternatively, recent advancements in the field of deep reinforcement learning (DRL) in combination with the availability of large datasets have been proven able to tackle very complex problems. At the end of this section we provide a short description of the underlying principles of DRL.

1.1.1 Storage participation in the electricity markets

In this thesis, we firstly address the problem of how to operate grid-connected storage capacity in today's electricity markets in a profitable way. The desired outcome from the participation of energy storage in the markets is to perform price arbitrage, i.e. to buy (charge) energy from the market when the prices are low and to sell (discharge) energy when the prices are high. In electricity markets, low-price periods usually coincide with low-demand periods and inversely, high-price periods coincide with high-demand periods. Therefore, the price arbitrage can be considered to be also energy arbitrage through the underlying market mechanism. The effective market value capture for energy storage devices depends on:

- the technical characteristics of the storage unit. A storage device is typically characterized by its power capacity (MW), its energy capacity (MWh), and its roundtrip efficiency[14]. The round-trip efficiency has a large impact on the value of storage due to the fact that a more inefficient device not only needs to charge more hours, but these added hours are typically more expensive [15]. Additionally, the relative size of the storage unit with respect to the rest of the market participants affects the value of arbitrage. A comparatively large storage unit has the potential to shift the prices in an unfavourable way (price maker), thus reducing the value that can be captured, while a

[14] The fraction of energy added into the storage that can be retrieved.

small storage unit has a negligible impact impact on the prices and thus cannot affect the potential value (price taker).

- the energy mix, the fuel prices, as well as the hourly load profile. More specifically, the price-setting units at peak vs off-peak periods define in a straightforward way the price spread that generates the arbitrage value of storage. The price spread depends largely on the underlying fuel mix of the supply curve and the hourly off- and on-peak loads. Thus, storage can be more valuable in regions where cheap nuclear, hydroelectric, and coal are available for off-peak electricity generation while expensive gas is the marginal fuel during peak periods. Additionally, the marginal price of the price-setting fuels (i.e. gas and coal) significantly impacts the potential benefits that can be captured by energy storage.
- the market mechanism characteristics, i.e. the rules and the regulatory framework that define the way in which electricity is exchanged. For instance, the design of a market that operates close to real-time with available products that offer refined granularity can create a level-playing field for fast, flexible storage units.
- the operational control strategy that is applied. The hourly operation of storage typically depends on market price patterns that are highly correlated to load patterns. There are two main load seasonality patterns in modern energy systems, namely i) the daily pattern that consists of peak and off-peak periods and ii) the weekday vs weekend patterns that lead to different load levels and subsequently price levels. Therefore, in principle, a good storage control policy is predictable to obtain and the arbitrage value can be estimated in a straightforward manner. However, unexpected short-term changes in the weather, the supply and the load can substantially increase the arbitrage value captured by storage, as well as the complexity of finding a good control policy [15].

Price arbitrage has been extensively studied in the literature for the case where energy storage units participate in the short-term electricity markets either self-standing or in combination with VRES. In the first case, price uncertainties are the main source of risk when attempting to identify the optimal bidding strategies for storage. In particular, the arbitrage potential for PHES and CAES in European markets is analysed in [16]. The authors point out a number of factors that influence the value of arbitrage such as the market integration, the market efficiency and the market competition levels as well as the amount of existing flexibility. Among those factors, one of the most critical, is the adopted operational strategy used to control the storage unit. There exists a wide range of methods used for optimizing the operational strategy for storage units in short-term markets. A rather simple backcasting strategy is considered in [15], where the operation of the next 2-weeks period is defined by the optimal operation plan of the previous 2-weeks period. A stochastic optimization framework is proposed in [17], where the participation of a storage unit in the day-ahead market and real-time market is considered. The results indicate that by taking explicitly into consideration the uncertainty regarding the market prices leads to increased revenues for the storage operator in comparison to the deterministic approach. Alternatively, stochastic dynamic programming (SDP) [18], stochastic dual dynamic programming (SDDP) [19] and approximate dual dynamic programming (ADDP) [20] are different methodologies that can tackle sequential decision-making problems under uncertainty in day-ahead markets with storage. The main differences between these three methods stem from the representation of the problem (state and action spaces) and the way the updates of the value functions are performed. These methods are shown to lead to optimal charging/discharging decisions for storage units, while accounting for market and system uncertainty, however they scale unfavourably with the size of the state/action space and the number of decision steps, so they usually come at a high computational cost for real life problems. In [21], the authors propose an analytical solution method to the multi-stage energy arbitrage problem under price uncertainty, that has increased computational performance compared to the SDDP benchmark. In [22], approximate dynamic programming (ADP) is proposed for optimizing real-time decisions for a storage unit participating in the hour-ahead market organized by the New York Independent System Operator. In [23], the authors tackle the problem of real-time price arbitrage using reinforcement learning. They propose a novel reward function that not only reflects the instant profit of charge/discharge decisions but also the historical information from past trades. The proposed method leads to significant performance improvements when compared to existing benchmarks. Alternatively, market participation of storage can be considered in combination with VRES. in this case, uncertainties in the decision-making process originate from both the market prices and the variability of renewable generation. In [24], the day-ahead bidding problem of a wind farm coupled with storage is formulated as a robust optimization model where uncertainties regarding prices and wind generation are considered. The conditional value at risk (CVaR) is used as a measure to determine the worst-case scenarios and the resulting decisions yield improved revenues when compared to a deterministic benchmark.

1.1.2 Storage in the context of microgrids

In the second part of this thesis, we focus on the energy arbitrage that storage can provide in order to facilitate the penetration and the utilization of VRES in the energy mix. In addition to grid-scale energy storage, flexibility in future power systems will be also provided in a decentralized manner by integrating microgrids at the customer or community level. Microgrids are small electrical networks composed of distributed energy resources and electricity loads that are controlled and operated locally. Recent technological advances and the development of smart microgrid control strategies enabled the efficient utilization of RES and enhanced security of supply. Microgrids, when connected to the main grid, can be operated either interconnected or in islanded mode [25]. When connected to the main grid, microgrids can facilitate the operation of the power system by providing flexibility. On the other hand, in the event of a black-out, microgrids can disconnect and operate in islanded mode and later assist with the power restoration process. The benefits from energy arbitrage are demonstrated in [26], where the authors investigate the factors that impact the self-sufficiency rate and, as such, the economic benefits that can be achieved by a grid-connected solar photovoltaic microgrid. The simulation results show that increasing the installed capacity of energy storage installed (Li-ion batteries in particular) leads to increased levels of self-sufficiency rates. However, the authors conclude that, for these high self-sufficiency rates to be attained, the battery cost should be well below the value of the considered level of battery prices (2016). Additionally, in many cases, microgrids can be installed and operated completely off-grid. Off-grid microgrids are receiving a growing interest for rural electrification purposes in developing countries due to their ability to ensure affordable, sustainable and reliable energy services [27], [28]. Off-grid microgrids rely on VRES coupled with storage systems to supply the electricity consumption. The inherent uncertainty introduced by VRES, as well as the stochastic nature of the electrical demand in rural contexts pose significant challenges to the efficient lifelong control of off-grid microgrids [27]. A critical issue in the lifelong microgrid operation is that the optimal operational strategy changes over its lifetime due to permanent shifts in the consumption profile [29]. For instance, the population of the rural area where the microgrid is installed can progressively increase because more people want to have access to electricity and thus, are connect to the microgrid. Additionally, the change in the routines of people can have similar impact, e.g. selecting electrical stoves instead of wood-fire ovens can introduce changes in the shape of the daily demand profile. Moreover, the degradation or damage of the various components such as the storage devices or the photovoltaic panels affect accordingly the operational strategy that needs to be adapted in order to maintain the safe operation of the microgrid.

Overall, the methods proposed for tackling the operational control in the context of an off-grid microgrid are similar to the ones used for price arbitrage. Fundamentally, the main difference between the two problems lies in the fact that the uncertainty does not originate from the market price formation process (as it is the case in price arbitrage) but from the underlying variability of the distributed VRES and the load. To this end, a simple set of expertly engineered rules can be proven to be a quite effective solution to the energy management problem of a microgrid [30]. As an extension to the rule-based control, a control strategy based on fuzzy logic is proposed in [31]. The logic implemented is similar to human reasoning in a way that it tolerates uncertainties and imprecision. A more complex approach, the so-called model predictive control (MPC), is based on solving an optimization model and requires forecasts of the uncertainty (typically induced by load and VRES variability). The output of the forecasting models, in combination with the system parameters, are used to compute the optimal control actions that need to be taken. The optimization of the operational control actions can be performed using the simulation model of the microgrid. MPC is a feedback control law that is meant to compensate for the realization of uncertainty and is often used for achieving economic efficiency in microgrid operation management [32], [33]. A comparison between the two distict methods, namely rule-based and MPC can be found in [34]. Probabilistic forecasting models attempt not only to provide the best point forecast[15] but instead capture the distribution of the uncertainty. The output of these models can be used to solve stochastic variants of MPC [35], [36]. In [35], a two-stage stochastic programming approach is applied to efficiently optimize microgrid operations while satisfying a time-varying request and operation constraints. Depending on the reliability concerns related to the microgrid use case, robust MPC can provide more secure ways of dealing with uncertainty [37].

1.1.3 Reinforcement learning

Reinforcement learning is a methodology that lies at the intersection between optimal control theory and machine learning. It is a branch of machine learning that deals with ways to learn control laws (known as policies) through experience. Reinforcement learning provides a framework to study and to optimize sequential decision making problems. It is based on trial and error and on the notion of receiving positive/negative feedback after each interaction of an agent with its environment. The considered agent learns a good control strategy or a

[15] The prediction of the expected value of a random variable.

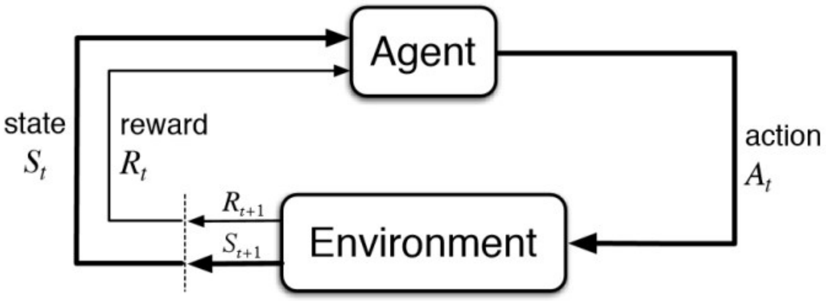


FIGURE 1.1: Schematic of the interactions between an agent with its environment in reinforcement learning.

good set of actions through positive and negative reinforcement. The main difference with respect to optimal control theory is that reinforcement learning relies on the availability of a simulator (that generates data from the interactions) for the design of a good control law, whereas dynamic programming requires a model of the environment.

There exist two broad classifications of the reinforcement learning algorithms. Depending on whether or not we have prior knowledge of the environment, we can categorize reinforcement learning methods in *offline* or *online*. The former class is used to train the agent on data that are generated in advance. After the agent is trained, it can start interacting with the real environment. On the other hand, online methods are applied when there is no data from the environment beforehand and the agent learns during the interactions with the environment. When a model (or any approximation) of the environment dynamics is available, it can be used (online or offline) to generate data which in turn can be used to accelerate learning and to speed-up the performance improvement of the agent. Depending on whether we have in our possession a model of the environment or not we can categorize reinforcement learning methods in *model-free* or *model-based*.

Background

Let us consider an agent that is interacting in an environment as illustrated in Figure 1.1. The agent, at each discrete decision step t, measures its current state s_t in the environment and takes a new action a_t in the environment. Subsequently, the environment performs a transition to the next state s_{t+1} and yields a scalar reward r_t. The transitions can be deterministic or stochastic. Depending on the actual application that is considered, the rewards that are received can be dense or sparse. This means that in some cases the agent may receive a reward signal only when the desirable task is achieved (e.g. escaping a maze), as opposing to receiving rewards for every action that is taken. The dynamics of the environment, the function that generates the reward signal, as well as the sets of states and actions, constitute what is known as a Markov decision process (MDP). The agent is considered to take an action a given the observed state s, following a policy π according to $a \sim \pi(s)$. Assuming an infinite horizon problem, the value V of being at the initial state s and following a policy π is defined as the expected cumulative reward collected and is given by:

$$V^{\pi}(s) = \mathbb{E}\left(\sum_{t=0}^{\infty} \gamma^{t} r_t | s_0 = s\right) \tag{1.1}$$

The parameter γ is called the discount factor and assumes values in the interval $(0,1]$. When the considered task has a finite horizon then we can consider $\gamma = 1$. In the case that the horizon is infinite, the value of γ is strictly less than one and attempts to emulate the fact that rewards collected far into the future are less important than immediate rewards. The value function V^{π} of a current policy π can be computed using trajectories that are produced by taking actions according to the considered policy. Once we compute the value function in practice we can evaluate how much better off (aligned with the agent goals) the agent is by being in one state versus another.

The goal in reinforcement learning is to optimize the policy function π in order to maximize the cumulative discounted rewards (value function) obtained by the interaction of the agent with the environment as:

$$\pi^{*} = \arg\max_{\pi} V^{\pi} \tag{1.2}$$

Reinforcement learning algorithms generally attempt to solve this optimization problem by estimating and optimizing these two elements, namely the value function and/or the policy. Depending on the way they accomplish that, they can be broken down into the following three subclasses:

- Value iteration algorithms that search for the optimal value function V^{*}, that represents the maximal cumulative rewards from every state that is visited. Subsequently, the optimal value function is used to compute an optimal policy according to equation (1.2).

- Policy iteration algorithms that proceed in two steps. First, they evaluate an existing policy by computing its value function, and subsequently they use this value function to update/improve the existing policy. These steps, namely policy evaluation and policy improvement are performed iteratively until convergence.

- Policy search algorithms that use optimization techniques to directly search for an optimal policy.

These algorithms have convergence and optimality guarantees when an exact (tabular) representation can be used for the value function, that is when the state and action spaces are discrete and low dimensional. However, this condition does not hold for many problems where the state and/or action spaces are continuous. For these problems, function approximators can be used instead of the exact representations of value and/or policy functions. An approximate version of the mentioned algorithms can then be applied in order to obtain approximately optimal policies [38]. By doing so, many issues arise regarding the convergence guarantees of the resulting approximate algorithms and the optimality gap of the obtained policies. Additionally, the selection process of an appropriate function approximator for a given problem is not a trivial task.

This section intends to provide an overview of the reinforcement learning framework and the existing categories of algorithms. The interested reader can refer to [39] for a detailed description of the basic reinforcement learning concepts and algorithms. Additionally, great resources for exact and approximate dynamic programming can be found in [40], [41]. The links between all different stochastic optimization methods, including reinforcement learning, can be found in [42]. A valuable source on the application of function approximation to reinforcement learning algorithms can be found in [38].

Recent advancements

While reinforcement learning and dynamic programming methods date back to the 1950s, they have recently received increasing attention. The main reason for that stems from the recent advancements in the field of Deep Learning (DL). DL is a subfield of machine learning (ML) where models (function approximators) are represented as a network of artificial neurons. Each neuron performs a linear algebraic operation to the incoming signal and, in combination with a non-linear activation function it can represent highly non-linear functions, given sufficient data. Research in artificial neural networks is a fairly old field of research as well. However, developments in recent years of new software regarding back-propagation in combination with the acceleration gained by using graphical processing units has significantly reduced the computational time. This has led to a substantial increase in the size of the neural networks i.e. the number of layers and the number of neurons in each layer, leading to the construction of deep neural networks with millions of tunable parameters. The increase in model complexity (number of parameters) that deep neural networks introduce is translated into the capacity to approximate very complex non-linear functions. This in turn, in conjunction with the increasing availability of large datasets, has led to impressive results in different ML tasks and has given birth to new types of neurons such as the convolutional or recurrent cells that are specialized in the way they process information for dedicated tasks such as computer vision or language translation.

The use of novel deep learning architectures in the field of reinforcement learning have given rise to a new field of research: deep reinforcement learning (DRL). In this field, the previously discussed components such as the value, the policy and the model are now approximated using deep neural networks. This has led to a number of very complex problems being successfully solved using DRL. In the past years, there has been an upsurge of novel algorithms starting from the deep Q-networks (DQN) [43] algorithm. The DQN agent, without any prior experience and only using raw pixels on the screen as its state, managed to reach human level performance in half of the 50 Atari games to which it was applied. Following this breakthrough, many improvements to this algorithm followed, such as further stabilising the learning dynamics [44], prioritising the replayed experiences [45], normalising [46], aggregating [47] and re-scaling [48] the outputs. The combination of these improvements has led to a large improvement in mean score across 50 Atari games. A number of additional improvements, have allowed the DRL agent to reach human-level performance in almost all of the Atari games [49]. Furthermore, another asynchronous and distributed algorithm, so-called asynchronous advantage actor-critic (A3C) [50], has managed to largely decrease the computation time while reaching new records in performances not only in Atari games but also on many Labyrinth tasks.

In March 2016 AlphaGo [51], a computer program that uses a combination of deep neural networks with a state-of-the-art tree search, defeated Lee Sedol, the world grandmaster in the game of Go. Later in 2018, an extension of AlphaGo called AlphaZero [52] managed to master the games of chess, shogi, and Go, beating a world-champion program in each case only by having knowledge of the rules of each game and no prior training. Later in 2020, a new model based algorithm called MuZero [53] managed to tackle all three games without any information about the rules of each game.

Besides its enormous success in games, deep reinforcement learning has been applied successfully to several real world problems. For instance, it was recently shown to have several appications in fluid mechanics [54]. In particular, DRL has been used for automating turbulence modelling with plenty of practical applications in aircraft design, weather forecasting and climate prediction [55]. Additionally, DRL has multiple applications in the field of robotics [56]. For instance, in [57] the authors present a DRL algorithm based on off-policy training of deep Q-functions that can scale to complex 3D manipulation tasks and can learn deep neural network policies in a scalable way so that they can be trained on real physical robots. Additionally, a novel algorithm for managing dynamic tasks like table tennis by robotic arms is proposed in [58]. The algorithm combines simulation and real training by randomly replaying recorded ball trajectories in simulation and applying actions to the real robot. Another quite promising field of application for DRL is healthcare [59], [60]. In particular, DRL has been proposed for the development of dynamic treatment strategies based on registry data [61] and for learning treatment policies for sepsis [62]. The potential of DRL has been recently exploited in various applications in the energy field. For instance, DRL agents demonstrated impressive results in an open challenge called Learn to Run a Power Network, organized by RTE (the French TSO) [63]. The goal of this challenge was to optimize the operation of a high voltage network while avoiding blackouts.

1.2 Contributions and outline of this thesis

In view of the great successes of DRL in solving challenging tasks, the goal of this thesis is to investigate its potential in solving complex problems related to the control of storage in modern energy systems. In particular, we investigate and develop novel operating strategies for energy storage units at different scales (i.e. grid-scale or distributed) and in different settings (i.e. interconnected or off-grid). Subsequently, we highlight the importance of jointly optimizing the size and the control of a storage system and propose an novel algorithm to address this problem. Moreover, in our work we develop modeling frameworks for the problems at hand that allow practitioners from various disciplines (i.e. computer science, engineering) to join forces and progressively tackle these problems.

We start in Chapter 2 of this thesis by addressing the energy arbitrage problem of a storage unit that participates in the short-term electricity markets. In particular, it is expected that energy transactions will take place closer to real time in order to reward flexibility resources and to enable better forecasting and control of VRES and electricity demand [64]. Motivated by this, we select to study, in the context of this thesis, the price arbitrage opportunities for storage units in the European Continuous Intraday (CID) market. We develop an operational strategy in order to maximize its arbitrage value. To this end, a novel modeling framework for the strategic participation of energy storage in the European CID market is proposed, where exchanges occur through a process similar to the stock market. A detailed description of the market mechanism and the storage system management is provided. The assumptions that allow the formulation of the problem of market participation for storage devices as a Markov Decision Process (MDP) are elaborated. A set of necessary simplifications that constitute the problem tractable are described. The resulting problem is solved using a state-of-the-art DRL algorithm. The outcome of the proposed method is compared with the state-of-the-art industrial practices and the resulting policy is found able to outperform this benchmark. Additionally, we discuss a number of limitations arising from the proposed implementation that are related to: i) the insufficient amount of relevant information contained in the state variable and ii) the limited state space exploration.

In Chapter 3 of this thesis, we address the limitations identified in Chapter 2, related to the state space exploration. In particular, we introduce a set of modifications to the described CID market participation problem that lead to a significant increase in the general performance of the proposed strategy. First, we motivate the use of a more compact state space representation and we propose the use of day-ahead prices in order to stationarize the states observed. We then proceed by normalizing the trading rewards in each day, by dividing them with the total profits obtained by the benchmark strategy. The proposed changes are evaluated in a new case study. In order to obtain a good grasp of the performance improvement potential we define a new benchmark that is anticipative, i.e. the policy has access to the future rewards and can act accordingly. The results demonstrate that our method can outperform the benchmark and reach a performance that is comparable to the anticipative policy.

In Chapter 4 of this thesis, we address the energy arbitrage problem faced by an off-grid microgrid operator in the context of rural electrification. In particular, we deal with the lifelong control problem of an isolated microgrid. The set of changes that may occur over its life span are categorized in progressive and abrupt changes. The main challenges for an effective control policy stem from the various changes that take place over time. Generally speaking, an operational strategy that relies on MPC has shown to be highly effective for the control of an off-grid microgrid. In this work, inspired by the comparison and the similarities between MPC and reinforcement learning, as they are presented in [65], we propose a novel model-based reinforcement learning algorithm that is able to address both types of changes. The algorithm demonstrates generalisation properties, transfer capabilities and better robustness in case of fast-changing system dynamics. The proposed algorithm is compared against two benchmarks, namely a rule-based and an MPC controller. The results show that the trained agent is able to outperform both benchmarks in the lifelong setting where the system dynamics are changing over time.

In Chapter 4, we also argue that in the context of an off grid-microgrid, the optimal size of the components (i.e. the capacity of photovoltaic (PV) panels, storage) depends heavily on the control policy applied. When the capacity of the installed components is large, a myopic policy can be as good as a look-ahead policy. On the other hand, a good policy that is able to anticipate changes and to act accordingly allows for the reduction of the components size and subsequently the investment cost. Generally speaking, the size of a system and the control that is applied to it are highly interdependent. In Chapter 5, we propose a new methodology for jointly sizing a dynamical system and designing its control law. First, the problem is formalized by considering parametrized reinforcement learning environments and parametrized policies. The objective of the optimization problem is to jointly find a control policy and an environment over the joint hypothesis space of parameters such that the sum of rewards gathered by the policy in this environment is maximal. The optimization problem is then addressed by generalizing the direct policy search algorithms to an algorithm we call Direct Environment Search with (projected stochastic) Gradient Ascent (DESGA). We illustrate the performance of DESGA on two benchmarks. First, we consider a parametrized space of Mass-Spring-Damper (MSD) environments and control policies. Then, we use our algorithm for optimizing the size of the components and the operation of a small-scale autonomous energy system, i.e. a solar off-grid microgrid, composed of photovoltaic panels, batteries. On both benchmarks, we compare the results of the execution of DESGA with a theoretical upper-bound on the expected return. Furthermore, the performance of DESGA is compared to an alternative algorithm. The latter performs a grid discretization of the environment's hypothesis space and applies the REINFORCE algorithm [66] to identify pairs of environments and policies resulting in a high expected return. The choice of this algorithm is also discussed and motivated. On both benchmarks, we show that DESGA and the alternative algorithm result in a set of parameters for which the expected return is nearly equal to its theoretical upper-bound. Nevertheless, the execution of DESGA is much less computationally costly.

Finally, in Chapter 6, we provide the general conclusions and remarks of this thesis and we propose a list of future research directions that emerge as an outcome of this work.

1.3 Publications

This thesis is based on a number of scientific articles in the field of DRL for the energy management of storage. The list of papers as well as a personal contribution statement for each one of them are hereby presented:

- [67] *A Deep Reinforcement Learning Framework for Continuous Intraday Market Bidding*, **Ioannis Boukas**, Damien Ernst, Thibaut Théate, Adrien Bolland, Alexandre Huynen, Martin Buchwald, Christelle Wynants and Bertrand Cornélusse, *Accepted* with minor revisions in Machine Learning Springer:
  Conceptualization, formal analysis, investigation, methodology, data curation, software, validation, writing - original draft, writing - review & editing, project administration

- [68] *Lifelong Control of Off-grid Microgrid with Model Based Reinforcement Learning*, Simone Totaro[16], **Ioannis Boukas**[16], Anders Jonsson and Bertrand Cornélusse, Under review in Energy Elsevier:
  Conceptualization, formal analysis, investigation, methodology, data curation, software, writing - original draft, writing - review & editing, project administration

- [69] *Learning optimal environments using projected stochastic gradient ascent*, Adrien Bolland, **Ioannis Boukas**, François Cornet, Mathias Berger and Damien Ernst, Submitted in Journal of Artificial Intelligence Research:
  Formal analysis, methodology, data curation, software, validation, writing - original draft

Additionally, research work in the context of this thesis has led to the publication/submission of the following articles that are not included in this manuscript:

- [70] *Intra-day bidding strategies for storage devices using deep reinforcement learning*, **Ioannis Boukas**, Damien Ernst, Anthony Papavasiliou and Bertrand Cornélusse, In 2018 15th International Conference on the European Energy Market (EEM),
  EEM 2018 Best student paper award

- [71] *Real-time bidding strategies from micro-grids using reinforcement learning*, **Ioannis Boukas**, Damien Ernst and Bertrand Cornélusse, In Proceedings of CIRED Workshop 2018

- [72] *Probabilistic Forecasting of Imbalance Prices in the Belgian Context*, Jonathan Dumas, **Ioannis Boukas**, Miguel Manuel de Villena, Sébastien Mathieu and Bertrand Cornélusse, In 2019 16th International Conference on the European Energy Market (EEM)

[16] Equal contribution

- [73] *Sizing and Operation of an Isolated Microgrid with Cold Storage*, Selmane Dakir, **Ioannis Boukas**, Vincent Lemort and Bertrand Cornélusse, In 2019 IEEE Milan PowerTech

- [28] *Sizing and Operation of an Isolated Microgrid With Building Thermal Dynamics and Cold Storage*, Selmane Dakir, **Ioannis Boukas**, Vincent Lemort and Bertrand Cornélusse, In IEEE Transactions on Industry Applications

- [74] *A Framework to Integrate Flexibility Bids into Energy Communities to Improve Self-Consumption*, Miguel Manuel de Villena, **Ioannis Boukas** and Sebastien Mathieu, Eric Vermeulen and Damien Ernst, In 2020 IEEE Power Energy Society General Meeting (PESGM)

- *Analyzing Trade in Continuous intra-day Electricity Market: An Agent-based Modeling Approach*, Priyanka Shinde [17], **Ioannis Boukas** [17], David Radu, Miguel Manuel de Villena, Mikael Amelin, Submitted in Energies

[17] Equal contribution

Chapter 2

A Deep Reinforcement Learning Framework for Continuous Intraday Market Bidding

In this chapter, we address the energy arbitrage problem of a storage unit that participates in the European CID market. In particular, we aim at developing an operational strategy in order to maximize its arbitrage value. To this end, a novel modeling framework for the strategic participation of energy storage in the European CID market is proposed, where exchanges occur through a process similar to the stock market. A detailed description of the market mechanism and the storage system management is provided. The assumptions that allow the formulation of the problem of market participation for storage devices as a Markov Decision Process (MDP) are elaborated. A set of necessary simplifications that make the problem tractable are described. The resulting problem is solved using a DRL algorithm. The outcome of the proposed method is compared with the state-of-the-art industrial practices and the resulting policy is found able to outperform this benchmark.

2.1 Introduction

The vast integration of renewable energy resources (RES) into (future) power systems, as directed by the recent worldwide energy policy drive [75], has given rise to challenges related to the security, sustainability and affordability of the power system ("The Energy Trilemma"). The impact of high RES penetration on the modern short-term electricity markets has been the subject of extensive research over the last few years. Short-term electricity markets in Europe are organized as a sequence of trading opportunities where participants can trade energy in the day-ahead market and can later adjust their schedule in the intraday market until the physical delivery. Deviations from this schedule are then corrected by the transmission system operator (TSO) in real time and the responsible parties are penalized for their imbalances [76].

Imbalance penalties serve as an incentive for all market participants to accurately forecast their production and consumption and to trade based on these forecasts [77]. Due to the variability and the lack of predictability of RES, the output planned in the day-ahead market may differ significantly from the actual RES output in real time [78]. Since the RES forecast error decreases substantially with a shorter prediction horizon, the intraday market allows RES operators to trade these deviations whenever an improved forecast is available [79]. As a consequence, intraday trading is expected to reduce the costs related to the reservation and activation of capacity for balancing purposes. The intraday market is therefore a key aspect towards the cost-efficient RES integration and enhanced system security of supply.

Owing to the fact that commitment decisions are taken close to real time, the intraday market is a suitable market floor for the participation of flexible resources (i.e. units able to rapidly increase or decrease their generation/consumption). However, fast-ramping thermal units (e.g. gas power plants) incur a high cost when forced to modify their output, to operate in part load, or to frequently start up and shut down. The increased cost related to the cycling of these units will be reflected to the offers in the intraday market [64]. Alternatively, flexible storage devices (e.g. pumped hydro storage units or batteries) with low cycling and zero fuel cost can offer their flexibility at a comparatively low price, close to the gate closure. Hence, they are expected to play a key role in the intraday market.

2.1.1 Intraday markets in Europe

In Europe, the intraday markets are organized in two distinct designs, namely auction-based or continuous trading.

In auction-based intraday markets, participants can submit their offers to produce or consume energy at a certain time slot until gate closure. After the gate closure, the submitted offers are used to form the aggregate demand and supply curves. The intersection of the aggregate curves defines the clearing price and quantity [80]. The clearing rule is uniform pricing, according to which there is only one clearing price at which all transactions occur. Participants are incentivized to bid at their marginal cost since they are paid at the uniform price. This mechanism increases price transparency, although it leads to inefficiencies, since imbalances after the gate closure can no longer be traded [81].

In continuous intraday (CID) markets, participants can submit at any point during the trading session orders to buy or to sell energy. The orders are treated according to the first come first served (FCFS) rule. A transaction occurs as soon as the price of a new “Buy" (“Sell") order is equal or higher (lower) than the price of an existing “Sell" (“Buy") order. Each transaction is settled following the pay-as-bid principle, stating that the transaction price is specified by the oldest order of the two present in the order book. Unmatched orders are stored in the order book and are accessible to all market participants. The energy delivery resolution offered by the CID market in Europe ranges between hourly, 30-minute and 15-minute products, and the gate closure takes place between five and 60 minutes before actual delivery. Continuous trading gives the opportunity to market participants to trade imbalances as soon as they appear [81]. However, the FCFS rule is inherently associated with lower allocative inefficiency compared to auction rules. This implies that, depending on the time of arrival of the orders, some trades with a positive welfare contribution may not occur while others with negative welfare contribution may be realised [82]. It is observed that a combination of continuous and auction-based intraday markets can increase the market efficiency in terms of liquidity and market depth, and results in reduced price volatility [80].

In practice, the available contracts (“Sell" and “Buy" orders) can be categorized into three types:

- The market order, where no price limit is specified (the order is matched at the best price)
- The limit order, which contains a price limit and can only be matched at that or at a better price
- The market sweep order, which is executed immediately (fully or partially) or gets cancelled.

Limit orders may appear with restrictions related to their execution and their validity. For instance, an order that carries the specification *Fill or Kill* should either be fully and immediately executed or cancelled. An order that is specified as *All or Nothing* remains in the order book until it is entirely executed [83].

The European Network Codes and specifically the capacity allocation and congestion management guidelines [76] (CACM GL) suggest that continuous trading should be the main intraday market mechanism. Complementary regional intraday auctions can also be put in place if they are approved by the regulatory authorities [76]. To that direction, the Cross-Border Intraday (XBID) Initiative [84] has enabled continuous cross-border intraday trading across Europe. Participants of each country have access to orders placed from participants of any other country in the consortium through a centralized order book, provided that there is available cross-border capacity.

2.1.2 Bidding strategies in literature

The strategic participation of power producers in short-term electricity markets has been extensively studied in the literature. In order to co-optimize the decisions made in the sequential trading floors from day-ahead to real time the problem has been traditionally addressed using multi-stage stochastic optimization. Each decision stage corresponds to a trading floor (i.e. day-ahead, capacity markets, real-time), where the final decisions take into account uncertainty using stochastic processes. In particular, the influence that the producer may have on the market price formation leads to the distinction between “price-maker" and “price-taker" and results in a different modelling of the uncertainty.

In [85], the optimization of a portfolio of generating assets over three trading floors (i.e. the day-ahead, the adjustment and the reserves market) is proposed, where the producer is assumed to be a “price-maker". The offering strategy of the producer is a result of the stochastic residual demand curve as well as the behaviour of the rest of the market players. On the contrary, a “price-taker" producer is considered in [86] for the first two stages of the problem studied, namely the day-ahead and the automatic generation control (AGC) market. However, since the third-stage (balancing market) traded volumes are small, the producer can negatively affect the prices with its participation. Price scenarios are generated using ARIMA models for the two first stages, whereas for the third stage a linear curve with negative slope is used to represent the influence of the producer’s offered capacity on the market price.

Hydro-power plant participation in short-term markets accounting for the technical constraints and several reservoir levels is formulated and solved in [87]. Optimal bidding curves for the participation of a “price-taker" hydro-power producer in the Nordic spot market are derived accounting for price uncertainty. In [88], the bidding strategy of a two-level reservoir plant is casted as a multi-stage stochastic program in order to represent the different sequential trading floors, namely the day-ahead spot market and the hour-ahead balancing market. The effects of coordinated bidding and the “price-maker" versus “price-taker" assumptions on the generated profits are evaluated. In [89], bidding strategies for a virtual power plant (VPP) buying and selling energy in the day-ahead and the balancing market in the form of a multi-stage stochastic optimization are investigated. The VPP aggregates a pumped hydro energy storage (PHES) unit as well as a conventional generator with stochastic intermittent power production and consumption. The goal of the VPP operator is the maximization of the expected profits under price uncertainty.

In these approaches, the intraday market is considered as auction-based and it is modelled as a single recourse action. For each trading period, the optimal offered quantity is derived according to the realization of various stochastic variables. However, in reality, for most European countries, according to the EU Network Codes [76], modern intraday market trading will primarily be a continuous process.

The strategic participation in the CID market is investigated for the case of an RES producer in [82] and [90]. In both works, the problem is formulated as a sequential decision-making process, where the operator adjusts its offers during the trading horizon, according to the RES forecast updates for the physical delivery of power. Additionally, in [91] the use of a PHES unit is proposed to undertake energy arbitrage and to offset potential deviations. The trading process is formulated as a Markov Decision Process (MDP) where the future commitment decision in the market is based on the stochastic realization of the intraday price, the imbalance penalty, the RES production and the storage availability.

The volatility of the CID prices, along with the quality of the forecast updates, are found to be key factors that influence the degree of activity and success of the deployed bidding strategies [82]. Therefore, the CID prices and the forecast errors are considered as correlated stochastic processes in [90]. Alternatively, in [82], the CID price is constructed as a linear function of the offered quantity with an increasing slope as the gate closure approaches. In this way, the scarcity of conventional units approaching real time is reflected. In [91], real weather data and market data are used to simulate the forecast error and CID price processes.

For the sequential decision-making problem in the CID market, the offered quantity of energy is the decision variable to be optimized [90]. The optimization is carried out using Approximate Dynamic Programming (ADP) methods, where a parameterised policy is obtained based on the observed stochastic processes for the price, the RES error and the level of the reservoir [91]. The ADP approach presented in [91] is compared in [92] to some threshold-based heuristic decision rules. The parameters are updated according to simulation-based experience and the obtained performance is comparable to the ADP algorithm. The obtained decision rules are intuitively interpretable and are derived efficiently through simulation-based optimization.

The bidding strategy deployed by a storage device operator participating in a slightly different real-time market organized by NYISO is presented in [22]. In this market, the commitment decision is taken one hour ahead of real-time and the settlements occur intra-hour every five minutes. In this setting, the storage operator selects two price thresholds at which the intra-hour settlements occur. The problem is formulated as an MDP and is solved using an ADP algorithm that exploits a particular monotonicity property. A distribution-free variant that assumes no knowledge of the price distribution is proposed. The optimal policy is trained using historical real-time price data.

Even though the focus of the mentioned articles lies on the CID market, the trading decisions are considered to take place in discrete time-steps. A different approach is presented in [93], where the CID market participation is modelled as a continuous time process using stochastic differential equations (SDE). The Hamilton Jacobi Bellman (HJB) equation is used for the determination of the optimal trading strategy. The goal is the minimization of the imbalance cost faced by a power producer arising from the residual error between the RES production and demand. The optimal trading rate is derived assuming a stochastic process for the market price using real market data and the residual error.

In the approaches presented so far, the CID price is modelled as a stochastic process assuming that the participating agent is a “price-taker". However, in the CID market, this assumption implies that the CID market is liquid and the price at which one can buy or sell energy at a given time are similar or the same. This assumption does not always hold, since the mean bid-ask spread in a trading session in the German intraday market for 2015 was several hundred times larger than the tick-size (i.e. the minimum price movement of a trading instrument) [94]. It is also reported in the same study that the spread decreases as trading approaches the gate closure.

An approach that explicitly considers the order book is presented in [95]. A threshold-based policy is used to optimize the bid acceptance for storage units participating in the CID market. A collection of different factors such as the time of the day are used for the adaptation of the price thresholds. The threshold policy is trained using a policy gradient method (REINFORCE) and the results show improved performance against the *rolling intrinsic* benchmark.

The *rolling intrinsic* benchmark was originally introduced in [96] as a gas storage valuation method and relies on repeated re-optimization as new price information arrives. According to this method, the trader starts with an initial position and when new information about the prices arrives it calculates whether the profit of (partially) changing its position and taking the optimal position based on these new prices outweighs the transaction costs. The *rolling intrinsic* strategy yields profits if the spread between different tradable products changes sign and if it makes sense to swap trading decisions. This strategy, although risk-free, is not fundamentally maximizing profit.

2.1.3 Contributions of the chapter

In this chapter, we focus on the sequential decision-making problem related to the optimal operation of a storage device participating in the CID market. Firstly, we present a novel modelling framework for the CID market, where the trading agents exchange energy via a centralized order book. Each trading agent is assumed to dynamically select the orders that maximize its benefits throughout the trading horizon. Secondly, we model the asset trading process and describe explicitly the dynamics of the storage system.

We elaborate on a set of assumptions that allow the formulation of the resulting problem as an MDP. In particular, we consider that the strategy of the trading agents is modeled by a stochastic process that depends on the previous order book observations. The exogenous information to the trading process is considered to be the outcome of a time-dependent stochastic model and the charging/discharging decisions of the storage unit are always such that they minimize any resulting imbalances. Additionally, in order to reduce the possible trading actions, we assume that the trading agent can only select existing orders and is not able to post new free-standing offers (aggressor). In order to fully comply with German regulation policies, we further restrict the agent to select orders if and only if it does not result in any imbalances. Lastly, since in practice the storage unit is used for other operational obligations (reserves etc.), we consider that its initial and final state of charge for each day are decided in advance and are fixed during each CID trading session. This assumption allows the decoupling of the full optimization horizon in smaller (daily) windows.

Due to the high-dimensionality and the dynamically evolving size of the order book, we propose a novel low-dimensional order book representation that allows to capture the relevant order book information about the arbitrage opportunities of a storage unit. In particular, we pool the available orders and we engineer features that serve as a proxy of the potential benefit from this order book configuration for a storage device. Additionally, due to the dynamically evolving size of the order book, the set of possible actions is still large despite our assumption on our agent being an aggressor. We thus define a set of two high level actions, i.e. “Trade" and “Idle". The new action space allows us to design a set of policies that are variants of the *rolling intrinsic* strategy, where instead of sequentially repeating the optimization steps as new information arrives, we introduce the possibility to wait.

In the absence of a realistic model for the rest of the participants in the market we use historical data, to construct the trading environment in which the storage agent engages. The CID market trading problem of a storage device is solved using Deep Reinforcement Learning techniques, specifically an asynchronous distributed variant of the fitted Q iteration RL algorithm with deep neural networks as function approximators [97]. The resulting policy is evaluated using real data from the German CID market [98]. The results suggest that the designed trading agent has the ability to identify the moments in which it would be better off by waiting based on a sequence of market indicators as well as other exogenous information. In summary, the contributions of this work are the following:

- We model the CID market trading process as an MDP where the energy exchanges occur explicitly through a centralized order book.
- We construct a novel state representation in order to provide a structured lower dimensional representation of the order book.
- We derive, using a batch-mode reinforcement algorithm, an operational policy that is able to identify the opportunity cost between trading and idling.

2.1.4 Outline of the chapter

The rest of the chapter is organized as follows. In Section 2.2, the CID market trading framework is presented. The interaction of the trading agents via a centralized order book is formulated as a dynamic process. All the available information for an asset trading agent is detailed and the objective is defined as the cumulative profits. In Section 2.3, all the assumptions necessary to formulate the bidding process in the CID market as an MDP are listed. The methodology utilised to find an optimal policy that maximizes the cumulative profits of the proposed MDP is detailed in Section 2.4. A case study using real data from the German CID market is performed in Section 2.5. The results as well as considerations about limitations of the developed methodology are discussed in Section 2.6. Finally, conclusions of this work are drawn and future recommendations are provided in Section 2.7. A detailed nomenclature is provided at the Appendix 2.8.

2.2 Continuous Intraday Bidding process

In this section, we firstly present a detailed description of the CID market mechanism. Secondly, we model the dynamics and the decision-making process of an asset trading agent

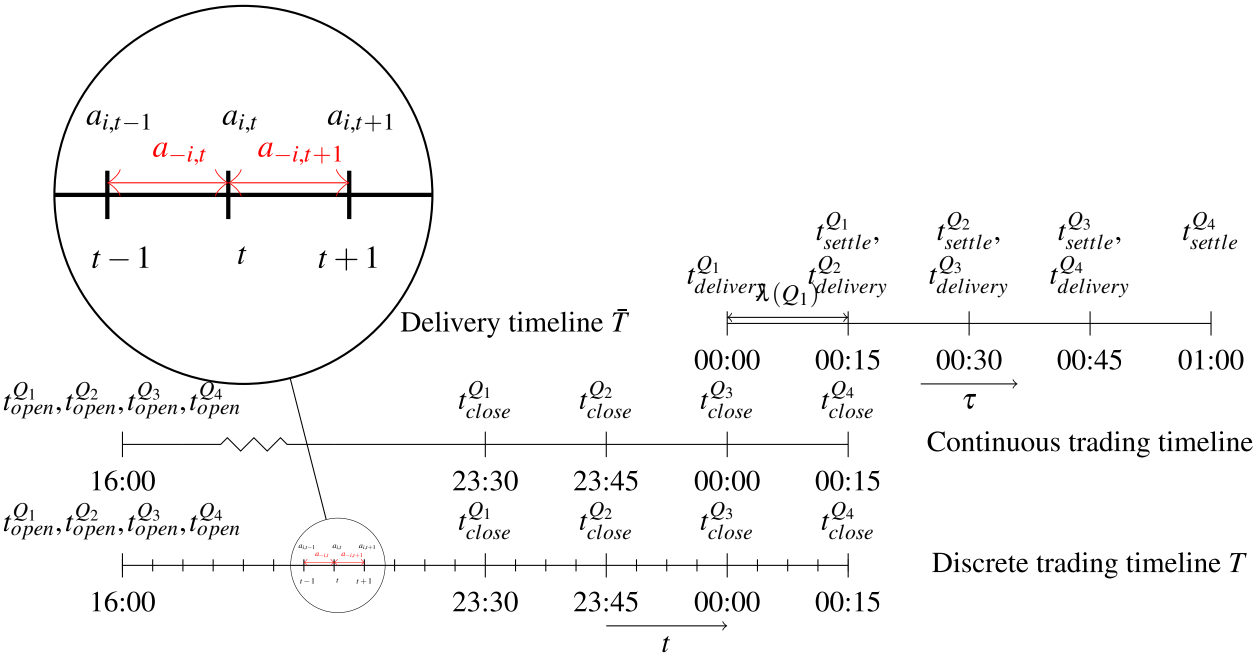


FIGURE 2.1: Trading (continuous and discrete) and delivery timelines for products Q_1 to Q_4

that participates in the CID market. The goal of the presented framework is to describe in a generic way the process under consideration. In the following sections, we introduce a number of assumptions and restrictions to this generic framework targeting a problem that can be tractable to solve.

2.2.1 Continuous Intraday market design

The participation in the CID market is a continuous process similar to the stock exchange. Each market product $x \in X$, where X is the set of all available products, is defined as the physical delivery of energy in a pre-defined time slot. The time slot corresponding to product x is defined by its starting point $t_{delivery}(x)$ and its duration $\lambda(x)$. The trading process for time slot x opens at $t_{open}(x)$ and closes at $t_{close}(x)$. During the time interval $t \in [t_{open}(x), t_{close}(x)]$, a participant can exchange energy with other participants for the lagged physical delivery during the interval $\delta(x)$, with:

$$\delta(x) = [t_{delivery}(x), t_{delivery}(x) + \lambda(x)].$$

The exchange of energy takes place through a centralized order book that contains all the unmatched orders o_j, where $j \in N_t$ corresponds to a unique index that every order receives upon arrival. The set $N_t \subseteq \mathbb{N}$ gathers all the unique indices of the orders available at time t.

We denote the status of the order book at time t by $s_t^{OB} = (o_j, \forall j \in N_t)$. As time progresses new orders appear and existing ones are either accepted or cancelled.

Trading for a set of products is considered to start at the gate opening of the first product and to finish at the gate closure of the last product. More formally, considering an ordered set of available products (hourly, half-hourly and quarter-hourly) $X = \{H_1,..,H_{24},HH_1,...,HH_{48}, Q_1,...,Q_{96}\}$, the corresponding trading horizon is defined as $T = [t_{open}(Q_1), t_{close}(Q_{96})]$. For instance, in the German CID market, trading of hourly (quarter-hourly) products for day D opens at 3 pm (4 pm) of day $D-1$ respectively. For each product x, the gate closes 30 minutes before the actual energy delivery at $t_{delivery}(x)$. The timeline for trading products Q_1 to Q_4 that correspond to the physical delivery in 15-minute time slots from 00:00 until 01:00, is presented in Figure 2.1. It can be observed that the agent can trade for all products until 23:30. After each subsequent gate closure the number of available products decreases and the commitment for the corresponding time slot is defined. Potential deviations during the physical delivery of energy are penalized in the imbalance market.

2.2.2 Continuous Intraday market environment

As its name indicates, the CID market is a continuous environment. In order to solve the trading problem presented in this chapter, it has been decided to perform a relevant discretization operation. As shown in Figure 1, the trading timeline is discretised in a high number of time-steps of constant duration Δt. Each discretised trading interval for product x can be denoted by the set of time-steps $T(x) = \{t_{open}(x), t_{open}(x) + \Delta t, ..., t_{close}(x) - \Delta t, t_{close}(x)\}$. Then, the discrete-time trading opportunities for the entire set of products X can be modelled such that the time-steps are defined as $t \in T = \bigcup_{x \in X} T(x)$. In the following, for the sake of clarity, the increment (decrement) operation $t+1$ $(t-1)$ will be used to model the discrete transition from time-step t to time-step $t + \Delta t$ $(t - \Delta t)$.

It is important to note that in theory the discretization operation leads to suboptimalities in the decision-making process. However, as the discretization becomes finer ($\Delta t \to 0$), the decisions taken can be considered near-optimal. Increasing the granularity of the decision time-line results in an increase of the number of decisions that can be taken and hence, the size of the decision-making problem. Thus, there is a clear trade-off between complexity and quality of the resulting decisions when using a finite discretization.

Let X_t denote the set of available products at time-step $t \in T$ such that:

$$X_t = \{x | x \in X, t \leq t_{close}(x)\}.$$

We define the state of the CID market environment at time-step t as $s_t^{OB} \in S^{OB}$. The state contains the observation of the order book at time-step $t \in T$ i.e. the unmatched orders for all the available products $x \in X_t \subset X$.

A set of n agents $I = \{1, 2, ..., n\}$ are continuously interacting in the CID environment exchanging energy. Each agent $i \in I$ can express its willingness to buy or sell energy by posting at instant t a set of new orders $a_{i,t} \in A_i$ in the order book, which results in the joint action $a_t = (a_{1,t}, ..., a_{n,t}) \in \prod_{i=1}^{n} A_i$.

The process of designing the set of new orders $a_{i,t}$ for agent i at instant t consists, for each new order, in determining the product $x \in X_t$, the side of the order $y \in \{\text{“}Sell\text{”}, \text{“}Buy\text{”}\}$, the volume $v \in \mathbb{R}^+$, the price level $p \in [p_{min}, p_{max}]$ of each unit offered to be produced or consumed, and the various validity and execution specifications $e \in E$. The index of each new order j belongs to the set $j \in N_t'$.

The set of new orders is defined as $a_{i,t} = ((x_j, y_j, v_j, p_j, e_j), \forall j \in N_t' \subseteq \mathbb{N})$. We will use the notation for the joint action $a_t = (a_{i,t}, a_{-i,t})$ to refer to the action that agent i selects $a_{i,t}$ and the joint action that all other agents use $a_{-i,t} = (a_{1,t}, ..., a_{i-1,t}, a_{i+1,t}, ..., a_{n,t})$.

TABLE 2.1: Order Book for Q_1 and time slot 00:00-00:15

i	**Side**	v **[MW]**	p **[€/MWh]**	
4	“Sell”	6.25	36.3	
2	“Sell”	2.35	34.5	⟵ ask
1	“Buy”	3.15	33.8	⟵ bid
3	“Buy”	1.125	29.3	
5	“Buy”	2.5	15.9	

The orders are treated according to the first come first served (FCFS) rule. Table 2.1 presents an observation of the order book for product Q_1. The difference between the most expensive “Buy" order (“bid") and the cheapest “Sell" order (“ask") defines the bid-ask spread of the product. A deal between two counter-parties is struck when the price p_{buy} of a “Buy” order and the price p_{sell} of a “Sell” order satisfy the condition $p_{buy} \geq p_{sell}$. This condition is tested at the arrival of each new order. The volume of the transaction is defined as the minimum quantity between the “Buy” and “Sell” order ($\min(v_{buy}, v_{sell})$). The residual volume remains available in the market at the same price. As mentioned in the previous section, each transaction is settled following the pay-as-bid principle, at the price indicated by the oldest order.

Finally, at each time-step t, every agent i observes the state of the order book s_t^{OB}, performs certain actions (posting a set of new orders) $a_{i,t}$, inducing a transition which can be represented by the following equation:

$$s_{t+1}^{OB} = f(s_t^{OB}, a_{i,t}, a_{-i,t}). \tag{2.1}$$

2.2.3 Asset trading

An asset optimizing agent participating in the CID market can adjust its position for product x until the corresponding gate closure $t_{close}(x)$. However, the physical delivery of power is decided at $t_{delivery}(x)$. An additional amount of information (potentially valuable for certain players) is received during the period $\{t_{close}(x), .., t_{delivery}(x)\}$, from the gate closure until the delivery of power. Based on this updated information, an asset-trading agent may need to or have an incentive to deviate from the net contracted power in the market.

Let $v_{i,t}^{con} = (v_{i,t}^{con}(x), \forall x \in X_t) \in \mathbb{R}^{|X_t|}$, gather the volumes of power contracted by agent i for the available products $x \in X_t$ at each time-step $t \in T$. In the following, we will adopt the convention for $v_{i,t}^{con}(x)$ to be positive when agent i contracts the net volume to sell (produce) and negative when the agent contracts the volume to buy (consume) energy for product x at time-step t.

Following each market transition as indicated by equation (2.1), the volumes contracted $v_{i,t}^{con}$ are determined based on the transactions that have occurred. The contracted volumes $v_{i,t}^{con}$ are derived according to the FCFS rule that is detailed in [99]. The mathematical formulation of the clearing algorithm is provided in [100]. The objective function of the clearing algorithm is comprised of two terms, namely the social welfare and a penalty term modelling the price-time priority rule. The orders that maximize this objective are matched, provided that they satisfy the balancing equations and constraints related to their specifications. The clearing rule is implicitly given by:

$$v_{i,t}^{con} = clear(i, s_t^{OB}, a_{i,t}, a_{-i,t}). \tag{2.2}$$

We denote as $P_{i,t}^{mar}(x) \in \mathbb{R}$ the net contracted power in the market by agent i for each product $x \in X$, which is updated at every time-step $t \in T$ according to:

$$P_{i,t+1}^{mar}(x) = P_{i,t}^{mar}(x) + v_{i,t}^{con}(x). \tag{2.3}$$
$$\forall x \in X_t$$

The discretization of the delivery timeline $\bar{T}$ is done with time-steps of duration $\Delta\tau$, equal to the minimum duration of delivery for the products considered. The discrete delivery timeline $\bar{T}$ is considered to start at the beginning of delivery of the first product τ_{init} and to finish at the end of the delivery of the last product τ_{term}. For the simple case where only four quarter-hourly products are considered, as shown in Figure 2.1, the delivery time-step is $\Delta\tau = 15min$ and the delivery timeline $\bar{T} = \{00:00, 00:15, ..., 01:00\}$, where $\tau_{init} = 00:00$ and $\tau_{term} = 01:00$. In general, when only one type of product is considered (e.g. quarter-hourly), there is a straightforward relation between time of delivery τ and product x, since $\tau = t_{delivery}(x)$ and $\Delta\tau = \lambda(x)$. Thus, terms x or τ can be used interchangeably. For the sake of keeping the notation relatively simple, we will only consider quarter-hourly products in the rest of the chapter. In such a context, the terms $P_{i,t}^{mar}(\tau)$ or $P_{i,t}^{mar}(x)$ can be used interchangeably to denote the net contracted power in the market by agent i at trading step t for delivery time-step τ (product x).

As the trading process evolves the set of delivery time-steps τ for which the asset-optimizing can make decisions decreases as trading time t crosses the delivery time τ. Let $\bar{T}(t) \subseteq \bar{T}$ be a function that yields the subset of delivery time-steps $\tau \in \bar{T}$ that follow time-step $t \in T$ such that:

$$\bar{T}(t) = \{\tau | \tau \in \bar{T} \setminus \{\tau_{term}\}, t \leq \tau\}.$$

The participation of an asset-optimizing agent in the CID market is composed of two coupled decision processes with different timescales. First, the trading process where a decision is taken at each time-step t about the energy contracted until the gate closure $t_{close}(x)$. During this process, the agent can decide about its position in the market and create scenarios/make projections about the actual delivery plan based on its position. Second, the physical delivery decision that is taken at the time of the delivery τ or $t_{delivery}(x)$ based on the total net contracted power in the market during the trading process.

An agent i participating in the CID market is assumed to monitor the state of the order book s_t^{OB} and its net contracted power in the market $P_{i,t}^{mar}(x)$ for each product $x \in X$, which becomes fixed once the gate closure occurs at $t_{close}(x)$. Depending on the role it presumes in the market, an asset-optimizing agent is assumed to monitor all the available information about its assets. We distinguish the three following cases among the many different roles that can be played by an agent in the CID market:

- *The agent controls a physical asset that can generate and/or consume electricity.* We define as $G_{i,t}(\tau) \in \left[\underline{G_i}, \overline{G_i}\right]$ the power production level for agent i at delivery time-step τ as computed at trading step t. In a similar way, we define the power consumption level $C_{i,t}(\tau) \in \left[\underline{C_i}, \overline{C_i}\right]$, where $\underline{C_i}, \overline{C_i}, \underline{G_i}, \overline{G_i} \in \mathbb{R}^+$. We further assume that the actual production $g_{i,t}(t')$ and consumption level $c_{i,t}(t')$ during the time-period of delivery $t' \in [\tau, \tau + \Delta\tau)$, is constant for each product x such that:

$$g_{i,t}(t') = G_{i,t}(\tau), \tag{2.4}$$

$$c_{i,t}(t') = C_{i,t}(\tau), \tag{2.5}$$

$$\forall t' \in [\tau, \tau + \Delta\tau).$$

At each time-step t during the trading process, agent i can decide to adjust its generation level by $\Delta G_{i,t}(\tau)$ or its consumption level by $\Delta C_{i,t}(\tau)$. According to these adjustments the generation and consumption levels can be updated at each time-step t according to:

$$G_{i,t+1}(\tau) = G_{i,t}(\tau) + \Delta G_{i,t}(\tau), \tag{2.6}$$

$$C_{i,t+1}(\tau) = C_{i,t}(\tau) + \Delta C_{i,t}(\tau), \tag{2.7}$$

$$\forall \tau \in \bar{T}(t).$$

Let $w_{i,t}^{exog}$ denote any other relevant exogenous information to agent i such as the RES forecast, a forecast of the actions of other agents, or the imbalance prices. The computation of $\Delta G_{i,t}(\cdot)$ and $\Delta C_{i,t}(\cdot)$ depends on the market position, the technical limits of the assets, the state of the order book and the exogenous information $w_{i,t}^{exog}$. We define the residual production $P_{i,t}^{res}(\tau) \in \mathbb{R}$ at delivery time-step τ as the difference between the production and the consumption levels and can be computed by:

$$P_{i,t}^{res}(\tau) = G_{i,t}(\tau) - C_{i,t}(\tau). \tag{2.8}$$

We note that the amount of residual production $P_{i,t}^{res}(\tau)$ aggregates the combined effects that $G_{i,t}(\tau)$ and $C_{i,t}(\tau)$ have on the revenues made by agent i through interacting with the markets (intraday/imbalance).

The level of generation and consumption for a market period τ can be adjusted at any time-step t before the physical delivery τ, but it becomes binding when $t = \tau$. We denote as $\Delta_{i,t}(\tau)$ the deviation from the market position for each time-step τ, as scheduled at time t, after having computed the variables $G_{i,t}(\tau)$ and $C_{i,t}(\tau)$, as follows:

$$P^{mar}_{i,t}(\tau) + \Delta_{i,t}(\tau) = P^{res}_{i,t}(\tau), \qquad (2.9)$$
$$\forall \tau \in \bar{T}(t).$$

The term $\Delta_{i,t}(\tau)$ represents the imbalance for market period τ as estimated at time t. This imbalance may evolve up to time $t = \tau$. We denote by $\Delta_i(\tau) = \Delta_{i,t=\tau}(\tau)$ the final imbalance for market period τ.

The power balance of equation (2.9) written for time-step $t+1$ is given by:

$$P^{mar}_{i,t+1}(\tau) + \Delta_{i,t+1}(\tau) = G_{i,t+1}(\tau) - C_{i,t+1}(\tau) \qquad (2.10)$$
$$\forall \tau \in \bar{T}(t+1).$$

It can be observed that by substituting equations (2.3), (2.6) and (2.7) in equation (2.10) we have:

$$\begin{aligned} P^{mar}_{i,t}(\tau) + v^{con}_{i,t}(\tau) + \Delta_{i,t+1}(\tau) = \\ G_{i,t}(\tau) + \Delta G_{i,t}(\tau) - (C_{i,t}(\tau) + \Delta C_{i,t}(\tau)) \\ \forall \tau \in \bar{T}(t). \end{aligned} \qquad (2.11)$$

The combination of equations (2.8) and (2.9) with equation (2.11) yields the update of the imbalance vector according to:

$$\Delta_{i,t+1}(\tau) = \Delta_{i,t}(\tau) + \Delta G_{i,t}(\tau) - \Delta C_{i,t}(\tau) - v^{con}_{i,t}(\tau) \qquad (2.12)$$
$$\forall \tau \in \bar{T}(t).$$

- *The agent does not own any physical asset (market maker).* It is equivalent to the first case with $\underline{C_i} = \overline{C_i} = \underline{G_i} = \overline{G_i} = 0$. The net imbalance $\Delta_{i,t}(\tau)$ is updated at every time-step $t \in T$ according to:

$$P^{mar}_{i,t}(\tau) + \Delta_{i,t}(\tau) = 0, \qquad (2.13)$$
$$\forall \tau \in \bar{T}(t).$$

- *The agent controls a storage device that can produce, store and consume energy.* We can consider an agent controlling a storage device as an agent that controls generation and production assets with specific constraints on the generation and the consumption level related to the nature of the storage device. Following this argument, let $G_{i,t}(\tau)$ ($C_{i,t}(\tau)$) refer to the level of discharging (charging) of the storage device for delivery time-step τ, updated at time t. Obviously, if $G_{i,t}(\tau) > 0$ ($C_{i,t}(\tau) > 0$), then we automatically have $C_{i,t}(\tau) = 0$ ($G_{i,t}(\tau) = 0$) since a battery cannot charge and discharge energy at the same time. In this case, agent i can decide to adjust its discharging (charging) level by $\Delta G_{i,t}(\tau)$ ($\Delta C_{i,t}(\tau)$). Let $SoC_{i,t}(\tau)$ denote the state of charge of the storage unit at delivery time-step $\tau \in \bar{T}$ as it is computed at time-step t, where $SoC_{i,t}(\tau) \in \left[\underline{SoC_i}, \overline{SoC_i}\right]$. The evolution of the state of charge during the delivery timeline can be updated at decision time-step t as:

$$\begin{aligned} SoC_{i,t}(\tau+\Delta\tau) = SoC_{i,t}(\tau) + \\ \Delta\tau \cdot \left(\eta C_{i,t}(\tau) - \frac{G_{i,t}(\tau)}{\eta}\right), \\ \forall \tau \in \bar{T}(t). \end{aligned} \tag{2.14}$$

Parameter η represents the charging and discharging efficiencies of the storage unit which, for simplicity, we assume are equal. We note that for batteries, charging and discharging efficiencies may be a function of the battery conditions. As can be observed from equation (2.14), time-coupling constraints are imposed on $C_{i,t}(\tau)$ and $G_{i,t}(\tau)$ in order to ensure that the amount of energy that can be discharged during some period already exists in the storage device. Additionally, constraints associated with the maximum charging power $\overline{C}_i$ and discharging power $\overline{G}_i$, as well as the maximum and minimum energy level ($\overline{SoC_i}$, $\underline{SoC_i}$) are considered in order to model the operation of the storage device.

Equation (2.14) can be written for time-step $t+1$ as:

$$\begin{aligned} SoC_{i,t+1}(\tau+\Delta\tau) = SoC_{i,t+1}(\tau) + \\ \Delta\tau \cdot \left(\eta C_{i,t+1}(\tau) - \frac{G_{i,t+1}(\tau)}{\eta}\right), \\ \forall \tau \in \bar{T}(t+1). \end{aligned} \tag{2.15}$$

Combining equations (2.14) and (2.15) we can derive the updated vector of the state of charge at time-step $t+1$ depending on the decided adjustments $(\Delta G_{i,t}(\tau), \Delta C_{i,t}(\tau))$ as:

$$\begin{aligned} SoC_{i,t+1}(\tau+\Delta\tau) - SoC_{i,t+1}(\tau) = & \\ & SoC_{i,t}(\tau+\Delta\tau) - SoC_{i,t}(\tau) + \\ & \Delta\tau \cdot (\eta \Delta C_{i,t}(\tau) - \frac{\Delta G_{i,t}(\tau)}{\eta}), \\ & \forall \tau \in \bar{T}(t). \end{aligned} \tag{2.16}$$

The state of charge $SoC_{i,t}(\tau)$ at delivery time-step τ can be updated until $t = \tau$. Let us also observe that there is a bijection between $P^{res}_{i,t}(\tau)$ and the terms $C_{i,t}(\tau)$ and $G_{i,t}(\tau)$ or, in other words, determining $P^{res}_{i,t}$ is equivalent to determining $C_{i,t}(\tau)$ and $G_{i,t}(\tau)$ and vice versa. The deviation from the committed schedule $\Delta_{i,t+1}(\tau)$ at delivery time-step τ at each time-step $t+1$ can be computed by equation (2.12).

All the new information arriving at time-step t for an asset-optimizing agent i (controlling a storage device) is gathered in variable:

$$\begin{aligned} s_{i,t} = (& s^{OB}_t, \\ & (P^{mar}_{i,t}(\tau), \Delta_{i,t}(\tau), G_{i,t}(\tau), C_{i,t}(\tau), SoC_{i,t}(\tau), \forall \tau \in \bar{T}), \\ & w^{exog}_{i,t}) \in S_i. \end{aligned}$$

The control action applied by an asset-optimizing agent i trading in the CID market at time-step t consists of posting new orders in the CID market and adjusting its production/consumption level or equivalently its charging/discharging level for the case of the storage device. The control actions can be summarised in variable $u_{i,t} = (a_{i,t}, (\Delta C_{i,t}(\tau), \Delta G_{i,t}(\tau), \forall \tau \in \bar{T}))$.

In this chapter, we consider that the trading agent adopts a simple strategy for determining, at each time-step t, the variables $\Delta C_{i,t}(\tau)$, $\Delta G_{i,t}(\tau)$ once the trading actions $a_{i,t}$ have been selected. In this case, the decision regarding the trading actions $a_{i,t}$ fully defines action $u_{i,t}$ and thus the notation $u_{i,t}$ will not be further used. This strategy will be referred to in the rest of the chapter as the “default" strategy for managing the storage device. According to this strategy, the agent aims at minimizing any imbalances $(\Delta_{i,t+1}(\tau))$ and therefore we use the following decision rule:

$$(\Delta C_{i,t}(\tau), \Delta G_{i,t}(\tau), \forall \tau \in \bar{T}) = \arg\min \sum_{\tau \in \bar{T}} \mid \Delta_{i,t+1}(\tau) \mid,$$

$$\text{s.t. (2.2), (2.3), (2.8), (2.9), (2.12), (2.14).} \tag{2.17}$$

One can easily see that from equation (2.11) this decision rule is equivalent to imposing $P^{res}_{i,t+1}(\tau)$ as close as possible to $P^{mar}_{i,t+1}(\tau)$, given the operational constraints of the device. We will elaborate later in this chapter on the fact that adopting such a strategy is not suboptimal in a context where the agent needs to be balanced for every market period while being an aggressor in the CID market.

For the sake of simplicity, we assume that the decision process of an asset-optimizing agent terminates at the gate closure $t_{close}(x)$ along with the trading process. Thus, the final residual production $P^{res}_i(\tau)$ for delivery time-step τ is given by $P^{res}_i(\tau) = P^{res}_{i,t=t_{close}(x)}(\tau)$. Similarly, the final imbalance is provided by $\Delta_i(\tau) = \Delta_{i,t=t_{close}(x)}(\tau)$.

Although this approach can be used for the optimization of a portfolio of assets, in this chapter, the focus lies on the case where the agent is operating a storage device. We note that this case is particularly interesting in the context of energy transition, where storage devices are expected to play a key role in the energy market.

2.2.4 Trading rewards

The instantaneous reward signal collected after each transition for agent i is given by:

$$r_{i,t} = R_i\left(t, s_{i,t}, a_{i,t}, a_{-i,t}\right), \tag{2.18}$$

where $R_i : T \times S_i \times A_1 \times ... \times A_n \rightarrow \mathbb{R}$.

The reward function R_i is composed of the following terms:

i. The trading revenues obtained from the matching process of orders at time-step t, given by ρ where ρ is a stationary function $\rho : S^{OB} \times A_1 \times ... \times A_n \rightarrow \mathbb{R}$,

ii. The imbalance penalty for deviation $\Delta_i(\tau)$ from the market position for delivery time-step τ at the imbalance price $I(\tau)$. The imbalance settlement process for product $x \in X$ (delivery time-step τ) takes place at the end of the physical delivery $t_{settle}(x)$ (i.e. at $\tau + \Delta\tau$), as presented in Figure 2.1. We define the imbalance settlement timeline T^{Imb}, as

$T^{Imb} = \{\tau + \Delta\tau, \forall \tau \in \bar{T}\}$. The imbalance penalty[1] is only applied when time instance t is an element of the imbalance settlement timeline.

The function R_i is defined as:

$$R_i\left(t, s_{i,t}, a_{i,t}, a_{-i,t}\right) =$$
$$\rho\left(s_t^{OB}, a_{i,t}, a_{-i,t}\right) + \begin{cases} \Delta_i(\tau) \cdot I(\tau) & ,\text{if } t \in T^{Imb}, \\ 0 & ,\text{otherwise} \end{cases}. \tag{2.19}$$

2.2.5 Trading policy

All the relevant information that summarises the past and that can be used to optimize the market participation is assumed to be contained in the history vector $h_{i,t} = (s_{i,0}, a_{i,0}, r_{i,0}, ..., s_{i,t-1}, a_{i,t-1}, r_{i,t-1}, s_{i,t}) \in H_i$. Trading agent i is assumed to select its actions following a non-anticipative history-dependent policy $\pi_i(h_{i,t}) \in \Pi$ from the set of all admissible policies Π, according to: $a_{i,t} \sim \pi_i(\cdot|h_{i,t})$.

2.2.6 Trading objective

The return collected by agent i in a single trajectory $\zeta = (s_{i,0}, a_{i,0}, ..., a_{i,K-1}, s_{i,K})$ of $K-1$ time-steps, given an initial state $s_{i,0} = s_i \in S_i$, which is the sum of cumulated rewards over this trajectory is given by:

$$G^{\zeta}(s_i) = \sum_{t=0}^{K-1} R_i\left(t, s_{i,t}, a_{i,t}, a_{-i,t}\right) |s_{i,0} = s_i. \tag{2.20}$$

The sum of returns collected by agent i, where each agent i is following an arbitrary policy $\pi_i \in \Pi$ are consequently given by:

$$V^{\pi_i}(s_i) = \underset{a_{i,t} \sim \pi_i, a_{-i,t} \sim \pi_{-i}}{\mathbb{E}} \left\{ \sum_{t=0}^{K-1} R_i\left(t, s_{i,t}, a_{i,t}, a_{-i,t}\right) |s_{i,0} = s_i \right\}. \tag{2.21}$$

The goal of the trading agent i is to identify an optimal policy $\pi_i^* \in \Pi$ that maximizes the expected sum of rewards collected along a trajectory. An optimal policy is obtained by:

$$\pi_i^* = \arg\max_{\pi_i \in \Pi} V^{\pi_i}(s_i). \tag{2.22}$$

[1] The imbalance price $I(\tau)$ is defined by a process that depends on a plethora of factors among which is the net system imbalance during delivery period τ, defined by the imbalance volumes of all the market players ($\sum^I \Delta_i(\tau)$). For the sake of simplicity we will assume that it is randomly sampled from a known distribution over prices that is not conditioned on any variable.

2.3 Markov Decision Process Formulation

In this section, we propose a series of assumptions that allow us to formulate the previously introduced problem of a storage device operator trading in the CID market using a reinforcement learning (RL) framework. Based on these assumptions, the decision-making problem is cast as an MDP; the action space is tailored in order to represent a particular market player and additional restrictions on the operation of the storage device are introduced.

2.3.1 Assumptions on the decision process

Assumption 1 (*Behaviour of the other agents*). The other agents $-i$ interact with the order book in between two discrete time-steps in such a way that agent i is the only agent interacting with the CID market at each time-step t. Moreover, it is assumed that the other agents $-i$ can only react in the market according to the previously observed order book states. More precisely their actions $a_{-i,t}$ depend strictly on the history of order book states s^{OB}_{t-1} and thus by extension on the history $h_{i,t-1}$ for every time-step t:

$$a_{-i,t} \sim P_{a_{-i,t}}(\cdot|h_{i,t-1}). \tag{2.23}$$

Assumption (1) suggests that the agents engage in a way that is very similar to a Markov Game [101]. The process under consideration is such that it interleaves between agent i taking actions $a_{i,t}$ followed by its opponents $-i$ taking actions $a_{-i,t}$. Furthermore, the joint strategy of the opponents is modeled with Equation (2.23) such that the agent i is involved in an MDP. This behaviour is illustrated in Figure 2.1 (magnified area). Given this assumption, the notation $a_{-i,t}$ can also be seen as referring to actions selected during the interval $(t-\Delta t, t)$.

Assumption 2 (*Exogenous information*). The exogenous information $w^{exog}_{i,t}$ is given by a stochastic model that depends solely on k past values, where $0 < k \leq t$ and a random disturbance $e_{i,t}$ according to:

$$w^{exog}_{i,t} = b(w^{exog}_{i,t-1}, ..., w^{exog}_{i,t-k}, e_t), \tag{2.24}$$

$$e_{i,t} \sim P_{e_{i,t}}(\cdot|h_{i,t}). \tag{2.25}$$

Assumption 3 (*Strategy for storage control*). The control decisions related to the charging ($\Delta C_{i,t}(\tau)$) or discharging ($\Delta G_{i,t}(\tau)$) power to/from the storage device are made based on the “default" strategy described in Section 2.2.3.

As described in Section 2.2.3, the original control action that can be applied at each time-step t is $u_{i,t} = (a_{i,t}, (\Delta C_{i,t}(\tau), \Delta G_{i,t}(\tau), \forall \tau \in \bar{T}))$. It can be observed that with such an assumption, the storage control decisions ($\Delta C_{i,t}(\tau)$ and $\Delta G_{i,t}(\tau)$) are obtained as a direct consequence of the trading decisions $a_{i,t}$. Indeed, after the trading decisions are submitted and the market position is updated, the storage control decisions are subsequently derived following the “default" strategy. Assumption (3) results in reducing the dimensionality of the action space and consequently the complexity of the decision-making problem.

2.3.2 Decision process

Following Assumptions (1), (2) and (3), one can simply observe that the decision-making problem faced by an agent i operating a storage device and trading energy in the CID market can be formalised as a fully observable finite-time MDP with the following characteristics:

- *Discrete time-step* $t \in T$, where T is the optimization horizon.
- *State space* H_i, where the state of the system $h_{i,t} \in H_i$ at time t summarises all past information that is relevant for future optimization.
- *Action space* A_i, where $a_{i,t} \in A_i$ is the set of new orders posted by agent i at time-step t.
- *Transition probabilities* $h_{i,t+1} \sim P(\cdot|h_{i,t}, a_{i,t})$, that can be inferred by the following processes:

  1. $a_{-i,t} \in A_{-i}$ is drawn according to equation (2.23)
  2. The state of the order book s^{OB}_{t+1} follows the transition given by equation (2.1)
  3. The exogenous information $w^{exog}_{i,t}$ is given by equation (2.24) and the noise by (2.25)
  4. The variable $s_{i,t+1}$ that summarises the information of the storage device optimizing agent follows the transition given by equations (2.1), (2.6)-(2.12) (2.24), (2.25) and (2.16)
  5. The instantaneous reward $r_{i,t}$ collected after each transition is given by equations (2.18) and (2.19).

  The elements resulting from these processes can be used to construct $h_{i,t+1}$ in a straightforward way.

2.3.3 Assumptions on the trading actions

Assumption 4 (*Aggressor*). The trading agent can only submit new orders that match already existing orders at their price (i.e. aggressor or liquidity taker).

Let A_i^{red} be the space that contains only actions that match pre-existing orders in the order book. According to Assumption (4), the i^{th} agent, at time-step t, is restricted to select actions $a_{i,t} \in A_i^{red} \subset A_i$. Let $s_t^{OB} = ((x_j^{OB}, y_j'^{OB}, v_j^{OB}, p_j^{OB}, e_j^{OB}), \forall j \in N_t)$ be the order book observation at trading time-step t. We use y'^{OB} to denote that the new orders have the opposite side (“Buy" or “Sell") than the existing orders. We denote as $a_{i,t}^j \in [0,1]$ the fraction of the volume accepted from order j. The reduced action space A_i^{red} is then defined as:

$$A_i^{red} = \{(x_j^{OB}, y_j'^{OB}, a_{i,t}^j \cdot v_j^{OB}, p_j^{OB}, e_j^{OB}), a_{i,t}^j \in [0,1], \forall j \in N_t\}.$$

At this point, posting a new set of orders $a_{i,t} \in A_i^{red}$ boils down to simply specifying the vector of fractions:

$$\bar{a}_{i,t} = \left(a_{i,t}^j, \forall j \in N_t\right) \in \bar{A}_i^{red}$$

that define the partial or full acceptance of the existing orders. The action $a_{i,t}$ submitted by an aggressor is a function l of the observed order book s_t^{OB} and the vector of fractions $\bar{a}_{i,t}$ and is given by:

$$a_{i,t} = l(s_t^{OB}, \bar{a}_{i,t}). \tag{2.26}$$

2.3.4 Restrictions on the storage operation

Assumption 5 (*No imbalances permitted*). The trading agent can only accept an order to buy or sell energy if and only if it does not result in any imbalance for the remaining delivery periods.

According to Assumption (5) the agent is completely risk-averse in the sense that, even if it stops trading at any given point, its position in the market can be covered without causing any imbalance. This assumption is quite restrictive with respect to the full potential of an asset-optimizing agent in the CID market. We note that, according to the German regulation policies (see [102]), the imbalance market should not be considered as an optimization floor and the storage device should always be balanced at each trading time-step t ($\Delta_{i,t}(\tau) = 0, \forall \tau \in \bar{T}$). In this respect, we can view Assumption 5 as a way to comply with the German regulation policies in a risk-free context where each new trade should not create an imbalance that would have to be covered later.

Assumption 6 (*Optimization decoupling*). The storage device has a given initial value for the storage level SoC_i^{init} at the beginning of the delivery timeline. Moreover, it is constrained to terminate at a given level SoC_i^{term} at the end of the delivery timeline.

Under Assumption (6) the optimization of the storage unit over a long trading horizon can be decomposed into shorter optimization windows (e.g. of one day). In the simulation results reported later in this chapter, we will choose $SoC_i^{init} = SoC_i^{term}$.

2.4 Methodology

In this section, we describe the methodology that has been applied for tackling the MDP problem described in subsection 2.3. We consider that, in reality, an asset-optimizing agent has at its disposal a set of trajectories (one per day) from participating in the CID market in the past years. The process of collecting these trajectories and their structure is presented in Section 2.4.1. Based on this dataset, we propose in subsection 2.4.2 the deployment of the fitted Q iteration algorithm as introduced in [97]. This algorithm belongs to the class of batch-mode RL algorithms that make use of all the available samples at once for updating the policy. This class of algorithms is known to be very sample efficient.

Despite the different assumptions made on the operation of the storage device and the way it is restricted to interact with the market, the dimensionality of the action space still remains very high. Due to limitations related to the function approximation architecture used to implement the fitted Q iteration algorithm, a low-dimensional and discrete action space is necessary, as discussed in subsection 2.4.3. Therefore, as part of the methodology, in subsection 2.4.4 we propose a way for reducing the action space. Afterwards, in subsection 2.4.5, a more compact representation of the state space is proposed in order to reduce the computational complexity of the training process and increase the sample efficiency of the algorithm.

Finally, the low number of available samples (one trajectory per day) gives rise to issues related to the limited exploration of the agent. In order to address these issues, we generate a large number of trading trajectories of our MDP according to an ε-greedy policy, using historical trading data. In the last part of this section, we elaborate on the strategy that is used in this chapter for generating the trajectories and the limitations of this procedure.

2.4.1 Collection of trajectories

As previously mentioned, an asset-optimizing agent can collect a set of trajectories from previous interactions with the CID market. Based on Assumption (6), each day can be optimized separately and thus, trading for one day corresponds to one trajectory. We consider that the trading horizon defined in Section 2.2.2 consists of K discrete trading time-steps such that $T = \{0, ..., K\}$. A single trajectory sampled from the MDP described in Section 2.3 is defined as:

$$\zeta_m = \left(h_{i,0}^m, a_{i,0}^m, r_{i,0}^m, ..., h_{i,K-1}^m, a_{i,K-1}^m, r_{i,K-1}^m, h_{i,K}^m\right).$$

A set of M trajectories can be then defined as:

$$F = \{\zeta_m, m = 1, ..., M\}.$$

The set of trajectories F can be used to generate the set of sampled one-step system transitions F' defined as:

$$F' = \begin{Bmatrix} (h_{i,0}^1, a_{i,0}^1, r_{i,0}^1, h_{i,1}^1), & \cdots & (h_{i,K-1}^1, a_{i,K-1}^1, r_{i,K-1}^1, h_{i,K}^1), \\ \vdots & \ddots & \vdots \\ (h_{i,0}^M, a_{i,0}^M, r_{i,0}^M, h_{i,1}^M), & \cdots & (h_{i,K-1}^M, a_{i,K-1}^M, r_{i,K-1}^M, h_{i,K}^M) \end{Bmatrix}.$$

The set F' is split into K sets of one-step system transitions F'_t defined as:

$$F'_t = \left\{(h_{i,t}^m, a_{i,t}^m, r_{i,t}^m, h_{i,t+1}^m), m = 1, ..., M\right\}_t,$$
$$\forall t \in \{0, ..., K-1\}.$$

In the following subsection, the type of RL algorithm used for inferring a high-quality policy from this set of one-step system transitions is explained in detail.

2.4.2 Batch-mode reinforcement learning

Q-functions and Dynamic Programming: In this section, the fitted Q iteration algorithm is proposed for the optimization of the MDP defined in Section 2.3, using a set of collected trajectories. In order to solve the problem, we first define the Q-function for each state-action pair $(h_{i,t}, a_{i,t})$ at time t as proposed in [40] as:

$$Q_t(h_{i,t}, a_{i,t}) = \underset{a_{-i,t},\ e_{i,t}}{\mathbb{E}} \left\{ r_{i,t} + V_{t+1}(h_{i,t+1}) \right\}, \qquad (2.27)$$

$$\forall t \in \{0, ..., K-1\}.$$

A time-variant policy $\pi = \{\mu_0, ..., \mu_{K-1}\} \in \Pi$, consists in a sequence of functions μ_t, where $\mu_t : H_i \to A_i^{red}$. An action $a_{i,t}$ is selected from this policy at each time-step t, according to $a_{i,t} = \mu_t(h_{i,t})$. We denote as $\pi^{t+1} = \{\mu_{t+1}, ..., \mu_{K-1}\}$ the sequence of functions μ_t from time-step $t+1$ until the end of the horizon. Standard results from dynamic programming (DP) show that for the finite time MDP we are addressing in this chapter, there exists at least one such time-variant policy which is an optimal policy as defined by equation (2.22). Therefore, we focus on the computation of such an optimal time-variant policy. We define the value function V_{t+1} as the optimal expected cumulative rewards from stage $t+1$ until the end of the horizon K given by:

$$V_{t+1}(h_i) =$$

$$\max_{\pi^{t+1} \in \Pi} \underset{\substack{(a_{-i,t+1}, e_{i,t+1}) \\ \cdots \\ (a_{-i,K-1}, e_{i,K-1})}}{\mathbb{E}} \left\{ \sum_{k=t+1}^{K-1} R_{i,k}\left(h_{i,k}, \mu_k(h_{i,k}), a_{-i,k}\right) | h_{i,t+1} = h_i \right\}. \qquad (2.28)$$

We observe that $Q_t(h_{i,t}, a_{i,t})$ is the value attained by taking action $a_{i,t}$ at state $h_{i,t}$ and subsequently using an optimal policy. Using the dynamic programming algorithm [40] we have:

$$V_t(h_{i,t}) = \max_{a_{i,t} \in A_i^{red}} Q_t(h_{i,t}, a_{i,t}). \qquad (2.29)$$

Equation (2.27) can be written in the following form that relates Q_t and Q_{t+1}:

$$Q_t(h_{i,t}, a_{i,t}) = \underset{a_{-i,t},\ e_{i,t}}{\mathbb{E}} \left\{ r_{i,t} + \max_{a_{i,t+1} \in A_i^{red}} Q_{t+1}(h_{i,t+1}, a_{i,t+1}) \right\}. \qquad (2.30)$$

An optimal time-variant policy $\pi^* = \left\{\mu_0^*, ..., \mu_{K-1}^*\right\}$ can be identified using the Q-functions as following:

$$\mu_t^* = \arg\max_{a_{i,t} \in A_i^{red}} Q_t(h_{i,t}, a_{i,t}), \qquad (2.31)$$
$$\forall t \in \{0, ..., K-1\}.$$

Computing the Q-functions from a set of one-step system transitions: In order to obtain the optimal time-variant policy π^*, the effort is focused on computing the Q-functions defined in equation (2.30). However, two aspects render the use of the standard value iteration algorithm impossible for solving the MDP defined in Section 2.3. First, the transition probabilities of the MDP defined in Section 2.3 are not known. Instead, we can exploit the set of collected historical trajectories to compute the exact Q-functions using an algorithm such as Q-learning (presented in [103]). Q-learning is designed for working only with trajectories, without any knowledge of the transition probabilities. Optimality is guaranteed given that all state-action pairs are observed infinitely often within the set of the historical trajectories and that the successor states are independently sampled at each occurrence of a state-action pair [40]. In Section 2.4.6 we discuss the validity of this condition and we address the problem of limited exploration by generating additional artificial trajectories. Second, due to the continuous nature of the state and action spaces a tabular representation of the Q-functions used in Q-learning is not feasible. In order to overcome this issue, we use a function approximation architecture to represent the Q-functions [104].

The computation of the approximate Q-functions is performed using the fitted Q iteration algorithm [97]. We present the algorithm for the case where a parametric function approximation architecture ($Q_t(h_{i,t}, a_t; \theta_t)$) is used (e.g. neural networks). In this case, the algorithm is used to compute, recursively, the parameter vectors θ_t starting from $t = K-1$. However, it should be emphasized that the fitted Q iteration algorithm can be adapted in a straightforward way to the case in which a non-parametric function approximation architecture is selected.

The set of M samples of quadruples $F_t' = \{(h_{i,t}^m, a_{i,t}^m, r_t^m, h_{i,t+1}^m), m = 1, ..., M\}$ obtained from previous experience is exploited in order to update the parameter vectors θ_t by solving the supervised learning problem presented in equation (2.32). The target vectors y_t are computed using the Q-function approximation of the next stage ($Q_{t+1}(h_{i,t+1}, a_{t+1}; \theta_{t+1})$) according to equation (2.33). The Q-function for the terminal state is set to zero ($\hat{Q}_K \equiv 0$) and the algorithm iterates backwards in the time horizon T, producing a sequence of approximate Q-functions denoted by $\hat{Q} = \{\hat{Q}_0, ..., \hat{Q}_{K-1}\}$ until termination at $t = 0$.

$$\theta_t = \arg\min_{\theta_t} \sum_{m=1}^{M} (Q_t(h_{i,t}^m, a_t^m; \theta_t) - y_t^m)^2 \tag{2.32}$$

$$y_t^m = r_t^m + \max_{a_{i,t+1} \in A_i^{red}} Q_{t+1}(h_{i,t+1}^m, a_{i,t+1}; \theta_{t+1}) \tag{2.33}$$

Once the parameters θ_t are computed, the time-variant policy $\hat{\pi}^* = \left\{\hat{\mu}_0^*, ..., \hat{\mu}_{K-1}^*\right\}$ is obtained as:

$$\hat{\mu}_t^*(h_{i,t}) = \arg\max_{a_{i,t} \in A_i^{red}} Q_t(h_{i,t}, a_{i,t}; \theta_t), \quad \forall t \in \{0, ..., K-1\}. \tag{2.34}$$

In practice, a new trajectory is collected after each trading day. The set of collected trajectories F is consequently augmented. Thus, the fitted Q iteration algorithm can be used to compute a new optimal policy when new data arrive.

2.4.3 Limitations

The fitted Q iteration algorithm, described in the previous section, can be used to provide a trading policy based on the set of past trajectories at the disposal of the agent. Even though, this approach is theoretically sound, in practice there are several limitations to overcome. The efficiency of the described fitted Q iteration algorithm is overshadowed by the high-dimensionality of the state and the action space.

The state variable

$$h_{i,t} = (s_{i,0}, a_{i,0}, r_{i,0}, ..., s_{i,t-1}, a_{i,t-1}, r_{i,t-1}, s_{i,t}) \in H_i$$

is composed of :

- The entire history of actions $(a_{i,0}, ..., a_{i,t-1})$ before time t
- The entire history of rewards $(r_{i,0}, ..., r_{i,t-1})$ before time t

- The history of order book states $\left(s_0^{OB},...,s_t^{OB}\right)$ up to time t and, of the private information $(s_{i,0}^{private},...,s_{i,t}^{private})$ up to time t, where:

$$\begin{aligned} s_{i,t}^{private} =&((P_{i,t}^{mar}(\tau),\Delta_{i,t}(\tau),\\ &G_{i,t}(\tau),C_{i,t}(\tau),SoC_{i,t}(\tau),\forall\tau\in\bar{T}),\\ &w_{i,t}^{exog}). \end{aligned}$$

The state space H_i as well as the action space A_t^{red}, as described in Section 2.3.3, depend explicitly on the content of the order book s_t^{OB}. The dimension of these spaces at each time-step t depends on the total number of available orders $\mid N_t \mid$ in the order book. However, the total number of orders is changing at each step t. Thus, both the state and the action spaces are high-dimensional spaces of variable size. In order to reduce the complexity of the decision-making problem, we have chosen to reduce these spaces so as to work with a small action space of constant size and a compact state space. In the following, we describe the procedure that was carried out for the reduction of the state and action spaces.

2.4.4 Action space reduction: High-level actions

In this section, we elaborate on the design of a small and discrete set of actions that is an approximation of the original action space. Based on Assumptions (1), (2), (3), (4), (5) and (6), a new action space A_i' is proposed, which is defined as $A_i' = \{\text{“}Trade\text{”},\text{“}Idle\text{”}\}$. The new action space is composed of two high-level actions $a_{i,t}' \in A_i'$. These high-level actions are transformed to an original action through mapping $p : A_i' \rightarrow A_i^{red}$, from space A_i' to the reduced action space A_i^{red}. The high-level actions are defined as follows:

“Trade”

At each time-step t, agent i selects orders from the order book with the objective of maximizing the instantaneous reward under the constraint that the storage device can remain balanced for every delivery period, even if no further interaction with the CID market occurs. As a reminder, this constraint was imposed by Assumption (5).

Under this assumption, the instantaneous reward signal $r_{i,t}$, presented in equation (5.78), consists only of the trading revenues obtained from the matching process of orders at time-step t. We will further assume that mapping $u : \mathbb{R}^+ \times \{\text{“}Sell\text{”},\text{“}Buy\text{”}\} \rightarrow \mathbb{R}$ that adjusts the sign of the volume v^{OB} of each order according to their side y^{OB}. Orders posted for buying energy will be associated with positive volume and orders posted for selling energy with negative volume, or equivalently:

$$u(v^{OB}, y^{OB}) = \begin{cases} v^{OB}, & \text{if } y^{OB} = \text{“}Buy\text{”}, \\ -v^{OB}, & \text{if } y^{OB} = \text{“}Sell\text{”}. \end{cases} \tag{2.35}$$

Consequently, the reward function ρ defined in Section 2.2.4 is adapted according to the proposed modifications. The new reward function ρ, where $\rho : S^{OB} \times \bar{A}_i^{red} \to \mathbb{R}$, is a stationary function of the orders observed at each time-step t and the agent's response to the observed orders. An analytical expression for the instantaneous reward collected is given by:

$$r_{i,t} = \rho\left(s_t^{OB}, \bar{a}_{i,t}\right) = \sum_{j=1}^{N_t} a_{i,t}^{j} \cdot u(v_j^{OB}, y_j^{OB}) \cdot p_j^{OB}. \tag{2.36}$$

The High-level action “Trade" amounts to solving the bid acceptance optimization problem presented in Model 2. The objective function of the problem, formulated in equation (2.37), consists of the revenues arising from trading. It is important to note that the operational constraints guarantee that no order will be accepted if it causes any imbalance. We denote as $N_\tau \subset \mathbb{N}$ the set of unique indices of the available orders that correspond to delivery time-step τ and $N_t = \bigcup_{\tau \in \bar{T}} N_\tau$. In equation (2.38), the energy purchased and sold ($\sum_{j \in N_\tau} a_{i,t}^{j} u(v_j^{OB})$), the past net energy trades ($P_{i,t}^{mar}(\tau)$) and the energy discharged by the storage ($G_{i,t}(\tau)$) must match the energy charged by the storage ($C_{i,t}(\tau)$) for every delivery time-step τ. The energy balance of the storage device, presented in equation (2.39), is responsible for the time-coupling and the arbitrage between two products x (delivery time-steps τ). The technical limits of the storage level and the charging and discharging process are described in equations (2.40) to (2.44). The binary variables $k_{i,t} = (k_{i,t}(\tau), \forall \tau \in \bar{T})$ restrict the operation of the unit for each delivery period in only one mode, either charging or discharging.

The optimal solution to this problem yields the vector of fractions:

$$\bar{a}_{i,t} = \left(a_{i,t}^{j}, \forall j \in N_t\right) \in \bar{A}_i^{red}$$

that are used in equation (2.26) to construct the action $a_{i,t} \in A_i^{red}$. The optimal solution also defines at each time-step t the adjustments in the level of the production (discharge) $\Delta G_{i,t} = (\Delta G_{i,t}(\tau), \forall \tau \in \bar{T}(t))$ and the consumption (charge) $\Delta C_{i,t} = (\Delta C_{i,t}(\tau), \forall \tau \in \bar{T}(t))$. The evolution of the state of charge $SoC_{i,t+1} = (SoC_{i,t+1}(\tau), \forall \tau \in \bar{T}(t))$ of the unit as well as the production $G_{i,t+1} = (G_{i,t+1}(\tau), \forall \tau \in \bar{T}(t))$ and consumption $C_{i,t+1} = (C_{i,t+1}(\tau), \forall \tau \in \bar{T}(t))$ levels are computed for each delivery period.

Algorithm 2 "Trade"

Input: t, s_t^{OB}, $P_{i,t}^{mar}$, $\underline{SoC_i}$, $\overline{SoC_i}$, $\underline{C_i}$, $\overline{C_i}$, $\underline{G_i}$, $\overline{G_i}$, SoC_i^{init}, SoC_i^{term},τ_{init}, τ_{term},$G_{i,t}$, $C_{i,t}$
Output: $\bar{a}_{i,t}$, $SoC_{i,t+1}$, $G_{i,t+1}$, $C_{i,t+1}$, $\Delta G_{i,t}$, $\Delta C_{i,t}$, $k_{i,t+1}$, $r_{i,t}$
Solve:

$$\max_{\substack{\bar{a}_{i,t},\, SoC_{i,t+1}\\ G_{i,t+1},\, C_{i,t+1}\\ \Delta G_{i,t},\, \Delta C_{i,t}\\ k_{i,t+1},\, r_{i,t}}} \sum_{j \in N_t} a_{i,t}^j \cdot u(v_j^{OB}, y_j^{OB}) \cdot p_j^{OB} \tag{2.37}$$

$$\begin{aligned}
\text{s.t.} \sum_{j \in N_\tau} a_{i,t}^j u(v_j^{OB}, y_j^{OB}) + P_{i,t}^{mar}(\tau) + & \\
C_{i,t+1}(\tau) = G_{i,t+1}(\tau), & \qquad \forall \tau \in \bar{T}(t) \qquad (2.38)\\
SoC_{i,t+1}(\tau + \Delta\tau) = SoC_{i,t+1}(\tau) + & \\
\Delta\tau \cdot \left(\eta \cdot C_{i,t+1}(\tau) - \frac{G_{i,t+1}(\tau)}{\eta}\right), & \qquad \forall \tau \in \bar{T}(t) \qquad (2.39)\\
\underline{SoC_i} \leq SoC_{i,t+1}(\tau) \leq \overline{SoC_i}, & \qquad \forall \tau \in \bar{T}(t) \qquad (2.40)\\
SoC_i^{init} = SoC_{i,t+1}(\tau_{init}), & \qquad (2.41)\\
SoC_i^{term} = SoC_{i,t+1}(\tau_{term}), & \qquad (2.42)\\
\underline{C_i} \leq C_{i,t+1}(\tau) \leq k_{i,t+1}(\tau) \cdot \overline{C_i}, & \qquad \forall \tau \in \bar{T}(t) \qquad (2.43)\\
\underline{G_i} \leq G_{i,t+1}(\tau) \leq (1 - k_{i,t+1}(\tau))\, \overline{G_i}, & \qquad \forall \tau \in \bar{T}(t) \qquad (2.44)\\
G_{i,t+1}(\tau) = G_{i,t}(\tau) + \Delta G_{i,t}(\tau), & \qquad \forall \tau \in \bar{T}(t) \qquad (2.45)\\
C_{i,t+1}(\tau) = C_{i,t}(\tau) + \Delta C_{i,t}(\tau), & \qquad \forall \tau \in \bar{T}(t) \qquad (2.46)\\
k_{i,t+1}(\tau) \in \{0, 1\}, & \qquad \forall \tau \in \bar{T}(t) \qquad (2.47)\\
a_{i,t}^j \in [0, 1], & \qquad \forall j \in N_t \qquad (2.48)
\end{aligned}$$

"Idle"

No transactions are executed, and no adjustment is made to the previously scheduled quantities. Under this action, the vector of fractions $\bar{a}_{i,t}$ is a zero vector. The discharge and charge as well as the state of charge of the storage device remain unchanged ($\Delta G_{i,t} \equiv 0$ and $\Delta C_{i,t} \equiv 0$) and we have:

$$G_{i,t+1}(\tau) = G_{i,t}(\tau), \forall \tau \in \bar{T}(t), \tag{2.49}$$

$$C_{i,t+1}(\tau) = C_{i,t}(\tau), \forall \tau \in \bar{T}(t), \tag{2.50}$$

$$SoC_{i,t+1}(\tau) = SoC_{i,t}(\tau), \forall \tau \in \bar{T}(t). \tag{2.51}$$

With such a reduction of the action-space, the agent can choose at every time-step t between the two described high-level actions ($a'_{i,t} \in A'_i = \{$"$Trade$","$Idle$"$\}$). Note that when the agent learns to idle, given a current situation, it does not necessarily mean, that if it had chosen to "$Trade$" instead, he would not make a positive immediate reward. Indeed, the agent would choose "$Idle$" if it believes that there may be a better market state emerging, i.e. the agent would learn to wait for the ideal opportunity of orders appearing in the order book at subsequent time-steps. We compare this approach to an alternative, which we refer to as the "rolling intrinsic" policy. According to this policy, at every time-step t of the trading horizon the agent selects the combination of orders that optimises its operation and profits, based on the current information assuming that the storage device must remain balanced for every delivery period as presented in [105]. The "rolling intrinsic" policy is, thus, equivalent to sequentially selecting the action "$Trade$" (Algorithm 2), as defined in this framework. The algorithm proposed later in this chapter exploits the experience that the agent can gain through (artificial) interaction with its environment, in order to learn the value of trading or idling at every different state that agent may encounter.

2.4.5 State space reduction

In this section, we propose a more compact and low-dimensional representation of the state space H_i. The state $h_{i,t}$, as explained in Section 2.4.3, contains the entire history of all the relevant information available for the decision-making process up to time t. As such, the information contained in the trajectories is represented as unstructured sets. We consider each one of the components of the state $h_{i,t}$, namely the entire history of actions, order book states and private information, and we provide an alternative form. This alternative form is engineered with the aim to capture the structure between observed bids in the order book.

First, the vector containing the entire history of actions is reduced to a vector of binary variables after the modifications introduced in Section 2.4.4.

Second, the vector containing the history of order book states is reduced into a vector of engineered features. We start from the order book state $s_t^{OB} = ((x_j^{OB}, y_j'^{OB}, v_j^{OB}, p_j^{OB}, e_j^{OB}), \forall j \in N_t \subseteq \mathbb{N}) \in S^{OB}$ that is defined in Sections 2.2.1 and 2.2.2 as a high-dimensional continuous vector used to describe the state of the CID market. Owing to the variable (non-constant) and large amount of orders $|N_t|$, the space S^{OB} has a non-constant size with high-dimensionality.

In order to overcome this issue, we proceed as following. First, we consider the market depth curves for each product x. The market depth of each side ("Sell" or "Buy") at a time-step t, is defined as the total volume available in the order book per price level for product x. The market depth for the "Sell" ("Buy") side is computed by stacking the existing orders in ascending (descending) price order and accumulating the available volume. The market depth for each of the quarter-hourly products Q_1 to Q_6 at time instant t is illustrated in Figure 2.2a using data from the German CID market. The market depth curves serve as a visualization of the order book that provides information about the liquidity of the market. Moreover, it provides information about the maximum (minimum) price that a trading agent will have to pay in order to buy (sell) a certain volume of energy. If we assume a fixed-price discretization, certain upper and lower bounds on the prices and interpolation of the data in this price range, the market depth curves of each product x can be approximated by a finite and constant set of values.

Even though this set of values has a constant size, it can still be extremely large. Its dimension is not a function of the number of existing orders any more, but it depends on the resolution of the price discretization, the price range considered, and the total number of products in the market. Instead of an individual market depth curve for each product x, we consider a market depth curve for all the available products, i.e. existing orders in ascending (descending) price order and accumulating the available volumes for all the products. In this way we can construct the aggregated market depth curve, presented in Figure 2.2b. The aggregated market depth curve illustrates the total available volume ("Sell" or "Buy") per price level for all products.

The motivation for considering the aggregated curves comes from the very nature of a storage device. The main profit-generating mechanism of a storage device is the arbitrage between two delivery periods. Its functionality involves the purchasing (charging) of electricity during periods of low prices and the selling (discharging) during periods of high prices.

For instance, in Figure 2.2a, a storage device would buy volume for product Q_4 and sell volume back for product Q_5. The intersection of the "Sell" and "Buy" curves in Figure 2.2b defines the maximum volume that can be arbitraged by the storage device if no operational constraints were considered and serves as an upper bound for the profits at each step t. Alternatively, the market depth for the same products Q_1 to Q_6 at a different time-step of the trading horizon is presented in Figure 2.3a. As illustrated in Figure 2.3b, there is no arbitrage opportunity between the products, hence the aggregated curves do not intersect. Thus, we assume, that the aggregated curves provide a sufficient representation of the order book.

At this point, considering a fixed-price discretization and a fixed price range would yield a constant set of values able to describe the aggregated curves. However, in order to further decrease the size of the set of values with sufficient price discretization, we motivate the use of a set of distance measures between the two aggregated curves that succeed in capturing the arbitrage potential at each trading time-step t as state variables, as presented in Figures 2.2b and 2.3b.

For instance, we define as $D1$ the signed distance between the 75th percentile of "Buy" price and the 25th percentile of "Sell" price and as $D2$ the absolute distance between the mean value of “Buy” and “Sell” volumes. Other measures used are the signed price difference and absolute volume difference between percentiles (25%, 50%, 75%) and the bid-ask spread. A detailed list of the distance measures is provided in Table 2.2.

TABLE 2.2: Order book features used for the state reduction.

Symbol	Definition	Description
$D1$	$p^{Buy}_{max} - p^{Sell}_{min}$	Signed diff. between the maximum “Buy” price and the minimum “Sell” price
$D2$	$p^{Buy}_{mean} - p^{Sell}_{mean}$	Signed diff. between the mean “Buy” price and the mean “Sell” price
$D3$	$p^{Buy}_{25\%} - p^{Sell}_{75\%}$	Signed diff. between the 25th percentile “Buy” price and the 75th percentile “Sell” price
$D4$	$p^{Buy}_{50\%} - p^{Sell}_{50\%}$	Signed diff. between the 50th percentile “Buy” price and the 50th percentile “Sell” price
$D5$	$p^{Buy}_{75\%} - p^{Sell}_{25\%}$	Signed diff. between the 75th percentile “Buy” price and the 25th percentile “Sell” price
$D6$	$\|v^{Buy}_{min} - v^{Sell}_{min}\|$	Abs. diff. between the minimum “Buy” cum. volume and the maximum “Sell” cum. volume
$D7$	$\|v^{Buy}_{mean} - v^{Sell}_{mean}\|$	Abs. diff. between the mean “Buy” cum. volume and the mean “Sell” cum. volume
$D8$	$\|v^{Buy}_{25\%} - v^{Sell}_{25\%}\|$	Abs. diff. between the 25th percentile “Buy” cum. volume and the 25th percentile “Sell” cum. volume
$D9$	$\|v^{Buy}_{50\%} - v^{Sell}_{50\%}\|$	Abs. diff. between the 50th percentile “Buy” cum. volume and the 50th percentile “Sell” cum. volume
$D10$	$\|v^{Buy}_{75\%} - v^{Sell}_{75\%}\|$	Abs. diff. between the 75th percentile “Buy” cum. volume and the 75th percentile “Sell” cum. volume

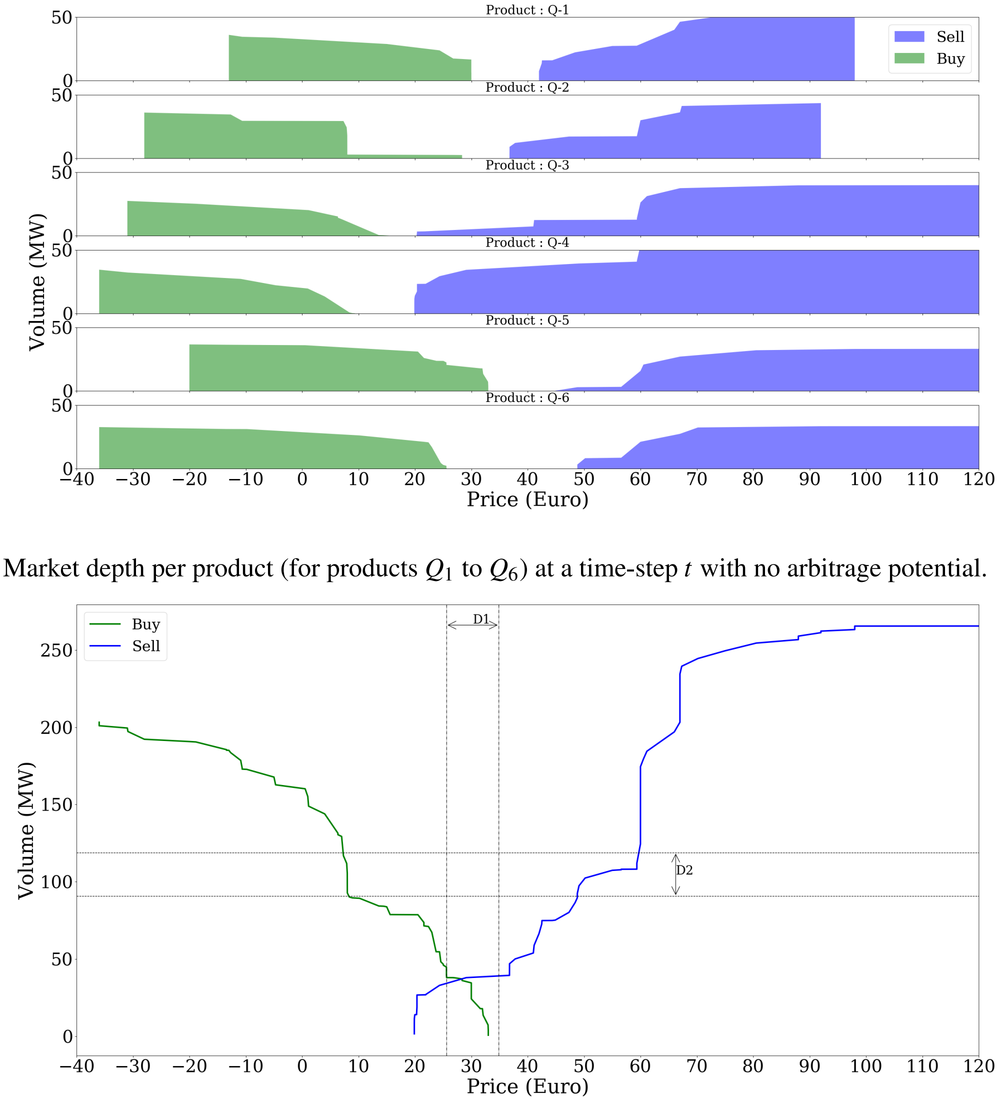
The new, continuous, low-dimensional observation of the order book $s'^{OB}_t \in S'^{OB} = \{D1,..,D10\}$ is used to represent the state of the order book and, in particular, its profit potential. It is important to note that in contrast to $s^{OB}_t \in S^{OB}$, the new order book observation $s'^{OB}_t \in S'^{OB}$ does not depend on the number of orders in the order book and therefore has a constant size, i.e. the cardinality of S'^{OB} is constant over time.

Finally, the history of the private information of agent i, that is not publicly available, is a vector that contains the high-dimensional continuous variables $s^{private}_{i,t}$ related to the operation of the storage device. As described in Section 2.4.3, $s^{private}_{i,t}$ is defined as:

$$\begin{aligned} s^{private}_{i,t} =& ((P^{mar}_{i,t}(\tau), \Delta_{i,t}(\tau), \\ & G_{i,t}(\tau), C_{i,t}(\tau), SoC_{i,t}(\tau), \forall \tau \in \bar{T}), \\ & w^{exog}_{i,t}). \end{aligned}$$

According to Assumption (5), the trading agent cannot perform any transaction if it results in imbalances. Therefore, it is not relevant to consider the vector $\Delta_{i,t}$ since it will always be zero according to the way the high-level actions are defined in Section 2.4.4. Additionally, Assumption (3) regarding the default strategy for storage control in combination with Assumption (5) yields a direct correlation between vectors $P^{mar}_{i,t}$ and $G_{i,t}$, $C_{i,t}$, $SoC_{i,t}$. Thus, it is considered that $P^{mar}_{i,t}$ contains all the required information and thus vectors $G_{i,t}$, $C_{i,t}$ and $SoC_{i,t}$ can be dropped.

Following the previous analysis we can define the low-dimensional pseudo-state $z_{i,t} =$



(A) Market depth per product (for products Q_1 to Q_6) at a time-step t with no arbitrage potential.

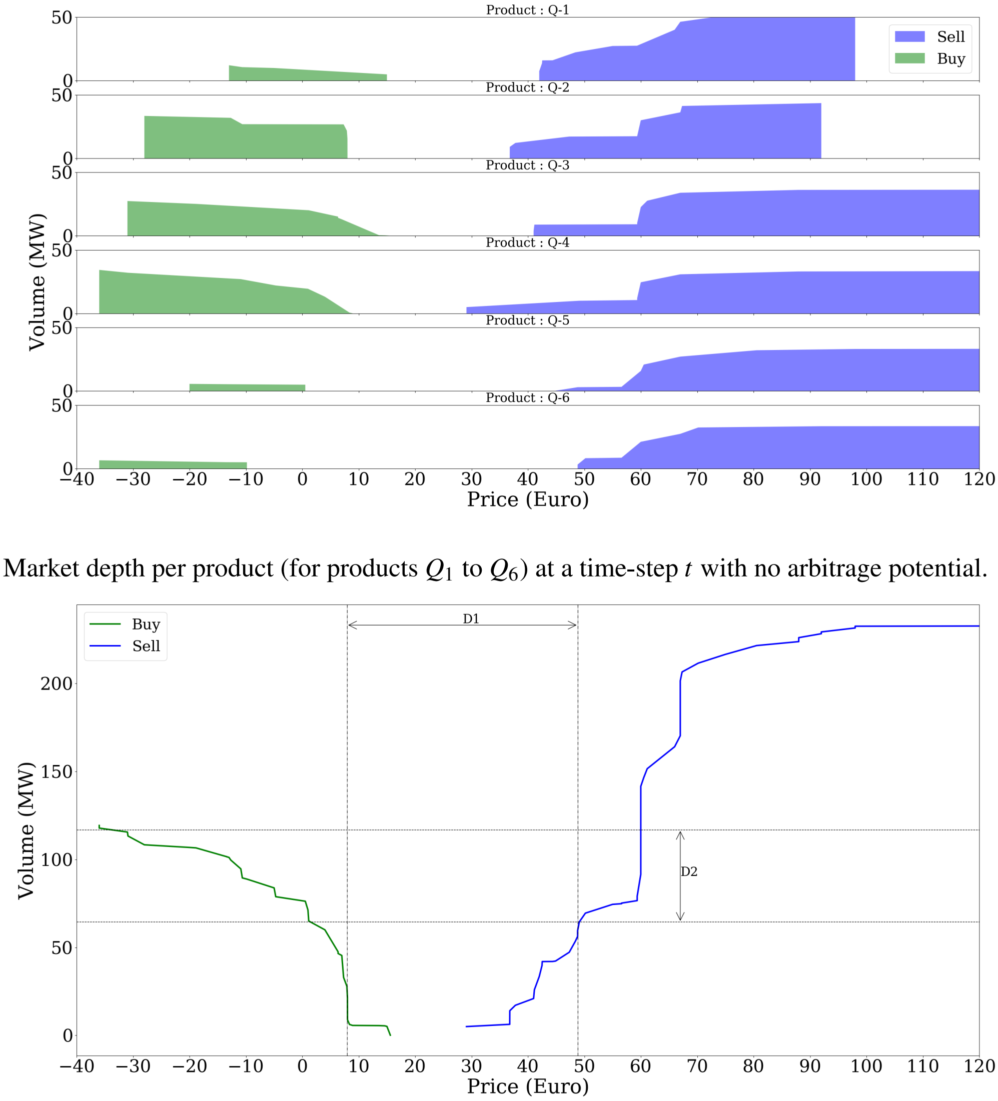
(B) The corresponding aggregated curves for a non profitable order book.

FIGURE 2.2

$(s'_{i,0}, a'_{i,0}, r_{i,0}, ..., a'_{i,t-1}, r_{i,t-1}, s'_{i,t}) \in Z_i$, where $s'_{i,t} = (s'^{OB}_t, P^{mar}_{i,t}, w^{exog}_{i,t}) \in S'_i$. This pseudo-state can be seen as the result of applying an encoder $enc : H_i \rightarrow Z_i$ which maps a true state $h_{i,t}$ to pseudo-state $z_{i,t}$.

In the following, it is considered that the pseudo-state $z_{i,t} \in Z_i$ contains all the relevant information for the optimization of the CID market trading of an asset-optimizing agent. Thus, replacing the true state $h_{i,t}$ with pseudo-state $z_{i,t}$ is not considered to lead to a sub-optimal policy. The resulting decision process after the state and action spaces reductions is illustrated in Figure 2.4.

2.4.6 Generation of artificial trajectories



(A) Market depth per product (for products Q_1 to Q_6) at a time-step t with no arbitrage potential.

(B) The corresponding aggregated curves for a non profitable order book.

FIGURE 2.3

In this section, the generation of artificial trajectories for addressing exploration issues in an offline setting is discussed. Indeed, if we were to implement an agent that selects at every time-step among the “Idle" and “Trade" actions, we would collect a certain number of trajectories (one per day) over a certain period of interactions with the real market. The collected dataset could be used to train a policy using a batch mode RL algorithm, as described in Section 2.4.2. Every time a new trajectory would arrive, it would be appended in the previous set of trajectories and the entire dataset could be used to improve the trading policy.

As discussed in Section 2.4.2, sufficient exploration of the state and action spaces is a key requirement for converging to a near-optimal policy. The RL agent needs to explore unknown grounds in order to discover interesting policies (exploration). It should also apply

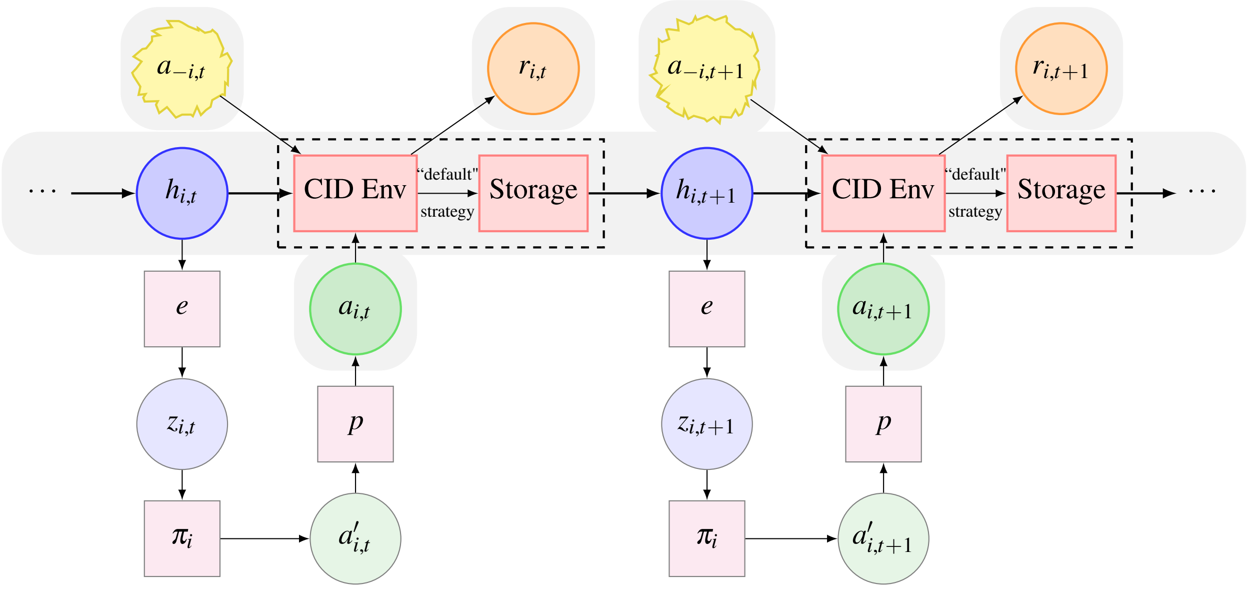
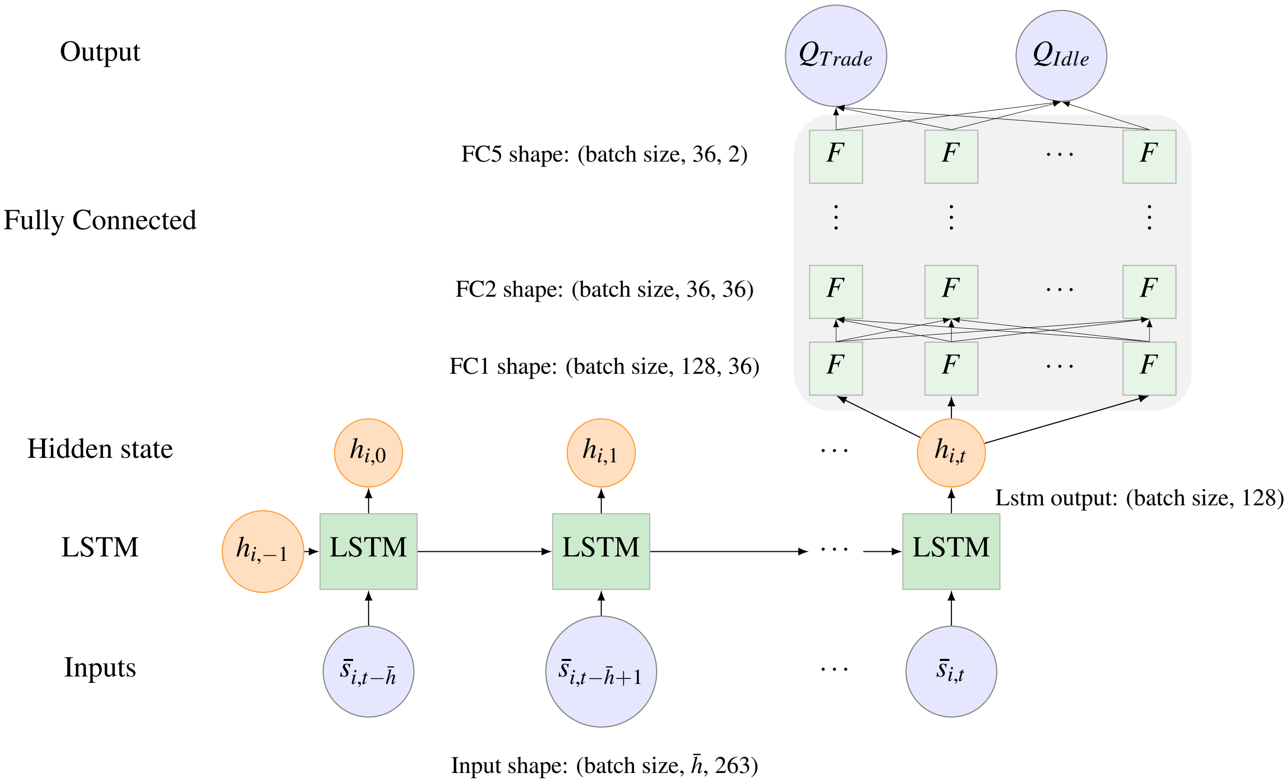


FIGURE 2.4: Schematic of the decision process. The original MDP is highlighted in a gray background. The state of the original MDP $h_{i,t}$ is encoded in pseudo-state $z_{i,t}$. Based on $z_{i,t}$, agent i takes an high-level action $a'_{i,t}$, according to its policy π_i. This action $a'_{i,t}$ is mapped to an original action $a_{i,t}$ and submitted to the CID market. The CID market makes a transition based on the action of agent i and the actions of the other agents $a_{-i,t}$. After this transition, the market position of agent i is defined and the control actions for storage device are derived according to the "default" strategy. Each transition yields a reward $r_{i,t}$ and a new state $h_{i,t}$.



Output
Q_{Trade}
Q_{Idle}
FC5 shape: (batch size, 36, 2)
Fully Connected
F
$\cdots$
FC2 shape: (batch size, 36, 36)
FC1 shape: (batch size, 128, 36)
Hidden state
$h_{i,0}$
$h_{i,1}$
$\cdots$
$h_{i,t}$
Lstm output: (batch size, 128)
LSTM
$h_{i,-1}$
LSTM
Inputs
$\bar{s}_{i,t-\bar{h}}$
$\bar{s}_{i,t-\bar{h}+1}$
$\bar{s}_{i,t}$
Input shape: (batch size, $\bar{h}$, 263)

FIGURE 2.5: Schematic of the neural network architecture.

Algorithm 3 Generation of artificial trajectories

```
Input: $L^{train}$, $E$, $ep$, $\varepsilon$, $decay$
Output: $\hat{Q}$, $F$
Initialize $\hat{Q} \equiv 0$
$M \leftarrow E \cdot |L^{train}|$
$m \leftarrow 0$
while $m \geq M$ do
    for $iter_j \leftarrow 0$ to $ep$ do
        $d \leftarrow rand(L^{train})$        ▷ Randomly pick a day $d$ from train set $L^{train}$
        $\zeta_m \leftarrow simulate(d, \varepsilon - greedy(\hat{Q}))$
        ▷ Generate trajectory $\zeta_m$ by simulating day $d$ using $\varepsilon$-greedy policy
        $F.add(\zeta_m)$        ▷ Append trajectory from day $d$ to set $F$
        $\varepsilon \leftarrow anneal(\varepsilon, decay, iter_i)$        ▷ Anneal the value of $\varepsilon$ based on $decay$ parameter
        $m \leftarrow m + 1$
    end for
end while
Update $\hat{Q}$ using set $F$ according to equations (2.32), (2.33)        ▷ Fit new $\hat{Q}$ functions
return $\hat{Q}, F$
```

these learned policies to get high rewards (exploitation). However, since the set of collected trajectories would come from a real agent, the visitation of many different states is expected to be limited.

Furthermore, the aforementioned approach requires the direct interaction with the CID market in order to collect samples from the unknown initial state distribution and from the opponents' actions. In the RL context, exploration is then performed when the agent selects a different action than the one that, according to its experience, will yield the highest rewards. In real life, it is unlikely for a trader to select such actions, and potentially bear negative revenues, for the sake of gaining more experience. This leads to limited exploration of the learning process and would result in a suboptimal policy.

Assumption 7 (*No impact on the behaviour of the rest of the agents*). The actions of trading agent i do not influence the future actions of the rest of the agents $-i$ in the CID market. In this way, agent i is not capable of influencing the market.

Assumption (7) implies that each of the agents $-i$ entering in the market would post orders solely based on their individual needs. Furthermore, its actions are not considered as a reaction to the actions of the other market players.

Leveraging Assumption (7) allows one to tackle the exploration issues discussed previously in an offline setting by generating several artificial trajectories using historical order book data. An artificial trajectory is generated as follows. At each time t, the agent i takes an action according to the current state of the order book. Under Assumption (7), the next state of the order book is then the historical state at time $t+1$ from which the bids accepted by agent i

have been removed. Finally, in this framework, such an artificial trajectory corresponds to a trajectory sampled from the CID model developed. We denote by E the number of episodes (times) each day from historical data is repeated and by L^{train} the set of trading days used to train the agent. We can then obtain the total number of trajectories M as $M = E \cdot |L^{train}|$.

The simulation of trajectories is performed according to the process described in Figure 1 in [97]. Nevertheless, in this framework, trajectories are generated artificially as described previously rather than directly sampled from the system. This process interleaves the generation of trajectories with the computation of an approximate Q-function using the trajectories already generated. As shown in Algorithm 3, for a number of episodes ep, we randomly select days from the training set which we simulate using an ε-greedy policy. According to this policy, an action is chosen at random with probability ε and according to the available Q-functions with probability $(1-\varepsilon)$. The generated trajectories are added to the set of trajectories. The second step consists of updating the Q-function approximation using the set of collected trajectories. This process is terminated when the total number of episodes has reached the specified number E.

This process introduces parameters L^{train}, E, ep, ε and $decay$. The selection of these parameters impacts the training progress and the quality of the resulting policy. The set of days considered for training (L^{train}) is typically selected as a proportion (e.g. 70%) of the total set of days available. The total number of episodes E should be large enough so that convergence is achieved and is typically tuned based on the application. The frequency with which the trajectory generation and the updates are interleaved is controlled by parameter ep. A small number of ep results in a large number of updates. Parameter ε is used to address the trade-off between exploration-exploitation during the training process. As the training evolves, this parameter is annealed based on some predefined parameter $decay$, in order to gradually reduce exploration and to favour exploration along the (near-)optimal trajectories. In practice, the size of the buffer F cannot grow infinitely due to memory limitations, so typically a limit on the number of trajectories stored in the buffer is imposed. Once this limit is reached, the oldest trajectories are removed as new ones arrive. The buffer is a double-ended queue of fixed size.

2.4.7 Neural Network architecture

As described in Section 2.4.5, pseudo-state $z_{i,t}$ contains a sequence of variables whose length is proportional to t. This motivates the use of Recurrent Neural Networks (RNNs), that are known for being able to efficiently process variable-length sequences of inputs. In particular, we use Long Short-term Memory (LSTM) networks [107], a type of RNNs where a gating mechanism is introduced to regulate the flow of information to the memory state.

All the networks in this study have the architecture presented in Figure 2.5. It is composed of one LSTM layer with 128 neurons followed by five fully connected layers with 36 neurons where “ReLU" was selected as the activation function. The structure of the network (number of layers and neurons) was selected after cross-validation.

Theoretically, the length of the sequence of features that is provided as input to the neural network can be as large as the total number of trading steps in the optimization horizon. In practice though, there are limitations with respect to the memory that is required to store a tensor of this size. As we can observe in Figure 2.5, each sample in the batch contains a vector of size 249 for each time-step. Assuming a certain batch size, there is a certain limit to the number of steps that can be stored in the memory. Therefore, for practical reasons and due to hardware limitations, we assume a history length $\bar{h}$ defined as $z_{i,t} = (a'_{i,t-\bar{h}-1}, r_{i,t-\bar{h}-1}, s'_{i,t-\bar{h}}, a'_{i,t-\bar{h}}, r_{i,t-\bar{h}}, ..., a'_{i,t-1}, r_{i,t-1}, s'_{i,t}) \in Z_i$. At each step t, the history length $\bar{h}$ takes the minimum value between the time-step t and $\bar{h}_{max}$, ($\bar{h} = min(t, \bar{h}_{max})$). Additionally, we provide the variable $\bar{s}_t = (a'_{i,t-1}, r_{i,t-1}, s'_{i,t})$, as a fixed size input for each step t of the LSTM. Consequently, the pseudo-state can be written as $z_{i,t} = (\bar{s}_{t-\bar{h}}, ..., \bar{s}_t)$.

2.4.8 Asynchronous Distributed Fitted Q iteration

The exploration requirements of the continuous state space, as defined previously introduce the necessity for collecting a large number of trajectories M. The total time required for gathering these trajectories heavily depends on the simulation time needed for one episode. In this particular setting developed, the simulation time can be quite long since, at each decision step, if the action selected is “Trade", an optimization model is constructed and solved.

In order to address this issue, we resort to an asynchronous architecture, similar to the one proposed in [108], presented in Figure 2.6. The two processes, described in Section 2.4.6, namely generation of trajectories and computation of the Q-functions, run concurrently with no high-level synchronization.

Multiple actors that run on different threads are used to generate trajectories. Each actor contains a copy of the environment, an individual ε-greedy policy based on the latest version of the Q functions and a local buffer. The actors use their ε-greedy policy to perform transitions in the environment. The transitions are stored in the local buffer. When the local buffer of each actor is filled, it is appended to the global buffer, the agent collects the latest Q-functions

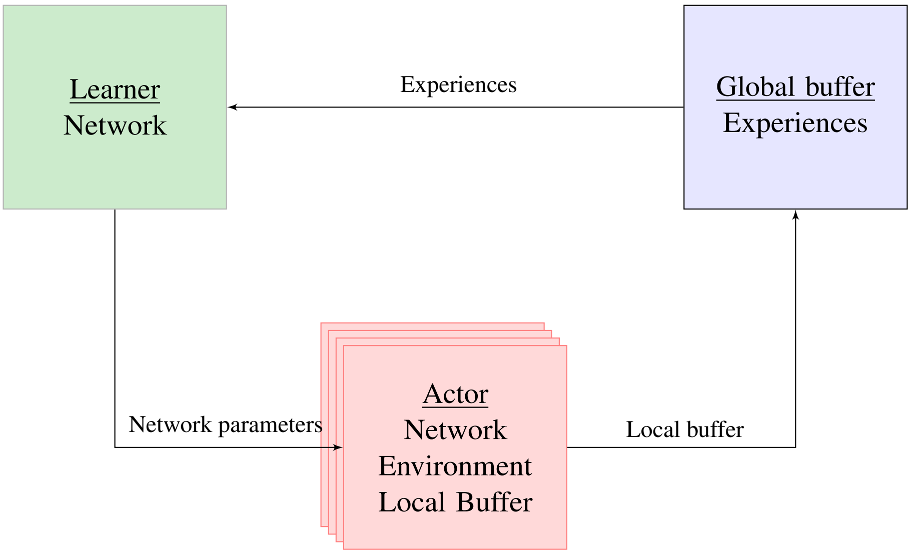


FIGURE 2.6: Schematic of the asynchronous distributed architecture. Each actor runs on a different thread and contains a copy of the environment, an individual ε-greedy policy based on the latest version of the network parameters and a local buffer. The actors generate trajectories that are stored in their local buffers. When the local buffer of each actor is filled, it is appended to the global buffer and the agent collects the latest network parameters from the learner. A single learner runs on a separate thread and is continuously training using experiences from the global buffer.

from the learner and continues the simulation. A single learner continuously updates the Q-functions using the simulated trajectories from a global buffer.

The benefits from asynchronous methods in Deep Reinforcement Learning (DRL) are elaborated in [50]. Each actor can use a different exploration policy (different initial ε value and decay) in order to enhance diversity in the collected samples which leads to a more stable learning process. Additionally, it is shown that the total computational time scales linearly with the number of threads considered. Another major advantage is that distributed techniques were shown to have a super-linear speedup for one-step methods that are not only related to computational gains. It is argued that, the positive effect of having multiple threads leads to a reduction of the bias in one-step methods [50]. In this way, these algorithms are shown to be much more data efficient than the original versions.

2.5 Case study

The proposed methodology is applied to the case of a PHES unit. Firstly, the parameters and the exogenous information used for the optimization of the CID market participation of a PHES operator are described. Secondly, the benchmark strategy used for comparison purposes and the process that was carried out for validation are presented. Finally, performance results of the obtained policy are presented and discussed.

2.5.1 Parameters specification

The proposed methodology is applied for an instance of a PHES unit[2] participating in the German CID market with the following characteristics:

- $\overline{SoC_i} = 40$ MWh,
- $\underline{SoC_i} = 0$ MWh,
- $SoC_i^{init} = SoC_i^{term} = \left(\overline{SoC_i} - \underline{SoC_i}\right)/2$,
- $\overline{C_i} = \overline{G_i} = 8$ MW,
- $\underline{C_i} = \underline{G_i} = 0$ MW,
- $\eta = 90\%$.

The discrete trading horizon has been selected to be the full day, i.e. $T = \{16:00, ..., 00:00, ..., 23:15\}$. The trading time interval is selected to be $\Delta t = 15$ min. Thus the trading process takes $K = 124$ steps until termination. Moreover, all 96 quarter-hourly products of the day, $X = \{Q_1, .., Q_{96}\}$, are considered. Consequently, the delivery timeline is $\bar{T} = \{00:00, ..., 23:45\}$, with $\tau_{init} = 00:00$ and $\tau_{term} = 24:00$ and the delivery time interval is $\Delta\tau = 15$ min. Each product can be traded until 30 minutes before the physical delivery of electricity begins (e.g. $t_{close}(Q_1) = 23:30$ etc.).

For the construction of the training/test sets, we proceed as following. Due to the high computational burden, we train our algorithm on a period of $|L^{train}| = 36$ days in which a high variance in prices is observed and therefore high profit potential. Subsequently, we evaluate its performance in the following $|L^{test}| = 110$ days. The total number of simulated episodes was selected to be $E = 10000$ episodes for the artificial trajectories generation process, described in Section 2.4.6. During the trajectories generation process the high-level actions (“Trade" or “Idle") were chosen following an ε-greedy policy. As described in Section 2.4.8, each of the actor threads is provided with a different exploration parameter ε that is initialised with a random uniform sample in the range $[0.1, 0.5]$. The parameter ε is then annealed exponentially until a zero value is reached.

[2] A small instance of the storage unit was selected due to the low volumes available in the historical order book data used.

The pseudo-state $z_{i,t} = (s'_{i,0}, a'_{i,0}, r_{i,0}, ..., a'_{i,t-1}, r_{i,t-1}, s'_{i,t}) \in Z_i$ is composed of the entire history of observations and actions up to time-step t, as described in Section 2.4.5. For the sake of memory requirements, as explained in Section 2.4.7, we assume that the last ten trading steps contain sufficient information about the past. Thus, the pseudo-state is transformed in sequences of fixed length $\bar{h}_{max} = 10$.

2.5.2 Exogenous variable

The exogenous variable $w_{i,t}^{exog}$ represents any relevant information available to agent i about the system. In this case study, we assumed that the variable $w_{i,t}^{exog}$ contains:

- The 24 values of the Day-ahead price for the entire trading day
- The Imbalance price and the system Imbalance for the four quarters preceding each time-step t
- The 96 values of the intraday auction prices for the entire trading day
- Time features: i) the month and ii) whether the traded day is a weekday or weekend

2.5.3 Benchmark strategy

The strategy selected for comparison purposes is the *rolling intrinsic* policy [106], denoted by π^{RI}. According to this policy, the agent selects at each trading time-step t the action “Trade", as described in Section 2.4.4. This benchmark is selected since it represents the current practice in some industrial applications for the optimization of PHES unit market participation. Additionally, the benchmark presented in [95] could be used for comparison purposes. However, the basis of our analysis is significantly different. In particular, the assumptions related to the storage operation (Assumptions 5 and 6) as well as the fact that quarterly products are considered (instead of hourly) in this chapter, constitute the comparison impossible.

2.5.4 Validation process

The performance of the policy obtained using the fitted Q iteration algorithm, denoted by π^{FQ}, is evaluated on test set L^{test} that contains historical data from 110 days. These days are not used during the training process. This process of backtesting a strategy on historical data is widely used because it can provide a measure of how successful a strategy would be if it had been executed in the past. However, there is no guarantee that this performance can be expected in the future. This validation process heavily relies on Assumption (7) about the inability of the agent to influence the behaviour of the other players in the market. It can still provide an approximation on the results of the obtained policy before deploying it in real life. However, the only way to evaluate the exact viability of a strategy is to deploy it in real life.

We compare the performances of the policy obtained by the fitted Q iteration algorithm π^{FQ} and the *rolling intrinsic* policy π^{RI}. The comparison is based on the computation of the return of the policies on each day. For a given policy, the return over a day is simply computed by running the policy on the day and summing up the rewards obtained.

Our learning algorithm has two sources of variance, namely those related to the generation of the new trajectories and those related to the learning of the Q-functions from the set of trajectories. Hence, we perform several runs and average the performances of the policies learned. In the following, when we report the performance of a fitted Q iteration policy over a dataset, we will actually report the average performances of ten learned policies over this dataset.

We describe the different indicators that will be used afterwards to assess the performance of our method. These indicators are computed for both the training set and the test set, but are detailed hereafter when they are computed for the test set. It is straightforward to adapt the procedure for computing the indicators for the training set.

Let $V_d^{\pi^{FQ}}$ and $V_d^{\pi^{RI}}$ denote the total return of the fitted Q and the *rolling intrinsic* policy for day d, respectively. We gather the obtained returns of each policy for each day $d \in L^{test}$. We sort the returns in ascending order, and we obtain an ordered set containing a number of $|L^{test}|$ values for each policy. We provide descriptive statistics about the distribution of the returns of each policy $V_d^{\pi^{FQ}}$ and $V_d^{\pi^{RI}}$ on the test set L^{test}. In particular, we report the mean, the minimum and maximum values achieved for the set considered. Moreover, we provide the values obtained for each of the quartiles (25%, 50% and 75%) of the set.

Additionally, we compute the sum of returns over the entire set of days as follows:

$$V^{\pi^{FQ}} = \sum_{d \in L^{test}} V_d^{\pi^{FQ}}, \tag{2.52}$$

$$V^{\pi^{RI}} = \sum_{d \in L^{test}} V_d^{\pi^{RI}}. \tag{2.53}$$

An alternative performance indicator considered is the discrepancy of the returns coming from the fitted Q policy with respect to the risk-averse *rolling intrinsic* policy. We define the profitability ratio r_d for each day $d \in L^{test}$, that corresponds to the signed percentage difference between the two policies as follows:

$$r_d = \frac{V_d^{\pi^{FQ}} - V_d^{\pi^{RI}}}{V_d^{\pi^{RI}}} \cdot 100\%. \tag{2.54}$$

In a similar fashion, we sort the profitability ratios obtained for each day in the test set and we provide descriptive statistics about its distribution across the set. The mean, minimum and maximum values of the profitability ratio as well as the values of each quartile are reported. Finally, we compute the profitability ratio for the sum of returns over the entire set between the two policies, as:

$$r_{sum} = \frac{V^{\pi^{FQ}} - V^{\pi^{RI}}}{V^{\pi^{RI}}} \cdot 100\%. \tag{2.55}$$

2.5.5 Results

The performance indicators described previously are computed for both the training and the test set. The results obtained are summarised in Tables 2.3 and 2.4. Descriptive statistics about the distribution of the returns from both policies as well as the profitability ratio are presented for each dataset.

It can be observed that on average π^{FQ} yields better returns than π^{RI} both on the training and the test set. More specifically, on the training set, the obtained policy performs, on average 7.6% better than the *rolling intrinsic* policy. For the top 50% of the training days the profitability ratio is higher than 3.5% and in some cases it even exceeds 10%. Overall, the total profits coming from the fitted Q policy add up to €14523.1, yielding a difference of €1144. (8.5%) more than the profits from the *rolling intrinsic* for the set of 36 days considered.

TABLE 2.3: Descriptive statistics of the returns obtained on the days of the training set for policies π^{FQ} and π^{RI}. The last column also provides the corresponding profitability ratios.

	π^{FQ} **returns (€)**	π^{RI} **returns (€)**	**r (%)**
mean	403.4	371.6	7.6
min	232.2	114.4	−12.7
25%	298.9	202.1	−2.2
50%	351.5	287.1	3.5
75%	463.9	345.1	7.5
max	1064.6	416.5	69.0
sum	14523.1	13378.5	8.5

The fitted Q policy yields on average a 2.25% greater profit on the test set with respect to the returns of the *rolling intrinsic* policy. It is important to highlight that for 50% of the test set, the profits from the fitted Q policy are higher than 1% in comparison to the *rolling intrinsic*. The difference between the total profits resulting from the two policies over the set of 110 days considered amounts to €901.4 (2.16%).

TABLE 2.4: Descriptive statistics of the returns obtained on the days of the test set for policies π^{FQ} and π^{RI}. The last column also provides the corresponding profitability ratios.

	π^{FQ} **returns (€)**	π^{RI} **returns (€)**	**r (%)**
mean	401.5	392.9	2.25
min	121.3	126.8	−4.7
25%	272.7	266.9	−1.5
50%	347.4	345.5	1.7
75%	463.9	465.9	4.8
max	1351.4	465.9	19.5
sum	42559.2	41657.7	2.16

The distribution of training and test set samples according to the obtained profitability ratio is presented in Figure 2.7. It can be observed that most samples are spread in the interval between $0-5\%$ and that the distribution has a positive skew. From the standpoint of practical implementation this result allows us to construct a wrapper around the current industrial standard practices and expect an average improved performance of 2%. However, as discussed earlier, the back-testing of a strategy in historical data may differ from the outcomes in real deployment for various reasons.

The evolution of the expected return of the fitted Q iteration policy $V^{\pi^{FQ}}$ as function of the training episodes (number of trajectories collected) is presented in Figure 2.8. We can observe that at the early steps of the training the fitted Q policy performs very similar to the *rolling intrinsic*. Later in the training process it progressively learns the right moments to idle in order to increase its returns. The progressive evaluation of the fitted Q policy in the test set is illustrated in Figure 2.8. The shaded area in both graphs represents the variance obtained between the ten different runs.

2.6 Discussion

In this section, we provide some remarks related to the practical challenges encountered and the validity of the assumptions considered throughout this chapter.

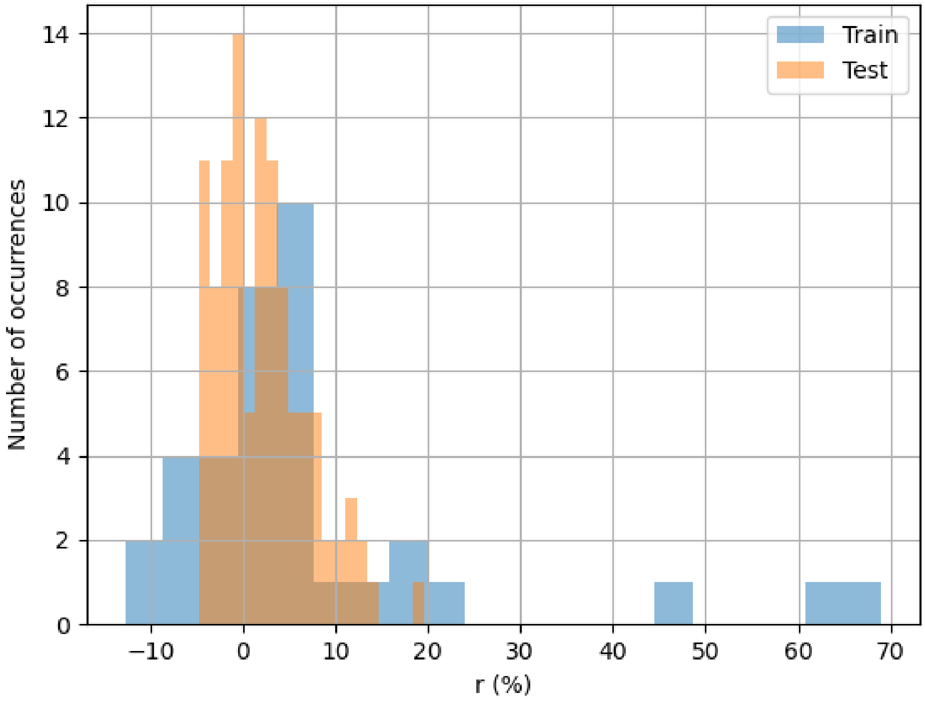


FIGURE 2.7: Profitability ratio.

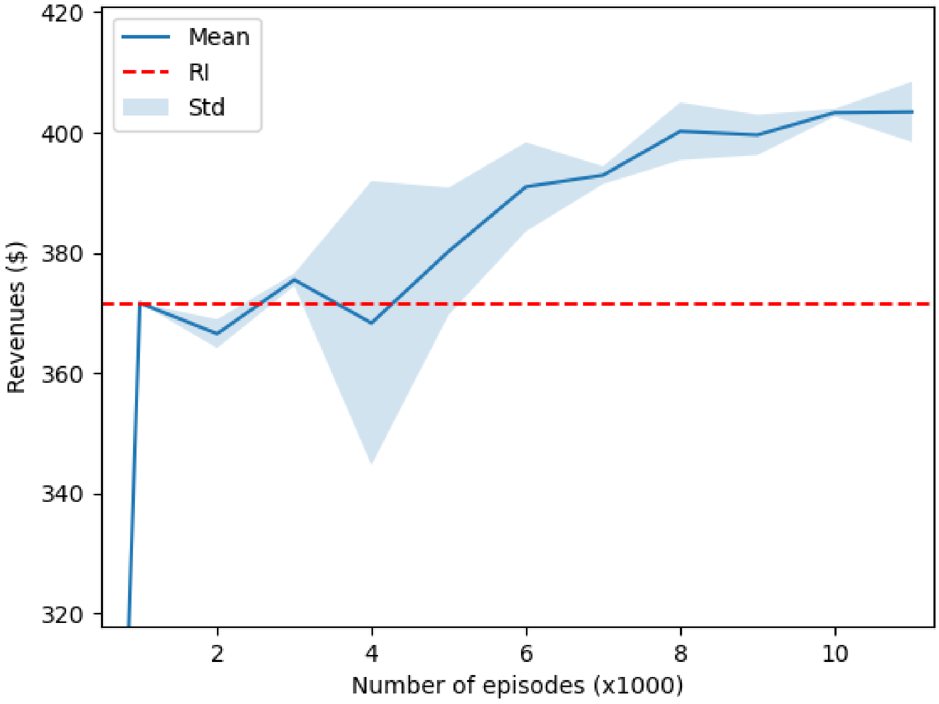


FIGURE 2.8: Progressive evaluation in the train set.

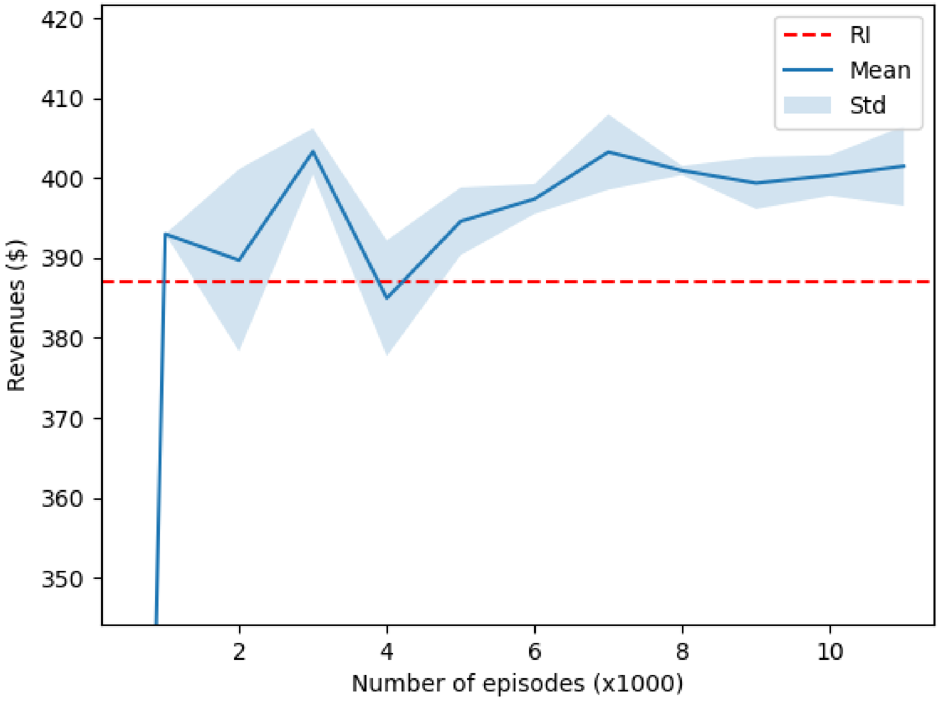


FIGURE 2.9: Progressive evaluation in the test set.

2.6.1 Behaviour of the rest of the agents

In this chapter, we assumed (Assumption 1) that the rest of the agents $-i$ post orders in the market based on their needs and some historical information of the state of the order book. In reality, the available information that the other agents possess is not accessible by agent i. This fact gives rise to issues related to the validity of the assumption that the process is Markovian.

We further assumed (Assumption 7) in Section 2.4.6 that the behaviour of agent i does not influence the strategy of the other agents $-i$. Based on this assumption the training and the validation process were performed using historical data. However, the strategy of each of the market participants is highly dependent on the actions of the rest participants, especially in a market with limited liquidity such as the CID market.

These assumptions, although slightly unrealistic and optimiztic, provide us with a meaningful testing protocol for a trading strategy. The actual profitability of a strategy can be obtained by deploying the strategy in real-time. However, it is important to show that the strategy is able to obtain substantial profits in back-testing first.

2.6.2 Partial observability of the process

In Section 2.3, the decision-making problem studied in this chapter was framed as an MDP after considering certain assumptions. Theoretically, this formulation is very convenient, but does not hold in practice. In particular, the reduced pseudo-state may not contain all the relevant information required.

Indeed, the trading agents do not have access to all the information required. For instance, a real agent does not know how many other agents are active in the market. They do not know the strategy of each agent either. There is also a lot of information gathered by w^{exog} which is not available for the agent. Finally, the fact that the state space was reduced results in an inevitable loss of information.

Therefore, it would be more accurate to consider a Partially Observable Markov Decision Process (POMDP) instead. In a POMDP, the real state is hidden and the agent only has access to observations. For an RL algorithm to properly work with a POMDP, the observations have to be representative of the real hidden states.

2.6.3 Action space reduction

The presented action space (High-level actions) is rather restricted in the sense that the storage unit will buy energy for a product only if it can sell it back to another product at the same instant. According to this definition of the action space, there is no risk of buying energy without using it later. However, as expected, this strategy results in reduced profits eventually. The action space reduction performed leads to a rather constrained set of admissible policies. The restrictions arise from the imposed rule that no trade of energy is allowed if it cannot be physically backed (Assumption 5). Although this assumption is made to fully comply with the German regulation, it is rather restrictive on the profits that can be achieved by the storage unit.

Alternatively, one can relax this assumption and use the reduced action space A^{red}. This would significantly increase the dimensionality of the action space and the need for exploration. Additionally, that would imply the need for a risk measure in order to quantify and control the freedom to which the resulting policy is operating.

2.6.4 Exploration

There are two main issues related to the state space exploration that result in the somewhat limited performance of the obtained policy. First, in the described setting, the way in which we generate the artificial trajectories is very important for the success of the method. The generated states must be “representative" in the sense that the areas around these states are visited often under a near optimal policy [40]. In particular, the frequency of appearance of these areas of states in the training process should be proportional to the probability of occurrence under the optimal policy. However, in practice, we are not in a position to know which areas are visited by the optimal policy. In that respect, the asynchronous distributed algorithm used in this chapter was found to successfully address the issue of state exploration.

Second, the assumptions (Assumptions 3, 4, 5) related to the operation of the storage device according to the “default" strategy without any imbalances allowed, as well as the participation of the agent as an aggressor, are restrictive with respect to the set of all admissible policies. Additionally, the adoption of the reduced discrete action space described in Section 2.4.4 introduces further restrictions on the set of available actions. Although having a small and discrete space is convenient for the optimization process, it leads to limited state exploration. For instance, the evolution of the state of charge of the storage device is always given as the output of the optimization model based on the order book data. Thus, in this configuration, it is not possible to explore all areas of the state space (storage levels) but only certain areas driven by the historical order book data. However, evaluating the policy on a different dataset might lead to areas of the state space (e.g. storage level) that are never visited during training, leading to poor performance. Potential mitigations of this issue involve diverse data augmentation techniques and/or different representation of the action space.

2.7 Conclusions and future work

In this chapter, a novel RL framework for the participation of a storage device operator in the CID market is proposed. The energy exchanges between market participants occur through a centralized order book. A series of assumptions related to the behaviour of the market agents and the operation of the storage device are considered. Based on these assumptions, the sequential decision-making problem is cast as an MDP. The high dimensionality of both the action and the state spaces increase the computational complexity of finding a policy. Thus, we motivate the use of discrete high-level actions that map into the original action space. We further propose a more compact state representation. The resulting decision process is solved using fitted Q iteration, a batch mode reinforcement learning algorithm. The results illustrate that the obtained policy is a low-risk policy that is able to outperform on average the state of the art for the industry benchmark strategy (*rolling intrinsic*) by 2.2% on the test set. The proposed method can serve as a wrapper around the current industrial practices that provides decision support to energy trading activities with low risk.

The main limitations of the developed strategy originate from: i) the insufficient amount of relevant information contained in the state variable, either because the state reduction proposed leads to a loss of information or due to the unavailability of information and ii) the limited state space exploration as a result of the proposed high-level actions in combination with the use of historical data. To this end and as future work, a more detailed and accurate representation of the state should be devised. This can be accomplished by increasing the amount of information considered, such as RES forecasts, and by improving the order book representation. We propose the use of continuous high-level actions in an effort to gain state exploration without leading to very complex and high-dimensional action space.

Appendix

2.8 Nomenclature

Acronyms

ADP	Approximate Dynamic Programming.
CID	Continuous Intraday.
DRL	Deep Reinforcement Learning.
FCFS	First Come First Served.
MDP	Markov Decision Process.
OB	Order Book.
PHES	Pumped Hydro Energy Storage.
RES	Renewable Energy Sources.

Sets and indexes

Name	Description
i	Index of an agent.
$-i$	Index of all the agents except agent i.

j	Index of an order.
m	Index of a sample of quadruples.
d	Index of a day in a set.
t	Trading time-step.
τ	Discrete time-step of delivery.
A	Joint action space for all the agents.
A_i	Action space of agent i.
A_{-i}	Action space of the rest of the agents $-i$.
A_i^{red}	Reduced action space of agent i.
A_i'	Set of high-level actions for agent i.
$\bar{A}_i$	Set of all factors for the partial/full acceptance of orders by agent i.
E	Set of conditions that can apply to an order.
F	Set of all sampled trajectories.
F'	Set of sampled one-step transitions.
F_t'	Set of sampled one-step transitions for time t.
H_i	Set of all histories for agent i.
I	Set of agents.
L^{train}	Set of trading days used to train the agent.
L^{test}	Set of trading days used to evaluate the agent.
N_t	Set of all available order unique indexes at time t.
N_t'	Set of all the unique indexes of new orders posted at time t.
N_τ	Set of all the unique indexes of orders for delivery at τ.
O_t	Set of all available orders in the order book at time t.
S^{OB}	Set of all available orders in the order book.
S'^{OB}	Low dimensional set of all available orders in the order book.
S_i	State space of agent i.
T	Trading horizon, i.e. time interval between first possible trade and last possible trade.
$T(x)$	Discretization of the trading timeline for product x.
$\bar{T}$	Discretization of the delivery timeline.
$\bar{T}(t)$	Discretization of the delivery timeline at trading step t.

T^{Imb}	Discretization of the imbalance settlement timeline.
X	Set of all available products.
X_t	Set of all available products at time t.
Z_i	Set of pseudo-states for agent i.
Π	Set of all admissible policies.

Parameters

Name	Description
$\overline{C_i}$	Maximum consumption level for the asset of agent i.
$\underline{C_i}$	Minimum consumption level for the asset of agent i.
E	Number of episodes.
e	Conditions applying on an order other than volume and price.
ep	Number of simulations between two successive Q function updates.
$decay$	Parameter for the annealing of ε.
$\overline{G_i}$	Maximum production level for the asset of agent i.
$\underline{G_i}$	Minimum production level for the asset of agent i.
$\bar{h}$	Sequence length of past information.
$\bar{h}_{max}$	Maximum sequence length of past information.
$I(\tau)$	Imbalance price for delivery period $\delta(x)$.
K	Number of steps in the trading period.
M	Number of samples of quadruples.
n	Number of agents.
o_t	Market order.
p	Price of an order.
$p_{\max}$	Maximum price of an order.
$p_{\min}$	Minimum price of an order.
$\overline{SoC_i}$	Maximum state of charge of storage device.
$\underline{SoC_i}$	Minimum state of charge of storage device.
SoC_i^{init}	State of charge of storage device at the beginning of the delivery timeline.
SoC_i^{term}	State of charge of storage device at the end of the delivery timeline.
$t_{close}(x)$	End of trading period for product x.
$t_{delivery}(x)$	Start of delivery of product x.

$t_{open}(x)$	Start of trading period for product x.
$t_{settle}(x)$	Time of settlement for product x.
v	Volume of an order.
x	Market product.
y	Side of an order (“Sell" or “Buy").
y_t^m	Target computed for sample m at time t.
$\delta(x)$	Time interval covered by product x (delivery).
Δt	Time interval between trading time-steps.
$\Delta \tau$	Time interval between delivery time-steps.
ε	Parameter for the ε-greedy policy.
η	Charging/discharging efficiency of storage device.
θ_t	Parameters vector of function approximation at time t.
$\lambda(x)$	Duration of time-interval $\delta(x)$.
ζ	A single trajectory.
ζ_m	A single indexed trajectory.
τ_{init}	Initial time-step of the delivery timeline.
τ_{term}	Terminal time-step of the delivery timeline.

Variables

Name	Description
a_t	Joint action from all the agents at time t.
$a_{i,t}$	Action of posting orders by agent i at time t.
$a_{-i,t}$	Action of posting orders by the rest of the agents $-i$ at time t.
$a'_{i,t}$	High-level action by agent i at time t.
$a^j_{i,t}$	Acceptance (partial/full) factor for order j by agent i at time t.
$\bar{a}_{i,t}$	Factors for the partial/full acceptance of all orders by agent i at time t.
$C_{i,t}(\tau)$	Consumption level at delivery time-step τ computed at time t.
$c_{i,t}(t')$	Consumption level during the delivery interval.
$e_{i,t}$	Random disturbance for agent i at time t.
$G_{i,t}(\tau)$	Generation level at delivery time-step τ computed at t.
$g_{i,t}(t')$	Generation level during the delivery interval.
$h_{i,t}$	History vector of agent i at time t.

$k_{i,t}(\tau)$	Binary variable that enforces either charging or discharging of the storage device.
$P^{mar}_{i,t}(x)$	Net contracted power of agent i for product x (delivery time-step τ) at time t.
$P^{res}_{i,t}(\tau)$	Residual production of agent i delivery time-step τ (for product x) at time t.
$P^{res}_{i}(\tau)$	Final residual production of agent i for product
$r_{i,t}$	Instantaneous reward of agent i at time t.
r_d	Profitability ratio at day d.
r_{sum}	Profitability ratio for the sum of returns over set.
$s_{i,t}$	State of agent i at time t.
$SoC_{i,t}(\tau)$	State of charge of device at delivery time-step τ computed at t.
s^{OB}_t	State of the order book at time t.
s'^{OB}_t	Low dimensional state of the order book at time t.
$s^{private}_{i,t}$	Private information of agent i at time t.
$\bar{s}_t$	Triplet of fixed size, part of pseudo-state $z_{i,t}$ that serves as an input at LSTM at time t.
$u_{i,t}$	Aggregate (trading and asset) control action of the asset trading agent i at time t.
$v^{con}_{i,t}(x)$	Volume of product x contracted by agent i at time t.
$w^{exog}_{i,t}$	Exogenous information of agent i at time t.
$z_{i,t}$	Pseudo-state for agent i at time t.
$\Delta_{i,t}(\tau)$	Imbalance for delivery time τ for agent i computed at time t.
$\Delta_i(\tau)$	Final imbalance for delivery time τ for agent i.
$\Delta G_{i,t}$	Change in the production level for the asset of agent i at time t.
$\Delta C_{i,t}$:	Change in the consumption level for the asset of agent i at time t.

Functions

Name	Description
$clear(\cdot)$	Market clearing function.
$b(\cdot)$	Univariate stochastic model for exogenous information.
$enc(\cdot)$	Encoder that maps from the original state space H_i to pseudo-state space Z_i.
$f(\cdot)$	Order book transition function.
$G^{\zeta}(\cdot)$	Revenue collected over a trajectory.

$g(\cdot)$	System dynamics of the MDP.
$k(\cdot)$	System dynamics of asset trading process.
$l(\cdot)$	Reduced action space construction function.
$P_{a_{-i,t}}(\cdot)$	Probability distribution function for the actions of the rest of the agents $-i$.
$P_{e_t}(\cdot)$	Random disturbance probability distribution function.
$P(\cdot)$	Transition probabilities of the MDP.
$P_{FQ}(\cdot)$	The stochastic process (algorithm) of fitted Q iteration.
$P_{\theta_{t,0}}(\cdot)$	Distribution of the initial parameters $\theta_{t,0}$.
$p(\cdot)$	Mapping from high-level actions A_i' to the reduced action space A_i^{red}.
$Q_t(\cdot,\cdot)$	State-action value function at time t.
$\hat{Q}(\cdot,\cdot)$	Sequence of Q-function approximations.
$R(\cdot)$	Reward function.
$u(\cdot)$	Signing convention for the volume wrt. the side (‘Buy" or ‘Sell") of each order.
$V^{\pi_i}(\cdot)$	Total expected reward function for policy π_i.
$V_d^{\pi^{FQ}}(\cdot)$	Return of the fitted Q policy π_i^{FQ} for day d.
$V_d^{\pi^{RI}}(\cdot)$	Return of the “rolling" intrinsic policy π_i^{RI} for day d.
$\mu_t(\cdot)$	Policy function at time t.
$\pi_i(\cdot)$	Policy followed by agent i.
$\rho(\cdot)$	Trading revenue function.

Chapter 3

Expanding the scope of the CID agent

In this chapter, we introduce a set of modifications to the described CID market participation problem that lead to a significant increase in the general performance of the proposed strategy. First, we motivate the use of a more compact state space representation. Moreover, we propose a process of scaling the observed states so as to have a stationary input to the function approximator predicting the optimal action. To achieve that, we propose the use of the respective day-ahead prices for each trading day, in order to scale the states coming from different trading days. Additionally, we introduce a scaling of the rewards coming from each trading day, based on the returns obtained by the *rolling intrinsic* benchmark policy in this days. The proposed changes are evaluated in a new case study. In order to obtain a good grasp of the performance improvement potential of these changes, we also define a new benchmark policy that is anticipative, i.e. the policy has access to the future and thus, can act in a near-optimal way. This policy cannot be implemented in practice because it relies on future information that a storage operator would not have in its possession in real-time. However, it is useful for quantifying the performance gap between the proposed policy and a near-optimal one. The results demonstrate that our method can outperform the *rolling intrinsic* benchmark and reach a performance that is comparable to the one of the anticipative policy.

3.1 New state space representation

As discussed in Section 2.6.4, one of the key factors curbs the performance of the proposed method arises from the limited exploration of the state space. It is observed, that the states visited during the testing phase are not “similar" (close) to the states visited during training. More precicely, the states visited in the trajectories of the historical dataset did not seem to be drawn from a stationary process as hypothesised in Section 2.3. Intuitively, it means that states visited within each trading day are drawn from a different environment (distribution). This, in turn, limits the performance that can be obtained by the proposed fitted Q iteration method that relies on visiting similar states during training and testing. In particular, the DRL agent has limited generalization capabilities due to the fact that the states visited during training come from a different distribution that the states visited during testing. In this section, we will follow a two-step process by introducing modifications to the state space that attempt to address this problem. Firstly, we propose a new, more compact state space representation that still originates from the same principles used to build the one in Section 2.4.5. However, instead of instead of a separate account of the market information and the market position, the new state vector contains these two components in one compact representation we hereby call the potential profit. Secondly, we propose a new way of scaling states (i.e. the potential profits). In particular, it was empirically observed that the daily profits collected presented large variance, i.e. the amount of profits collected would vary significantly from one day to another. To address that issue, we use the day-ahead profits that the storage unit would collect if it participated in the day-ahead market, to scale the potential profits at each step, i.e. the new states.

3.1.1 Compact state space representation

The state reduction presented in Section 2.4.5 results in the state space vector $z_{i,t} = (s'_{i,0}, a'_{i,0}, r_{i,0}, ..., a'_{i,t-1}, r_{i,t-1}, s'_{i,t}) \in Z_i$, that contains (among others) past values of the variable $s'_{i,t} = (s_t'^{OB}, P^{mar}_{i,t}, w^{exog}_{i,t}) \in S'_i$. The first component of variable $s'_{i,t}$ is $s_t'^{OB}$, that contains statistics of the aggregated demand/supply curves. The second component of $s'_{i,t}$ is the market position $P^{mar}_{i,t}$ of the agent. The use of these two components is intended to provide a proxy of the potential profits that could be collected at each trading step t by the storage unit operator.

Instead of using these two components, we can directly compute the profits that the storage unit operator stands to make, should the operator selected the “Trade" action at time-step t i.e. the potential profits $\hat{r}_{i,t}$. This is a hypothetical computation and has no impact on the actual operation of the storage unit or on the market. The computation of the potential profits $\hat{r}_{i,t}$ is performed using the optimization problem defined in Algorithm 2 without applying the output actions ($\bar{a}_{i,t}$) to the real system. More specifically, we solve the optimization problem as it is presented in Algorithm 2 and the only information that we keep from the generated outputs is the value of the objective function that corresponds to the potential profits $\hat{r}_{i,t}$. Figure 3.1 presents the evolution of the potential profits $\hat{r}_{i,t}$ over the course of the trading horizon for two distinct traded days, namely the 1st and the 2nd of January 2015. This modification allows us to extract the useful information (how much profit would the operator make right a each trading step), from the observed market and storage unit situation. This feature engineering

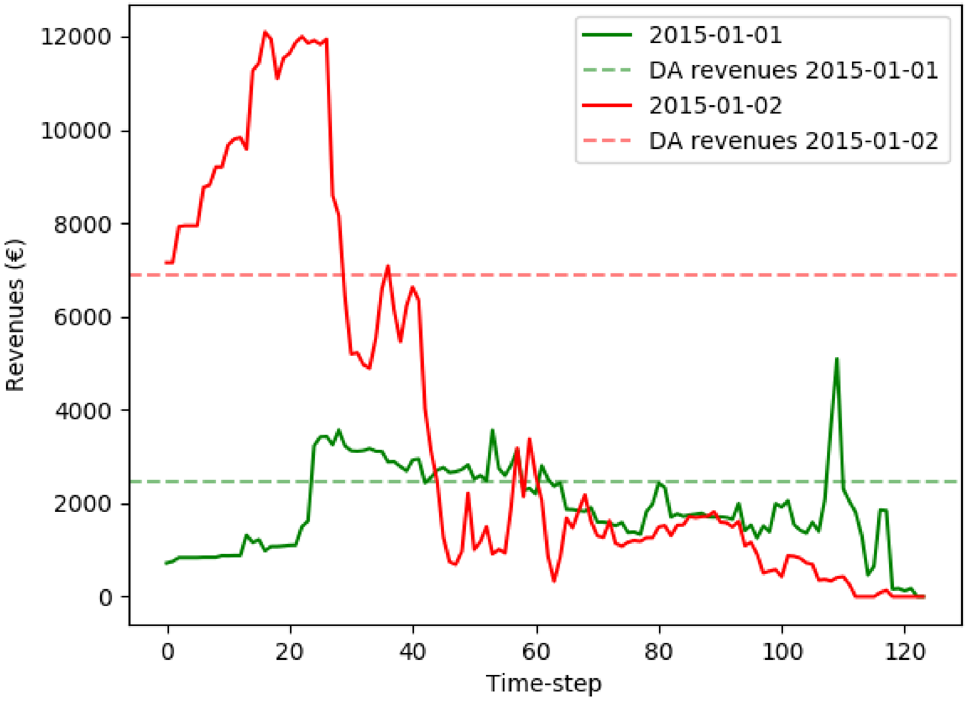


FIGURE 3.1: States from two different days.

represents a much more compact (low-dimensional) representation of the state that is expected to improve the performance of the trading agent and lead to computational performance gains.

3.1.2 Making the state stationary

As we can observe in Figure 3.1, the potential profits for two distinct days may be non-stationary (take values from different distributions). In particular, we notice that, for the 1st of January, the profits take values less than 4,000€for most of the (trading) horizon, while for the 2nd of January profits reach values up to 12,000 €. State variables with such distinct values are expected to lead to exploration issues both during training and testing. In order to mitigate this side effect, we proceed by scaling the potential profits. In particular, for each day d, we divide the potential profits $\hat{r}_{i,t}$ observed at each step t, by the revenues that would be collected by the same storage unit at day d, if the unit participated in the day-ahead market r_d^{DA} (dashed lines in Figure 3.1). We compute the scaled potential profit as:

$$\hat{r}'_{i,t} = \frac{\hat{r}_{i,t}}{r_d^{DA}} \tag{3.1}$$

For the computation of the day-ahead market revenues r_d^{DA}, we solve an optimization

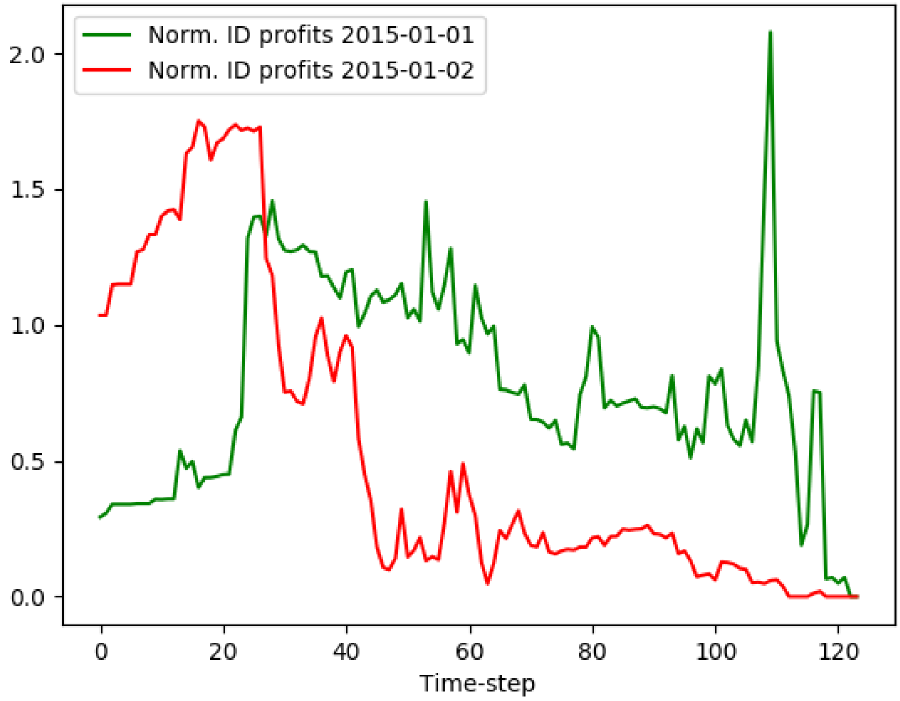


FIGURE 3.2: Normalized states from two different days.

problem[1] based on the the day-ahead prices for the trading day d, that exist in the exogenous part of the state $w_{i,t}^{exog}$. This scaling allows for comparing two days that may be very different in terms of the scale of profits collected and thus, allows for isolating the patterns of the potential profits in the day. For instance we can observe that a general pattern in the day is the increase of the potential profit in the first 20 trading steps (steps leading towards the first gate closure at 23:30). The new information variable is now represented by $s''_{i,t} = (\hat{r}'_{i,t}, w_{i,t}^{exog}) \in S''_i$ which is a much more compact representation of the previous high-dimensional vector $s'_{i,t}$. Therefore, the new low-dimensional pseudo-state is defined as $z'_{i,t} = (s''_{i,0}, a'_{i,0}, r_{i,0}, ..., a'_{i,t-1}, r_{i,t-1}, s''_{i,t}) \in Z'_i$. In the following, it is considered that the pseudo-state $z'_{i,t} \in Z'_i$ contains all the relevant information for the optimization of the CID market trading of an asset-optimizing agent.

3.2 New reward function

The large variance between the returns collected in different days can also influence the value functions that we attempt to learn with the fitted Q iteration method. In particular, it is observed that, for similar states that are visited in different days, the total returns can be in

[1] A profit maximization optimization model, considering the operational constraints of the storage unit.

different value ranges. This leads to large variance in the optimal Q values for each day. Since the Q functions represent the expected cumulative rewards given an observed state, as it is shown in equations (2.27) and (2.29), the resulting Q functions fail to sufficiently approximate the optimal Q functions.

3.2.1 Making the reward function stationary

To address this problem we design a reward function that divides the instantaneous profits from trading $r_{i,t}$, as it is computed by (2.36), to the returns of the *rolling intrinsic* $V_d^{\pi^{RI}}$ for the considered day d according to:

$$r'_{i,t} = \frac{r_{i,t}}{V_d^{\pi^{RI}}} \tag{3.2}$$

The goal of this approach is to learn a value function that maximizes the improvement in terms of revenues with respect to the *rolling intrinsic*. In this way, the value functions do not depend on the rewards collected on each day, but on how much improvement can be achieved, by taking a particular action at each state, with respect to the *rolling intrinsic*. The proposed reward scaling in performed only during the learning process of the policy. During evaluation in the test set this scaling does not occur.

3.3 Case study

The impact of the proposed changes in the state space and the reward function is evaluated in this section. Firstly, the new set of parameters and the exogenous information used for the optimization of the CID market participation of a PHES operator are described. Secondly, we present a new anticipative strategy that is used in addition to the *rolling intrinsic* benchmark for comparison purposes. Finally, performance results of the obtained policy are presented and discussed.

3.3.1 Parameters specification

The proposed methodology is applied for an instance of a PHES unit participating in the German CID market with the following characteristics:

- $\overline{SoC_i} = 400$ MWh,
- $\underline{SoC_i} = 0$ MWh,

- $SoC_i^{init} = SoC_i^{term} = \left(\overline{SoC_i} - \underline{SoC_i}\right)/2$,
- $\overline{C_i} = \overline{G_i} = 65$ MW,
- $\underline{C_i} = \underline{G_i} = 0$ MW,
- $\eta = 90\%$.

The discrete trading horizon has been selected in this case to be the full day, i.e. $T = \{16:00,...,00:00,...,22:30\}$, and the trading time interval is selected to be $\Delta t = 15$ min. Thus, the trading process takes $K = 124$ steps until termination. However, in this case study we use the dataset containing all 24 hourly products of the day, $X = \{H_1,..,H_{24}\}$. We select this dataset instead of the quarter-hourly products, due to the higher liquidity observed in the former. Due to the existing liquidity in the hourly-products dataset, we also increased the size of the storage unit, as compared to the PHES considered in Section 2.5. Consequently, the delivery timeline is $\bar{T} = \{00:00,...,23:00\}$, with $\tau_{init} = 00:00$ and $\tau_{term} = 22:30$ and the delivery time interval is $\Delta\tau = 1$ hour. Each product can be traded until 30 minutes before the physical delivery of electricity begins (e.g. $t_{close}(H_1) = 23:30$). For the construction of the training/test sets, we consider the first half of 2015 (i.e. 2015/01/01-2015/06/31) as train set and the second half of 2015 (i.e. 2015/07/01-2015/12/31) as test set. The pseudo-state $z'_{i,t} = (s''_{i,0}, a'_{i,0}, r_{i,0}, ..., a'_{i,t-1}, r_{i,t-1}, s''_{i,t}) \in Z'_i$ is composed of the entire history of observations and actions up to time-step t, as described in Section 3.1.1.

3.3.2 Exogenous variable

The exogenous variable $w_{i,t}^{exog}$ represents any relevant information available to agent i about the system. In this case study, we assumed that the variable $w_{i,t}^{exog}$ contains only the time features: i) the month and ii) whether the traded day is a weekday or weekend.

3.3.3 Anticipative benchmark strategy

In addition to the *rolling intrinsic* benchmark, we define an anticipative strategy which we call look-ahead policy π^{LA}. This policy cannot be implemented in practice because it relies on future information that a storage operator would not have in its possession in real-time. However, it can provide a good measure on how well the fitted Q iteration policy can anticipate future rewards. According to this policy, at each decision step t the agent can fast-forward a number of ψ steps into the future and compute the potential profits $\hat{r}_{i,t+\psi}$. If the future profits $\hat{r}_{i,t+\psi}$ are higher than the current potential profits $\hat{r}_{i,t}$, the agent selects to “Idle". On the opposite case the agent selects to “Trade". The look-ahead policy π^{LA} can be summarized by the following rule:

$$a'_{i,t} \sim \pi^{LA}(\hat{r}_{i,t+\psi}, \hat{r}_{i,t}) = \begin{cases} \text{“}Idle\text{”}, & \text{if } \hat{r}_{i,t+\psi} \geq \hat{r}_{i,t}, \\ \text{“}Trade\text{”}, & \text{otherwise .} \end{cases} \tag{3.3}$$

In the presented case study, we use the value $\psi = 1$. We use the profitability ratio denoted by r^{LA} to evaluate and compare the performance of the look-ahead policy with respect to the *rolling intrinsic* in a similar way to the one presented in Section 2.5.4. These profitability ratio r^{LA} is defined as the percentage difference between the returns collected by the look-ahead policy $V^{\pi^{LA}}$ and the ones collected by the *rolling intrinsic* $V^{\pi^{RI}}$, according to:

$$r^{LA} = \frac{V^{\pi^{LA}} - V^{\pi^{RI}}}{V^{\pi^{RI}}} \cdot 100\%. \tag{3.4}$$

3.3.4 Results

In this section, we present the results obtained, similarly to Section 2.5.5. Tables 3.1 and 3.2, contain descriptive statistics regarding the outcomes obtained in the train and the test set respectively by the compared policies (i.e. π^{FQ}, π^{RI} and π^{LA}). We can observe that π^{FQ} yields significant improvements with respect to the π^{RI} in both the training and the test set. In particular, the π^{FQ} was found to achieve an average increase of 18% on the returns collected in the train set in comparison to the π^{RI}. Additionally, in the test set, the π^{FQ} has managed to obtain an improvement of 19.6% with respect to the π^{RI}, similarly to the train set. We conjecture that the new modifications proposed in this chapter enable the fitted Q iteration algorithm to generalize better over unseen data. Moreover, the anticipative strategy π^{LA} with one step look-ahead results in 38.4% and 42.9% improvements in the train and test set respectively. We can observe that, while π^{FQ} is able to outperform the π^{RI}, it does not yet closely approach the performance of an anticipative policy.

The distribution of training and test set samples according to the obtained profitability ratio is presented in Figure 3.3. It can be observed that both the train and test set distributions have similar shapes. The vast majority of the samples are non-negative implying that following the π^{FQ} as a trading strategy in the CID market entails low risk.

The smoothened evolution of the expected return of the fitted Q iteration policy $V^{\pi^{FQ}}$ in the train set as function of the training episodes (number of trajectories collected) is presented in Figure 3.4. We observe that the π^{FQ} progressively learns how to outperform the π^{RI} and,

TABLE 3.1: Descriptive statistics of the returns obtained on the days of the training set for policies π^{FQ}, π^{RI} and π^{LA}.

	π^{FQ} **returns (€)**	π^{RI} **returns (€)**	**r (%)**	π^{LA} **returns (€)**	r^{LA} **(%)**
mean	3455.0	3020.7	18.0	4069.7	38.4
min	682.55	635.5	−14.8	730.2	−1.2
25%	2011.3	1699.6	7.5	2305.3	22.1
50%	3052.6	2602.2	15.6	3496.3	31.4
75%	4204.0	3497.9	25.2	4712.9	49.0
max	16448.5	1064.6	170.2	19600.0	168.0
sum	625,370.6	546,759.0	–	736,621.9	–

TABLE 3.2: Descriptive statistics of the returns obtained on the days of the test set for policies π^{FQ}, π^{RI} and π^{LA}.

	π^{FQ} **returns (€)**	π^{RI} **returns (€)**	**r (%)**	π^{LA} **returns (€)**	r^{LA} **(%)**
mean	3855.0	3365.4	19.6	4634.6	42.9
min	473.5	343.6	−18.5	668.8	−17.4
25%	2380.6	1978.0	7.2	2684.0	22.2
50%	3426.5	2931.3	15.4	4079.2	38.8
75%	4865.4	4297.9	28.3	5711.3	55.7
max	13963.1	13003.5	110.3	16208.3	203.8
sum	709,328.0	619,246.2	–	852,772.1	–

after a number of episodes, it stabilizes to a value higher than $V^{\pi^{RI}}$. Additionally, we can observe that the learned policy achieves similar results to the anticipative policy π^{LA}. The progressive evaluation of the fitted Q policy in the test set is illustrated in Figure 3.5. In addition to that, the π^{FQ} presents a very similar behaviour during test set evaluation, which suggests that indeed the proposed modifications have led to good generalization capabilities. The shaded area in both graphs represents the variance obtained between the ten different runs.

3.3.5 Policy analysis

In this section, we visualize the effect of the learned policy for each decision step in the trading horizon. In particular, for each step t we compute the percentage of days in which the policy suggests that the operator should “Trade". The results are presented in Figure 3.6. As we can observe, at the beginning of the trading horizon there is much lower probability to select the “Trade" action as compared to later in the day. Additionally, the spikes of high “Trade" probability correspond to gate closures during the trading horizon. This suggests that it is better to “Trade" in moments of high trading activity as there are high chances of capturing larger price spreads (profits).

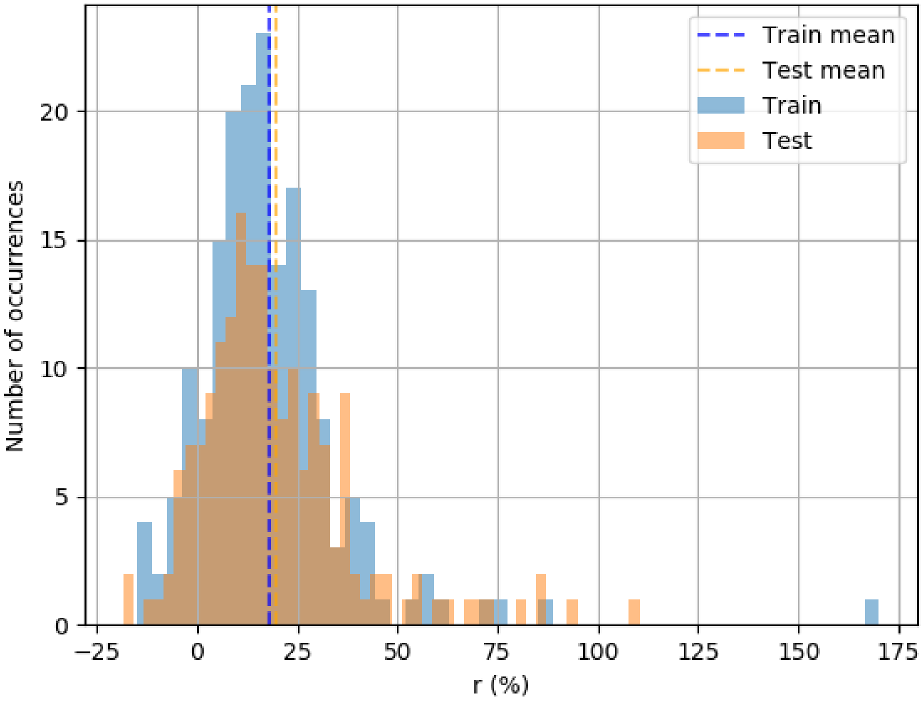


FIGURE 3.3: Profitability ratio.

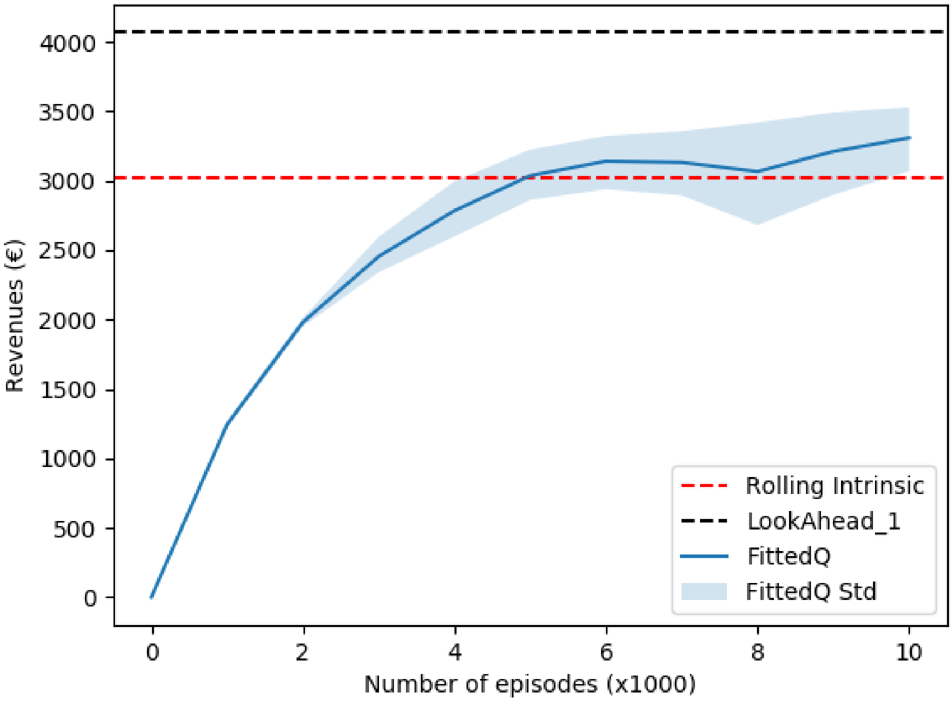


FIGURE 3.4: Progressive evaluation in the train set.

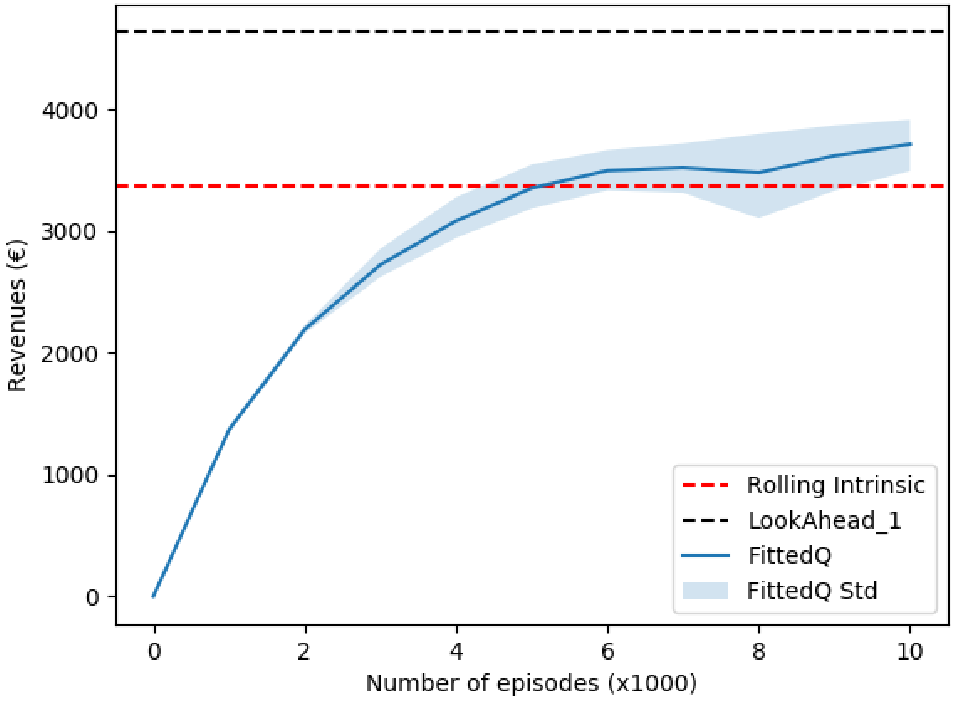


FIGURE 3.5: Progressive evaluation in the test set.

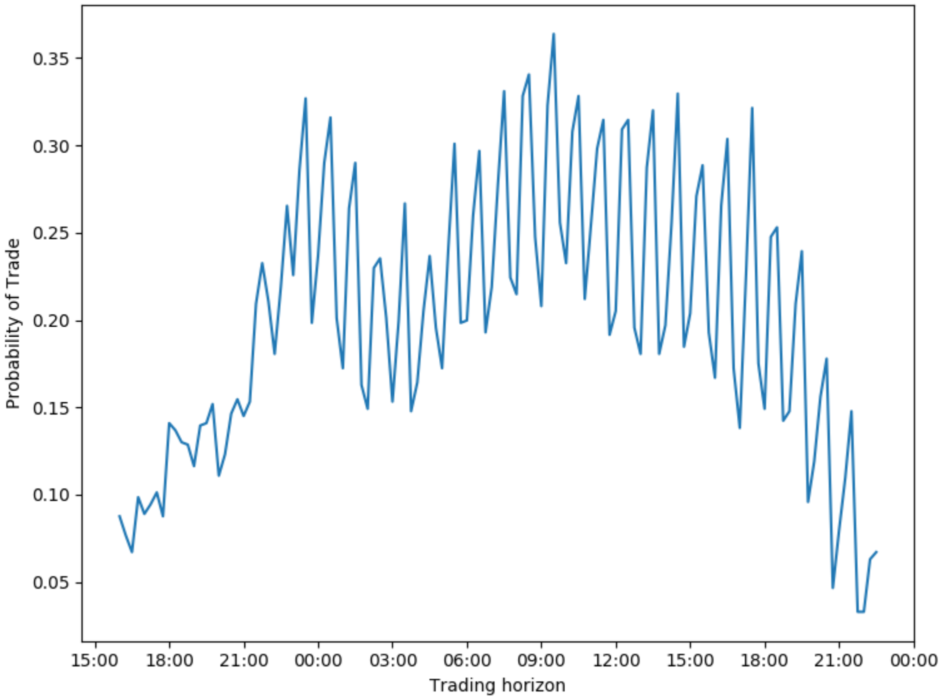


FIGURE 3.6: Probability of trading for each decision step according to π^{FQ}.

3.4 Conclusions

In this chapter, we propose a set of modifications to the original problem of the participation of a storage device operator in the CID market. We first propose a compact representation for the state in which the potential profits are directly computed instead of being inferred from raw market data and the agent's market position. Then, we proceed proceed by scaling the potential profits for each distinct day with its corresponding day-ahead returns. This leads to a state representation that can be observed during both train and test sets. Additionally, we propose a reward function that is defined as the ratio of the profits observed and the profits coming from the *rolling intrinsic* benchmark. This allows for a better representation of the value functions. The outcomes of these modifications are presented in a new case study and are compared against an anticipative policy we call the look-ahead policy. The results show that the proposed method yields significant improvements of approximately 19% on average, with respect to the *rolling intrinsic* benchmark. In addition, we can see that the proposed modifications allow for better generalization of the fitted Q method in out-of-sample data. Finally, the results illustrate that the obtained policy is a low-risk policy that is able to outperform on average the state of the art for the industrial *rolling intrinsic* benchmark strategy.

Chapter 4

Lifelong Control of Off-grid Microgrid with Model Based Reinforcement Learning

In this chapter, we address the energy arbitrage problem from the perspective of an off-grid microgrid operator in the context of rural electrification. In particular, we deal with the lifelong control problem of an isolated microgrid. The main challenges for an effective control policy stem from the various changes that take place over time. For the design of an effective control policy, we propose a novel model-based reinforcement learning algorithm that is able to address the different changes that are encountered over the lifetime of the microgrid. The algorithm demonstrates generalisation properties, transfer capabilities and better robustness in case of fast-changing system dynamics. The proposed algorithm is compared against two benchmarks, namely a rule-based and an MPC controller. The results show that the trained agent is able to outperform both benchmarks in the lifelong setting where the system dynamics are changing over time.

4.1 Introduction

Microgrids are small electrical networks composed of flexible consumption, distributed power generation (renewable and/or conventional) and storage devices. The operation of a microgrid is optimized in order to satisfy the demand while ensuring maximum reliability and power quality and to maximize the renewable energy harvested locally while minimizing the total system cost.

Centralized microgrid control is usually decomposed in four tasks: i) estimating the parameters of the microgrid devices (for instance the charge efficiency of a battery storage device as a function of the state of charge and temperature, or the actual capacity of a battery after a number of cycles), ii) forecasting the consumption and the renewable production, iii) operational planning to anticipate weather effects and human activities, and iv) real-time control to adapt planned decisions to the current situation. These tasks are preformed sequentially during the lifetime of a microgrid in order to achieve near optimal operation and to maximize the benefits arising from distributed generation.

After the initial parameters' estimation step, it is important for the efficient microgrid operation to incorporate in the decision making process all the sources of uncertainty. To this end, forecasting techniques are deployed for the stochastic production and consumption. There is a variety of forecasting techniques in the literature ranging from fundamental models of consumption and renewable energy production [109] to statistical models using measured data [27].

Subsequently, the outputs of the forecasting models in combination with the system parameters are used to compute the optimal control actions that need to be taken. The optimization of the control actions can be performed using the simulation model of the microgrid. Model predictive control (MPC), a feedback control law meant to compensate for the realization of uncertainty, is often used for achieving economic efficiency in microgrid operation management [32]. Probabilistic forecasting models attempt not only to provide the best point forecast but instead target the distribution of the uncertainty. The output of these models can be used to solve stochastic variants of MPC [36]. Depending on the reliability concerns related to the micorgrid use case, robust MPC can provide more secure ways of dealing with uncertainty [110]. Given the data availability, the two preceding tasks, namely forecasting and optimization, can be merged into one task and a control action can be derived directly from the data observed.

In this chapter, we present an open-source reinforcement framework for the modeling of an off-grid microgrid for rural electrification. Moreover, we formulate the control problem of an isolated microgrid as a Markov Decision Process (MDP). Due to the high-dimensional continuous action space we define a set of discrete meta-actions in a similar way to previous work [111].

The main challenge for the lifelong control of an off-grid microgrid arises from the uncertainty of the future renewable production and consumption. A critical issue in microgrid operation is that often-times the policy learned during training on a dataset does not perform well on unseen data. Additionally, the degradation or damage of the various components such as the storage devices or the photovoltaic panels cause the previously learned policies to become sub-optimal over time.

To address these challenges we propose a novel model-based reinforcement learning algorithm. In particular, this class of algorithms integrates planning and learning of an optimal control. They do so by firstly estimating a model of the environment from samples collected by interactions with the environment and by subsequently using this model to generate synthetic trajectories that accelerate the learning process of the control policy. The motivation for learning the dynamics of the environment stems from the enhanced exploration that yields generalization and transferability properties that are highly desired in the context where changes are introduced in the environment.

In particular, the proposed algorithm is an instance of DYNA [112], where the model is trained using distributional losses and the policy is optimized using the Proximal Policy Optimization (PPO) algorithm. A comprehensive description of the algorithms mentioned can be found in Sections 4.4.2 and 4.4.3. The values of the policy are updated based on the expectation computed over a set of states sampled from a model. This model is trained online using samples from the real environment. We illustrate that this algorithm allows for much better estimation of the values accounting for the uncertainties and yields enhanced exploration. Additionally, we show that the enhanced exploration gained using the model to sample states allows for better generalization to unseen data. Moreover, we show that the knowledge of a previously trained policy can be efficiently transferred when training on a new set of data. Finally, we demonstrate the ability of the controller to adapt to sudden changes such as damage of the equipment without explicit knowledge of the event.

A key advantage of the model-based approach is that it can help cope with rare events, in case those rare events have been experienced at least once before in the past. Since DYNA builds a model of the dynamics and simulates transitions using the learned model, it can repeat rare events many times in simulation and update the policy accordingly. The disadvantage is that the non-stationary time series of consumption and renewable production are difficult to predict, and that quantile regression may cause approximation errors.

To evaluate the performance of the obtained policy, we compare it with two benchmarks: i) a rule-based control that takes decisions in a myopic manner based only on current information; and ii) an optimization-based controller in which look-ahead is applied to forecast consumption and RES generation.

This chapter is organized as follows. Section 4.2 elaborates on state-of-the-art methods used for data-driven microgrid operation and control. In Section 4.3, the system dynamics of the microgrid are detailed. Section 4.4 provides the theoretical background used for the developed framework and the algorithm proposed. In Section 4.5, we formulate the lifelong control problem of an off-grid microgrid as an MDP. Section 4.6 presents the model-based algorithm used to solve the lifelong microgrid control problem. The proposed algorithm is compared against the two benchmark strategies presented in Section 4.7. Section 4.8 describes the case study and results obtained. Finally, Section 4.9 concludes the main findings and provides avenues for future research.

4.2 Related Work

A wide range of reinforcement learning techniques have been applied in the literature for optimizing energy systems operations. A vast share of these techniques include model-free reinforcement learning algorithms, where a controller is trained based solely on interactions with the physical environment and without using prior knowledge regarding system dynamics. For instance, the problem of controlling a storage device connected to the main grid in order to maximize its returns and to facilitate RES integration is proposed in [113]. A bias correction procedure of the classical Q-learning algorithm [114] was proposed and the results show a much faster and more stable converge than the vanilla version of the algorithm. The efficient storage control aiming at jointly mazimizing the usage of the battery during high electricity demand and the RES utilization in a grid connected microgrid setting is proposed in [115]. Predictions about the wind generation serve as input to a reinforcement learning controller that is responsible for the battery scheduling. In particular, the optimal control policy is computed using the Q-learning method. In both settings the state and action spaces are discretized in order to reduce the computational complexity and the results show increased utilization of renewable energy sources (RES) production. However, this reduction of complexity inherently puts a limit to the performance improvement margins.

Alternatively, the energy management of a grid-connected microgrid can be performed in a distributed way. Each entity of the microgrid that owns equipment (e.g. PV panels, storage etc.) acts as an autonomous agent who can learn the best response policy to increase their expected rewards in a microgrid market though interaction with other agents [116]. Reinforcement learning is used as a method to develop optimal strategies for energy management and the authors show that the proposed game converges to the Nash equilibrium. A similar multi-agent setting is adopted in [117] where the authors propose a fuzzy Q-Learning method to solve the control problem in a decentralized manner. Each component (i.e. storage, load, PV panels) act as independent learners and a common information state is shared between them in order to coordinate their behavior. Results show increased reliability and enhanced guarantees of energy supply by adopting this decentralized coordinated control approach. However, a centralized approach is more relevant in the context of rural electrification and generally leads to more tractable problems. In this chapter, we consider a central entity that owns an energy management system (EMS) and is making the control decisions over the controllable components.

Recent advancements in the field of Deep Learning (DL) have enabled the use of powerful function approximation techniques for representing value functions and policies that can support (as input/output) continuous and high dimensional state/action spaces. For instance, Q-learning combined with Artificial Neural Networks has been proposed for the optimal operation and maintenance of power grid [118], which is inherently a very large and complex problem. The proposed framework outperforms expert-based solutions to grid operation. Leveraging these techniques, researchers have also proposed a Deep Q-learning approach for the control of seasonal storage in an isolated microgrid [119]. In this framework, a specific deep learning structure is presented in order to extract information from the past RES production and consumption as well as the available forecasts. Despite the highly dimensional continuous state space, the authors obtain a control policy that is able to utilize the long-term storage in a meaningful way. However, in this approach it is assumed that the dynamics of the system are linear and that forecasts of the variable resources are available.

As it is shown, model-free reinforcement learning methods are able to tackle quite well the energy management problem in contexts (environments) where the dynamics remain unchanged. However, in this chapter we consider an environment in which changes (gradual and abrupt) are expected to occur during the operational horizon of the microgrid. To tackle this problem, we resort to model-based reinforcement learning, where a model of the system dynamics is learnt by interaction with the environment and then used for taking control actions. Model-based methods have shown better performance in terms of enhanced exploration [120], accelerated learning [121] and have proven to be suitable in real world non-stationary environments [120], [122].

In this direction, in [123], the energy scheduling in a residential microgrid is performed by adapting *MuZero* [124], a state of the art model-based reinforcement learning algorithm, to the problem. It incorporates the learning of a complex model that is then used for performing Monte-Carlo tree search (MCTS) for the selection of the optimal action. However, the MCTS algorithm relies heavily on the accuracy of the learnt model for simulating several steps in the future. In the context of a changing environment, this approach could lead to sub-optimalities driven by the increasing inaccuracy of the learnt model as the prediction length increases. Additionally, the search for the next action is performed online given some computational budget which might raise reliability issues. In the approach presented in this chapter, the training of the policy is assisted by a model of the system dynamics and it is performed off-line. The trained policy is then used in real time for dispatching the microgrid components.

The use of model-based reinforcement learning with Power Pinch Analysis (PoPA) has proven to be an effective way to coordinate several storage technologies with complementary features in order to enhance the reliability of intermittent renewable energy sources (RES) [125]. In particular, the authors propose an adaptive version of PoPA where DYNA-Q [126] is used to account for the variability introduced by RES and the demand. A physical model is used together with the RES predictions to simulate the future evolution of the system. Corrective actions are taken in order to avoid the violation of upper and lower limits set. In this chapter, we do not consider a physical model of the system. Instead, we attempt to learn an approximation (neural network) of the system dynamics based on simulated experience. In this way, any changes that might occur in the environment can be learnt through demonstration.

The contributions of this chapter are the following:

- We present an open-source reinforcement learning framework for the lifelong modeling of an off-grid microgrid for rural electrification. Any changes (gradual/abrupt) that may occur over the lifetime of the microgrid are incorporated in this framework.
- We propose a novel model-based reinforcement learning algorithm, where the model of the system is trained using quantile regression and the PPO algorithm is used for training the policy.
- The proposed algorithm is evaluated by its ability to perform well in changing conditions. To this end, we demonstrate that the algorithm is able to outperform the benchmark control algorithms in i) a setting where the electricity consumption is changing progressively and ii) a setting where a sudden failure of the storage device occurs.

4.3 Microgrid Description

In this section, we provide a detailed description of the system considered (Figure 4.1). The considered microgrid is composed of PV panels, a diesel generator, a battery and aims at providing energy to variable loads. The Energy Management System (EMS) is responsible for the interaction and the scheduling of the controllable components. As depicted in Figure

4.1, the EMS receives as inputs information about the production from the pv panels and the load consumption. Subsequently, the role of the EMS is to decide on whether to activate the diesel generator and/or to discharge/charge the battery. In the following we provide a detailed description of the components considered. Additionally, an off-grid microgrid designed for rural electrification is inherently characterized by changes occurring in different time-scales. We provide a formal description of the different types of changes and we motivate the need for a lifelong control that has the ability to adapt to these changes.

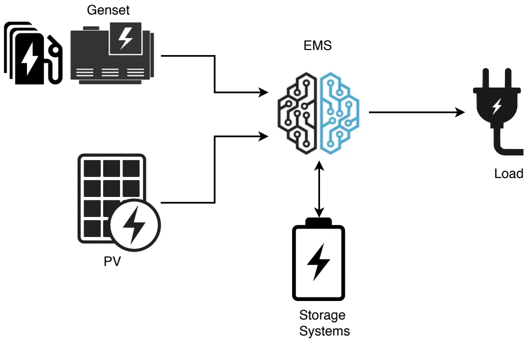


FIGURE 4.1: Schematic of the considered microgrid.

4.3.1 Components

An off-grid microgrid is composed of the following components:

Consumption

The consumption of the isolated microgrid C is considered to be non-flexible, meaning that there is a high cost associated to the energy non-served. The consumption C_t at each time-step t of the simulation is assumed to be a stochastic variable that is sampled from distribution P_t^C, given the h previous realizations, according to:

$$C_t \sim P_t^C(C_{t-1}, ..., C_{t-h}). \tag{4.1}$$

In this chapter, it is represented by real data gathered from an off-grid microgrid. The distribution P_t^C is indexed in time in order to indicate that changes occur in the aggregate consumption over the life-time of the microgrid. For instance, a change in the consumption profile can be caused by the fact that more users are progressively connected to the micro-grid.

Storage model

The modeling of the storage system can become quite complex and highly-nonlinear depending on the degree of accuracy required by each specific application. In this chapter, we use a linear “tank” model for the simulation of the battery since we assume that the simulation time-step size Δt is large enough (1 hour). The dynamics of a battery are given by:

$$SoC_{t+1} = SoC_t + \Delta t \cdot (\eta^{\text{ch}} P_t^{\text{ch}} - \frac{P_t^{\text{dis}}}{\eta^{\text{dis}}}), \tag{4.2}$$

where SoC_t denotes the state of charge at each time step t, P^{ch} and P^{dis} correspond to the charging and discharging power, respectively and η^{ch}, η^{dis} represent the charging and discharging efficiencies of the storage system. The charging (P^{ch}) and discharging (P^{dis}) power of the battery are assumed to be limited by a maximum charging rate $\overline{P}$ and discharging rate $\underline{P}$, respectively. Accounting for the storage system degradation, we consider that the maximum capacity $\overline{S}$ of the storage system as well as the charging and discharging efficiencies (η^{ch}, η^{dis}) are decreasing as a linear function of the number of cycles n_t that are performed at each time-step t. We have, $\forall t \in T$,

$$SoC_t, P_t^{\text{ch}}, P_t^{\text{dis}} \geq 0 \tag{4.3}$$

$$P_t^{\text{ch}} \leq \overline{P} \tag{4.4}$$

$$P_t^{\text{dis}} \leq \underline{P}, \tag{4.5}$$

$$SoC_t \leq \overline{S}, \tag{4.6}$$

$$\overline{S} = s(n_t). \tag{4.7}$$

Steerable generator model

Steerable generation is considered any type of conventional fossil-fuel-based generation that can be dispatched at any time-step t. When a generator is activated, it is assumed to operate at the output level P_t^{gen} that is ranging between the minimum stable generation $\underline{P^{\text{gen}}}$ and the maximum capacity $\overline{P^{\text{gen}}}$ such that:

$$\underline{P^{\text{gen}}} \leq P_t^{\text{gen}} \leq \overline{P^{\text{gen}}}. \tag{4.8}$$

The fuel consumption F_t related to the operation of the generator at time t is a function of the power output P_t^{gen} with parameters F_1, F_2 given by the manufacturer.

$$F_t = \begin{cases} F_1 + F_2 \cdot P_t^{\text{gen}} & \text{,if } P_t^{\text{gen}} > 0, \\ 0 & \text{,otherwise.} \end{cases} . \tag{4.9}$$

The fuel cost c_t^{fuel} accounting for the fuel price π^{steer} is then given by:

$$c_t^{\text{fuel}} = F_t \cdot \pi^{\text{fuel}}. \tag{4.10}$$

Non-steerable generators model

The level of non-steerable generation from renewable resources such as wind or solar is denoted by P^{res}. Similar to the non-flexible load case it is assumed that P_t^{res} at time-step t is sampled from a probability distribution $P_t^{P^{\text{res}}}$, given the h previous realizations, according to:

$$P_t^{\text{res}} \sim P_t^{P^{\text{res}}}(P_{t-1}^{\text{res}}, ..., P_{t-h}^{\text{res}}). \tag{4.11}$$

In this chapter, the renewable generation is represented by real data gathered from an off-grid microgrid. Similar to the case of the non-flexible load, the distribution $P_t^{P^{\text{res}}}$ is indexed by time t to indicate that changes in the renewable production might occur over time. These changes are mostly related to the progressive degradation of the equipment (solar panels).

Power balance

At each time-step t in the simulation horizon we compute the power balance between the injections and the off-takes. The residual power resulting from the mismatch between production and consumption is curtailed P_t^{curt} if its positive and shed P_t^{shed} if it is negative. We can formally define the power balance as:

$$\begin{aligned} & P_t^{\text{res}} + P_t^{\text{gen}} + P_t^{\text{dis}} + P_t^{\text{shed}} \\ & = P_t^{\text{ch}} + P_t^{\text{curt}} + C_t, \end{aligned} \tag{4.12}$$

with $P_t^{\text{curt}}, P_t^{\text{shed}} \geq 0$. The costs arising from the curtailment of generation or the shedding of non-flexible loads are given by:

$$c_t^{\text{curt}} = P_t^{\text{curt}} \cdot \pi^{\text{curt}} \tag{4.13}$$

$$c_t^{\text{shed}} = P_t^{\text{shed}} \cdot \pi^{\text{shed}} \tag{4.14}$$

4.3.2 Characterizing changes in the environment

Oftentimes in real-life applications the concept of interest depends on some underlying context that is not fully observable. Changes in this underlying concept might induce more or less radical changes in the concept of interest, which is formally known as concept drift [127]. For instance, in the off-grid microgrid under study the connection of new users and their habits have strong influence on distribution P_t^C. However, it is not possible to know exactly and to quantify the effect on the consumption a priori.

In this chapter, we deal with the following two distinct set of changes: 1) gradual changes that affect the non-controllable dynamics; and 2) sudden changes that affect the deterministic dynamics. As described in Section 4.5, one can decouple the two components of the state space. Gradual changes occurs in the stochastic component of the state space (4.29) while sudden changes occurs in the deterministic system dynamics (4.28).

Gradual changes

These are cases in which a slow concept drift occurs. The extent of the drift is bounded so that any learner can follow these changes successfully. A formal bound on the maximal rate of drift that is acceptable by a batch-based learner is given by Kuh, Petsche, and Rivest [128]. In this chapter, we assume that changes related to the consumption and renewable production profiles as well as degradation of the equipment (storage) belong to this category.

Sudden or abrupt changes

In our setting, sudden or abrupt changes are adversarial changes that affect the system dynamics, and for which the learner needs to find the best response. Robust MDPs [129] describe optimal control under such changes and recent work [130] shows that incorporating learning in such contexts can deliver policies as good as the minimax policy. Gajane, Ortner, and Auer [131] also propose an algorithm for detecting abrupt changes in MDPs. In the concept of an off-grid micro-grid this type of change would typically occur during equipment failure. In the case study presented in Section 4.8.6, we consider a sudden failure of the storage system. This event leads to a sudden change in the optimal control policy where the generator becomes the main source of power when the RES are not producing sufficiently. Another example of an abrupt change could be the sudden connection of a large industrial consumer to the microgrid. This would have a significant and direct impact to the control policy as well.

4.4 Reinforcement Learning Background

In this section, we provide the theoretical background used for the developed framework and the proposed methodology. We first introduce the Markov Decision Process that is the main framework on which we rely on for modeling the decision making process of an off-grid microgrid operator. We proceed by describing Dyna and Proximal Policy Optimization (PPO), which are the foundations of the proposed novel algorithm.

4.4.1 Markov Decision Process

We consider an infinite horizon discounted Markov Decision Process (MDP), defined by the tuple $\langle S, A, r, \{P_t\}_t, \gamma \rangle$ where S is the state space, A the action space, $r : S \times A \to \mathbb{R}$ is the Markovian cost function, $P_t : S \times A \to \Delta(S)$, $t \geq 0$, is the transition kernel at time t and $\gamma \in (0,1)$ is the discount factor. Here, $\Delta(S)$ is the probability simplex on S, i.e. the set of all probability distributions over S. At each time step t, the agent observes state $s_t \in S$, takes an action $a_t \in A$, obtains reward r_t with expected value $\mathbb{E}[r_t] = r(s_t, a_t)$, and transitions to a new state $s_{t+1} \sim P_t(\cdot|s_t, a_t)$. We refer to (s_t, a_t, r_t, s_{t+1}) as a *transition*. Note that the transition kernels may not be stationary.

Let π denote a stochastic policy $\pi : S \to \Delta(A)$ and $\eta(\pi)$ its expected discounted cumulative reward under some initial distribution $d_0 \in \Delta(S)$ over states:

$$\eta(\pi) = E_{s \sim d_0}[V^\pi(s)], \tag{4.15}$$

where $\tau = \{(s_t, a_t, r_t)\}_{t \geq 0}$ is a trajectory, $p(\tau)$ is the probability distribution over trajectories,

$$p(\tau) = d_0(s_0) \prod_{t=0}^{\infty} P_t(s_{t+1}|s_t, a_t)\pi(a_t|s_t), \tag{4.16}$$

Algorithm 4 DYNA

```
**Inputs**: MDP $M$, integers $T$, $B$, $N$
initialize policy $\pi_\theta$, model $M_\psi$
**for** $t = 0$ **to** $T - 1$ **do**
    $s \sim d_0$
    $a \sim \pi_\theta(\cdot|s)$
    $s', r \sim M(s,a)$
    $\pi_\theta = \text{UPDATEPOLICY}(s,a,r,s')$
    $M_\psi = \text{UPDATEMODEL}(s,a,r,s')$
    **if** $t \geq B$ **then**
        **for** $n = 0$ **to** $N - 1$ **do**
            $s \sim d_0$
            $a \sim \pi_\theta(\cdot|s)$
            $\hat{s}', \hat{r} \sim M_\psi(s,a)$
            $\pi_\theta = \text{UPDATEPOLICY}(s,a,\hat{r},\hat{s}')$
        **end for**
    **end if**
**end for**
```

and the value function V^π is defined for each state $s \in S$ as

$$V^\pi(s) = E_{p(\tau)}\left[\sum_{t=0}^{\infty} \gamma^t r_t(s_t,a_t) \middle| s_0 = s\right]. \tag{4.17}$$

The goal of the agent is to find a policy that maximizes the expected cumulative reward $\eta(\pi)$:

$$\eta^* = \max_\pi \eta(\pi), \tag{4.18}$$

$$\pi^* = \arg\max_\pi \eta(\pi). \tag{4.19}$$

4.4.2 Dyna

DYNA [112] is a model-based reinforcement learning architecture that aims to integrate learning and planning. It does so by performing online estimation of the transition kernel and reward function. Let $M_\psi = \langle P_\psi, r_\psi \rangle$ be a parametric model learned during training. Note that we estimate a single transition kernel P_ψ even though the true kernel may not be stationary.

Algorithm 4 outlines the DYNA algorithm in the parametric setting. For every transition (s,a,r,s') sampled from the environment M, we update the policy π_θ and parametric model M_ψ via update functions described in Algorithms 5 and 6. We remark that the policy update typically relies on a value function V_ϕ. Additionally, the value function V_ϕ and the components of the parametric model M_ψ are updated by minimizing a loss function.

Algorithm 5 UPDATEPOLICY

```
Input: transition $(s,a,r,s')$
$V_\phi = \arg\min_{V_\varphi} L^V(V_\varphi)$
$\pi_\theta = \arg\max_{\pi_\varphi} \eta(\pi_\varphi)$
```

Algorithm 6 UPDATEMODEL

```
Input: transition $(s,a,r,s')$
$P_\psi = \arg\min_{P_\varphi} L^P(P_\varphi)$
$r_\psi = \arg\min_{r_\varphi} L^r(r_\varphi)$
```

After the update step, we use the learned model to perform N updates of the policy π_θ, in the same way as one would using the true environment. At every step, we sample a state $s \sim d_0$, apply action $a \sim \pi_\theta(\cdot|s)$ and query the parametric model $\hat{s}', \hat{r} \sim M_\psi(s,a)$.

Note that there are two main differences during the planning phase. First, the transition $(s,a,\hat{r},\hat{s}')$ comes from the parametric model, and second, there is no structure in the sampling process, therefore in such an update the agent can experience any possible one step transition, even ones that are hard to gather under the current policy.

4.4.3 Proximal Policy Optimization

The Proximal Policy Optimization (PPO) algorithm [132] belongs to the family of policy gradient methods and can be used with both discrete and continuous action spaces. In the vanilla actor-critic method [133], a stochastic policy π_θ with parameters θ is optimized towards the following regularized objective:

$$\eta(\pi_\theta) = \mathbb{E}_{p(\tau)} \left\{ \frac{\pi_\theta(a_t|s_t)}{\pi_o(a_t|s_t)} \hat{A}_\phi(s_t, a_t) \right\} - \frac{1}{\beta} D(\pi_\theta || \pi_o), \tag{4.20}$$

where π_o is the old policy, $\hat{A}_\phi(s_t,a_t)$ is an estimator of the advantage function, D is a regularizer in the form of a Bregman divergence and β is a learning rate.

Since equation (4.20) is hard to optimize directly, the policy is repeatedly updated using stochastic gradient descent. Concretely, a gradient step is used to update of the parameters θ as

$$\theta_{new} = \theta + \alpha \nabla \hat{\eta}(\pi_\theta), \tag{4.21}$$

where α is a step size and the regularized objective for the individual transition (s,a,r,s') is estimated as

$$\hat{\eta}(\pi_\theta) = \frac{\pi_\theta(a|s)}{\pi_o(a|s)}\hat{A}_\phi(s,a) - \frac{1}{\beta}\sum_{a'}\pi_\theta(a'|s)\log\frac{\pi_\theta(a'|s)}{\pi_o(a'|s)}. \tag{4.22}$$

An unbiased estimator of the advantage function is given by

$$\hat{A}_\phi(s,a) = r + \gamma\hat{V}_\phi(s') - \hat{V}_\phi(s), \tag{4.23}$$

where the estimated value function $\hat{V}_\phi$ is obtained by minimizing the following loss:

$$L^V(\hat{V}_\phi) = \frac{1}{2}\mathbb{E}_{p(\tau)}[\hat{A}_{\phi_{old}}(s_t,a_t)^2]. \tag{4.24}$$

In practice, rather than performing updates for individual transitions, the algorithm performs multiple epochs of mini-batch updates of stochastic gradient descent of both the policy and value function. Both the policy and the value function are updated during the UPDATEPOLICY step in Algorithm 4. The way in which these updates are performed is presented in Algorithm 5.

4.4.4 Quantile Regression

The problem of estimating a model $M_\psi = \langle P_\psi, r_\psi\rangle$ is commonly cast as supervised learning, in which the components of M_ψ are computed by minimizing loss functions. One of the contributions of our proposed algorithm is to use distributional losses to estimate M_ψ in the parametric setting.

Distributional losses introduced by Bellemare, Dabney, and Munos [134] and expanded by Dabney, Rowland, Bellemare, *et al.* [135] achieve state of the art performance in several reinforcement learning benchmarks. Imani and White [136] discuss the importance of distributional losses for regression problems, arguing that such losses have locally stable gradients which improves generalization. Here we concisely describe the loss function that we use in our setting. For a more detailed description the reader can consult Dabney, Rowland, Bellemare, *et al.* [135].

Our goal is to learn the distribution of some random variable $z \sim F(z)$. To do so, it is known that the value of the quantile function $F_z^{-1}(\tau)$ is the minimizer of the quantile regression loss. This quantile regression loss acts as an asymmetric squared loss in an interval $(-k,k)$ around zero and reverts to a standard quantile loss outside this interval. The Quantile

Huber loss is defined as:

$$\rho_\tau(u) = |\tau - \delta_{\{u \leq 0\}}| L(u), \tag{4.25}$$

where $L(u)$ is given by

$$L(u) = \begin{cases} \frac{1}{2}u^2, & \text{if} |u| \leq k, \\ k(|u| - \frac{1}{2}k), & \text{otherwise.} \end{cases} \tag{4.26}$$

In Section 4.6 we show how to adapt this loss to learn the estimated transition kernel P_ψ and reward r_ψ.

4.5 Problem Statement

The operation of the system described in Section 4.3 can be modelled as a Markov Decision process as it is defined in Section 4.4. We consider that at each time-step $t \in T$ the state variable $s_t \in S$ is composed of a deterministic and a stochastic part as $s_t = (\underline{s}_t, \bar{s}_t) \in S$ and contains all the relevant information for the optimization of the system. The deterministic part $\underline{s}_t = (SoC_t) \in \underline{S}$ corresponds to the evolution of the state of charge of the storage device and can be fully determined by equations (4.2)-(4.7). The stochastic variable $\bar{s}_t$ represents the variable renewable production and consumption as $\bar{s}_t = \left((C_t, ..., C_{t-h}), (P_t^{\text{res}}, ..., P_{t-h}^{\text{res}}) \right) \in \bar{S}$ as defined in equations (4.1) and (4.11).

The available control action a_t that can be applied at each time-step t is defined as:

$$a_t = \left(P_t^{\text{ch}}, P_t^{\text{dis}}, P_t^{\text{gen}}\right) \in A, \tag{4.27}$$

and contains the charging/discharging decision for the storage system and the generation level of the steerable generators.

At each time-step t the system performs a transition based on the dynamics described in Section 4.3 according to

$$\underline{s}_{t+1} = f_t\left(s_t, a_t\right), \tag{4.28}$$

$$\bar{s}_{t+1} \sim \bar{P}_t\left(\bar{s}_t\right), \tag{4.29}$$

where f_t is a deterministic function and $\bar{P}_t$ is used to denote the joint probability distribution of the stochastic variables C, P^{res} as defined in equations (4.1) and (4.11). Note that, the transition function f_t is indexed in time to account for the changes (e.g. degradation) that may occur to the equipment. Equations (4.28) and (4.29) can fully determine the transition kernel of the MDP at each time step as $P_t : S \times A \rightarrow \Delta(S)$.

Each transition generates a non-positive reward signal (i.e. cost) r_t, that is composed of the fuel cost c_t^{fuel}, the cost of curtailment of RES generation c_t^{curt} and the cost of shedding of non-flexible loads c_t^{shed}. The reward function $r(s_t, a_t) \in \mathbb{R}$, can be defined as:

$$r_t = r(s_t, a_t) = -(c_t^{\text{fuel}} + c_t^{\text{curt}} + c_t^{\text{shed}}). \tag{4.30}$$

The problem of lifelong control of an off-grid microgrid is equivalent to finding a policy π that maximizes the total expected discounted cumulative reward $\eta(\pi)$ as defined in equations (4.15)-(4.19).

4.5.1 Microgrid Simulator

The described MDP for off-grid microgrid control is available as an open source simulator[1]implemented in OpenAI gym [137]. The simulator contains a detailed modelling of the microgrid components and allows for applying any control strategy. It receives as input the microgrid configuration (components size and parameters, time series representing the exogenous information, and simulation parameters) and simulates the operation for a predefined simulation horizon T.

4.6 Methodology

Real world applications are non-stationary, partially observable and high dimensional. A desirable algorithm should effectively deal with those challenges as well as provide basic safety guarantees [138].

Model-based RL algorithms are appealing for real world applications because they are sample efficient, they explicitly approximate the environment dynamics, and, when combined with powerful function approximation, they can scale to the high dimensional setting [139].

The key issue with model-based RL is learning the model sufficiently well to be useful for policy iteration. In real-world applications, this issue is exacerbated by the requirements of generalisation and sample efficiency. To address those challenges we propose a practical algorithm that builds upon the DYNA algorithm [112], as it is described in Section 4.4.2. We use a variant of PPO [132] to perform policy iteration, and quantile losses to approximate the model dynamics. A description of PPO can be found in Section 4.4.3. We have two quantile

[1] Available at `https://github.com/bcornelusse/microgridRLsimulator`.

losses, one for learning the transition kernel P_ψ and one for the learning the reward function r_ψ:

$$L^P(s) = \mathbb{E}[\sum_{i=1}^{q} \rho_{\tau_i}(s' - P_\psi(s,a))] \tag{4.31}$$

$$L^r(r) = \mathbb{E}[\sum_{i=1}^{q} \rho_{\tau_i}(r - r_\psi(s,a))] \tag{4.32}$$

Model-free updates are performed in PPO by sampling a partial trajectory and directly maximing (5). We use two seperate networks for the value V_ϕ and the policy π_θ, and we select the advantage estimator as in (4.23). Model-based updates are one-step simulated transitions. As noted in previous work [140], updating simulated states helps to empirically mitigate model error, constraining it to simulated states. Complementary work [141] shows that simulating one-step transitions provides a strong baseline with respect to partial or complete policy rollouts with a learned model, and PPO mantains its monotonic improvement property (Theorem 1, Schulman, Levine, Moritz, *et al.*, 2015).

In practice, in order to deal with the high dimensionality of the state and action space, we represent the model M_ψ as a neural network with shared parameters $\psi \in \Psi$ and two heads P_ψ and r_ψ. Each head outputs a vector of size $d \times q$ where d is the output dimension and q is the number of quantiles considered. The policy π_θ and the value function V_ϕ are represented using two different networks. Contrary to previous claims [132], sharing parameters does not improve learning in our experiments. Finally, we introduce a hyperparameter $B \in \mathbb{N}$ that is the minimum amount of optimisation performed with the model prior to allowing model-based updates. Empirically we found this to reduce the detrimental effect of model error on policy updates. We refer to the presented algorithm as D-DYNA.

4.7 Benchmark strategies

In this section, we introduce two control strategies used for comparison purposes. First, a myopic rule-based strategy is used to provide a lower bound of the total rewards in the period considered. The second strategy corresponds to a model-predictive control (MPC) with N-step look-ahead. We use MPC to compute an upper bound on the total reward that can be obtained by any policy, by considering a sufficiently large number of look-ahead steps and providing it with *perfect knowledge* about the future realization of the stochastic variables. In a realistic setting, no algorithm has access to perfect knowledge about the future, hence this upper bound is not attainable in practice.

4.7.1 Rule-based controller

The rule-based controller is a simple myopic controller that implements a set of decision rules to determine the control actions that need to be taken at each time-step t. It requires only data regarding the present condition of the microgrid. The logic that is implemented is the following:

1. First, the residual generation ΔP_t is computed as the difference between the current total renewable production and non-flexible demand as:

$$\Delta P_t = P_t^{\text{res}} - C_t$$

2. If ΔP_t is positive, the status of the battery is set to charge ("C") and the decision y_t is formed as:

$$y_t = \text{“}C\text{”}$$

3. If ΔP_t is negative, the status of the battery is set to discharge ("D") and the decision y_t is formed as:

$$y_t = \text{“}D\text{”}$$

4. When the decision y_t is made, the residual generation is dispatched over devices as presented in Algorithm 7, and the control action $a_t = \left(P_t^{\text{ch}}, P_t^{\text{dis}}, P_t^{\text{gen}}\right)$ containing to the storage device (P_t^{ch}, P_t^{dis}) and the generator (P_t^{gen}) is determined.

A detailed description of the rule-based controller is presented in the Appendix (Algorithm 7).

4.7.2 Model-predictive controller

The model-predictive controller (MPC) is used to define the control actions (P_t^{dis}, P_t^{ch}, P_t^{gen}) at each decision time-step t by solving an optimization problem with a look-ahead period of N steps. This controller receives as input the microgrid parameters and a forecast of the stochastic variables for the N following time steps. The forecast for the consumption, is denoted by $\widehat{C_t}$, and is given by $\widehat{C_t} = \left(\widehat{C_{t+k}}, \forall k \in \{0,...,N-1\}\right)$. Accordingly, the forecast of the renewable production is denoted by $\widehat{P_t^{\text{res}}}$, and is given by $\widehat{P_t^{\text{res}}} = \left(\widehat{P_{t+k}^{\text{res}}}, \forall k \in \{0,...,N-1\}\right)$.

The optimization problem that is solved at each time-step is presented in Algorithm 8. The objective function aims at minimizing the curtailment, load shedding and fuel cost subject to the operational constraints defined by a mixed-integer linear model of the microgrid. The integer variables n_{t+k} are used to ensure that when the generator is activated the generation level lies between its minimum stable generation level and its capacity.

The output of this controller is an open loop policy $a_t^N = ((P_{t+k}^{\mathrm{dis}}, P_{t+k}^{\mathrm{ch}}, P_{t+k}^{\mathrm{gen}}), \forall k \in \{0, ..., N-1\})$ for the subsequent N time-steps. At each control time-step t, only the first action from the sequence of computed actions is applied to the system $a_t = \left(P_t^{\mathrm{ch}}, P_t^{\mathrm{dis}}, P_t^{\mathrm{gen}}\right)$. The quality of this controller depends on the number of look-ahead steps N, the accuracy of the forecasts and the quality of the model considered.

4.8 Case study

In this section, we evaluate the performance of the proposed algorithm on a real-life off-grid microgrid. First, we define the microgrid specifications and the parameters used during our simulations. Subsequently, we define a set of Meta-Actions that simplify the policy search. The particular instances of the benchmarks (i.e. rule-based controller, MPC etc.) that are used in this case study to compare the performance of our algorithm are described. We proceed by defining three distinct experiments in order to evaluate the capability of the proposed algorithm to i) generalize in out-of-sample data, ii) to be robust in the event of sudden changes and to iii) transfer knowledge from one training session to another in order to accelerate learning.

4.8.1 System configuration

The evaluation of the developed methodology is performed using empirical data measured by the off-grid micro-grid system of the village “El Espino" (-19.188, -63.560), in Bolivia, installed in September 2015 and composed of photovoltaic (PV) panels, battery storage and a diesel generator. The system serves a community of 128 households, a hospital and a school, as well as the public lighting service. A comprehensive description of the system and of the data is available in previous work [143].

Aggregated electrical load data is available as an indirect measurement, i.e. as the sum of direct measurements retrieved from the PV arrays, the diesel generator and the battery by means of smart meters. In this chapter, we use the available measured data for the consumption and the PV production for the period January 2016 to July 2017, presented in Figures 4.2 and 4.3. We can observe the seasonality effects to both load and PV production as well as the constant increase of the load due to the gradual connection of households to the microgrid.

In an off grid-microgrid setting, the optimal size of the components depends heavily on the control policy applied. When the capacity of the installed components is large, a myopic

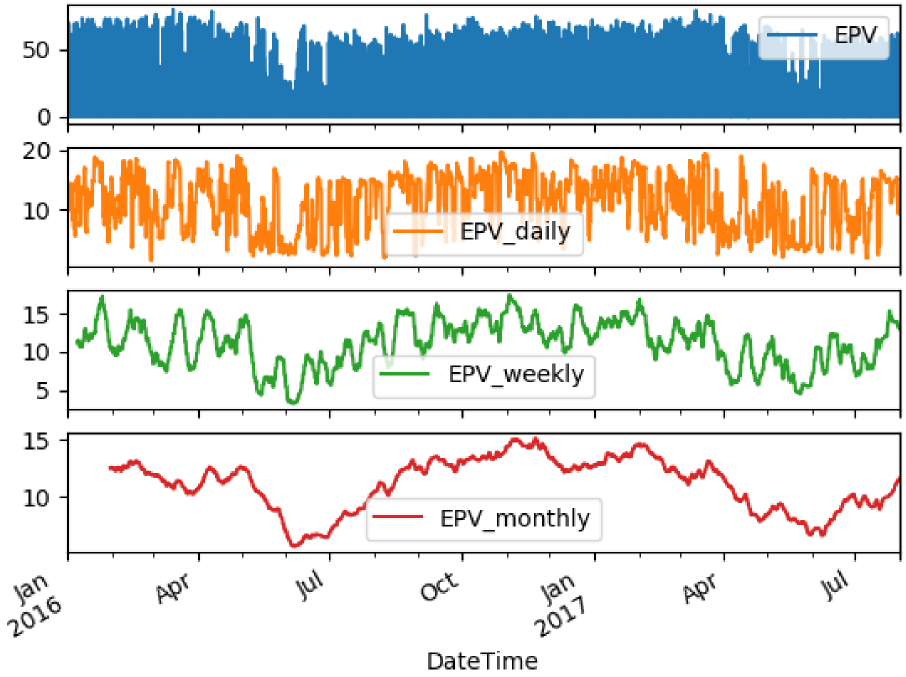


FIGURE 4.2: PV production and its daily, weekly and monthly rolling average.

policy can be as good as a look-ahead policy. On the other hand, a good policy that is able to anticipate changes and to act accordingly allows for the reduction of the components size and subsequently the installation cost.

The search for a good policy becomes much more relevant when the size of the components is constraining the operation of the microgrid. Therefore, in this chapter we consider a reduced installation for which the applied control policy really impacts the cost of operation for the microgrid. The parameters used for the microgrid configuration in this chapter are given in Table 4.1.

Additionally, the effect of different policies depend on the seasonality of solar irradiation and demand being observed. For instance, during the summer period (November through March in the case of Bolivia) there are high solar irradiation levels that can be used to charge fully the battery most of the days. During this period a myopic rule-based strategy has very similar outcomes with a look-ahead strategy. However, during the winter period (April to October), when solar irradiation is limited and the battery may not be fully charged, a more elaborate strategy is necessary in order to guarantee low-cost security of supply in the microgrid.

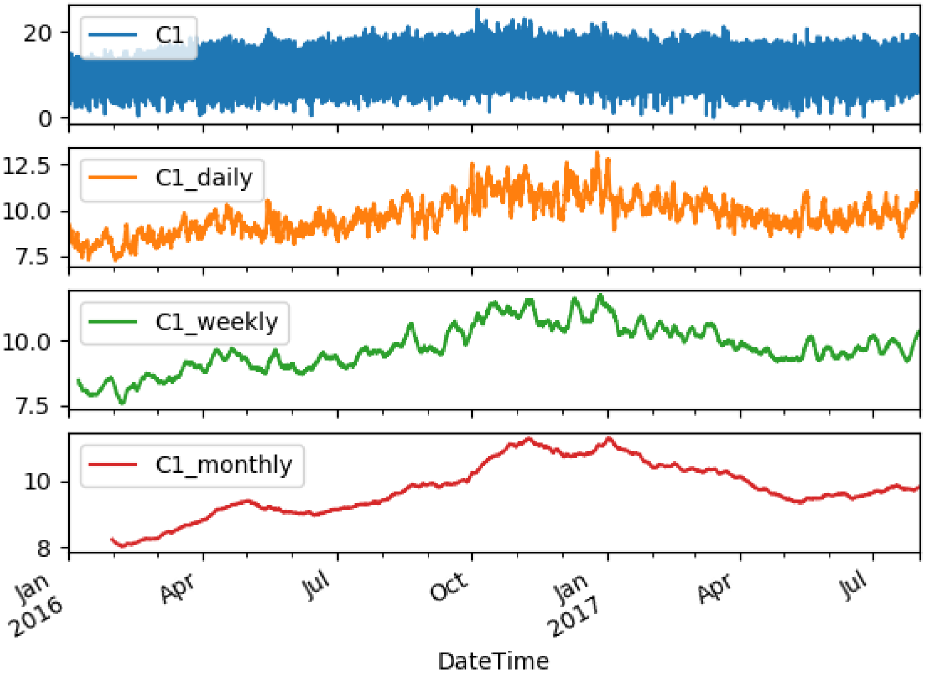


FIGURE 4.3: Electrical load and its daily, weekly and monthly rolling average.

4.8.2 Partial Observability

As described in Section 4.3, the process under consideration is non-stationary. The stochastic component of the transition kernel is known to be non-Markovian and the optimal decision requires knowledge of the next l time steps. In supervised learning problems this issue is commonly addressed by state-based networks [144]. However, in this chapter we take a similar approach as the one considered in the optimization-based controller (Section 4.7.2). We use the model M_ψ as a 1-step forecaster. After a number of warm-up iterations B, we use the model to produce a forecast of the state in the l following time-steps. This forecast is used to augment the actual state which is used to train the controller. A critical assumption of our approach is that the gradual changes to the system dynamics are sufficiently smooth for a single model M_ψ to successfully track these changes.

4.8.3 Action Space and Meta-Actions

Due to the continuous and high-dimensional nature of the state and the action spaces of the problem, reinforcement learning methods cannot be applied in their exact form. However,

TABLE 4.1: Input parameters.

$\overline{S}$	120	kWh
$\overline{P}, \underline{P}$	100	kW
$\eta^{\text{ch}}, \eta^{\text{dis}}$	75^2 %	
π^{fuel}	1	€/kWh
π^{curt}	1.5	€/kWh
π^{shed}	10	€/kWh
Δ_t	1	h
$\overline{P^{\text{res}}}$	120	kW
$\overline{P^{\text{gen}}}$	9	kW
$\underline{P^{\text{gen}}}$	0	kW

recent developments in the field of reinforcement learning have made possible the design of approximate optimal policies using function approximation.

In our setting, function approximation alone does not suffice. The action space visited by the optimal controller from Section 4.7.2 is constrained to a subspace of $\mathbb{R}^2$. Therefore, we elaborate on the design of a small and discrete set of actions A' that maps to the original action space A. This step is necessary for the use of policy-based algorithms, as the maximization problem defined in (4.19) is hard to solve.

The meta-action a'_t for each decision step t is defined as:

$$a'_t \in A' = \{\text{“}C\text{”}, \text{“}D\text{”}, \text{“}G\text{”}\}.$$

Meta-action “C” indicates the action to charge energy in the battery, when there is excessive renewable production ($\Delta P_t > 0$). With meta-action “D” we select to prioritize the discharge of the battery for covering the deficit of energy ($\Delta P_t < 0$) in the microgrid. In case the battery does not suffice for covering this deficit, the generator will be activated. Alternatively, meta-action “G” is used to prioritize the generator for supplying the deficit of energy and the battery will be discharged only in the case that the maximum generating limit ($\overline{P^{\text{gen}}}$) is reached.

In particular, at each decision step t we provide as inputs to the dispatch Algorithm 7, the observed residual generation ΔP_t and the meta-action $y_t = a'_t$. The residual generation ΔP_t is computed after the realization of the stochastic variables (P_t^{res}, C_t) as $\Delta P_t = P_t^{\text{res}} - C_t$.

Defining the action space in this way allows the use of the dispatch rule defined in Algorithm 7 to obtain the control actions $a_t = (P_t^{\text{ch}}, P_t^{\text{dis}}, P_t^{\text{gen}})$. The discrete action space A' simplifies the problem but restricts the class of possible policies, which sometimes harms the performance of the reinforcement learning methods. We leave the problem of directly optimizing continuous actions as future work.

4.8.4 Comparison with the benchmarks

The algorithm is compared against the two benchmarks described in Section 4.3 and the simple model-free version of PPO [132]. An optimization controller with perfect knowledge and 1 period of look-ahead (“MPC-1") is considered in order to obtain a fair comparison to the proposed algorithm. An optimization controller with 24 periods of look-ahead and perfect knowledge (“MPC-24") is used to provide an upper bound on the performance of any control strategy. Additionally, a myopic rule-based controller, indicated in the results as “heuristic", is used to provide a lower bound. We use PPO to denote the baseline algorithm which only performs model-free updates and D-DYNA to denote our method.

The label “training step" on the x-axis refers to the number of times a new set of trajectories has been used for computing one or multiple gradient steps. For a fair comparison we fix the total number of samples available for the agent and compute the number of samples per training update accounting for the number of gradient step and the number of planning steps. Finally, results are averaged for 10 random seeds in order to account for stochasticity.

4.8.5 Generalization

One of the challenges of real world applications is the occurrence of changes in the transition dynamics. As described in Section 4.3, the dynamics of the microgrid are composed of a deterministic part and a stochastic part. The stochastic part is not controllable and therefore constitutes a source of progressive change.

In this section, we evaluate the ability of model-based algorithm to adapt to gradual changes that occur in the state space. An algorithm that generalizes over unseen data distributions can provide a good initialization for fine-tuning the new controller. The following protocol was used for training and evaluation of the proposed algorithms. We split the original dataset in a training set and a test set: the training set ranges from January 2016 to December 2016, while the test set ranges from January 2017 to July 2017.

Figure 4.4 presents the cumulative returns (costs) collected in the test set by the compared algorithms as a function of the training progress (i.e. training steps). In other words, at each training step performed by the RL algorithms we perform an evaluation of all considered algorithms in the test set. We observe that the reinforcement learning methods approximately yield a 25% cost reduction in comparison to the rule-based controller and the model-based method is comparable to the upper bound set by MPC-24. As illustrated in Figure 4.4,

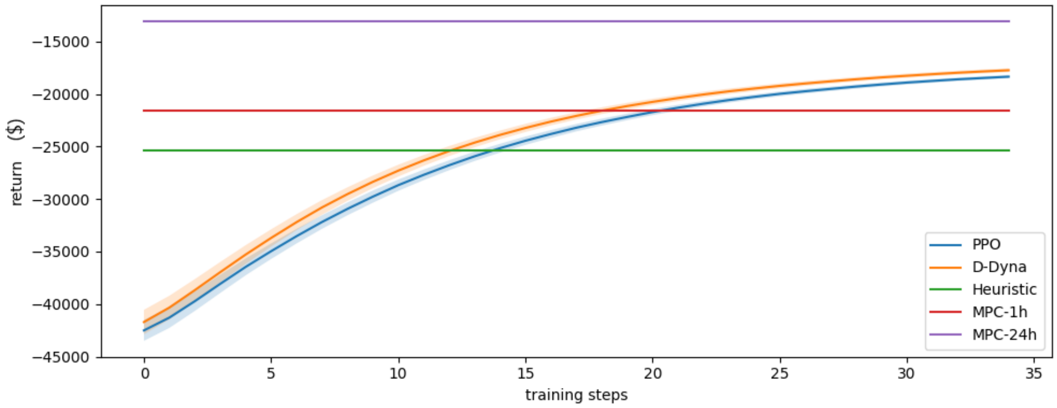


FIGURE 4.4: Cumulative returns (cost) on the test set as a function of the training progress of the RL algorithms.

introducing a model benefits generalization and both the baseline and the proposed algorithm are able to outperform the heuristic. We conjecture that using artificially generated states accelerates the learning process and provides a wider coverage of the state (exploration) and action space manifold, resulting in better generalization properties.

Additionally, we observe that the proposed model-based method is able to outperform the “MPC-1" benchmark. We can argue that the obtained policy manages to resemble a look-ahead policy that takes optimal actions with respect to several steps ahead. This outcome is rather valuable because by using such a policy we can reduce the investment cost for equipment (e.g. battery capacity or diesel generators), without jeopardising the security of supply in the microgrid. Additionally, the cost reduction achieved by the proposed algorithm mainly implies a reduction in the use of the diesel generator and the higher utilization of RES. This effect subsequently results in an overall reduction of CO_2 emissions and promotes sustainable energy utilization in the context of rural electrification.

4.8.6 Robustness

In this section, we evaluate the performance of the proposed model-based algorithm in sudden changes as defined in Section 4.3. An example of such a change is the abrupt failure of the storage system, where the battery capacity is suddenly unavailable.

We simulate this change in the following way. Let x_t be the random discrete variable taking at each time-step the value 0 if the battery has failed and the value 1 if the battery is still operational. We assume that x_t follows a Bernoulli probability distribution where $Pr(x_t = 1) = p_t$, with p_t following a linear decay in time and $p_0 = 0.99$. If the battery fails, then the maximum storage capacity is considered to be reduced to zero ($\overline{S} = 0$ kWh). After a

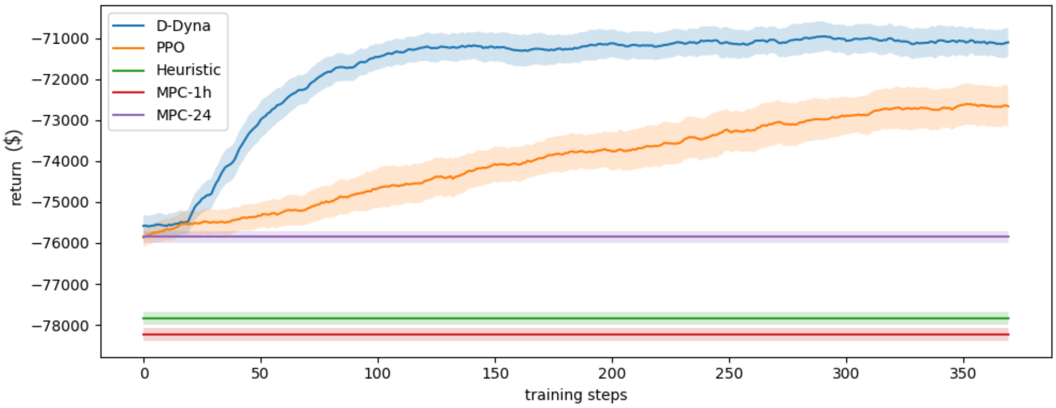


FIGURE 4.5: Cumulative returns (costs) when the battery is excluded as a function of the training progress of the RL algorithms.

failure, it is assumed that the battery equipment is fixed and the storage capacity is restored to its initial value in a period of $N = 370$ hours. Failures can occur during both training and testing.

For this experiment, we have increased the size of the generator at a level that covers the entire demand. In this way, we want to evaluate the capability of the proposed model-based method to switch from a regime where, the battery is mainly used when it is available, to only using the generator in the event of a battery failure.

Under this scenario, we evaluate the benchmark controllers, the model-free method as well as the model-based method. As we can see in Figure 4.5, all benchmarks perform poorly while the proposed algorithm is able to quickly adapt to the new drastically changing dynamics. The poor performance of the benchmark controllers is justified by the fact that there is no special equipment for the detection of the failure. The superiority of the proposed model-based algorithm stems from its ability to detect the change since the model has been exposed to similar incidents during training.

4.8.7 Transfer

In Reinforcement Learning, transfer learning is the ability of speeding up learning on new MDPs by reusing past experiences between similar MDPs. For real world applications, it would be desirable to obtain an algorithm that has the ability to learn off-line and adapt as the task changes.

A natural instance of such feature is to consider each month as a separate MDP and evaluate the ability to transfer knowledge across months. Note that each month has a different distribution of the stochastic component of the transition kernel.

We set up the following experimental protocol. We use January 2016 to pre-train the algorithms. Then we initiate the training process for February and August 2016 using the pre-trained model. Intuitively transfer should be easier if the data distributions are close in time, and harder otherwise.

The results of the described protocol are presented in Figures 4.6 and 4.7. As we can see, transferring the model and the control allows for better performance than learning from scratch. As illustrated, the model-based method can substantially speed up the learning process. The proposed method is shown to slightly outperform the “heuristic" as well as the “MPC-1h" benchmarks. However, in August the results are much better in that the model-based method is approaching the performance of the “MPC-24h" policy, while the rest of the benchmarks are falling behind.

As discussed in Section 4.8.1, the effect of different policies depends on the period of the year. We can observe that the results in February are substantially different in comparison to August. There is a small discrepancy between the returns from the myopic and the optimization-based controller with perfect knowledge during February. On the other hand, during August the two policies show an increased difference in returns.

4.9 Conclusions

In this chapter, a novel model-based reinforcement learning algorithm is proposed for the lifelong control of a microgrid. First, an open-source reinforcement framework for the modeling of an off-grid microgrid for rural electrification is presented. The control problem of an isolated microgrid is casted as a Markov Decision Process (MDP). The proposed algorithm learns a model online using the collected experiences. This model is used to sample states during the evaluation step of the proximal policy optimization (PPO) algorithm.

We compare the proposed algorithm to the standard benchmarks in the literature. Firstly, a rule-based control that takes decisions in a myopic manner based only on current information and secondly an optimization-based controller with look-ahead are considered for comparison purposes.

We evaluate the generalization capabilities of the proposed algorithm by comparing its performance in out-of-sample data to the benchmarks. It is found that the use of the model to create artificial states leads to improved exploration and superior performance compared to the myopic rule-based controller and the MPC with one step look-ahead.

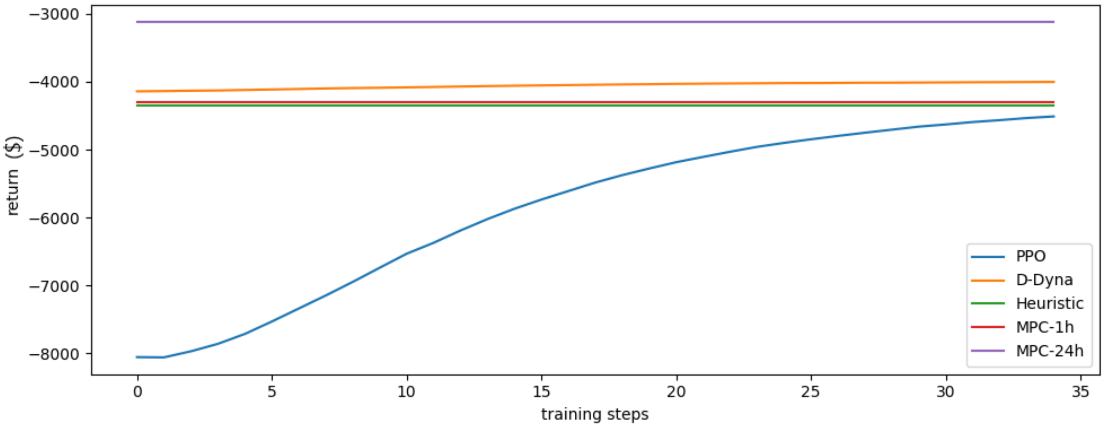


FIGURE 4.6: Cumulative returns (cost) during February as a function of the training progress of the RL algorithms.

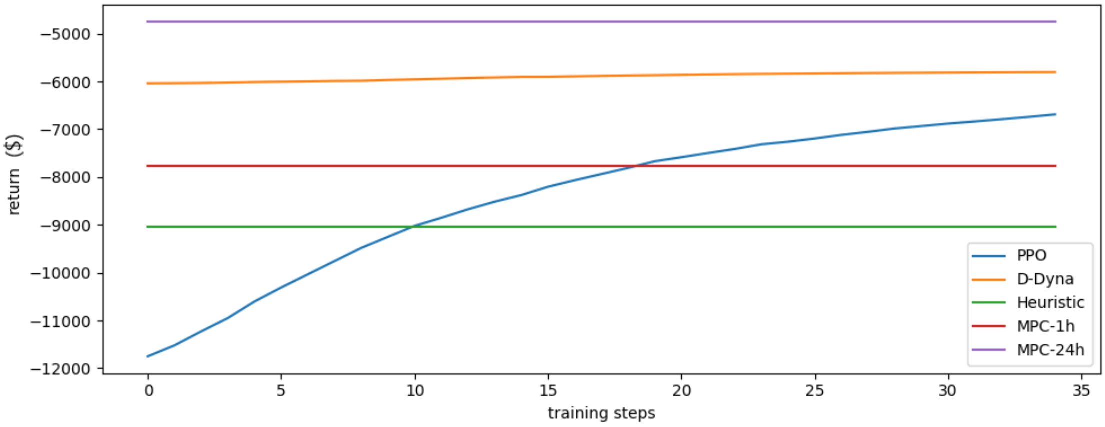


FIGURE 4.7: Cumulative returns (cost) during August as a function of the training progress of the RL algorithms.

We evaluate the robustness of the proposed algorithm when being subject to sudden changes in the transition dynamics, such as equipment failure. The results indicate that the model-based method has the ability to adapt rapidly to severe changes in contrast to the benchmarks that are unable to detect changes and perform poorly to the subjected task.

Finally, we evaluate the ability to transfer knowledge from one training session to the next. The results show large gains in computational time when initiating training on a new dataset with a pre-trained model.

One important conclusion is that the proposed model-based reinforcement learning method is able to adapt to changes, both gradual and abrupt. Overall, the proposed method succeeds in tackling the key challenges encountered in the lifelong control of an off-grid microgrid for rural electrification. Future work should be directed to the design of a low dimensional continuous action space in order to be able to obtain results similar to the optimization-based controller.

In future work, we plan to perform experiments directly with continuous actions. As explained in the chapter, discretizing the actions makes reinforcement learning and exploration much faster, but introduces approximation errors that may account for the slightly worse performance of reinforcement learning in some settings. Since actions are concentrated to restricted areas of the joint action space, we believe that it is necessary to impose constraints on the continuous actions during learning. Additionally, future work could be directed towards incorporating the effect of efficiency improvement on the microgrid components. For instance, improvements in the efficiency of consumer appliances are expected to progressively decrease the average demand profile. On the other hand, improvements in the efficiency of solar PV or in storage systems due to technological improvements are expected to have an impact on the control policy.

4.10 Appendix

4.10.1 Algorithms

Algorithm 7 Power dispatch.

```
**Inputs:** $\Delta P_t$ , $y_t$, $\overline{P}$, $\underline{P}$, $\overline{P^{\text{gen}}}$
**Initialize:**$P_t^{\text{dis}} \leftarrow 0$,$P_t^{\text{ch}} \leftarrow 0$, $P_t^{\text{gen}} \leftarrow 0$
**if** $\Delta P_t \geq 0$ **then**
    **if** $y_t =$ "$C$" **then**
        $P_t^{\text{ch}} = \min(P^{RES}, \overline{P})$
    **end if**
    $\Delta P_t \leftarrow \Delta P_t - P_t^{\text{ch}}$
**else**
    **if** $y_t =$ "$D$" **then**
        $P_t^{\text{dis}} = \min(-P^{RES}, \underline{P})$
        $\Delta P_t \leftarrow \Delta P_t + P_t^{\text{dis}}$
        $P_t^{\text{gen}} = \min(-P^{RES}, \overline{P^{\text{gen}}})$
    **end if**
    **if** $y_t =$ "$G$" **then**
        $P_t^{\text{gen}} = \min(-P^{RES}, \overline{P^{\text{gen}}})$
        $\Delta P_t \leftarrow \Delta P_t + P_t^{\text{gen}}$
        $P_t^{\text{dis}} = \min(-P^{RES}, \underline{P})$
    **end if**
**end if**
**Output:** $a_t$
```

4.10.2 Notation

Sets and indices

- t, decision time step
- k, look-ahead step
- $\mathscr{A}$, action space
- $\mathscr{A}'$, meta-action space
- $\mathscr{S}$, state space

Parameters

- F_1, F_2, fuel consumption parameters
- N, number of look-ahead periods
- $\widehat{C}$, load forecast (kW)
- $\overline{P}$, $\underline{P}$, maximum charge and discharge rate (kW)
- $\overline{P^{\text{res}}}$, non steerable generation (kW)

Algorithm 8 Model-predictive controller.

```
Inputs: $N$, $\pi^{\text{curt}}$ , $\pi^{\text{shed}}$, $\pi^{\text{fuel}}$,$F_1$, $F_2$, $\eta^{\text{ch}}$, $\eta^{\text{dis}}$,
    $\overline{P}$, $\underline{P}$, $\overline{S}$, $\overline{P^{\text{gen}}}$ , $\underline{P^{\text{gen}}}$, $\widehat{C}_t$,$\widehat{P_t^{\text{res}}}$
Solve:
    $\min \sum_{k=0}^{N} \Delta_t \left(c_t^{\text{fuel}} + c_t^{\text{curt}} + c_t^{\text{shed}}\right)$
    $s.t. \widehat{P_{t+k}^{\text{res}}} + P_{t+k}^{\text{gen}} + P_{t+k}^{\text{dis}} + P_{t+k}^{\text{shed}} =$
        $P_{t+k}^{\text{ch}} + P_{t+k}^{\text{curt}} + \widehat{C}_{t+k} \quad ,\forall k \in \{0,...,N-1\}$
        $c_{t+k}^{\text{curt}} = P_{t+k}^{\text{curt}} \cdot \pi^{\text{curt}} \quad ,\forall k \in \{0,...,N-1\}$
        $c_{t+k}^{\text{shed}} = P_{t+k}^{\text{shed}} \cdot \pi^{\text{shed}} \quad ,\forall k \in \{0,...,N-1\}$
        $c_{t+k}^{\text{fuel}} = F_{t+k} \cdot \pi^{\text{fuel}} \quad ,\forall k \in \{0,...,N-1\}$
        $F_{t+k} = F_1 + F_2 \cdot P_{t+k}^{\text{gen}} \quad ,\forall k \in \{0,...,N-1\}$
        $SoC_{t+k+1} = SoC_{t+k} + \Delta t \cdot (\eta^{\text{ch}} P_{t+k}^{\text{ch}} -$
            $\frac{P_{t+k}^{\text{dis}}}{\eta^{\text{dis}}}) \quad ,\forall k \in \{0,...,N-1\}$
        $SoC_{t+k}, P_{t+k}^{\text{ch}}, P_{t+k}^{\text{dis}} \geq 0 \quad ,\forall k \in \{0,...,N-1\}$
        $P_{t+k}^{\text{ch}} \leq \overline{P} \quad ,\forall k \in \{0,...,N-1\}$
        $P_{t+k}^{\text{dis}} \leq \underline{P} \quad ,\forall k \in \{0,...,N-1\}$
        $SoC_{t+k} \leq \overline{S} \quad ,\forall k \in \{0,...,N-1\}$
        $P_{t+k}^{\text{gen}} \leq \overline{P^{\text{gen}}} \cdot n_{t+k} \quad ,\forall k \in \{0,...,N-1\}$
        $P_{t+k}^{\text{gen}} \geq \underline{P^{\text{gen}}} \cdot n_{t+k} \quad ,\forall k \in \{0,...,N-1\}$
        $n_{t+k} \in \{0,1\} \quad ,\forall k \in \{0,...,N-1\}$
Output: $a_t^N$
```

- $\overline{P^{\text{gen}}}$, steerable generator capacity (kW)
- $\underline{P^{\text{gen}}}$, minimum steerable generation (kW)
- $\overline{S}$, $\underline{S}$, maximum and minimum battery capacity (kWh)
- $\widehat{P^{\text{res}}}$, renewable generation forecast (kW)
- Δ_t, simulation and control period duration (h)
- η^{ch}, η^{dis}, charge and discharge efficiency (%)
- π^{curt}, curtailment price (€/kWh)
- π^{fuel}, fuel price (€/kWh)
- π^{shed}, load shedding price (€/kWh)

Variables

- a, control actions vector
- a', meta-actions vector
- C, non-flexible load (kW)
- c^{fuel}, fuel cost (€)
- c^{curt}, curtailment cost (€)
- c^{shed}, lost load cost (€)
- F_t, fuel consumption (l)
- k, binary variable
- n_t, number of cycles of the battery
- P^{ch}, P^{dis}, charging and discharging power (kW)
- P^{shed}, load shed (kW)
- P^{curt}, generation curtailed (kW)
- P^{gen}, generation activated (kW)
- P^{res}, renewable generation (kW)
- P^{ch}, charged energy of battery (kWh)
- P^{dis}, discharged energy of battery (kWh)
- SoC, state of charge of battery (kWh)
- s, control state vector
- $\bar{s}$, stochastic state vector
- $\underline{s}$, deterministic state vector
- y_t, discrete decision about the use of the equipment
- ΔP_t, residual generation level (kW)

Functions

- $P_t^C(\cdot)$, load probability distribution
- $P_t^{P^{\text{res}}}(\cdot)$, renewable generation probability distribution
- $s(\cdot)$, storage capacity as a function of the number of cycles

Chapter 5

Learning optimal environments using projected stochastic gradient ascent

In the previous chapter, we argue that in the context of an off grid-microgrid, the optimal size of the components (i.e. the capacity of pv panels, storage) depends heavily on the control policy that is applied. When the capacity of the installed components is large, a myopic policy can be as good as a look-ahead policy. On the other hand, a good policy that is able to anticipate changes and to act accordingly allows for the reduction of the components size and subsequently the investment cost. Generally speaking, the size of a system and the control that is applied to it are highly interdependent. In this chapter, we propose a methodology for jointly sizing a dynamical system and designing its control law. First, the problem is formalized by considering parametrized reinforcement learning environments and parametrized policies. The objective of the optimization problem is to jointly find a control policy and an environment over the joint hypothesis space of parameters such that the sum of rewards gathered by the policy in this environment is maximal. The optimization problem is then addressed by generalizing the direct policy search algorithms to an algorithm we call Direct Environment Search with (projected stochastic) Gradient Ascent (DESGA). We illustrate the performance of DESGA on two benchmarks. First, we consider a parametrized space of Mass-Spring-Damper (MSD) environments and control policies. Then, we use our algorithm for optimizing the size of the components and the operation of a small-scale autonomous energy system, i.e. a solar off-grid microgrid, composed of photovoltaic panels, batteries. On both benchmarks, we compare the results of the execution of DESGA with a theoretical upper-bound on the expected return. Furthermore, the performance of DESGA is compared to an alternative algorithm. The latter performs a grid discretization of the environment's hypothesis space and applies the REINFORCE algorithm [66] to identify pairs of environments and policies resulting in a high expected return. The choice of this algorithm is also discussed and motivated. On both benchmarks, we show that DESGA and the alternative algorithm result in a set of parameters for which the expected return is nearly equal to its theoretical upper-bound. Nevertheless, the execution of DESGA is much less computationally costly.

5.1 Introduction

Problems where one has to design a system that has to be controlled afterwards are ubiquitous in the field of engineering. Common examples include the combined design and control of a robotic arm for achieving a specific goal [145], [146] or the sizing and the operation of a microgrid to minimize electricity costs [147]. System performance depends on both the system parameters and the method by which it is operated, and the interplay between the two should be properly accounted for when designing them [148].

This type of joint design and control problems can often be cast as multi-step optimization problems under uncertainty [149]. Roughly speaking, in this framework, an agent must take a decision at every step of a discretised time horizon in order to optimize a pre-specified criterion. Information about the underlying system is typically available in the form of a state-space representation, whose transition dynamics may be constrained and/or stochastic. Uncertainty is represented by stochastic processes, the outcomes of which may be conditioned on both states and decisions and usually become known immediately after decisions have been taken at every step of the time horizon. A reward (resp. cost) is associated with each pair of realizations and decisions, and solving the problem essentially consists in selecting a sequence of decisions maximizing (resp. minimizing) some function of the sum of rewards (resp. costs) collected at every step (e.g., its expectation). In our design and control problem, the first stages consist of the decisions regarding the design of the system and the following stages are concerned with its control over its lifetime. A variety of methods have been deployed to tackle such problems, as discussed next.

Firstly, multi-stage stochastic programming, which forms a subset of mathematical programming, has been widely used in the literature [150]. In this context, a mathematical model of the system is assumed to be available, in which the design and operational decisions as well as the system states are represented as optimization variables. Some model parameters are assumed to be uncertain and are represented as realizations of a stochastic process whose probability distribution is assumed to be known [151]. Moreover, the latter is usually assumed to be independent of decision variables, in which case the uncertainty is said to be *exogenous*. Conversely, the uncertainty may be *endogenous*, which implies that decisions have an influence on its probability distribution. In addition to constraints representing the dynamics and control of the system, *non-anticipativity constraints* are added to define the temporal structure of the uncertainty and specify how it is revealed over time. The main computational approach to solving multi-stage stochastic programming problems consists in approximating the uncertainty by a discrete stochastic process exhibiting a tree structure (resulting in a so-called *scenario tree*), and solving all scenarios at once via a large-scale mathematical program [152]. Clearly, the number of scenarios increases with the number of stages and the number of realizations required at each stage to properly approximate the original probability distribution. This can quickly lead to intractable problems and scenario tree reduction techniques are often used in practice [152]. Furthermore, considering nonlinear transition dynamics and control laws usually results in nonconvex optimization problems, which are notoriously difficult to solve to optimality [153]. Taking endogenous uncertainty into account usually involves additional nonconvexities [154], which further complicate matters. Hence, in practice, system design problems are often approximated using two-stage stochastic programs (possibly with recourse) [150], [155], [156]. In this setup, the first stage typically represents the design stage, while the second stage models system operation over its lifetime (or a representative truncated time horizon). This approach therefore reduces to having a star-shaped scenario tree, which limits the ability of these methods to properly represent short-term uncertainty and its impact on system operation. Once a system design has been identified, real-time operation is usually conducted using receding horizon control strategies such as model predictive control (MPC) [157]. In MPC, an optimization model representing short-term system operation is initialised with the current system state and solved online in order to identify a sequence of optimal (open-loop) control actions. A subset of these actions is then applied to the system before recovering the system state, and repeating the procedure. In other words, in such approaches, the original design and control problem is split into two separate sub-problems that are solved virtually independently.

A different approach proposed in the literature consists in specifying a control law *a priori*, selecting the system configuration and simulating system behaviour under this control law. During simulation, the system configuration is typically specified by a model whose parameters remain fixed. In addition, in order to perform these simulations, the uncertainty may be specified via its probability distribution or may be revealed through an oracle, which are sampled or queried online. Different system configurations can be tested in such fashion, and the configuration yielding the most desirable outcome is selected. To this end, derivative-free optimization methods and evolutionary algorithms are typically employed to this end.

Such methods have been applied to the design of electrical microgrids [158]–[160], where a rule-based controller is used and the system parameters are selected to minimize the expected cost over different operational scenarios. In some cases, the pre-specified control law may be defined implicitly by solving an optimization problem online, similarly to traditional MPC. In particular, an application to the design of smart buildings is given in [161]. Compared with applied multi-stage stochastic programming approaches, such methods are capable of better representing the uncertainty and its impact on system operation, since no *a priori* approximate representation of the uncertainty (in the form of a reduced scenario tree) is required in practice. However, the derivative-free strategies used to explore the space of system configurations can be ineffective and time consuming, especially in high dimensional spaces [162]. In addition, the fact that control laws are selected *a priori* may limit the ability of such methods to effectively capture the interplay between system configuration and control, and eventually result in system designs with lower performance. This crucial insight was made clear in [148], where a first attempt to address the issue was made by defining a parametric policy (e.g., in the form of a neural network) whose parameters were then jointly optimized with system parameters. This method was then applied to electrical energy storage system design and control. A genetic algorithm was used for the optimization, which suffers from the same drawbacks as the derivative-free methods discussed above [163] and has therefore commonly been substituted by derivative-based methods in machine learning applications [164].

On the other hand, reinforcement learning (RL) provides effective tools to design complex control policies adaptively while properly accounting for uncertainty, both endogenous and exogenous. In this setup, an active decision-making agent attempts to learn a policy in order to maximize its so-called *value function* through interaction with its environment [165]. During this interaction, the agent gathers experience that is used to improve its performance over time. The goal of the agent is defined by the reward signal collected after each interaction with the environment, and the value function is typically taken as the expected sum of rewards collected over the entire time horizon. In recent years, the subclass of solution methods known as direct policy search techniques have met with considerable success. These techniques essentially parametrize the policy and navigate in the space of candidate policies towards a (locally) optimal one by processing the information contained in trajectories generated throughout the optimization process. Typically, two main classes of direct policy search techniques can be distinguished, namely gradient-free and gradient-based methods. The first class uses derivative-free optimization techniques, e.g. the covariance matrix adaptation (CMA) [166] and the cross-entropy method (CEM) [167], [168]. The latter class of methods moves from one point to the next, in the space of candidate policies, through the reconstruction of a gradient of the objective from information contained in trajectories. Derivative-free methods are known to scale unfavourably with the number of policy parameters and do not perform well on large-scale problems [169]. On the other hand, gradient descent (or ascent) methods have been very successful at learning function approximators for supervised learning tasks with a large number of parameters [164], [170]. Gradient-based direct policy search methods extend these ideas to reinforcement learning and allow for efficient training of complex and powerful policies [171].

In the standard reinforcement learning setup, the environment is fixed and the agent merely seeks to learn an optimal control policy. From a modelling perspective, in order to extend reinforcement learning methods to joint design and control problems, the configuration of the system that an agent seeks to control may be encapsulated in the environment it faces. In this paper, we explore this idea and extend the standard deep RL framework by considering that, in addition to the policy, the environment (transition dynamics and reward signal) can be parametrized. The objective of our approach is to jointly optimize the environment and policy parameters in order to maximize the total expected cumulative rewards received. Our algorithm works as follows. Given an initial set of parameters, we compute the gradient of the expected cumulative rewards and perform a projected gradient ascent step in the space of environment and policy parameters. This procedure is then repeated a fixed number of times. We call this algorithm Direct Environment Search with (projected stochastic) Gradient Ascent (DESGA). Compared with methods previously introduced for solving joint design and control problems, this approach has several key advantages. It accurately represents uncertainty and its impact on system operation, allows for the definition of complex policies, and naturally accounts for the interplay between system configuration and control. Furthermore, it exploits gradients to explore the joint design and control hypothesis space, which have been shown to be very efficient on complex machine learning tasks [164].

The DESGA algorithm can be interpreted as an extension of gradient-based direct policy search techniques and more particularly the REINFORCE algorithm [66]. Our method also shares some similarities with model-based reinforcement learning algorithms [172]. In this sub-field of RL, a parametric model of a physical environment is learned from the trajectories collected from this environment. The models used range from parametrized stochastic processes to neural networks and parametrized dynamical systems. It is then possible to infer a control policy from the learned model. The latter class of methods has been successively applied on diverse problems [173]–[175]. However, in the DESGA algorithm, the environment parameters are learned to maximize the rewards collected by an optimal policy in this environment.

The rest of the paper is organized as follows. In Section 5.2, we present the theoretical background and the problem statement of optimizing over the joint environment and policy parameter space. In Section 5.3, the proposed methodology as well as the algorithmic implementation for direct environment search with gradient ascent (DESGA) are described. The experimental protocol for the evaluation of the proposed algorithm is introduced and the results are demonstrated in Section 5.4. Finally, the conclusions and certain considerations for future work are discussed in Section 5.5.

5.2 Theoretical background and problem statement

In this section, we provide a generic formulation for the optimal control problem of a discrete-time dynamical system with a finite-time optimization horizon. Then, we introduce a parametrization of both the dynamical system and the policy spaces. Subsequently, we formulate the problem of jointly optimizing the vector of parameters of the dynamical system and the policy with the goal to maximize the total expected rewards.

5.2.1 Discrete-time dynamical systems

Let us consider a discrete-time and time-invariant dynamical system defined as follows [40]. Let $T \in \mathbb{N}$ be the optimization horizon referring to the number of decisions to be taken in the control process. The system is defined by a state space $\mathscr{S}$, an action space $\mathscr{A}$, a disturbance space Ξ, a transition function $f : \mathscr{S} \times \mathscr{A} \times \Xi \rightarrow \mathscr{S}$, a bounded reward function $\rho : \mathscr{S} \times \mathscr{A} \times \Xi \rightarrow R \subsetneq \mathbb{R}$ and a conditional probability distribution P_ξ giving the probability $P(\xi_t|s_t, a_t)$ of drawing a disturbance $\xi_t \in \Xi$ when taking an action $a_t \in \mathscr{A}$ while being in a state $s_t \in S$. A probability measure P_0 yields the probability $P_0(s_0)$ of each state $s_0 \in \mathscr{S}$ to be the initial state. At time $t \in \{0, 1, \ldots, T-1\}$, the system moves from state $s_t \in \mathscr{S}$ to state $s_{t+1} \in \mathscr{S}$ under the effect of an action $a_t \in \mathscr{A}$ and a random disturbance $\xi_t \in \Xi$, drawn with probability $P_\xi(\xi_t|s_t, a_t)$, according the transition function f:

$$s_{t+1} = f(s_t, a_t, \xi_t) \ . \tag{5.1}$$

After each transition, a reward signal r_t is collected from the reward function according to $r_t = \rho(s_t, a_t, \xi_t)$ with $|r_t| \leq r_{max}$. The different elements of this optimal control problem are gathered in a tuple $(\mathscr{S}, \mathscr{A}, \Xi, P_0, f, \rho, P_\xi, T)$ referred to as the environment.[1]

We define a closed-loop policy $\pi \in \Pi$ as a function associating a probability distribution with support $\mathscr{A}$ to current state s_t of the system at a decision stage $t = 0, \ldots, T-1$. Applying the policy to the dynamical system consists in sampling an action a_t with probability $\pi(a_t|s_t, t)$ at each time t. A trajectory $\tau = (s_0, a_0, \xi_0, a_1, \xi_1, \ldots a_{T-1}, \xi_{T-1})$ contains the information collected from executing policy π over the horizon T. The cumulative reward $R(\tau)$ over trajectory τ can be computed as:

$$R(\tau) = \sum_{t=0}^{T-1} \rho(s_t, a_t, \xi_t) \,, \tag{5.4}$$

where $s_{t+1} = f(s_t, a_t, \xi_t)$. The expected cumulative reward associated to a policy π, and to a state $s_t \in \mathscr{S}$ at time t, is called the return of the policy and is given by:

$$V^\pi(s_t, t) = \sum_{t'=t}^{T-1} \underset{\substack{a_{t'} \sim \pi(\cdot|s_{t'}, t') \\ \xi_{t'} \sim P_\xi(\cdot|s_{t'}, a_{t'})}}{\mathbb{E}} \{\rho(s_{t'}, a_{t'}, \xi_{t'})\} \,. \tag{5.5}$$

Optimal policies are defined by the *principle of optimality* [40]. This principle states that a policy $\pi^*(\cdot|s_t, t) \in \Pi_{\mathscr{A}}$, where $\Pi_{\mathscr{A}}$ is the set of probability distribution functions with support $\mathscr{A}$, is optimal in a state s_t at a time t if it maximizes the expected reward-to-go from that state at that time. An optimal policy $\pi^* \in \Pi$ is thus such that $\forall s_t \in \mathscr{S}, \forall t = 0, \ldots, T-1$:

$$\pi^* \in \underset{\pi \in \Pi}{\text{argmax}} \{V^\pi(s_t, t)\} \,. \tag{5.6}$$

5.2.2 Problem statement: optimizing over a set of environments

We consider the environment $(\mathscr{S}, \mathscr{A}, \Xi, P_0, f_\psi, \rho_\psi, P_\xi, T)$, as defined in Section 5.2.1, with continuous state space $\mathscr{S} \subsetneq \mathbb{R}^{d_\mathscr{S}}$, action space $\mathscr{A} \subsetneq \mathbb{R}^{d_\mathscr{A}}$, disturbance space $\Xi \subsetneq \mathbb{R}^{d_\Xi}$, distribution P_0 over the initial states and horizon T; where $d_\mathscr{S}, d_\mathscr{A}, d_\Xi \in \mathbb{N}$. The state, action and disturbance spaces are assumed to be compact. The transition and reward functions are

[1] Let us note that from the environment $(\mathscr{S}, \mathscr{A}, \Xi, P_0, f, \rho, P_\xi, T)$, we can define an equivalent Markov Decision Process (MDP) with horizon T, state space $\mathscr{S}$, action space $\mathscr{A}$, initial probability distribution P_0, reward probability distribution r and transition probability distribution p such that:

$$r(r_t|s_t, a_t) = \underset{\xi_t \sim P_\xi(\cdot|s_t, a_t)}{\mathbb{E}} \{\delta_{\rho(s_t, a_t, \xi_t)}(r_t)\} \quad, \forall s_t, \in \mathscr{S}, a_t \in \mathscr{A}, r_t \in R \tag{5.2}$$

$$p(s_{t+1}|s_t, a_t) = \underset{\xi_t \sim P_\xi(\cdot|s_t, a_t)}{\mathbb{E}} \{\delta_{f(s_t, a_t, \xi_t)}(s_{t+1})\} \quad, \forall s_t, s_{t+1} \in \mathscr{S}, a_t \in \mathscr{A} \,, \tag{5.3}$$

where $\delta_y(x)$ is a function returning one if and only if x equals y and zero otherwise.

two parametric functions f_ψ and ρ_ψ, parametrized by the vector ψ defined over the compact $\Psi \subsetneq \mathbb{R}^{d_\Psi}$, with $d_\Psi \in \mathbb{N}$. Both functions are assumed continuously differentiable with respect to their parameters and to the state space for every action in $\mathscr{A}$ and every disturbance in Ξ. Additionally, we consider the parametric function π_θ to be a policy parametrized by the real vector θ in the compact $\Theta \subsetneq \mathbb{R}^{d_\Theta}$, with $d_\Theta \in \mathbb{N}$, and continuously differentiable with respect to its parameters Θ and to its domain $\mathscr{S}$ for every action in $\mathscr{A}$ and for every time t. We want to identify a pair of parameter vectors (ψ, θ) such that the policy π_θ maximizes the expected return, on expectation over the initial states, in the environment $(\mathscr{S}, \mathscr{A}, \Xi, P_0, f_\psi, \rho_\psi, P_\xi, T)$. We thus want to solve the following optimization problem:

$$\psi^*, \theta^* \in \underset{\psi \in \Psi, \theta \in \Theta}{\operatorname{argmax}} \; V(\psi, \theta) \tag{5.7}$$

$$V(\psi, \theta) = \underset{\substack{s_0 \sim P_0(\cdot) \\ a_t \sim \pi_\theta(\cdot|s_t, t) \\ \xi_t \sim P_\xi(\cdot|s_t, a_t)}}{\mathbb{E}} \left\{ \sum_{t=0}^{T-1} r_t \right\} \tag{5.8}$$

$$s_{t+1} = f_\psi(s_t, a_t, \xi_t) \tag{5.9}$$

$$r_t = \rho_\psi(s_t, a_t, \xi_t) \, . \tag{5.10}$$

5.3 Direct environment search with gradient ascent

In this section, we address the problem defined in Section 5.2.2. First, we show in Section 5.3.1 that the expected cumulative reward is differentiable with respect to the parameters of the system and the policy if the different parametric functions and the disturbance probability function are continuously differentiable. In such a context, we derive an analytical expression of the gradient. The results are also extended for discrete action and disturbance spaces. We also derive the expression of an unbiased estimator of the gradient from the differentiation of a loss function built from Monte-Carlo simulations. In Section 5.3.2, we present our Direct Environment Search with (projected stochastic) Gradient Ascent (DESGA) algorithm that uses a projected stochastic gradient ascent for optimizing both the parameters of the environment and the policy.

5.3.1 Gradient for learning optimal environments

In Theorem 1, we first prove the differentiability of the expected cumulative reward with respect to the policy and the environment parameters, assuming the functions composing the environment and the policy are continuously differentiable. We then extend these results in a straightforward way to the case where $\mathscr{A}$ and/or Ξ are discrete in Corollary 1. Corollaries 2 and 3 finally give the expressions of the gradients.

Theorem 1. Let $(\mathscr{S},\mathscr{A},\Xi,P_0,f_\psi,\rho_\psi,P_\xi,T)$ and π_θ be an environment and a policy as defined in Section 5.2.2. Additionally, let the functions f_ψ, ρ_ψ and P_ξ be continuously differentiable over their domain of definition. Let $V(\psi,\theta)$ be the expected cumulative reward of policy π_θ, averaged over the initial states, for all $(\psi,\theta) \in \Psi \times \Theta$, as defined in Eqn. (5.8).

Then, the function V exists, is bounded, and is continuously differentiable in the interior of $\Psi \times \Theta$.

Corollary 1. The function V, as defined in Theorem 1, exists, is bounded, and is continuously differentiable in the interior of $\Psi \times \Theta$ if $\mathscr{A}$ and/or Ξ are discrete.

Corollary 2. The gradient of the function V defined in Eqn. (5.8) with respect to the parameter vector ψ is such that:

$$\nabla_\psi V(\psi,\theta) = \mathop{\mathbb{E}}_{\substack{s_0 \sim P_0(\cdot) \\ a_t \sim \pi_\theta(\cdot|s,t) \\ \xi_t \sim P_\xi(\cdot|s_t,a_t)}} \Big\{ \Big(\sum_{t=0}^{T-1} \big(\nabla_s \log \pi_\theta(a_t|s,t)|_{s=s_t} + \nabla_s \log P_\xi(\xi_t|s,a_t)|_{s=s_t} \big) \cdot \nabla_\psi s_t \Big) \times \Big(\sum_{t=0}^{T-1} r_t \Big) + \Big(\sum_{t=0}^{T-1} \nabla_\psi \rho_\psi(s,a_t,\xi_t)|_{s=s_t} + \nabla_s \rho_\psi(s,a_t,\xi_t)|_{s=s_t} \cdot \nabla_\psi s_t \Big\}, \quad (5.11)$$

where:

$$\nabla_\psi s_t = (\nabla_s f_\psi)(s,a_{t-1},\xi_{t-1})|_{s=s_{t-1}} \cdot \nabla_\psi s_{t-1} + (\nabla_\psi f_\psi)(s,a_{t-1},\xi_{t-1})|_{s=s_{t-1}}\,, \quad (5.12)$$

with $\nabla_\psi s_0 = 0$.

Corollary 3. The gradient of the function V, defined in Eqn. (5.8), with respect to the parameter vector θ is given by:

$$\nabla_\theta V(\psi,\theta) = \mathop{\mathbb{E}}_{\substack{s_0 \sim P_0(\cdot) \\ a_t \sim \pi_\theta(\cdot|s_t,t) \\ \xi_t \sim P_\xi(\cdot|s_t,a_t)}} \{ (\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t,t)) (\sum_{t=0}^{T-1} r_t) \}\,. \quad (5.13)$$

Definition 1. Let $(\mathscr{S},\mathscr{A},\Xi,P_0,f,\rho,P_\xi,T)$ and π be an environment and a policy, respectively, as defined in Section 5.2. We call a history h of the policy in the environment, the sequence:

$$h = (s_0, a_0, \xi_0, r_0, a_1, \xi_1, r_1, \dots a_{T-1}, \xi_{T-1}, r_{T-1})\,, \tag{5.14}$$

where s_0 is an initial state sampled from P_0, and where, at time t, ξ_t is a disturbance sampled from P_ξ, a_t is an action sampled from π, and r_t is the reward observed.

For computing the gradients, our DESGA algorithm will exploit the following theorem that shows that an unbiased estimate of the gradients can be obtained by evaluating the gradients of a loss function computed from a set of histories. Automatic differentiation will later be used for computing these gradients in our simulations.

Theorem 2. Let $(\mathscr{S}, \mathscr{A}, \Xi, P_0, f_\psi, \rho_\psi, P_\xi, T)$ and π_θ be an environment and a policy, respectively, as defined in Section 5.2.2. Let $V(\psi, \theta)$ be the expected cumulative reward of policy π_θ averaged over the initial states, as defined in Eqn. (5.8). Let $\mathscr{D} = \{h^m | m = 0, \dots, M-1\}$ be a set of M histories sampled independently and identically from the policy π_θ in the environment. Let $\mathscr{L}$ be a loss function such that, $\forall(\psi, \theta) \in \Psi \times \Theta$:

$$\mathscr{L}(\psi,\theta) = -\frac{1}{M}\sum_{m=0}^{M-1}\Big(\Big(\sum_{t=0}^{T-1}\log\pi_\theta(a_t^m|s_t^m,t) + \log P_\xi(\xi_t^m|s_t^m,a_t^m)\Big) \times \big((\sum_{t=0}^{T-1} r_t^m) - B\big) + \big(\sum_{t=0}^{T-1}\rho_\psi(s_t^m,a_t^m,\xi_t^m)\big)\Big)\,, \tag{5.15}$$

where B is a constant value called the baseline.

The gradients with respect to ψ and θ of the loss function are unbiased estimators of the gradients of the function V as defined in Eqn. (5.8) with opposite directions, i.e. they are such that:

$$\mathop{\mathbb{E}}_{\substack{s_0 \sim P_0(\cdot)\\ a_t \sim \pi_\theta(\cdot|s_t,t)\\ \xi_t \sim P_\xi(\cdot|s_t,a_t)}} \{\nabla_\psi \mathscr{L}(\psi,\theta)\} = -\nabla_\psi V(\psi,\theta) \tag{5.16}$$

$$\mathop{\mathbb{E}}_{\substack{s_0 \sim P_0(\cdot)\\ a_t \sim \pi_\theta(\cdot|s_t,t)\\ \xi_t \sim P_\xi(\cdot|s_t,a_t)}} \{\nabla_\theta \mathscr{L}(\psi,\theta)\} = -\nabla_\theta V(\psi,\theta)\,. \tag{5.17}$$

Corollary 4. The gradient of the loss function, defined in Eqn. 5.15, with respect to θ corresponds to the opposite of the update direction computed with the REINFORCE algorithm [66] averaged over M simulations.

The proofs for the theorems and corollaries presented in this section are given in Appendix 5.6.

5.3.2 Parameter optimization with projected stochastic gradient ascent

In the previous section, we have developed an analytical expression for the computation of the gradients of the expected cumulative reward with respect to the parameters of the environment and of the policy. In order to allow for the event where these parameters belong to a constrained set, our DESGA algorithm will use the projected gradient ascent method [176].

Gradient ascent is an optimization technique where the optimized variables are updated at each iteration step k, by a fixed-size step that is proportional to the gradient of the objective function with respect to these variables. The size of the update can be controlled by parameter α, called the learning rate. In the problem defined by Eqn. (5.7), we aim to find a parameter vector $x = (\psi, \theta) \in X = \Psi \times \Theta \subsetneq \mathbb{R}^{d_\Psi + d_\Theta}$ that maximizes the expected cumulative reward. Gradient ascent updates the parameter vector x_k at time k as:

$$x_{k+1} \leftarrow x_k + \alpha \cdot \nabla_x V(x_k) \,. \tag{5.18}$$

The new point x_{k+1} computed by simple gradient ascent according to Eqn. (5.18), may not belong to the constraint set X. In projected gradient ascent, we choose the point nearest to x_{k+1}, according to the Euclidean distance, that is located in the set X i.e., the projection of x_{k+1} onto the set X. The projection Π_X of a point y onto a set X is defined as:

$$\Pi_X(y) = \arg\min_{x \in X} \frac{1}{2} \parallel x - y \parallel_2^2 \,. \tag{5.19}$$

Using projected gradient ascent, we first compute the update:

$$y_{k+1} = x_k + \alpha \cdot \nabla_x V(x_k) \,, \tag{5.20}$$

and then we project the new point y_{k+1} into the feasible set X, according to:

$$x_{k+1} \in \Pi_X(y_{k+1}) \,. \tag{5.21}$$

The projected gradient descent (or ascent) shares the same convergence rate and guarantees as the unconstrained case, under specific conditions on the smoothness and the convexity of the objective function [176]. However, the computational cost of the projection operation depends on the characteristics of the constrained space X. Let us also remark that, in practice, we assume the gradients to exist on the boundary of $\Psi \times \Theta$. If this assumption does not hold, we can consider a compact subset K of the interior of $\Psi \times \Theta$ such that Theorem 1 ensures the existence of the gradients on K.

The DESGA algorithm will update the vector of parameters ψ and θ according to Eqns. (5.20) and (5.21). In practice, the gradients are approximated using Theorem 2, such that projected stochastic gradient ascent is performed. Furthermore, we choose as the baseline the expected cumulative reward approximated by averaging the observed cumulative reward over the M histories h^m used for computing the loss function:

$$B = \frac{1}{M} \sum_{m=0}^{M-1} \sum_{t=0}^{T-1} r_t^m \,. \tag{5.22}$$

The execution of projected stochastic gradient ascent algorithm for optimizing the objective in Eqn. (5.7) is fully detailed in Algorithm 9 in Appendix 5.7.

5.4 Experiments

In this section, we first introduce the methodology used for assessing the performance of DESGA. Afterwards, we test the DESGA algorithm on two benchmarks, the Mass-Spring-Damper (MSD) environment and one related to the design of a solar off-grid microgrid. Both environments are fully described in Appendices 5.8 and 5.9. [2]

5.4.1 Methodology

When running the DESGA algorithm on a test problem, we will report the following results. First, at every iteration k of the algorithm we will compute the expected return of the policy on the environment for the current pair of parameter vectors (θ_k, ψ_k), that is $V(\psi_k, \theta_k)$. This value is computed by running 100 Monte-Carlo simulations. Since the DESGA algorithm is stochastic, we will actually report the average of this value obtained over 20 runs (random seeds) of the algorithm. The standard deviation over the 20 runs of the algorithm will also be reported.

[2] The implementation of our algorithm and of the different benchmarks are provided in the following github repository:
`https://github.com/adrienBolland/Direct-Environment-Search-with-Gradient-Ascent`

For every problem we will also compare the performance of DESGA with an algorithm based on a discretization Ψ_d of the environment's hypothesis space Ψ. This algorithm will run the REINFORCE algorithm for every value of $\psi_d \in \Psi_d$ and compute the expected return of the policy obtained using 100 Monte-Carlo simulations. The process will be repeated five times to estimate the average expected return that could be obtained by a policy learned by the REINFORCE algorithm for each ψ_d.

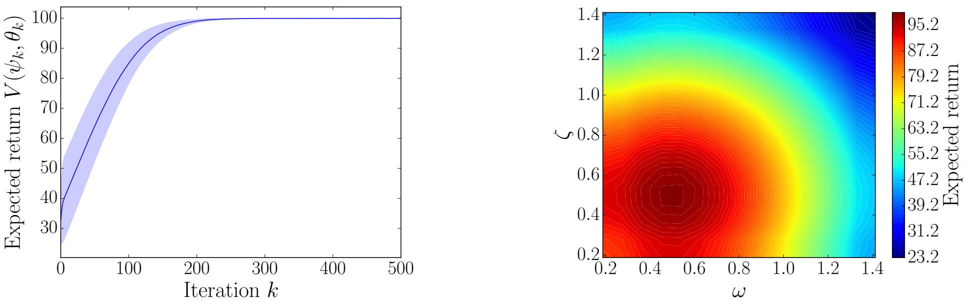
5.4.2 Mass-Spring-Damper environment

We consider here the MSD environment $(\mathscr{S}, \mathscr{A}, \Xi, P_0, f_\psi, \rho_\psi, P_\xi, T)$ described in detail in Appendix 5.8.

Hypothesis spaces. The environment is parametrized by the real vector $\psi = (\omega, \zeta, \phi_0, \phi_1, \phi_2) \in \Psi = [0.1,\, 1.5] \times [0.1,\, 1.5] \times [-2,\, 2] \times [-2,\, 2] \times [-2,\, 2] \subsetneq \mathbb{R}^5$. We will constrain the hypothesis space for the policies to time-invariant policies, meaning $\pi_\theta(a|s,t) = \pi_\theta(a|s,t'), \forall a \in \mathscr{A}, \forall s \in \mathscr{S}, \forall t,t' \in \{0,\dots,T-1\}$. Any of these policies is a multi-layer perceptron (MLP) with two inputs (one for each value of the state vector s), and with one hidden layer of 128 neurons with hyperbolic tangent activation functions. The MLP has five output neurons ($|\mathscr{A}| = 5$) from which a probability distribution over $\mathscr{A}$ will be inferred using a softmax function. All the possible values for the parameters of the MLP define the policy's hypothesis space Θ.

Parameters of the DESGA algorithm. The gradients are evaluated applying automatic differentiation on the loss function defined in Eqn. (5.15). Furthermore, the *Adam* algorithm is used for updating (ψ, θ). It is a variant on the vanilla stochastic gradient ascent given in Algorithm 9 which has proven to perform well on highly non-convex problems [170]. The gradients are estimated on batches of $M = 64$ trajectories and the stepsize α of the Adam algorithm is chosen equal to 0.005. We keep the default values for the other parameters of the Adam algorithm. Furthermore, the states are z-normalized by an average vector corresponding to the equilibrium position $(x_{eq}, 0)$ targeted by an optimal policy, as explained in Appendix 5.8. The standard deviation of the scaling is chosen equal to $(0.005, 0.02)$, an approximation of the standard deviation vector of the states collected over high-performing trajectories.

Performance of the DESGA algorithm. Figure 5.1a shows the evolution of the expected return, estimated with 100 Monte-Carlo samples, averaged over 20 runs of the DESGA algorithm. The standard deviation between the different runs is illustrated by the shaded area around the mean. As we can see, the DESGA algorithm converges towards a maximal expected return almost equal to 100. We note that 100 is an upper-bound on the return that can only be reached if at each time-step t, the position of the mass is at its equilibrium x_{eq}. The standard deviation also strongly decreases as the iterations go on. We discovered that by using time-variant policies, better results could not be obtained for this problem. Furthermore, Fig 5.1b shows the average expected return of 5 policies computed by the REINFORCE algorithm for each $\psi_d \in \Psi_d = \Omega_d \times Z_d \times \{c_0\} \times \{c_1\} \times \{c_2\}$ where $\Omega_d = Z_d = \{0.1 + k \cdot \Delta | k = 1, \dots, 15\}$ with $\Delta = 0.082$. We note that, c_0, c_1 and c_2 correspond to an optimal triplet of values for ϕ_1, ϕ_2 and ϕ_3, respectively, as described in Appendix 5.8. The highest average expected return of the policies occurs for $(\omega, \zeta) = (0.5, 0.5)$. Finally, the average expected return of the policies identified by the REINFORCE algorithm, for this value of $\psi = (0.5, 0.5, c_0, c_1, c_2)$, was almost identical to the expected returns obtained by the policies computed with the DESGA algorithm. We also note that the DESGA algorithm converged at every run towards a ψ whose ω and ζ components were both equal to 0.5 and whose triplet (ϕ_0, ϕ_1, ϕ_2) was always optimal, but not necessarily equal to (c_0, c_1, c_2).



(A) Evolution of the expected return

(B) Performance of REINFORCE on Ψ_d

FIGURE 5.1: Assessment of DESGA. Left: the average value of $V(\theta_k, \psi_k)$ and its standard deviation over 20 executions of the DESGA algorithm as a function of the number of iterations k of the algorithm. Right: the average expected return of five policies identified with the REINFORCE algorithm for every element $\psi_d \in \Psi_d$.

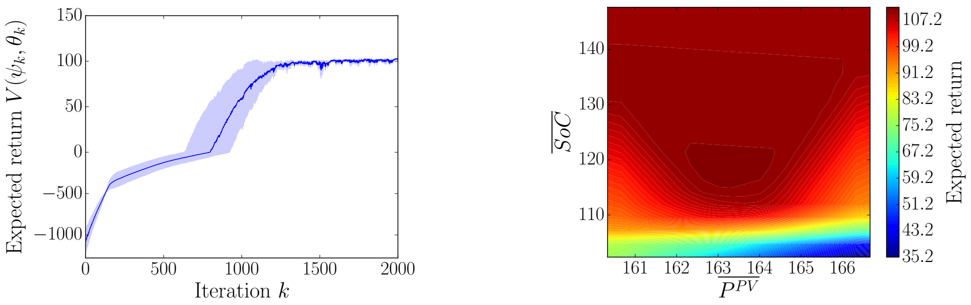
5.4.3 Sizing and operation of a solar off-grid microgrid

In this section, we consider the solar off-grid microgrid environment $(\mathscr{S}, \mathscr{A}, \Xi, P_0, f_\psi, \rho_\psi, P_\xi, T)$ presented in Appendix 5.9.

Hypothesis spaces. The environment is parametrized by the real vector $\psi = (\overline{SoC}, \overline{P^{PV}}) \in \Psi = [0,200] \times [0,300]$. We will constrain the hypothesis space for the policies to time-invariant Gaussian policies, i.e. policies such that $\pi_\theta(a|s,t) = \mathcal{N}(a|\mu_\theta(s), \sigma_\theta(s))$, $\forall a \in \mathcal{A}, \forall s \in \mathcal{S}, \forall t \in \{0,\ldots,T-1\}$ where $\mu_\theta(s)$ and $\sigma_\theta(s)$ are the expectation and the standard deviation of the normal distribution $\mathcal{N}$ in function of the state s and of the parameter vector θ, respectively. A MLP with four inputs (one for each value of the state vector s), and with one hidden layer of 128 neurons with hyperbolic tangent activation functions, outputs the two values $\mu_\theta(s)$ and $\sigma_\theta(s)$. All the possible values for the parameters of the MLP define the policy's hypothesis space Θ.

Parameters of the DESGA algorithm. The parameters related to the optimization process are the same as those used for the MSD environment in Section 5.4.2. The states are z-normalized by the average vector $(100, 12, 6.31, 6.48)$ and by the standard deviation vector $(50, 6, 8.9, 2)$. These values represent the mean and the standard deviation of the state vector for a microgrid configuration where $\psi = (200, 300)$. The rewards collected are scaled linearly from the interval $[-5000, 0]$ to the interval $[0, 1]$. Moreover, the vector ψ is scaled from $[0,200] \times [0,300]$ to $[0,1] \times [0,1]$ in the interest of keeping the optimization variables in a small range.

Performance of the DESGA algorithm. Similar to Section 5.4.2, Figure 5.2a presents the evolution of the average expected (scaled) return collected in the solar off-grid microgrid environment, averaged over 20 runs of the DESGA algorithm. As we can see, the DESGA algorithm converges towards a maximal expected return that stands around a value of 100. We note that 120 is an upper bound on the expected return that can only be reached if, during the entire horizon ($T = 120$), the instantaneous reward takes the value one. The standard deviation also strongly decreases as the iterations go on. Furthermore, Fig 5.2b shows the average expected return of five policies computed with the REINFORCE algorithm at each point $\psi^d \in \Psi^d = \{0,2,...,200\} \times \{0,3,...,300\}$, where Ψ^d is a discrete subset of the hypothesis space Ψ that forms a mesh 100×100. We also note that the DESGA algorithm is converging at every run towards a value of $\psi = (\overline{SoC}, \overline{P^{PV}}) = (114, 165)$, which is very close to the value of $\psi^d = (114, 166)$ that leads to the highest average expected return of policies computed with the REINFORCE algorithm.



(A) Evolution of the expected return

(B) Performance of REINFORCE on Ψ_d

FIGURE 5.2: Assessment of DESGA on the solar off-grid microgrid environment. Left: the average value of $V(\theta_k, \psi_k)$ and its standard deviation over 20 executions of the DESGA algorithm as a function of the number of iterations k of the algorithm. Right: the average expected return of five policies identified with the REINFORCE algorithm for every element $\psi_d \in \Psi_d$. A magnified area of the original graph is presented, where the maximum values are located.

5.4.4 Discussion on the alternative to the DESGA algorithm

In order to compare the DESGA algorithm, we have decided to discretize the hypothesis space of environments Ψ and to apply the REINFORCE algorithm on each environment ψ_d of the discretized set Ψ_d. In the following, we will describe some implications of this choice and justify this procedure. First, we used the REINFORCE algorithm instead of any other policy gradient method. This choice is motivated by Corollary 4 stating that for a fixed environment, the DESGA algorithm is equivalent to the REINFORCE algorithm. The Figures 5.1b and 5.2b thus provide the best possible average performance of the DESGA algorithm assuming the discrete set Ψ_d is precise enough. Also, since this method enabled us to approach the theoretical (tight) upper bounds on the return of any policies in the environments for both benchmarks, it was not necessary to use any other policy gradient algorithm to provide a clear view on the maximal performance one could achieve without the DESGA algorithm. This method based on a discretization has as only drawback that it is computationally inefficient and not scalable to larger problems.

5.5 Conclusions and Future Work

In this paper, we propose an algorithm that can jointly optimize an RL environment and a policy with maximal expected return over a joint hypothesis space of environments and policies. This algorithm is suited to cases in which the design of the environment and the applied policy are interdependent. We demonstrate the performance of DESGA on the design of an MSD environment and on the sizing of an autonomous energy system. The results show that the DESGA algorithm outputs a solution which is equivalent in terms of performance to the one obtained by the REINFORCE algorithm run for every element of a finely discretized environment's hypothesis space.

In this paper, the DESGA algorithm was designed in the context of jointly optimizing the design of a discrete-time dynamical system and its policy. This algorithm could be extended to the case where the environment is a finite-time Markov Decision Process (MDP) performing a similar development as the one presented in Section 5.3.1. The approach could also be extended to environments with infinite-time horizons.

Future work could also be directed on an approximation of the gradients. With the computational complexity of the automatic differentiation being proportional to the optimization horizon, the problem may become intractable for long horizons. An analytical bound on the error when performing this approximation would be valuable for striking a trade-off between computational efficiency and the quality of the solution.

Additionally, as future work, the proposed method could also be combined with recent research in gradient-based direct policy search. The use of actor-critic methods, proximal policy optimization, etc., that are shown to result in stable learning and efficient exploration, could lead to better performance. This would come at the expense of involving the additional approximation architecture (set of parameters) of a value function.

Finally, in this paper we assumed that we have direct access to the parametrized dynamics of the system, the reward function, and the disturbance function. In the event these assumptions do not hold, we propose constructing an approximation of these functions by a differentiable function approximator as future work. This would introduce an additional learning step, in order to obtain a good approximation architecture from observations, which would then be used in the proposed algorithm.

Appendices

5.6 Analytical derivation of the gradient for learning optimal environments

Theorem 1. Let $(\mathscr{S},\mathscr{A},\Xi,P_0,f_\psi,\rho_\psi,P_\xi,T)$ and π_θ be an environment and a policy as defined in Section 5.2.2. Additionally, let the functions f_ψ, ρ_ψ and P_ξ be continuously differentiable over their domain of definition. Let $V(\psi,\theta)$ be the expected cumulative reward of policy π_θ, averaged over the initial states, for all $(\psi,\theta) \in \Psi \times \Theta$ as defined in Eqn. (5.8).

Then, the function V exists, is bounded, and is continuously differentiable in the interior of $\Psi \times \Theta$.

Proof. Let us first define the random variable associating the cumulative reward to a realization of a trajectory sampled from a policy in the environment for fixed parameter vectors $(\psi,\theta) \in \Psi \times \Theta$. We prove its expectation exists and is bounded for all $(\psi,\theta) \in \Psi \times \Theta$. Furthermore, $V(\psi,\theta)$ is defined by a parametric integral which we prove to be continuously differentiable for all $(\psi,\theta) \in \Psi \times \Theta$.

Let $\mathscr{R}_{\psi,\theta}$ be the real random variable that associates to the realization of a trajectory given $\psi \in \Psi$ and $\theta \in \Theta$ its cumulative reward . Given a trajectory τ, the random variable $\mathscr{R}_{\psi,\theta}$ takes as values $R_{\psi,\theta}(\tau)$ as defined in Eqn. (5.4). Let $P_{\mathscr{R}_{\psi,\theta}}$ be the induced probability of this random variable. We can write:

$$\begin{aligned} P_{\mathscr{R}_{\psi,\theta}} &= P_{\psi,\theta}(s_0, a_0, \xi_0, a_1, \xi_1, \ldots, a_{T-1}, \xi_{T-1}) && (5.23)\\ &= P_0(s_0) \prod_{t=0}^{T-1} \pi_\theta(a_t|s_t,t) P_\xi(\xi_t|s_t,a_t)\,, && (5.24) \end{aligned}$$

where $s_{t+1} = f_\psi(s_t, a_t, \xi_t)$. The expected cumulative reward given in Eqn. (5.8) is the expectation of the random variables $\mathscr{R}_{\psi,\theta}$. If the expectation exists, it can therefore be written as:

$$\begin{aligned} V(\psi,\theta) = \int \Big(P_0(s_0) \prod_{t=0}^{T-1} \pi_\theta(a_t|s_t,t) P_\xi(\xi_t|s_t,a_t)\Big) \\ \Big(\sum_{t=0}^{T-1} \rho_\psi(s_t,a_t,\xi_t)\Big)\, ds_0 da_0 \ldots da_{T-1} d\xi_0 \ldots d\xi_{T-1}\,, \quad (5.25) \end{aligned}$$

or, more simply, as:

$$V(\psi,\theta) = \int P_{\mathscr{R}_{\psi,\theta}}(\tau) R_{\psi,\theta}(\tau) d\tau\,. \quad (5.26)$$

The integration theory has shown that a measurable function upper-bounded in norm almost-everywhere by an integrable function on a domain is itself integrable on this domain. Moreover, a random variable is measurable by definition and the cumulative reward is such that:

$$\int |P_{\mathscr{R}_{\psi,\theta}} R_{\psi,\theta}(\tau)| d\tau \leq \int P_{\mathscr{R}_{\psi,\theta}} T\, r_{max} d\tau \leq T\, r_{max}\,. \tag{5.27}$$

The integral defined by Eqn. (5.26) thus exists and the function V is bounded for all $(\psi,\theta) \in \Psi \times \Theta$.

As a corollary to the Leibniz integral rule, a function defined as in Eqn. (5.26) is continuously differentiable on the interior of the set $\Psi \times \Theta$ if $P_{\mathscr{R}_{\psi,\theta}} R_{\psi,\theta}(\tau)$ is continuously differentiable on the compact $\Psi \times \Theta \times X$ where $X = \mathscr{S} \times (\mathscr{A} \times \Xi)^T$ is the set of all trajectories. The latter is true by hypothesis. Furthermore, it implies that the partial derivative of the integral equals the integral of the partial derivative of the integrand.

□

Corollary 1. The function V, as defined in Theorem 1, exists, is bounded and is continuously differentiable on the interior of $\Psi \times \Theta$ if $\mathscr{A}$ and/or Ξ are discrete.

Proof. Let us write the expression of the expectation (5.8) in the three cases depending on whether $\mathscr{A}$ and/or Ξ are discrete and show that the different results of Theorem 1 are still valid.

1. If $\mathscr{A}$ is discrete:

$$V(\psi,\theta) = \int \sum_{(a_0,\ldots a_{T-1}) \in \mathscr{A}^T} \big(P_0(s_0) \prod_{t=0}^{T-1} \pi_\theta(a_t|s_t,t) P_\xi(\xi_t|s_t,a_t)\big) \\ \big(\sum_{t=0}^{T-1} \rho_\psi(s_t,a_t,\xi_t)\big)\, ds_0 d\xi_0 \ldots d\xi_{T-1}\,. \tag{5.28}$$

2. If Ξ is discrete:

$$V(\psi,\theta) = \int \sum_{(\xi_0,\ldots \xi_{T-1}) \in \Xi^T} \big(P_0(s_0) \prod_{t=0}^{T-1} \pi_\theta(a_t|s_t,t) P_\xi(\xi_t|s_t,a_t)\big) \\ \big(\sum_{t=0}^{T-1} \rho_\psi(s_t,a_t,\xi_t)\big)\, ds_0 da_0 \ldots da_{T-1}\,. \tag{5.29}$$

3. If $\mathscr{A}$ and Ξ are discrete:

$$V(\psi,\theta)=\int \sum_{(a_0,\dots a_{T-1})\in\mathscr{A}^T}\ \sum_{(\xi_0,\dots \xi_{T-1})\in\Xi^T}\Big(P_0(s_0)\prod_{t=0}^{T-1}\pi_\theta(a_t|s_t,t)P_\xi(\xi_t|s_t,a_t)\Big)\Big(\sum_{t=0}^{T-1}\rho_\psi(s_t,a_t,\xi_t)\Big)\,ds_0\,. \tag{5.30}$$

In the three cases, we can still bound the integral as in Eqn. (5.27) and apply the corollary of the Leibniz integral rule if the integrand is continuously differentiable for all discrete values. Finally, by linearity of the differential operator, the operator can be distributed on the terms of the different sums when computing the derivative of the function V.

□

Corollary 2. The gradient of the function V defined in Eqn. (5.8) with respect to the parameter vector ψ is such that:

$$\nabla_\psi V(\psi,\theta)=\mathop{\mathbb{E}}_{\substack{s_0\sim P_0(\cdot)\\ a_t\sim\pi_\theta(\cdot|s,t)\\ \xi_t\sim P_\xi(\cdot|s_t,a_t)}}\Big\{\Big(\sum_{t=0}^{T-1}\big(\nabla_s\log\pi_\theta(a_t|s,t)|_{s=s_t}+\nabla_s\log P_\xi(\xi_t|s,a_t)|_{s=s_t}\big)\cdot\nabla_\psi s_t\Big) \times\Big(\sum_{t=0}^{T-1}r_t\Big)+\Big(\sum_{t=0}^{T-1}\nabla_\psi\rho_\psi(s,a_t,\xi_t)|_{s=s_t}+\nabla_s\rho_\psi(s,a_t,\xi_t)|_{s=s_t}\cdot\nabla_\psi s_t\Big\}\,, \tag{5.31}$$

where:

$$\nabla_\psi s_t=(\nabla_s f_\psi)(s,a_{t-1},\xi_{t-1})|_{s=s_{t-1}}\cdot\nabla_\psi s_{t-1}+(\nabla_\psi f_\psi)(s,a_{t-1},\xi_{t-1})|_{s=s_{t-1}}\,, \tag{5.32}$$

with $\nabla_\psi s_0=0$.

Proof. To compute this gradient, we first apply the product rule for gradients to Eqn. (5.8). Afterwards, we exploit the equality $\nabla f=f\nabla\log f$ that holds if f is a continuously differentiable function.

$$\nabla_\psi V(\psi,\theta)=\int(\nabla_\psi P_{\mathscr{R}_{\psi,\theta}}(\tau))R_{\psi,\theta}(\tau)d\tau+\int P_{\mathscr{R}_{\psi,\theta}}(\tau)(\nabla_\psi R_{\psi,\theta}(\tau))d\tau \tag{5.33}$$

$$=\int P_{\mathscr{R}_{\psi,\theta}}(\tau)(\nabla_\psi\log P_{\mathscr{R}_{\psi,\theta}}(\tau))R_{\psi,\theta}(\tau)d\tau+\int P_{\mathscr{R}_{\psi,\theta}}(\tau)(\nabla_\psi R_{\psi,\theta}(\tau))d\tau \tag{5.34}$$

$$=\mathop{\mathbb{E}}_{\substack{s_0\sim P_0(\cdot)\\ a_t\sim\pi_\theta(\cdot|s_t,t)\\ \xi_t\sim P_\xi(\cdot|s_t,a_t)}}\{(\nabla_\psi\log P_{\mathscr{R}_{\psi,\theta}}(\tau))R_{\psi,\theta}(\tau)+(\nabla_\psi R_{\psi,\theta}(\tau))\}\,. \tag{5.35}$$

By applying the logarithmic operator to both sides of Eqn. (5.24), we have:

$$\log P_{\mathscr{R}_{\psi,\theta}}(\tau) = \log P_0(s_0) + \sum_{t=0}^{T-1} \log \pi_\theta(a_t|s_t,t) + \sum_{t=0}^{T-1} \log P_\xi(\xi_t|s_t,a_t)\,. \tag{5.36}$$

Let $\cdot$ denote the dot product operator. Using the chain rule formula together with Eqn. (5.4), we can write:

$$\nabla_\psi \log \pi_\theta(a_t|s_t,t) = \nabla_s \log \pi_\theta(a_t|s,t)|_{s=s_t} \cdot \nabla_\psi s_t \tag{5.37}$$

$$\nabla_\psi \log P_\xi(\xi_t|s_t,a_t) = \nabla_s \log P_\xi(\xi_t|s,a_t)|_{s=s_t} \cdot \nabla_\psi s_t \tag{5.38}$$

$$\nabla_\psi \rho_\psi(s_t,a_t,\xi_t) = \nabla_\psi \rho_\psi(s,a_t,\xi_t)|_{s=s_t} + \nabla_s \rho_\psi(s,a_t,\xi_t)|_{s=s_t} \cdot \nabla_\psi s_t\,, \tag{5.39}$$

where:

$$\nabla_\psi s_t = (\nabla_s f_\psi)(s,a_{t-1},\xi_{t-1})|_{s=s_{t-1}} \cdot \nabla_\psi s_{t-1} + (\nabla_\psi f_\psi)(s,a_{t-1},\xi_{t-1})|_{s=s_{t-1}}\,, \tag{5.40}$$

with $\nabla_\psi s_0 = 0$.

Finally, combining the previous results with Eqns. (5.35) and (5.36), we have:

$$\nabla_\psi V(\psi,\theta) = \mathop{\mathbb{E}}_{\substack{s_0\sim P_0(\cdot)\\ a_t\sim\pi_\theta(\cdot|s,t)\\ \xi_t\sim P_\xi(\cdot|s_t,a_t)}} \left\{ \Big(\sum_{t=0}^{T-1} \left(\nabla_s \log \pi_\theta(a_t|s,t)|_{s=s_t} + \nabla_s \log P_\xi(\xi_t|s,a_t)|_{s=s_t} \right) \cdot \nabla_\psi s_t \Big) \times \Big(\sum_{t=0}^{T-1} r_t \Big) + \Big(\sum_{t=0}^{T-1} \nabla_\psi \rho_\psi(s_t,a_t,\xi_t) \Big) \right\}. \tag{5.41}$$

□

Corollary 3. The gradient of the function V defined in Eqn. (5.8) with respect to the parameter vector θ is given by:

$$\nabla_\theta V(\psi,\theta) = \mathop{\mathbb{E}}_{\substack{s_0\sim P_0(\cdot)\\ a_t\sim\pi_\theta(\cdot|s_t,t)\\ \xi_t\sim P_\xi(\cdot|s_t,a_t)}} \{ (\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t,t)) (\sum_{t=0}^{T-1} r_t) \}\,. \tag{5.42}$$

Proof. Using similar derivations as for the Corollary 1, we have for the gradient with respect to θ:

$$\nabla_\theta V(\psi,\theta) = \mathop{\mathbb{E}}_{\substack{s_0\sim P_0(\cdot)\\ a_t\sim\pi_\theta(\cdot|s_t,t)\\ \xi_t\sim P_\xi(\cdot|s_t,a_t)}} \{(\nabla_\theta \log P_{\mathscr{R}_{\psi,\theta}}(\tau))R_{\psi,\theta}(\tau)\} \tag{5.43}$$

$$= \mathop{\mathbb{E}}_{\substack{s_0\sim P_0(\cdot)\\ a_t\sim\pi_\theta(\cdot|s_t,t)\\ \xi_t\sim P_\xi(\cdot|s_t,a_t)}} \{(\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t,t))(\sum_{t=0}^{T-1} r_t)\}\,. \tag{5.44}$$

□

Theorem 2. Let $(\mathscr{S},\mathscr{A},\Xi,P_0,f_\psi,\rho_\psi,P_\xi,T)$ and π_θ be an environment and a policy as defined in Section 5.2.2. Let $V(\psi,\theta)$ be the expected cumulative reward of policy π_θ averaged over the initial states as defined in Eqn. (5.8). Let $\mathscr{D}=\{h^m|m=0,\dots,M-1\}$ be a set of M histories sampled independently and identically from the policy π_θ in the environment. Let $\mathscr{L}$ be a loss function such that, $\forall(\psi,\theta)\in\Psi\times\Theta$:

$$\mathscr{L}(\psi,\theta) = -\frac{1}{M}\sum_{m=0}^{M-1}\Big(\sum_{t=0}^{T-1}\log\pi_\theta(a_t^m|s_t^m,t)+\log P_\xi(\xi_t^m|s_t^m,a_t^m)) \\ \times\big((\sum_{t=0}^{T-1} r_t^m)-B\big)+\big(\sum_{t=0}^{T-1}\rho_\psi(s_t^m,a_t^m,\xi_t^m))\Big)\,, \tag{5.45}$$

where B is a constant value called the baseline.

The gradients with respect to ψ and θ of the loss function are unbiased estimators of the gradients of the function V as defined in Eqn. (5.8), with opposite directions, i.e. such that:

$$\mathop{\mathbb{E}}_{\substack{s_0\sim P_0(\cdot)\\ a_t\sim\pi_\theta(\cdot|s_t,t)\\ \xi_t\sim P_\xi(\cdot|s_t,a_t)}} \{\nabla_\psi\mathscr{L}(\psi,\theta)\} = -\nabla_\psi V(\psi,\theta) \tag{5.46}$$

$$\mathop{\mathbb{E}}_{\substack{s_0\sim P_0(\cdot)\\ a_t\sim\pi_\theta(\cdot|s_t,t)\\ \xi_t\sim P_\xi(\cdot|s_t,a_t)}} \{\nabla_\theta\mathscr{L}(\psi,\theta)\} = -\nabla_\theta V(\psi,\theta)\,. \tag{5.47}$$

Proof. Let us first rewrite the loss function using the notations of Theorem 1. We have:

$$\mathscr{L}(\psi,\theta) = -\frac{1}{M}\sum_{m=0}^{M-1}(\log P_{\mathscr{R}_{\psi,\theta}}(\tau^m)-\log P_0(s_0^m))\big((\sum_{t=0}^{T-1} r_t^m)-B\big)+(R_{\psi,\theta}(\tau^m))\,. \tag{5.48}$$

The expectation of the gradient with respect to ψ is given by:

$$\underset{\substack{s_0 \sim P_0(\cdot) \\ a_t \sim \pi_\theta(\cdot|s_t,t) \\ \xi_t \sim P_\xi(\cdot|s_t,a_t)}}{\mathbb{E}} \{\nabla_\psi \mathscr{L}(\psi,\theta)\} = \underset{\substack{s_0 \sim P_0(\cdot) \\ a_t \sim \pi_\theta(\cdot|s_t,t) \\ \xi_t \sim P_\xi(\cdot|s_t,a_t)}}{\mathbb{E}} \{-\frac{1}{M}\sum_{m=0}^{M-1} \nabla_\psi(\log P_{\mathscr{R}_{\psi,\theta}}(\tau^m) - \log P_0(s_0^m)) \\ \times \big((\sum_{t=0}^{T-1} r_t^m) - B\big) + \nabla_\psi(R_{\psi,\theta}(\tau^m))\} \,. \tag{5.49}$$

Observing that every term in the sum has the same expectation and that $\nabla_\psi \log P_0(s_0^m) = 0$, we can write:

$$\underset{\substack{s_0 \sim P_0(\cdot) \\ a_t \sim \pi_\theta(\cdot|s_t,t) \\ \xi_t \sim P_\xi(\cdot|s_t,a_t)}}{\mathbb{E}} \{\nabla_\psi \mathscr{L}(\psi,\theta)\} = -\underset{\substack{s_0 \sim P_0(\cdot) \\ a_t \sim \pi_\theta(\cdot|s_t,t) \\ \xi_t \sim P_\xi(\cdot|s_t,a_t)}}{\mathbb{E}} \{\nabla_\psi(\log P_{\mathscr{R}_{\psi,\theta}}(\tau)) \times \big((\sum_{t=0}^{T-1} r_t) - B\big) + \nabla_\psi(R_{\psi,\theta}(\tau))\} \,. \tag{5.50}$$

Moreover:

$$\underset{\substack{s_0 \sim P_0(\cdot) \\ a_t \sim \pi_\theta(\cdot|s_t,t) \\ \xi_t \sim P_\xi(\cdot|s_t,a_t)}}{\mathbb{E}} \{\nabla_\psi(\log P_{\mathscr{R}_{\psi,\theta}}(\tau))B\} = \nabla_\psi \underset{\substack{s_0 \sim P_0(\cdot) \\ a_t \sim \pi_\theta(\cdot|s_t,t) \\ \xi_t \sim P_\xi(\cdot|s_t,a_t)}}{\mathbb{E}} \{B\} = 0 \,, \tag{5.51}$$

such that:

$$\underset{\substack{s_0 \sim P_0(\cdot) \\ a_t \sim \pi_\theta(\cdot|s_t,t) \\ \xi_t \sim P_\xi(\cdot|s_t,a_t)}}{\mathbb{E}} \{\nabla_\psi \mathscr{L}(\psi,\theta)\} = -\nabla_\psi V(\psi,\theta) \,. \tag{5.52}$$

Equivalently, the expectation of the gradient with respect to θ is given by:

$$\underset{\substack{s_0 \sim P_0(\cdot) \\ a_t \sim \pi_\theta(\cdot|s_t,t) \\ \xi_t \sim P_\xi(\cdot|s_t,a_t)}}{\mathbb{E}} \{\nabla_\theta \mathscr{L}(\psi,\theta)\} = -\underset{\substack{s_0 \sim P_0(\cdot) \\ a_t \sim \pi_\theta(\cdot|s_t,t) \\ \xi_t \sim P_\xi(\cdot|s_t,a_t)}}{\mathbb{E}} \{\nabla_\theta(\log P_{\mathscr{R}_{\psi,\theta}}(\tau)) \\ \times \big((\sum_{t=0}^{T-1} r_t) - B\big) + \nabla_\theta(R_{\psi,\theta}(\tau))\} \,. \tag{5.53}$$

The expectation of the term relative to the baseline is zero:

$$\underset{\substack{s_0 \sim P_0(\cdot) \\ a_t \sim \pi_\theta(\cdot|s_t,t) \\ \xi_t \sim P_\xi(\cdot|s_t,a_t)}}{\mathbb{E}} \{\nabla_\theta(\log P_{\mathscr{R}_{\psi,\theta}}(\tau))B\} = \nabla_\theta \underset{\substack{s_0 \sim P_0(\cdot) \\ a_t \sim \pi_\theta(\cdot|s_t,t) \\ \xi_t \sim P_\xi(\cdot|s_t,a_t)}}{\mathbb{E}} \{B\} = 0 \,. \tag{5.54}$$

Furthermore, the gradient of the reward function with respect to θ is zero:

$$\mathop{\mathbb{E}}_{\substack{s_0\sim P_0(\cdot)\\ a_t\sim\pi_\theta(\cdot|s_t,t)\\ \xi_t\sim P_\xi(\cdot|s_t,a_t)}} \{\nabla_\theta(R_{\psi,\theta}(\tau))\} = \mathop{\mathbb{E}}_{\substack{s_0\sim P_0(\cdot)\\ a_t\sim\pi_\theta(\cdot|s_t,t)\\ \xi_t\sim P_\xi(\cdot|s_t,a_t)}} \{(\sum_{t=0}^{T-1}\nabla_\theta\rho_\psi(s_t,a_t,\xi_t)\} = 0\,. \tag{5.55}$$

We thus have that:

$$\mathop{\mathbb{E}}_{\substack{s_0\sim P_0(\cdot)\\ a_t\sim\pi_\theta(\cdot|s_t,t)\\ \xi_t\sim P_\xi(\cdot|s_t,a_t)}} \{\nabla_\theta\mathscr{L}(\psi,\theta)\} = -\mathop{\mathbb{E}}_{\substack{s_0\sim P_0(\cdot)\\ a_t\sim\pi_\theta(\cdot|s_t,t)\\ \xi_t\sim P_\xi(\cdot|s_t,a_t)}} \{\nabla_\theta(\log P_{\mathscr{R}_{\psi,\theta}}(\tau))\times\big(\sum_{t=0}^{T-1} r_t\big)\} \tag{5.56}$$

$$= -\mathop{\mathbb{E}}_{\substack{s_0\sim P_0(\cdot)\\ a_t\sim\pi_\theta(\cdot|s_t,t)\\ \xi_t\sim P_\xi(\cdot|s_t,a_t)}} \{\nabla_\theta(\sum_{t=0}^{T-1}\log\pi_\theta(a_t|s_t,t))\times\big(\sum_{t=0}^{T-1} r_t\big)\}\,. \tag{5.57}$$

Finally, we have that:

$$\mathop{\mathbb{E}}_{\substack{s_0\sim P_0(\cdot)\\ a_t\sim\pi_\theta(\cdot|s_t,t)\\ \xi_t\sim P_\xi(\cdot|s_t,a_t)}} \{\nabla_\theta\mathscr{L}(\psi,\theta)\} = -\nabla_\theta V(\psi,\theta)\,. \tag{5.58}$$

□

Corollary 4. The gradient of the loss function, defined in Eqn. 5.15, with respect to θ corresponds to the opposite of the update direction computed with the REINFORCE algorithm [66] averaged over M simulations.

Proof. The gradient of the loss function with respect to θ is given by:

$$\nabla_\theta\mathscr{L}(\psi,\theta) = -\sum_{m=0}^{M-1}\Big(\nabla_\theta(\log P_{\mathscr{R}_{\psi,\theta}}(\tau^m))\times\big(R_{\psi,\theta}(\tau^m)-B\big)\Big)\,. \tag{5.59}$$

The gradient is the opposite of the average over M trajectories of the update direction of the REINFORCE algorithm [66].

□

5.7 Direct environment search with (projected stochastic) gradient ascent

Algorithm 9 DESGA

```
function Optimize($(\mathscr{S}, \mathscr{A}, \Xi, P_0, f_\psi, \rho_\psi, P_\xi, T)$, $\pi_\theta$, $\Pi_\Psi$, $\Pi_\Theta$)
Parameter Number of gradient steps $N$
Parameter Batch size $M$
Parameter Learning rate $\alpha$
for $n \in \{0, \ldots, N-1\}$ do
    for $m \in \{0, \ldots, M-1\}$ do
        $h =$ GenerateHistory($(\mathscr{S}, \mathscr{A}, \Xi, P_0, f_\psi, \rho_\psi, P_\xi, T)$, $\pi_\theta$)
        Add $h$ to the set $\mathscr{D}$
    end for
    Compute the baseline using the histories $B = \frac{1}{m} \sum_{m=0}^{M-1} \sum_{t=0}^{T-1} r_t$
    Differentiate Eqn. (5.15) for estimating the gradients Eqns. (5.11) and (5.13) using $\mathscr{D}$
    $(\psi, \theta) =$ VanillaGradientAscent($\psi$, $\theta$, $\alpha$, $\hat{\nabla}_\psi V(\psi, \theta)$, $\hat{\nabla}_\theta V(\psi, \theta)$)
    $\psi \leftarrow \Pi_\Psi(\psi)$
    $\theta \leftarrow \Pi_\Theta(\theta)$
end for

Output: $(\psi, \theta)$

function GenerateHistory($(\mathscr{S}, \mathscr{A}, \Xi, P_0, f_\psi, \rho_\psi, P_\xi, T)$, $\pi_\theta$)
Sample an initial state: $s_0 \sim P_0(\cdot)$
for $t \in \{0, \ldots, T-1\}$ do
    $a_t \sim \pi_\theta(\cdot | s_t, t)$
    $\xi_t \sim P_\xi(\cdot | s_t, a_t)$
    $s_{t+1} = f_\psi(s_t, a_t, \xi_t)$
    $r_t = \rho_\psi(s_t, a_t, \xi_t)$
end for
$h = (s_0, a_0, \xi_0, r_0, a_1, \xi_1, \ldots, a_{T-1}, \xi_{T-1}, r_{T-1})$

Output: $h$

function VanillaGradientAscent($\psi$, $\theta$, $\alpha$, $\hat{\nabla}_\psi V(\psi, \theta)$, $\hat{\nabla}_\theta V(\psi, \theta)$)
$\psi \leftarrow \psi + \alpha \cdot \hat{\nabla}_\psi V(\psi, \theta)$
$\theta \leftarrow \theta + \alpha \cdot \hat{\nabla}_\theta V(\psi, \theta)$
```

5.8 Mass-Spring-Damper environment

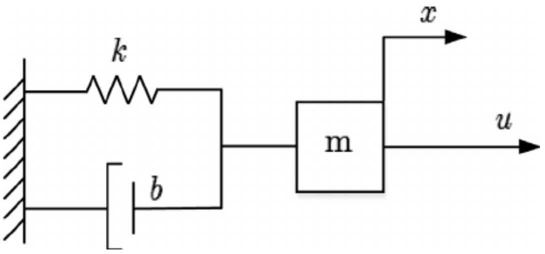


FIGURE 5.3: Mass-Spring-Damper system.

Let us consider a Mass-Spring-Damper (MSD) system defined as follows. A point mass m is attached to a spring and a damper. The spring has a Hooke constant k and the damping is proportional to the speed through the damping constant b. The damping force acts in the direction opposite to the motion. Furthermore, the system is subject to an external force u. Let x denote the position of the mass. The continuous-time system dynamics is described by Newton's second law as:

$$m\ddot{x} = -kx - b\dot{x} + u\,, \tag{5.60}$$

which can equivalently be written as:

$$\ddot{x} + 2\zeta\omega\dot{x} + \omega^2 x = a\,, \tag{5.61}$$

where:

$$\omega = \sqrt{\frac{k}{m}} \tag{5.62}$$

$$\zeta = \frac{b}{2m\omega} \tag{5.63}$$

$$a = \frac{u}{m}\,. \tag{5.64}$$

The evolution of the position x of the mass is thus described by the position itself and the speed v as:

$$\begin{cases} \dot{x} &= \quad v \\ \dot{v} &= \quad a - 2\zeta\omega v - \omega^2 x\,. \end{cases} \tag{5.65}$$

Optimization horizon. The optimization horizon T refers to the number of actions to be taken in the discrete process.

State space. The state is described at every time t by two variables: the position x_t and the speed v_t. The state space of the system is:

$$\mathscr{S} = \mathbb{R}^2 \,. \tag{5.66}$$

Initial state distribution. The initial states x_0 and v_0 are uniformly drawn from the intervals $[x_{0,min}, x_{0,max}]$ and $[v_{0,min}, v_{0,max}]$.

Action space. In its most general setting, the system can be submitted to any external acceleration a. However, we will only consider a discrete action space defined as follows:

$$\mathscr{A} = \{-0.3, -0.1, 0, 0.1, 0.3\} \,. \tag{5.67}$$

Disturbance space. We will consider a stochastic version of the problem where a real disturbance ξ_t is added to the action a_t such that an acceleration $a_t + \xi_t$ is applied to the system. In such a context, we have:

$$\xi_t \in \Xi = \mathbb{R} \,. \tag{5.68}$$

Disturbance distribution. The disturbance is sampled at time t from a Normal distribution centred at the current position x_t, and whose standard deviation is a linear combination of the magnitude of the action a_t and of the speed v_t:

$$P_\xi\left(\xi_t | s_t, a_t\right) = \mathscr{N}\left(\xi_t | x_t, 0.1 \times |a_t| + |s_t| + \varepsilon\right) , \tag{5.69}$$

where ε is a constant equal to 10^{-6}.

Discrete dynamics. The discrete-time process comes from a discretization of the continuous process defined by Eqn. (5.65) with a discretization time-step $\Delta = 50$ms. The discrete dynamics f is the function computing the position and speed after a period Δ during which the constant acceleration $a_t + \xi_t$ is applied. The position x_{t+1} and the speed v_{t+1} can be computed from x_t and v_t using these analytical expressions:

$$x_{t+1} = g(x_t, v_t, a_t + \xi_t, \Delta), \tag{5.70}$$

$$v_{t+1} = \frac{\partial g}{\partial t}(x_t, v_t, a_t + \xi_t, t)|_{t=\Delta} \,, \tag{5.71}$$

where:

$$g(x_t, v_t, a, t) = \frac{a}{\omega^2} + \exp(-\zeta \omega t) \times$$
$$\begin{cases} (x_t - \frac{a}{\omega^2}) \cosh(\sqrt{\zeta^2 - 1}\omega t) + \frac{\frac{v_t}{\omega} + \zeta(x_t - \frac{a}{\omega^2})}{\sqrt{\zeta^2 - 1}} \sinh(\sqrt{\zeta^2 - 1}\omega t) & \text{, if} \quad \zeta > 1 \\ (x_t - \frac{a}{\omega^2}) + \big(v_t + \omega(x_t - \frac{a}{\omega^2})\big)t & \text{, if} \quad \zeta = 1 \\ (x_t - \frac{a}{\omega^2}) \cos(\sqrt{1 - \zeta^2}\omega t) + \frac{\frac{v_t}{\omega} + \zeta(x_t - \frac{a}{\omega^2})}{\sqrt{1 - \zeta^2}} \sin(\sqrt{1 - \zeta^2}\omega t) & \text{, if} \quad 0 < \zeta < 1 \,. \end{cases} \tag{5.72}$$

Reward function. The reward function is defined as:

$$\rho(a_t, s_t, \xi_t) = \exp\Big(- |x_t - x_{eq}| - (\omega - c_\omega)^2 - (\zeta - c_\zeta)^2 - \prod_{k=1}^{K} (\phi_k - c_k)^2 \Big) \,, \tag{5.73}$$

where ω, ζ and ϕ_k are parameters of the system that need to be optimized. Furthermore x_{eq}, c_ω, c_ζ, K and c_k are constant values. Let us also remark that the reward function does not depend on the disturbance.

The first term of the exponential will be minimized if the mass is stabilized at the position x_{eq}. The second and third terms are minimized if the parameters ω and ζ are equal to c_ω and c_ζ, respectively. The last term is a strictly positive function minimized if, at least one of the parameters ϕ_k equals the value c_k. Minimizing these terms results in maximizing the reward. Furthermore, since the the reward function is the exponential of a negative value, the reward is bounded by $r_{max} = 1$.

Parametrized MSD environment. A parametrized MSD environment is an environment $(\mathscr{S}, \mathscr{A}, \Xi, P_0, f_\psi, \rho_\psi, P_\xi, T)$ parametrized by the real vector $\psi = (\omega, \zeta, \phi_0, \phi_1, \phi_2) \in \mathbb{R}^5$.

Numerical values. In this work, we will consider the values given in Table 5.1 for the constant parameters.

TABLE 5.1: Parameters for the MSD.

Symbol	Value
$x_{0,min}$	0.198
$x_{0,max}$	0.202
$v_{0,min}$	−0.010
$v_{0,max}$	0.010
x_{eq}	0.200
c_{ω}	0.500
c_{ζ}	0.500
K	3.000
c_0	0.500
c_1	−0.300
c_2	0.200
T	100

5.9 Optimal design of a solar off-grid microgrid

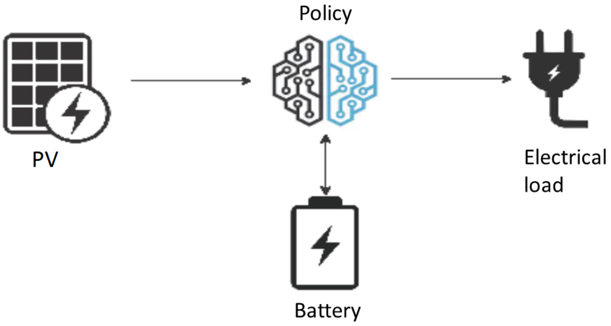


FIGURE 5.4: Microgrid configuration

A solar off-grid microgrid is a small-scale electrical grid composed of photovoltaic (PV) panels (converting solar energy into electricity) and a battery for ensuring the supply of an electrical load. A schematic of the considered configuration is presented in Fig 5.4. The total cost of the microgrid is the sum of the investment costs and the penalties obtained for shedding the load if there is insufficient electricity available. In this section, we are interested in sizing the microgrid components, i.e. identifying the optimal investment in equipment that leads to the least total cost over the investment lifetime, assuming that the microgrid is operated in an optimal way.

This problem is therefore related to the one addressed in this paper, by noticing that finding the optimal investment (i.e., the size of the PV panels and the battery) is equivalent to optimizing both the "solar off-grid microgrid" environment and the policy at the same time. We note that the actions that can be taken by the policy are related to the charging/discharging power of the battery. An optimal policy should, in principle, charge the battery when there is an excess of solar power generated by the PV panels, and discharge that power from the battery when the electrical demand cannot be fully covered by the PV panels.

We will now provide, hereafter, a formalization of this problem that exactly fits the generic problem tackled in this paper. We note that more generic formalizations may exist, as for example those where the load consumption and the PV production cannot be considered as variables fully conditioned on the hour of the day, as will be assumed here. Those stand beyond the scope of this paper, even if they could lead to other interesting problem statements. Before carefully defining this benchmark problem, let us emphasize that we will use the notation $[\cdot]$ to indicate the corresponding unit of the symbol preceding it. In this section, $[W]$ denotes instantaneous power production in Watts, $[W_p]$ denotes nameplate (manufacturer) power capacity, $[Wh]$ denotes energy in Watt-hours and $[Wh_p]$ denotes nameplate (manufacturer) energy capacity. We now define the different elements of this learning optimal environment type of problem.

Optimization horizon. The optimization horizon is denoted by the value T.

State space. The state of the system can be fully described by $s_t = (SoC_t, h_t, \bar{P}_t^{C,h}, \bar{P}_t^{PV,h}) \in \mathscr{S} = [0, \overline{SoC}] \times \{0,...,23\} \times \mathbb{R}^+ \times \mathbb{R}^+$, where, at time t:

- $SoC_t\,[Wh] \in [0, \overline{SoC}]$ denotes the state of charge of the battery. The installed capacity of the battery is denoted by $\overline{SoC}\,[Wh_p] \in \mathbb{R}^+$.
- $h_t\,[h] \in \{0,...,23\}$ denotes the hour of the day.
- $\bar{P}_t^{C,h}\,[W] \in \mathbb{R}^+$ denotes the expected value of the electrical consumption level during hour h that is considered to be known.
- $\bar{P}_t^{PV,h}\,[W] \in \mathbb{R}^+$ denotes the expected value of the PV power generation during hour $h = h_t$ that is also considered to be known.

Initial state distribution. The initial state of charge SoC_0 is drawn uniformly from the interval $\left[0,\overline{SoC}\right]$ and the initial hour h_0 takes the value zero with probability one. The initial value for $\bar{P}_0^{C,h}$ is given by the first line of Table 5.3 in the corresponding column. Let $\overline{P^{PV}}\,[W_p] \in \mathbb{R}^+$ denote the capacity of PV panels installed, the column $\bar{p}^{PV,h}$ in Table 5.3 gives the average PV production per installed capacity (%). Subsequently, the initial value for $\bar{P}_0^{PV,h}$ is given by the product of $\overline{P^{PV}}$ and the first element of column $\bar{p}^{PV,h}$ in Table 5.3.

Action space. As previously described, the available actions correspond to defining the charging/discharging power of the storage system. The charging power is denoted by $P^B \in \left[-\overline{P^B},\overline{P^B}\right]$, which will be positive during charging and negative during discharging. The charging/discharging limit $\overline{P^B} \in \mathbb{R}^+$ is assumed to be a proportion p (%) of the battery capacity as $\overline{P^B} = p \cdot \overline{SoC}$.

We therefore consider the continuous action space:

$$\mathscr{A} = \left[-\overline{P^B},\overline{P^B}\right] . \tag{5.74}$$

Disturbance space. We consider as disturbance the variable $\xi_t = E_t^{C,h} \in \Xi \subseteq \mathbb{R}$, the stochastic deviation from the expected consumption for hour h_t.

Disturbance distribution. The disturbance is sampled at time t from a Normal distribution centred at zero with standard deviation $\sigma_{C,h}$ depending on the hour $h = h_t$:

$$P_\xi\left(\xi_t|s_t,a_t\right) = \mathscr{N}\left(\xi_t|0,\sigma_{C,h}\right) . \tag{5.75}$$

The values of the standard deviations $\sigma_{C,h}$ are given in Table 5.3 for every hour h of the day.

Transition function. We use a discretization time-step Δt of one hour for defining the discrete-time dynamics. For the state variable h we have therefore:

$$h_{t+1} = \left(h_t + 1\right) \mod 24 . \tag{5.76}$$

The state of charge of the battery is updated using a linear water tank model [67]. With this tank model, the value of SoC_{t+1} at time $t+1$, if there were no limits on it, would be equal to A_{t+1} defined as follows:

$$A_{t+1} = SoC_t + \Delta t \cdot \begin{cases} \eta_{ch} \cdot P_t^B & ,\text{if } P_t^B \geq 0 \\ P_t^B / \eta_{dis} & ,\text{if } P_t^B < 0 \,, \end{cases} \tag{5.77}$$

where $\eta_{ch} \in [0,1]$, $\eta_{dis} \in [0,1]$ represent the charging and discharging efficiencies of the storage system. Given the fact that the state of charge of the battery lies within predefined limits, its state of charge at time $t+1$ is therefore defined as:

$$SoC_{t+1} = \begin{cases} 0 & ,\text{if } A_{t+1} < 0 \\ \overline{SoC} & ,\text{if } A_{t+1} \geq \overline{SoC} \\ A_{t+1} & \text{otherwise} \,. \end{cases} \tag{5.78}$$

The variable $\bar{P}_{t+1}^{C,h}$ takes the value reported in Table 5.3 at the line corresponding to the hour $h = h_{t+1}$. Finally, the variable $\bar{P}_{t+1}^{PV,h}$ is updated as:

$$\bar{P}_{t+1}^{PV,h} = \bar{p}^{PV,h} \cdot \overline{P^{PV}} \,, \tag{5.79}$$

where $\bar{p}^{PV,h}$ take the values reported in Table 5.3 at the line corresponding to the hour $h = h_{t+1}$.

Reward function. The reward signal is, in this case, a cost function composed of two parts, namely the investment cost and the operational cost. The reward signal is given by:

$$r_t = \rho(s_t, a_t, \xi_t) = -(c_t^{fix} + c_t^{shed}) \,, \tag{5.80}$$

where c_t^{fix} [\$] $\in \mathbb{R}^+$ represents a fixed hourly payment for settling the initial investment cost and c_t^{shed} [\$] $\in \mathbb{R}^+$ corresponds to the cost of shedding load at each time-step t.

In order to compute the fixed cost term c_t^{fix} we proceed as follows. Let c^{PV} $[\$/W_p] \in \mathbb{R}^+$ denote the cost per unit of PV capacity installed. The total installation cost for PV I^{PV} [\$] $\in \mathbb{R}^+$ is defined as:

$$I^{PV} = c^{PV} \cdot \overline{P^{PV}} \,. \tag{5.81}$$

Let $c^B\,[\$/Wh_p] \in \mathbb{R}^+$ denote the cost per unit of storage capacity installed. The total installation cost for battery storage $I^B\,[\$] \in \mathbb{R}^+$ is defined as:

$$I^B = c^B \cdot \overline{SoC}\,. \tag{5.82}$$

The investment cost I is the sum of the investment costs for each component of the microgrid defined as:

$$I = I^B + I^{PV}\,. \tag{5.83}$$

This payment occurs once in the beginning of the investment. In this case, we assume this investment to be a loan in its entirety. A fixed yearly payment P over the lifetime of the investment for settling the initial loan, is given by the following amortization formula:

$$P = I\frac{r(1+r)^n}{(1+r)^n - 1}\,, \tag{5.84}$$

where n is the number of years considered for the lifetime of the investment and $r(\%)$ is the interest rate considered. By noting that a common (non-leap) year has 8760 hours, we define the fixed hourly cost as:

$$c_t^{fix} = \frac{P}{8760}\,. \tag{5.85}$$

In order to compute the shedding cost term c_t^{shed} we proceed as follows. The realization of the consumption $P_t^{C,h}\,[W] \in \mathbb{R}^+$, after an action is taken at each time-step $t \in T$, corresponds to the actual consumption level in the interval $]t, t+1]$, i.e. for hour h_t. This variable takes the value:

$$P_t^{C,h} = \bar{P}_t^{C,h} + E_t^{C,h}\,, \tag{5.86}$$

where $h = h_t$ is the hour of the day at time t.

We denote by $\tilde{P}_t^B$ the actual charging power that can be applied to the battery considering its limited capacity. Given an action to charge P_t^B, the actual charge $\tilde{P}_t^B$ is constrained by the battery capacity limit for charging the available energy stored in the battery for discharging, according to:

$$\tilde{P}_t^B = \begin{cases} (\overline{SoC} - SoC_t)\,/\,\eta_{ch} & \text{,if } P_t^B > (\overline{SoC} - SoC_t)\,/\,\eta_{ch} \\ -(SoC_t) \cdot \eta_{dis} & \text{,if } P_t^B < -(SoC_t) \cdot \eta_{dis} \\ P_t^B & \text{otherwise .} \end{cases} \tag{5.87}$$

At each time-step t in the simulation horizon, there exists a power balance between the injections and the off-takes. The residual power resulting from the mismatch between production and consumption is curtailed $P_t^{curtail}\,[W] \in \mathbb{R}^+$. Formally the power balance is given by:

$$P_t^{curtail} = \bar{P}_t^{PV,h} - P_t^{C,h} - \tilde{P}_t^B \,. \tag{5.88}$$

If $P_t^{curtail}$ is positive, the excess of generation is simply lost (curtailed). If $P_t^{curtail}$ is negative, there is a lack of generation and a part of the load has to the shed. This is associated with a cost of shedding load $c_t^{shed}[\$] \in \mathbb{R}^+$ equal to:

$$c_t^{shed} = -\min(0, P_t^{curtail}) \cdot \pi^{shed} \,, \tag{5.89}$$

where $\pi^{shed}\,[\$/W] \in \mathbb{R}^+$ corresponds to the penalty per unit of power shed.

Parametrized environment. The off-grid microgrid environment $(\mathscr{S}, \mathscr{A}, \Xi, P_0, f_\psi, \rho_\psi, P_\xi, T)$ will be parametrized by the vector $\psi = (\overline{SoC}, \overline{P^{PV}}) \in \mathbb{R}^{+2}$.

Numerical values. Table 5.2 summarises the parameter values used in the experiments presented in this paper.

TABLE 5.2: Parameters for the solar off-grid microgrid.

Symbol	Value	Unit
η_{ch}, η_{dis}	75	%
σ^C, σ^{PV}	0.01	Wh
p	100	%
Δt	1	hour
c^{PV}	1	$\$/W_p$
c^B	1	$\$/W_p$
r	7	%
n	2	years
π^{shed}	10	$\$/Wh$
T	120	hour

TABLE 5.3: Electrical load consumption and PV production power factor data.

Hour	$\bar{P}^{C,h}$	$\sigma^2_{C,h}$	$\bar{p}^{PV,h}$
0	6.9	0.55	0.
1	6.4	0.50	0.
2	6.1	0.43	0.
3	5.9	0.39	0.
4	5.7	0.39	0.
5	5.4	0.37	0.
6	4.8	0.37	0.
7	4.5	0.36	0.
8	4.6	0.40	0.
9	4.6	0.43	0.04
10	4.7	0.44	0.08
11	4.9	0.47	0.12
12	5.1	0.42	0.14
13	5.3	0.40	0.15
14	5.4	0.42	0.14
15	5.4	0.47	0.12
16	5.4	0.43	0.08
17	5.8	0.44	0.04
18	8.4	0.81	0.
19	10.6	0.60	0.
20	11.0	0.55	0.
21	10.5	0.57	0.
22	9.2	0.60	0.
23	7.8	0.59	0.

Chapter 6

Concluding remarks and future work

In this chapter, we firstly provide a summary of the contributions of this thesis. Subsequently, we provide a list of potential future research directions that derive as natural continuation of the work presented.

6.1 Conclusions

The main goal of this thesis has been to investigate the potential of deep reinforcement learning (DRL) in solving complex problems related to the control of storage devices in modern energy systems aiming at maximizing the value they can provide by performing arbitrage.

In Chapter 2 of this thesis, we address the energy arbitrage problem of a storage unit that participates in the European Continuous Intraday (CID) market. To that end, we develop a novel modeling framework where exchanges (energy and financial) occur through a process similar to the stock market. A detailed description of the CID market mechanism and the storage management process is provided. We formulate this problem as a Markov Decision Process MDP, detailing the assumptions that allow for this type of formulation in this particular problem. Furthermore, a set of necessary simplifications that constitute the problem tractable are described. The resulting problem is solved using a state-of-the-art DRL algorithm. The results suggest that the obtained policy is a low-risk policy that is able to outperform, on average, the state-of-the-art for the industry benchmark strategy (*rolling intrinsic*). In particular, we observe improvements of up to 2.2% on unseen data using our algorithm with respect to the *rolling intrinsic*. In this way, the proposed DRL method is shown to increase the arbitrage value for storage units participating in the CID market. The proposed method can serve as a wrapper around the current industrial practices that provides decision support to energy trading activities with low risk. However, the insufficient amount of relevant information contained in the state variable, as well as the limited state space exploration, are identified as key limitations for the performance of the proposed method.

In Chapter 3 of this thesis, we address these limitations related to the state space exploration. We introduce a set of modifications to the described CID market participation problem that lead to a significant increase in the general performance of the proposed strategy. First, we motivate the use of a more compact state space representation and we propose the use of day-ahead prices in order to stationarize the states observed. To that end, we proceed by normalizing the trading rewards in each day, by dividing them with the total profits obtained by the benchmark strategy. The results show that the proposed method yields significant improvements. More precisely, these improvements amount to approximately 19% with respect to the *rolling intrinsic* benchmark. In addition, the proposed modifications allow for better generalization of the fitted Q method in out-of-sample (unseen) data. In addition, the results illustrate that the obtained policy is low-risk, and can outperform on average the state of the art for the industrial *rolling intrinsic* benchmark strategy. In conclusion, it is shown that using the proposed DRL method we were able to obtain a control strategy that can significantly improve the value of storage when performing price arbitrage in the European CID market.

In Chapter 4 of this thesis, we address the energy arbitrage problem faced by an off-grid microgrid operator in the context of rural electrification. In particular, we deal with the lifelong control problem of an isolated microgrid. The main challenges for an effective control policy stem from the various changes that take place over the life span of the microgrid. These changes can be categorized in progressive and abrupt changes. In this work, we propose a novel model-based DRL algorithm that is able to address both types of changes. The algorithm demonstrates generalization properties, transfer capabilities and robustness in case of fast-changing system dynamics. The proposed algorithm is compared against two benchmarks, namely a rule-based and an model predictive controller (MPC). The results show that the trained agent yields approximately a 25% cost reduction in comparison to the rule-based controller, and that its performance is comparable to the upper bound set by an MPC controller. Moreover, the results indicate that, the proposed model-based reinforcement learning method is able to adapt to changes, both gradual and abrupt. Overall, the proposed DRL method succeeds in tackling the key challenges encountered in the lifelong control of an off-grid microgrid for rural electrification. Additionally, the cost reduction achieved by the proposed algorithm mainly implies a reduction in the use of the diesel generator and a higher utilization of RES. This effect subsequently results in an overall reduction of CO_2 emissions and promotes sustainable energy utilization in the context of rural electrification. It can be thus concluded that, DRL is proven to be a highly effective method for maximizing the value of energy arbitrage in an off-grid microgrid context.

Finally in Chapter 5, we propose a new DRL methodology for jointly sizing a dynamical system and designing its control law. First, the problem is formalized by considering parametrized reinforcement learning environments and parametrized policies. The objective of the optimization problem is to jointly find a control policy and an environment over the joint hypothesis space of parameters such that the sum of rewards gathered by the policy in this environment is maximal. The optimization problem is then addressed by generalizing the direct policy search algorithms to an algorithm we call Direct Environment Search with (projected stochastic) Gradient Ascent (DESGA). We illustrate the performance of DESGA on two benchmarks. First, we consider a parametrized space of Mass-Spring-Damper environments and control policies. Then, we use our algorithm for optimizing the size of the components and the operation of a small-scale autonomous energy system, i.e. a solar off-grid microgrid, composed of photovoltaic panels, batteries. Also, on both benchmarks, we compare the results of the execution of DESGA with a theoretical upper-bound on the expected return. On both benchmarks, we show that DESGA results in a set of parameters for which the expected return is nearly equal to its theoretical upper-bound.

6.2 Future work

In this thesis, we have proposed detailed modeling frameworks for two important energy management problems in the context of the Energy Transition. Subsequently, we have solved the developed problems using DRL techniques. However, to obtain a tractable solution, we have performed an intermediate step, that is, we have reduced the problem complexity by decreasing the dimensionality of the action spaces. A key challenge from the practitioner's perspective is to strike the right balance between problem complexity and optimality. In that respect, future work should be directed toward the design of low dimensional continuous action spaces that are not restrictive, i.e. contain the optimal solution of the original problem.

Many of the state-of-the-art reinforcement learning algorithms have demonstrated large success in solving problems that are stationary (such as Atari games). In Chapter 4, we have highlighted the increasing importance and the motivation for addressing problems in which changes occur over time. In order to address these changes, in this thesis, we have proposed a model based algorithm that has demonstrated generalization, robustness and transfer capabilities. Future work should be directed toward creating a more adaptive version of this algorithm, one that is able to track occurring changes and can be automatically retrained. Each new training step should rely on more recent data that better represent the underlying processes without forgetting critical knowledge about the considered system. In this way, we could eventually be able to address the problem of lifelong control in the context of energy management.

Finally, the DESGA algorithm presented in Chapter 5 has demonstrated the potential to jointly optimize a system and its corresponding policy for the case of an isolated microgrid. Future work should be directed toward using this algorithm for optimizing more complex systems. For instance, including in the existing case study other controllable or variable components such as diesel generators or wind turbines would result in a more complex joint environment and policy hypothesis space. Increasing the complexity of the investigated problems is expected to bring new challenges and high potential for improvements for the DESGA algorithm.

Bibliography

[1] NASA. (2021). The effects of climate change, [Online]. Available: `https://climate.nasa.gov/effects/`.

[2] V. Masson-Delmotte, P. Zhai, H.-O. Pörtner, D. Roberts, J. Skea, P. R. Shukla, A. Pirani, W Moufouma-Okia, C Péan, R Pidcock, *et al.*, “Ipcc, 2018: Summary for policymakers”, *Global Warming of 1.5°C. An IPCC Special Report on the impacts of global warming of 1.5°C above pre-industrial levels and related global greenhouse gas emission pathways, in the context of strengthening the global response to the threat of climate change, sustainable development, and efforts to eradicate poverty*, vol. 1, pp. 1–9, 2018.

[3] (2019). Paris agreement, [Online]. Available: `https://treaties.un.org/pages/ViewDetails.aspx?src=TREATY&mtdsg_no=XXVII-7-d&chapter=27&clang=_en` (visited on 03/28/2019).

[4] E. Commission. (2019). Clean energy for all europeans, [Online]. Available: `https://op.europa.eu/en/publication-detail/-/publication/b4e46873-7528-11e9-9f05-01aa75ed71a1`.

[5] ENTSOE and ENTSOG. (2020). Tyndp 2020 scenario report, [Online]. Available: `https://www.entsos-tyndp2020-scenarios.eu/wp-content/uploads/2019/10/TYNDP_2020_Scenario_Report_entsog-entso-e.pdf`.

[6] IRENA, *Global Renewables Outlook: Energy transformation 2050*. 2020, p. 292, ISBN: 9789292602383. [Online]. Available: `https://www.irena.org/publications/2020/Apr/Global-Renewables-Outlook-2020`.

[7] P. Denholm, M. O’Connell, G. Brinkman, and J. Jorgenson, “Overgeneration from solar energy in california. a field guide to the duck chart”, National Renewable Energy Lab.(NREL), Golden, CO (United States), Tech. Rep., 2015.

[8] H. Chandler, *A Guide to the Balancing Challenge*. 2011, pp. 1–234, ISBN: 9789264111387. [Online]. Available: `www.iea.org`.

[9] IEA. (2019). The california duck curve, [Online]. Available: https://www.iea.org/data-and-statistics/charts/the-california-duck-curve.

[10] M. A. Gonzalez-Salazar, T. Kirsten, and L. Prchlik, “Review of the operational flexibility and emissions of gas-and coal-fired power plants in a future with growing renewables”, *Renewable and Sustainable Energy Reviews*, vol. 82, pp. 1497–1513, 2018.

[11] F. Ess and F. Peter, “Flexibility in thermal power plants–with a focus on existing coal-fired power plants”, Agora Energiewende, Tech. Rep., 2017.

[12] ENTSOE. (2019). Imbalance netting, [Online]. Available: https://www.entsoe.eu/network_codes/eb/imbalance-netting/.

[13] W. Cole and A. W. Frazier, “Cost Projections for Utility- Scale Battery Storage Cost Projections for Utility- Scale Battery Storage”, *National Renewable Energy Laboratory*, no. June, NREL/TP–6A20–73222, 2019. [Online]. Available: https://www.nrel.gov/docs/fy19osti/73222.pdf.

[14] N. Günter and A. Marinopoulos, “Energy storage for grid services and applications: Classification, market review, metrics, and methodology for evaluation of deployment cases”, *Journal of Energy Storage*, vol. 8, pp. 226–234, 2016, ISSN: 2352-152X. DOI: https://doi.org/10.1016/j.est.2016.08.011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352152X16301141.

[15] R. Sioshansi, P. Denholm, T. Jenkin, and J. Weiss, “Estimating the value of electricity storage in PJM: Arbitrage and some welfare effects”, *Energy Economics*, vol. 31, no. 2, pp. 269–277, 2009, ISSN: 01409883. DOI: 10.1016/j.eneco.2008.10.005. [Online]. Available: http://dx.doi.org/10.1016/j.eneco.2008.10.005.

[16] D. Zafirakis, K. J. Chalvatzis, G. Baiocchi, and G. Daskalakis, “The value of arbitrage for energy storage: Evidence from European electricity markets”, *Applied Energy*, vol. 184, pp. 971–986, 2016, ISSN: 03062619. DOI: 10.1016/j.apenergy.2016.05.047. [Online]. Available: http://dx.doi.org/10.1016/j.apenergy.2016.05.047.

[17] D. Krishnamurthy, C. Uckun, Z. Zhou, P. R. Thimmapuram, and A. Botterud, “Energy storage arbitrage under day-ahead and real-time price uncertainty”, *IEEE Transactions on Power Systems*, vol. 33, no. 1, pp. 84–93, 2018. DOI: 10.1109/TPWRS.2017.2685347.

[18] X. Xi, R. Sioshansi, and V. Marano, “A stochastic dynamic programming model for co-optimization of distributed energy storage”, *Energy Systems*, vol. 5, no. 3, pp. 475–505, 2014, ISSN: 18683975. DOI: 10.1007/s12667-013-0100-6.

[19] A. Shapiro, W. Tekaya, J. P. da Costa, and M. P. Soares, “Risk neutral and risk averse stochastic dual dynamic programming method”, *European journal of operational research*, vol. 224, no. 2, pp. 375–391, 2013.

[20] N. Löhndorf, D. Wozabal, and S. Minner, “Optimizing trading decisions for hydro storage systems using approximate dual dynamic programming”, *Operations Research*, vol. 61, no. 4, pp. 810–823, 2013.

[21] B. Xu, A. Botterud, and M. Korpås, “Operational valuation for energy storage under multi-stage price uncertainties”, *arXiv*, 2019, ISSN: 23318422. DOI: 10.1109/cdc42340.2020.9304081. arXiv: 1910.09149.

[22] D. R. Jiang and W. B. Powell, “Optimal hour-ahead bidding in the real-time electricity market with battery storage using Approximate Dynamic Programming”, pp. 1–28, 2014, ISSN: 1091-9856. DOI: 10.1287/ijoc.2015.0640. arXiv: 1402.3575. [Online]. Available: http://arxiv.org/abs/1402.3575.

[23] H. Wang and B. Zhang, “Energy storage arbitrage in real-time markets via reinforcement learning”, in *2018 IEEE Power Energy Society General Meeting (PESGM)*, 2018, pp. 1–5. DOI: 10.1109/PESGM.2018.8586321.

[24] A. A. Thatte, L. Xie, D. E. Viassolo, and S. Singh, “Risk measure based robust bidding strategy for arbitrage using a wind farm and energy storage”, *IEEE Transactions on Smart Grid*, vol. 4, no. 4, pp. 2191–2199, 2013. DOI: 10.1109/TSG.2013.2271283.

[25] T. L. Vandoorn, B. Meersman, J. D. De Kooning, and L. Vandevelde, “Transition from islanded to grid-connected mode of microgrids with voltage-based droop control”, *IEEE transactions on power systems*, vol. 28, no. 3, pp. 2545–2553, 2013.

[26] S. Quoilin, K. Kavvadias, A. Mercier, I. Pappone, and A. Zucker, “Quantifying self-consumption linked to solar home battery systems: Statistical analysis and economic assessment”, *Applied Energy*, vol. 182, pp. 58–67, 2016, ISSN: 0306-2619. DOI: https://doi.org/10.1016/j.apenergy.2016.08.077. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306261916311643.

[27] F. Lombardi, S. Balderrama, S. Quoilin, and E. Colombo, “Generating high-resolution multi-energy load profiles for remote areas with an open-source stochastic model”, *Energy*, vol. 177, pp. 433–444, 2019, ISSN: 03605442. DOI: `10.1016/j.energy.2019.04.097`. [Online]. Available: `https://doi.org/10.1016/j.energy.2019.04.097`.

[28] S. Dakir, I. Boukas, V. Lemort, and B. Cornélusse, “Sizing and operation of an isolated microgrid with building thermal dynamics and cold storage”, 5, vol. 56, 2020, pp. 5375–5384. DOI: `10.1109/TIA.2020.3005370`.

[29] N. Stevanato, F. Lombardi, G. Guidicini, L. Rinaldi, S. L. Balderrama, M. Pavičević, S. Quoilin, and E. Colombo, “Long-term sizing of rural microgrids: Accounting for load evolution through multi-step investment plan and stochastic optimization”, *Energy for Sustainable Development*, vol. 58, pp. 16–29, 2020, ISSN: 0973-0826. DOI: `https://doi.org/10.1016/j.esd.2020.07.002`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0973082620302635`.

[30] C. Sun, G. Joos, S. Q. Ali, J. N. Paquin, C. M. Rangel, F. Al Jajeh, I. Novickij, and F. Bouffard, “Design and real-time implementation of a centralized microgrid control system with rule-based dispatch and seamless transition function”, *IEEE Transactions on Industry Applications*, vol. 56, no. 3, pp. 3168–3177, 2020.

[31] M. Jafari, Z. Malekjamshidi, J. Zhu, and M.-H. Khooban, “A novel predictive fuzzy logic-based energy management system for grid-connected and off-grid operation of residential smart microgrids”, *IEEE Journal of Emerging and Selected Topics in Power Electronics*, vol. 8, no. 2, pp. 1391–1404, 2018.

[32] A. Parisio, E. Rikos, G. Tzamalis, and L. Glielmo, “Use of model predictive control for experimental microgrid optimization”, *Applied Energy*, vol. 115, pp. 37–46, 2014, ISSN: 03062619. DOI: `10.1016/j.apenergy.2013.10.027`. [Online]. Available: `http://dx.doi.org/10.1016/j.apenergy.2013.10.027`.

[33] J. Sachs and O. Sawodny, “A two-stage model predictive control strategy for economic diesel-pv-battery island microgrid operation in rural areas”, *IEEE Transactions on Sustainable Energy*, vol. 7, no. 3, pp. 903–913, 2016. DOI: `10.1109/TSTE.2015.2509031`.

[34] A. Kanwar, D. I. H. Rodríguez, J. von Appen, and M. Braun, *A comparative study of optimization-and rule-based control for microgrid operation*. Universitatsbibliothek Dortmund, 2015.

[35] A. Parisio and L. Glielmo, “Stochastic model predictive control for economic/environmental operation management of microgrids”, in *2013 European Control Conference (ECC)*, 2013, pp. 2014–2019. DOI: `10.23919/ECC.2013.6669807`.

[36] G. Bruni, S. Cordiner, V. Mulone, V. Sinisi, and F. Spagnolo, “Energy management in a domestic microgrid by means of model predictive controllers”, *Energy*, vol. 108, pp. 119–131, 2016, Sustainable Energy and Environmental Protection 2014, ISSN: 0360-5442. DOI: `https://doi.org/10.1016/j.energy.2015.08.004`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0360544215010488`.

[37] E. Kuznetsova, C. Ruiz, Y.-F. Li, and E. Zio, “Analysis of robust optimization for decentralized microgrid energy management under uncertainty”, *International Journal of Electrical Power & Energy Systems*, vol. 64, pp. 815–832, 2015.

[38] L. Busoniu, R. Babuska, B. D. Schutter, and D. Ernst, *Reinforcement Learning and Dynamic Programming Using Function Approximators*, 1st. USA: CRC Press, Inc., 2010, ISBN: 1439821089.

[39] R. S. Sutton, A. G. Barto, *et al.*, *Introduction to reinforcement learning*. MIT press Cambridge, 1998, vol. 135.

[40] D. P. Bertsekas, *Dynamic programming and optimal control*, 3. Athena scientific Belmont, MA, 2005, vol. 1.

[41] ——, “Approximate dynamic programming”, 2008.

[42] W. B. Powell, “A unified framework for stochastic optimization”, *European Journal of Operational Research*, vol. 275, no. 3, pp. 795–821, 2019.

[43] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, “Human-level control through deep reinforcement learning”, *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015, ISSN: 00280836. [Online]. Available: `http://dx.doi.org/10.1038/nature14236`.

[44] H. van Hasselt, A. Guez, and D. Silver, *Deep reinforcement learning with double q-learning*, 2015. arXiv: `1509.06461 [cs.LG]`.

[45] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, “Prioritized experience replay”, *arXiv preprint arXiv:1511.05952*, 2015.

[46] Z. Wang, T. Schaul, M. Hessel, H. van Hasselt, M. Lanctot, and N. de Freitas, *Dueling network architectures for deep reinforcement learning*, 2016. arXiv: `1511.06581 [cs.LG]`.

[47] I. Osband, C. Blundell, A. Pritzel, and B. Van Roy, "Deep exploration via bootstrapped dqn", *arXiv preprint arXiv:1602.04621*, 2016.

[48] H. van Hasselt, A. Guez, M. Hessel, V. Mnih, and D. Silver, "Learning values across many orders of magnitude", *arXiv preprint arXiv:1602.07714*, 2016.

[49] A. P. Badia, B. Piot, S. Kapturowski, P. Sprechmann, A. Vitvitskyi, D. Guo, and C. Blundell, *Agent57: Outperforming the atari human benchmark*, 2020. arXiv: `2003.13350 [cs.LG]`.

[50] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning", in *International conference on machine learning*, 2016, pp. 1928–1937.

[51] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *et al.*, "Mastering the game of go with deep neural networks and tree search", *nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[52] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan, and D. Hassabis, "A general reinforcement learning algorithm that masters chess, shogi, and go through self-play", *Science*, vol. 362, no. 6419, pp. 1140–1144, 2018, ISSN: 0036-8075. DOI: `10.1126/science.aar6404`. eprint: `https://science.sciencemag.org/content/362/6419/1140.full.pdf`. [Online]. Available: `https://science.sciencemag.org/content/362/6419/1140`.

[53] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, *et al.*, "Mastering atari, go, chess and shogi by planning with a learned model", *Nature*, vol. 588, no. 7839, pp. 604–609, 2020.

[54] S. L. Brunton, B. R. Noack, and P. Koumoutsakos, "Machine learning for fluid mechanics", *Annual Review of Fluid Mechanics*, vol. 52, pp. 477–508, 2020.

[55] G. Novati, H. L. de Laroussilhe, and P. Koumoutsakos, "Automating turbulence modelling by multi-agent reinforcement learning", *Nature Machine Intelligence*, vol. 3, no. 1, pp. 87–96, 2021.

[56] H. Nguyen and H. La, "Review of deep reinforcement learning for robot manipulation", in *2019 Third IEEE International Conference on Robotic Computing (IRC)*, IEEE, 2019, pp. 590–595.

[57] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates", in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 3389–3396. DOI: 10.1109/ICRA.2017.7989385.

[58] D. Büchler, S. Guist, R. Calandra, V. Berenz, B. Schölkopf, and J. Peters, *Learning to play table tennis from scratch using muscular robots*, 2020. arXiv: 2006.05935 [cs.RO].

[59] C. Yu, J. Liu, and S. Nemati, *Reinforcement learning in healthcare: A survey*, 2020. arXiv: 1908.08796 [cs.LG].

[60] A. Jonsson, "Deep reinforcement learning in medicine", *Kidney Diseases*, vol. 5, no. 1, pp. 18–22, 2019.

[61] N. Liu, Y. Liu, B. Logan, Z. Xu, J. Tang, and Y. Wang, *Deep reinforcement learning for dynamic treatment regimes on medical registry data*, 2018. arXiv: 1801.09271 [cs.AI].

[62] A. Raghu, M. Komorowski, I. Ahmed, L. Celi, P. Szolovits, and M. Ghassemi, "Deep reinforcement learning for sepsis treatment", *arXiv preprint arXiv:1711.09602*, 2017.

[63] A. Marot, B. Donnot, G. Dulac-Arnold, A. Kelly, A. O'Sullivan, J. Viebahn, M. Awad, I. Guyon, P. Panciatici, and C. Romero, *Learning to run a power network challenge: A retrospective analysis*, 2021. arXiv: 2103.03104 [cs.LG].

[64] I. Pérez Arriaga, C. Knittel, C. Batle, T. Gómez, J. Chaves, P. Rodilla, I. Herrero, P. Dueñas, C. Vergara Ramírez, A. Bharatkumar, S. Burger, S. Hungtinton, J. D. Denkins, M. Luke, R. Miller, R. Tabors, and N. et al. Xu, *Envisioning a Future with Distributed Energy Resources*. 2016, p. 382, ISBN: 9780692808245. [Online]. Available: energy.mit.edu/uof.

[65] D. Ernst, M. Glavic, F. Capitanescu, and L. Wehenkel, "Reinforcement learning versus model predictive control: A comparison on a power system problem", *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 2, pp. 517–529, 2008.

[66] R. J. Williams, “Simple statistical gradient-following algorithms for connectionist reinforcement learning”, *Machine learning*, vol. 8, no. 3-4, pp. 229–256, 1992.

[67] I. Boukas, D. Ernst, T. Théate, A. Bolland, A. Huynen, M. Buchwald, C. Wynants, and B. Cornélusse, *A deep reinforcement learning framework for continuous intraday market bidding*, 2020. arXiv: `2004.05940 [q-fin.TR]`.

[68] S. Totaro, I. Boukas, A. Jonsson, and B. Cornélusse, *Lifelong control of off-grid microgrid with model based reinforcement learning*, 2020. arXiv: `2005.08006 [eess.SY]`.

[69] A. Bolland, I. Boukas, F. Cornet, M. Berger, and D. Ernst, *Learning optimal environments using projected stochastic gradient ascent*, 2020. arXiv: `2006.01738 [cs.LG]`.

[70] I. Boukas, D. Ernst, A. Papavasiliou, and B. Cornélusse, “Intra-day bidding strategies for storage devices using deep reinforcement learning”, in *International Conference on the European Energy Market, Łódź 27-29 June 2018*, 2018, p. 6.

[71] I. Boukas, D. Ernst, and B. Cornélusse, “Real-time bidding strategies from micro-grids using reinforcement learning”, in *Proceedings of CIRED Workshop 2018*, 2018.

[72] J. Dumas, I. Boukas, M. M. de Villena, S. Mathieu, and B. Cornélusse, “Probabilistic forecasting of imbalance prices in the belgian context”, in *2019 16th International Conference on the European Energy Market (EEM)*, IEEE, 2019, pp. 1–7.

[73] S. Dakir, I. Boukas, V. Lemort, and B. Cornélusse, “Sizing and operation of an isolated microgrid with cold storage”, in *2019 IEEE Milan PowerTech*, 2019, pp. 1–6. DOI: `10.1109/PTC.2019.8810700`.

[74] M. M. de Villena, I. Boukas, S. Mathieu, E. Vermeulen, and D. Ernst, “A framework to integrate flexibility bids into energy communities to improve self-consumption”, in *2020 IEEE Power Energy Society General Meeting (PESGM)*, 2020, pp. 1–5. DOI: `10.1109/PESGM41954.2020.9282036`.

[75] The European Commission. (2017). 2030 energy strategy, [Online]. Available: `https://ec.europa.eu/energy/en/topics/energy-strategy-and-energy-union/2030-energy-strategy`.

[76] B. L. Meeus and T. Schittekatte, “The EU Electricity Network Codes: Course text for the Florence School of Regulation online course”, no. October, 2017.

[77] R. Scharff and M. Amelin, "Trading behaviour on the continuous intraday market Elbas", *Energy Policy*, vol. 88, pp. 544–557, 2016, ISSN: 03014215. DOI: 10.1016/j.enpol.2015.10.045.

[78] F. Karanfil and Y. Li, "The role of continuous intraday electricity markets: The integration of large-share wind power generation in Denmark", *Energy Journal*, vol. 38, no. 2, pp. 107–130, 2017, ISSN: 01956574. DOI: 10.5547/01956574.38.2.fkar.

[79] F. Borggrefe and K. Neuhoff, "Balancing and intraday market design: Options for wind integration", eng, Berlin, DIW Discussion Papers 1162, 2011. [Online]. Available: http://hdl.handle.net/10419/61319.

[80] K. Neuhoff, N. Ritter, A. Salah-Abou-El-Enien, and P. Vassilopoulos, "Intraday markets for power: discretizing the continuous trading?", 2016.

[81] S. Hagemann, "Price determinants in the German intraday market for electricity: An empirical analysis", *Journal of Energy Markets*, 2015.

[82] A. Henriot, "Market design with centralized wind power management: Handling low-predictability in intraday markets", *Energy Journal*, vol. 35, no. 1, pp. 99–117, 2014, ISSN: 01956574. DOI: 10.5547/01956574.35.1.6.

[83] C. Balardy, "German continuous intraday market: Orders book's behavior over the trading session", in *Meeting the Energy Demands of Emerging Economies, 40th IAEE International Conference, June 18-21, 2017*, International Association for Energy Economics, 2017.

[84] N. Spot. (2018). Xbid cross-border intra day market project, [Online]. Available: https://www.nordpoolspot.com/globalassets/download-center/xbid/xbid-qa_final.pdf (visited on 10/19/2018).

[85] A. Baillo, M. Ventosa, M. Rivier, and A. Ramos, "Optimal offering strategies for generation companies operating in electricity spot markets", *IEEE Transactions on Power Systems*, vol. 19, no. 2, pp. 745–753, 2004, ISSN: 0885-8950. DOI: 10.1109/TPWRS.2003.821429.

[86] M. A. Plazas, A. J. Conejo, and F. J. Prieto, "Multimarket optimal bidding for a power producer", *IEEE Transactions on Power Systems*, vol. 20, no. 4, pp. 2041–2050, 2005, ISSN: 0885-8950. DOI: 10.1109/TPWRS.2005.856987.

[87] S. E. Fleten and T. K. Kristoffersen, “Stochastic programming for optimizing bidding strategies of a Nordic hydropower producer”, *European Journal of Operational Research*, vol. 181, no. 2, pp. 916 –928, 2007, ISSN: 0377-2217. DOI: http://dx.doi.org/10.1016/j.ejor.2006.08.023.

[88] T. K. Boomsma, N. Juul, and S.-E. Fleten, “Bidding in sequential electricity markets: The Nordic case”, *European Journal of Operational Research*, vol. 238, no. 3, pp. 797 –809, 2014, ISSN: 0377-2217. DOI: http://dx.doi.org/10.1016/j.ejor.2014.04.027. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0377221714003695.

[89] H. PandŽić, J. M. Morales, A. J. Conejo, and I. Kuzle, “Offering model for a virtual power plant based on stochastic programming”, *Applied Energy*, vol. 105, pp. 282–292, 2013, ISSN: 03062619. DOI: 10.1016/j.apenergy.2012.12.077. arXiv: 0611061v2 [arXiv:quant-ph].

[90] E. Garnier and R. Madlener, “Balancing forecast errors in continuous-trade intraday markets”, *Energy Systems*, vol. 6, no. 3, pp. 361–388, 2015.

[91] J. Gönsch and M. Hassler, “Sell or store? An ADP approach to marketing renewable energy”, *OR Spectrum*, vol. 38, no. 3, pp. 633–660, 2016, ISSN: 14366304. DOI: 10.1007/s00291-016-0439-x.

[92] M. Hassler, “Heuristic decision rules for short-term trading of renewable energy with co-located energy storage”, *Computers and Operations Research*, vol. 83, 2017, ISSN: 03050548. DOI: 10.1016/j.cor.2016.12.027.

[93] R. Aïd, P. Gruet, and H. Pham, “An optimal trading problem in intraday electricity markets”, *Mathematics and Financial Economics*, vol. 10, no. 1, pp. 49–85, 2016.

[94] C. Balardy, “An analysis of the bid-ask spread in the german power continuous market”, in *Heading Towards Sustainable Energy Systems: Evolution or Revolution?, 15th IAEE European Conference, Sept 3-6, 2017*, International Association for Energy Economics, 2017.

[95] G. Bertrand and A. Papavasiliou, “Adaptive trading in continuous intraday electricity markets for a storage unit”, *IEEE Transactions on Power Systems*, pp. 1–1, 2019, ISSN: 1558-0679. DOI: 10.1109/TPWRS.2019.2957246.

[96] J. Gray and P. Khandelwal, “Towards a realistic gas storage model”, *Commodities Now*, vol. 7, no. 2, pp. 1–4, 2004.

[97] D. Ernst, P. Geurts, and L. Wehenkel, "Tree-based batch mode reinforcement learning", *Journal of Machine Learning Research*, vol. 6, no. Apr, pp. 503–556, 2005.

[98] EPEXSPOT. (2017). Market data intraday continuous, [Online]. Available: `http://www.epexspot.com/en/market-data/intradaycontinuous`.

[99] EPEXSPOT. (2019). EPEXSPOT Operational rules, [Online]. Available: `https://www.epexspot.com/document/40170/EPEX%20SPOT%20Market%20Rules`.

[100] H. L. Le, V. Ilea, and C. Bovo, "Integrated European intra-day electricity market: Rules, modeling and analysis", *Applied Energy*, vol. 238, no. June 2018, pp. 258–273, 2019, ISSN: 03062619. DOI: `10.1016/j.apenergy.2018.12.073`. [Online]. Available: `https://doi.org/10.1016/j.apenergy.2018.12.073`.

[101] M. L. Littman, "Markov games as a framework for multi-agent reinforcement learning", in *Machine learning proceedings 1994*, Elsevier, 1994, pp. 157–163.

[102] S. Braun and R. Hoffmann, "Intraday Optimization of Pumped Hydro Power Plants in the German Electricity Market", in *Energy Procedia*, vol. 87, 2016, pp. 45–52. DOI: `10.1016/j.egypro.2015.12.356`.

[103] C. J. Watkins and P. Dayan, "Q-learning", *Machine learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[104] L. Busoniu, R. Babuska, B. De Schutter, and D. Ernst, *Reinforcement learning and dynamic programming using function approximators*. CRC press, 2017.

[105] N Lohndorf and D. Wozabal, "Optimal gas storage valuation and futures trading under a high-dimensional price process", Technical report, Tech. Rep., 2015.

[106] N. Löhndorf and D. Wozabal, "Gas storage valuation in incomplete markets", *Eur. J. Oper. Res.*, vol. 288, pp. 318–330, 2020.

[107] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, `http://www.deeplearningbook.org`.

[108] D. Horgan, J. Quan, D. Budden, G. Barth-Maron, M. Hessel, H. Van Hasselt, and D. Silver, "Distributed prioritized experience replay", *arXiv preprint arXiv:1803.00933*, 2018.

[109] A. Dolara, S. Leva, and G. Manzolini, "Comparison of different physical models for PV power output prediction", *Solar Energy*, vol. 119, pp. 83–99, 2015, ISSN: 0038092X. DOI: `10.1016/j.solener.2015.06.017`. [Online]. Available: `http://dx.doi.org/10.1016/j.solener.2015.06.017`.

[110] Y. Zhang, L. Fu, W. Zhu, X. Bao, and C. Liu, “Robust model predictive control for optimal energy management of island microgrids with uncertainties”, *Energy*, vol. 164, pp. 1229–1241, 2018, ISSN: 0360-5442. DOI: `https://doi.org/10.1016/j.energy.2018.08.200`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0360544218317365`.

[111] I. Boukas, D. Ernst, A. Papavasiliou, and B. Cornelusse, “Intra-day bidding strategies for storage devices using Deep Reinforcement Learning”, vol. 14, no. October, 2018.

[112] R. S. Sutton, C. Szepesvári, A. Geramifard, and M. P. Bowling, “Dyna-style planning with linear function approximation and prioritized sweeping”, *arXiv preprint arXiv:1206.3285*, 2012.

[113] D. Lee and W. B. Powell, “An intelligent battery controller using bias-corrected q-learning.”, 2012.

[114] C. J. Watkins and P. Dayan, “Q-learning”, *Machine learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[115] E. Kuznetsova, Y. F. Li, C. Ruiz, E. Zio, G. Ault, and K. Bell, “Reinforcement learning for microgrid energy management”, *Energy*, vol. 59, pp. 133–146, 2013, ISSN: 03605442. DOI: `10.1016/j.energy.2013.05.060`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0360544213004817`.

[116] E. Foruzan, L.-K. Soh, and S. Asgarpoor, “Reinforcement learning approach for optimal distributed energy management in a microgrid”, *IEEE Transactions on Power Systems*, vol. 33, no. 5, pp. 5749–5758, 2018.

[117] P. Kofinas, A. Dounis, and G. Vouros, “Fuzzy q-learning for multi-agent decentralized energy management in microgrids”, *Applied Energy*, vol. 219, pp. 53–67, 2018, ISSN: 0306-2619. DOI: `https://doi.org/10.1016/j.apenergy.2018.03.017`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0306261918303465`.

[118] R. Rocchetta, L. Bellani, M. Compare, E. Zio, and E. Patelli, “A reinforcement learning framework for optimal operation and maintenance of power grids”, *Applied Energy*, vol. 241, pp. 291–301, 2019, ISSN: 0306-2619. DOI: `https://doi.org/10.1016/j.apenergy.2019.03.027`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0306261919304222`.

[119] V. François-Lavet, D. Taralla, D. Ernst, and R. Fonteneau, "Deep reinforcement learning solutions for energy microgrids management", in *European Workshop on Reinforcement Learning*, 2016.

[120] M. Lopes, T. Lang, M. Toussaint, and P.-Y. Oudeyer, "Exploration in model-based reinforcement learning by empirically estimating learning progress", in *Neural Information Processing Systems (NIPS)*, 2012.

[121] L. Kaiser, M. Babaeizadeh, P. Milos, B. Osinski, R. H. Campbell, K. Czechowski, D. Erhan, C. Finn, P. Kozakowski, S. Levine, *et al.*, "Model-based reinforcement learning for atari", *arXiv preprint arXiv:1903.00374*, 2019.

[122] A. S. Polydoros and L. Nalpantidis, "Survey of model-based reinforcement learning: Applications on robotics", *Journal of Intelligent & Robotic Systems*, vol. 86, no. 2, pp. 153–173, 2017.

[123] H. Shuai, H. He, and J. Wen, "On-line scheduling of a residential microgrid via monte-carlo tree search and a learned model", *arXiv preprint arXiv:2005.06161*, 2020.

[124] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, T. Lillicrap, and D. Silver, *Mastering atari, go, chess and shogi by planning with a learned model*, 2019. arXiv: `1911.08265 [cs.LG]`.

[125] B. E. Nyong-Bassey, D. Giaouris, C. Patsios, S. Papadopoulou, A. I. Papadopoulos, S. Walker, S. Voutetakis, P. Seferlis, and S. Gadoue, "Reinforcement learning based adaptive power pinch analysis for energy management of stand-alone hybrid energy storage systems considering uncertainty", *Energy*, vol. 193, p. 116 622, 2020, ISSN: 0360-5442. DOI: `https://doi.org/10.1016/j.energy.2019.116622`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0360544219323175`.

[126] R. S. Sutton, "Integrated architectures for learning, planning, and reacting based on approximating dynamic programming", in *Machine Learning Proceedings 1990*, B. Porter and R. Mooney, Eds., San Francisco (CA): Morgan Kaufmann, 1990, pp. 216–224, ISBN: 978-1-55860-141-3. DOI: `https://doi.org/10.1016/B978-1-55860-141-3.50030-4`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/B9781558601413500304`.

[127] A. Tsymbal, “The problem of concept drift: Definitions and related work”, *Computer Science Department, Trinity College Dublin*, vol. 106, no. 2, p. 58, 2004.

[128] A. Kuh, T. Petsche, and R. L. Rivest, “Learning time-varying concepts”, in *Advances in Neural Information Processing Systems*, 1991, pp. 183–189.

[129] A. Nilim and L. El Ghaoui, “Robust control of markov decision processes with uncertain transition matrices”, *Operations Research*, vol. 53, no. 5, pp. 780–798, 2005.

[130] S. H. Lim, H. Xu, and S. Mannor, “Reinforcement learning in robust markov decision processes”, in *Advances in Neural Information Processing Systems*, 2013, pp. 701–709.

[131] P. Gajane, R. Ortner, and P. Auer, “A sliding-window approach for reinforcement learning in mdps with arbitrarily changing rewards and transitions”, in *Proceedings of the 2nd ICML/IJCAI Workshop on Lifelong Learning: A Reinforcement Learning Approach (LLARLA 2018)*, 2018.

[132] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal Policy Optimization Algorithms”, pp. 1–12, 2017. arXiv: 1707.06347. [Online]. Available: http://arxiv.org/abs/1707.06347.

[133] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, “Policy gradient methods for reinforcement learning with function approximation”, in *Advances in neural information processing systems*, 2000, pp. 1057–1063.

[134] M. G. Bellemare, W. Dabney, and R. Munos, “A distributional perspective on reinforcement learning”, in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, JMLR. org, 2017, pp. 449–458.

[135] W. Dabney, M. Rowland, M. G. Bellemare, and R. Munos, “Distributional reinforcement learning with quantile regression”, in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[136] E. Imani and M. White, “Improving regression performance with distributional losses”, *arXiv preprint arXiv:1806.04613*, 2018.

[137] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, “OpenAI gym”, *arXiv preprint arXiv:1606.01540*, 2016.

[138] G. Dulac-Arnold, D. Mankowitz, and T. Hester, “Challenges of real-world reinforcement learning”, *arXiv preprint arXiv:1904.12901*, 2019.

[139] A. Nagabandi, G. Kahn, R. S. Fearing, and S. Levine, “Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning”, in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2018, pp. 7559–7566.

[140] H. P. van Hasselt, M. Hessel, and J. Aslanides, “When to use parametric models in reinforcement learning?”, in *Advances in Neural Information Processing Systems*, 2019, pp. 14 322–14 333.

[141] M. Janner, J. Fu, M. Zhang, and S. Levine, “When to trust your model: Model-based policy optimization”, in *Advances in Neural Information Processing Systems*, 2019, pp. 12 498–12 509.

[142] J. Schulman, S. Levine, P. Moritz, M. Jordan, and P. Abbeel, “Trust Region Policy Optimization”, in *ICML 2015-Proceedings of the 31st International Conference on Machine Learning*, 2015.

[143] S. Balderrama, F. Lombardi, F. Riva, W. Canedo, E. Colombo, and S. Quoilin, “A two-stage linear programming optimization framework for isolated hybrid microgrids in a rural context: The case study of the “El Espino” community”, *Energy*, vol. 188, p. 116 073, 2019, ISSN: 03605442. DOI: 10.1016/j.energy.2019.116073. [Online]. Available: https://doi.org/10.1016/j.energy.2019.116073.

[144] S. J. Taylor and B. Letham, “Forecasting at scale”, *The American Statistician*, vol. 72, no. 1, pp. 37–45, 2018.

[145] C. Castejón, G. Carbone, J. García Prada, and M Ceccarelli, “A multi-objective optimization of a robotic arm for service tasks.”, *Strojniski Vestnik/Journal of Mechanical Engineering*, vol. 56, no. 5, 2010.

[146] S. A. Ajwad, J. Iqbal, R. U. Islam, A. Alsheikhy, A. Almeshal, and A. Mehmood, “Optimal and robust control of multi dof robotic manipulator: Design and hardware realization”, *Cybernetics and Systems*, vol. 49, no. 1, pp. 77–93, 2018.

[147] V. François-Lavet, Q. Gemine, D. Ernst, and R. Fonteneau, “Towards the minimization of the levelized energy costs of microgrids using both long-term and short-term storage devices”, *Smart Grid: Networking, Data Management, and Business Models*, pp. 295–319, 2016.

[148] T. K. Brekken, A. Yokochi, A. Von Jouanne, Z. Z. Yen, H. M. Hapke, and D. A. Halamay, “Optimal energy storage sizing and control for wind power applications”, *IEEE Transactions on Sustainable Energy*, vol. 2, no. 1, pp. 69–77, 2010.

[149] H. Bakker, F. Dunke, and S. Nickel, “A structuring review on multi-stage optimization under uncertainty: Aligning concepts from theory and practice”, *Omega*, vol. 96, p. 102 080, 2020.

[150] S. W. Wallace and S.-E. Fleten, “Stochastic programming models in energy”, *Handbooks in operations research and management science*, vol. 10, pp. 637–677, 2003.

[151] J. R. Birge and F. Louveaux, *Introduction to stochastic programming*. Springer Science & Business Media, 2011.

[152] H. Heitsch and W. Roemisch, “Scenario tree modelling for multistage stochastic programs”, *Mathematical Programming*, vol. 118, pp. 371–406, 2009.

[153] A. S. Nemirovsky and D. B. Yudin, “Problem complexity and method efficiency in optimization.”, 1983.

[154] V. Goel and I. E. Grossmann, “A class of stochastic programs with decision dependent uncertainty”, *Mathematical programming*, vol. 108, no. 2-3, pp. 355–394, 2006.

[155] M. Marufuzzaman, S. D. Eksioglu, and Y. (Eric) Huang, “Two-stage stochastic programming supply chain model for biodiesel production via wastewater treatment”, *Computers & Operations Research*, vol. 49, pp. 1 –17, 2014, ISSN: 0305-0548. DOI: `https://doi.org/10.1016/j.cor.2014.03.010`.

[156] A. Schwele, J. Kazempour, and P. Pinson, “Do unit commitment constraints affect generation expansion planning? a scalable stochastic model”, *Energy Systems*, vol. 11, no. 2, pp. 247–282, 2020.

[157] E. F. Camacho and C. B. Alba, *Model predictive control*. Springer Science & Business Media, 2013.

[158] B. Zhao, X. Zhang, P. Li, K. Wang, M. Xue, and C. Wang, “Optimal sizing, operating strategy and operational experience of a stand-alone microgrid on dongfushan island”, *Applied Energy*, vol. 113, pp. 1656–1666, 2014.

[159] Y. Zhang, A. Lundblad, P. E. Campana, F Benavente, and J. Yan, “Battery sizing and rule-based operation of grid-connected photovoltaic-battery system: A case study in sweden”, *Energy conversion and management*, vol. 133, pp. 249–263, 2017.

[160] S. Dakir, I. Boukas, V. Lemort, and B. Cornélusse, "Sizing and operation of an isolated microgrid with building thermal dynamics and cold storage", *IEEE Transactions on Industry Applications*, 2020.

[161] A. Baniasadi, D. Habibi, W. Al-Saedi, M. A. Masoum, C. K. Das, and N. Mousavi, "Optimal sizing design and operation of electrical and thermal energy storage systems in smart buildings", *Journal of Energy Storage*, vol. 28, p. 101 186, 2020.

[162] K. G. Jamieson, R. Nowak, and B. Recht, "Query complexity of derivative-free optimization", *Advances in Neural Information Processing Systems*, vol. 25, pp. 2672–2680, 2012.

[163] P. S. Oliveto and C. Witt, "Improved time complexity analysis of the simple genetic algorithm", *Theoretical Computer Science*, vol. 605, pp. 21–41, 2015.

[164] L. Bottou, "Large-scale machine learning with stochastic gradient descent", in *Proceedings of COMPSTAT'2010*, Springer, 2010, pp. 177–186.

[165] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey", *Journal of artificial intelligence research*, vol. 4, pp. 237–285, 1996.

[166] N. Hansen and A. Ostermeier, "Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation", in *Proceedings of IEEE International Conference on Evolutionary Computation*, 1996, pp. 312–317.

[167] I. Szita and A. Lörincz, "Learning tetris using the noisy cross-entropy method", *Neural computation*, vol. 18, no. 12, pp. 2936–2941, 2006.

[168] L. Buşoniu, D. Ernst, B. De Schutter, and R. Babuška, "Cross-entropy optimization of control policies with adaptive basis functions", *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 41, no. 1, pp. 196–209, 2011.

[169] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization", in *International conference on machine learning*, 2015, pp. 1889–1897.

[170] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization", *arXiv preprint arXiv:1412.6980*, 2014.

[171] I. Grondman, L. Busoniu, G. A. Lopes, and R. Babuska, "A survey of actor-critic reinforcement learning: Standard and natural policy gradients", *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 1291–1307, 2012.

[172] T. M. Moerland, J. Broekens, and C. M. Jonker, “Model-based reinforcement learning: A survey”, *arXiv preprint arXiv:2006.16712*, 2020.

[173] I. V. Serban, C. Sankar, M. Pieper, J. Pineau, and Y. Bengio, “The bottleneck simulator: A model-based deep reinforcement learning approach”, *Journal of Artificial Intelligence Research*, vol. 69, pp. 571–612, 2020.

[174] S. Bechtle, Y. Lin, A. Rai, L. Righetti, and F. Meier, “Curious ilqr: Resolving uncertainty in model-based rl”, in *Conference on Robot Learning*, PMLR, 2020, pp. 162–171.

[175] G. Wu, B. Say, and S. Sanner, “Scalable planning with deep neural network learned transition models”, *Journal of Artificial Intelligence Research*, vol. 68, pp. 571–606, 2020.

[176] K. Cohen, A. Nedic, and R. Srikant, “On projected stochastic gradient descent algorithm with weighted averaging for least squares regression”, *arXiv preprint arXiv:1606.03000*, 2016.