Balancing Imbalances

On using reinforcement learning to increase stability in smart electricity grids

Marten Schutten

22nd June 2016

Master's thesis
Artificial Intelligence
Department of Artificial Intelligence
University of Groningen, The Netherlands

Internal supervisor:
Dr. M.A. Wiering (Artificial Intelligence, University of Groningen)

External supervisor:
P. MacDougall, PhD (Smart Electricity Grids, TNO)

university of groningen / faculty of mathematics and natural sciences / artificial intelligence

Abstract

Over the years interest in renewable energy sources is growing and the amount of electricity provided by such resources, such as photovoltaic (PV) or wind energy is increasing. This results in a number of changes in the way that electricity is both produced and consumed in society. The supply of renewable energy sources is highly dependent on environmental changes and therefore hard to predict and adjust. As a response new methods have been proposed to shift this control to the demand side of the electricity grid by the means of Demand Side Management. Furthermore it can be seen that there is a shift from a situation in which the supply of electricity is managed by a small group of very large suppliers to a larger group of smaller suppliers. This, among others because of the increase of for example wind farms and the opportunity for residentials to generate their own power (by, for example, installing PV panels).

One of the responsibilities of the larger producers is to maintain the balance between demand and supply across the electricity grid. This means that these parties should be able to ramp up or ramp down their production whenever the demand is higher or lower than expected in different regions. These adjustments happen on the so-called balancing market. Due to minimum production rates these markets are only open to larger suppliers and not to the group of smaller prosumers (parties that both consume and produce electricity). However, as this group becomes larger this group would be a great addition to the electricity markets when it comes to maintaining stability between the supply and demand in the electricity grid.

One solution to add these smaller parties to the electricity grid is to bundle groups of small prosumers (a party that both produces and consumes electricity) into a cluster, controlled by a so-called aggregator. The aggregator can then offer a certain range of power within which it is able to ramp the amount of consumption (or production) up or down. However, ramping up or down too much at a given moment, might result in an imbalance in the opposite direction at a later point in time. This means that any imbalances that might be restored by the aggregator initially might come back even more extreme in the near future.

In this thesis it is shown how reinforcement learning can be applied to successfully learn the boundary conditions within which an aggregator can safely ramp up or down its consumption. Furthermore Neural Fitted CACLA (NFCACLA) is proposed as a new reinforcement learning algorithm, which is a neural fitted variant on the existing CACLA algorithm.

ACKNOWLEDGEMENTS

I would like to thank both of my supervisors, Marco and Pamela, for the guidance they have provided me with during the course of this project. Especially for providing me with the space and freedom they allowed me to determine the course of the project, while steering me into the right directions.

Secondly I would like to thank TNO for providing me with the opportunity for this project, and especially to Wilco Wijbrandi and Johan van der Geest who were always ready to help me out, whenever they had some time left and when they did not.

Finally and most importantly I would like to thank my parents for enabling me to pursue the degree of my choice and for always supporting me in whatever way they were able to.

Contents

Chapter 1

Introduction

Over the years, focus on sustainable energy solutions has been increased. By an increase in the use of Renewable Energy Resources (RES) such as photo voltaic (solar) and wind energy, the supply of electricity becomes more variable and harder to control and predict. On the other hand, devices are being developed that consume electricity in a more flexible way. Traditionally energy is supplied by a limited number of suppliers to a wide range of consumers; however, the amount of so-called *prosumers* [60, 37], individuals that not only consume but also produce electricity, is increasing. As a result the supply of energy becomes increasingly distributed and is transferred slowly from larger parties to smaller groups of individuals.

The current electricity market setup consists of a number of liberalized markets that function on different time scales at which suppliers can offer to sell their electricity [19]. Generally these markets are only open to large-scale producers of electricity. This means that the ever growing group of smaller producers of energy are not able to participate in this market on an individual level. As a result different market adaptations have been proposed that allow these smaller parties to participate in the electricity market on an individual level [76, 49, 11]. Additionally, the addition of aggregators, or *Virtual Power Plants* (VPPs), [9, 46] introduces a method for a group of smaller producers to bundle their resources in order to participate in the electricity market as a cluster of small suppliers or prosumers.

However, it remains a challenge to determine the extent to which it is desired for a (group of) prosumer(s) to participate in these markets. For example, when a prosumer produces a large amount of electricity, he (or she) might be willing to trade it. However, he might have a larger demand in the near future, or maybe his supply will diminish due to weather influences. When this is the case it might not be beneficial to sell your electricity since you might have to buy it back, possibly even at a greater cost than the profit that was made initially.

In this master's thesis we will take on the point of view of an aggregator, trying to participate in electricity markets on behalf of a cluster of small individual prosumers. The aim is to explore the suitability of Reinforcement Learning (RL) techniques [68, 79] to learn the boundaries within which an aggregator can make beneficial trades by taking into account the weather forecasts and the expected demand profile of the cluster. This is done by comparing an online RL technique (CACLA [73]) and a novel batch learning technique: Neural Fitted CACLA (NFCACLA).

1.1 Flexibility in the Electricity Grid

Within the current electricity markets there is a shift in which we go from a situation in which energy is generated from non-renewable energy sources produced by a small amount of larger parties, towards an increased use of renewable energy sources (RES) by a larger amount of smaller producers. By 2020, the European Union aims to increase the use of RES to 20% of all electricity consumption [15]. Currently the main increase in RES lies in the generation of wind and photovoltaic (PV) energy [14], but other forms, such as biomass and tidal sources, contribute as well [70]. The European Wind Energy Association expects that 14.9% of the electrical demand can be fulfilled by wind energy by 2020 [20], which is expected to increase to 24% by 2030 [21].

One of the disadvantages of using renewable sources is that their supply is dependent on environmental influences, which calls for a more efficient and flexible way in which electricity is consumed [70, 18]. In order to enable the power grid to incorporate this increase in flexibility, different smart grid technologies, such as [34] and [3] were introduced. These technologies aim to spread out the demand over a broader time-scale, in such a way that the peaks at moments of high demand are flattened. This can be done by utilizing different types of flexibility that can be identified in any device connected to the power grid.

1.1.1 Flexible Devices

Each device (or appliance) has its own form and amount of flexibility. These flexibilities can be in the range of completely stochastic and uncontrollable to freely controllable devices [36]. Only the completely stochastic devices lack flexibility and can not be controlled. Examples of such devices are PV panels or wind turbines, which supply an uncontrollable amount of electricity based on weather conditions. On the other hand, freely controllable devices have no (or barely any) restrictions on when they are able to run.

Rumph, Huitema and Verhoosel [61] identify the following types of flexibility in devices:

Devices with operational deadlines These devices need a fixed amount of time and electricity in order to perform a certain task (e.g. washing machines or dish washers). The use of these devices can be shifted towards beneficial moments, however generally a deadline needs to be taken into account that decides when the device ultimately should be finished performing its task.

Buffering devices These devices generally perform a continuous task, such as heating or cooling (e.g. refrigerators or heat pump devices). The objective is then to keep the temperature level between two limits. Whenever the temperature rises or drops outside of the limits, the device is obligated to either turn on or off, but as long as it remains within its limits, the device has the freedom to switch on or off at desired moments.

Storage devices These devices are able to store energy in order to resupply it later on (e.g. batteries).

Freely controllable devices These devices can be run at any desired time, generally within certain limits [36] (e.g. diesel generator).

Finally one final type of devices is mentioned in [36], which does not fall under one of these categories: the **user-action device**. This type of device runs as the direct result of a user action, such as switching on the lights or turning on the television. These actions are comparable to stochastic devices, since they have no flexibility and are required to directly follow the demand of the user.

1.2 Electricity Markets

In the current electricity market suppliers can trade power on different time scales in order to meet expected future demand from customers [34]. These time scales are divided amongst different markets and combined they form the so-called 'Electricity Wholesale Market'. Spread over the different time scales the participants can offer both supply and demand on these markets. A utility would aim to forecast its future supply and purchase it on the different wholesale markets, which generates an expected consumption profile: the amount of demand that the utility expects to fulfill. With the increase in RES, the expected supply might be estimated too high and and suppliers will have to estimate the size of this surplus and correct their expected supply with respect to that. The final profile that is traded on these markets is referred to as the load profile and reflects the overall allocation of all participants on the markets. Any deviations from these load profiles (i.e. when the demand does not meet the supply) are being equalized by a so-called balancing market.

In this section a short overview will be given of the different markets that correspond to the different time scales on the wholesale market followed by an overview of the balancing market. The descriptions of these markets ore roughly based on the descriptions given in [34] and [19].

1.2.1 The Wholesale Market

The wholesale market consists of three different sub-markets, each corresponding to a different time frame before the delivery of electricity. These markets are (1) The futures market, which takes place in a range of five years to a couple of days before the moment of delivery; (2) the Day-ahead market, which takes place the day before delivery, until noon; and (3) the intraday market, in which suppliers can make some final trades to match their expected demand as closely as possible. Each of these will be explained shortly.

Futures Market

In the futures market suppliers can purchase load blocks from power plant owners or future-market operators. Initially these purchases can be made from five to one year(s) in advance and become smaller over time. So initially suppliers buy a number of load blocks on year-basis. These load blocks can later be adjusted once year-quarter blocks become available. This refinement can later on be done for the period of a month, followed by weeks and finally until load blocks are traded for individual days.

A distinction is made between two types of load blocks: (1) *base load* blocks and (2) *peak load* blocks. The former represents a load that is to be supplied constantly for the period for which a sale is made. The latter is limited to certain time windows, in which the supplier expects a general increase in demand in comparison to the rest of the day (e.g. for 8 AM until 11 PM).

Day-ahead Market

On the day-ahead market (also referred to as spot market) suppliers can refine their load blocks to match the expected demand profile as closely as possible in hourly blocks. Weather forecasts become less uncertain and any other impactful events are generally known. The trades take place on a *power exchange* market, run by a market operator (this is the APX for the Netherlands), where all expected supplies and demands are pooled for each of the 24 hours of the next day (from midnight until midnight). Each day suppliers can bid until noon for power blocks that they wish to supply during the next day.

Intraday Market

Once the Spot market closes, still a couple of hours remain until electricity is being delivered. Until a prespecified time period before the trade (the so-called *gate closure* time) suppliers can still trade hourly load blocks either through bi-lateral trades with other market parties, or via a market operator.

1.2.2 The Balancing Market

At the moment of delivery, electricity is being exchanged with the electricity grid according to the load blocks that were purchased by the suppliers. When the suppliers purchased the wrong amount of energy there is a discrepancy between the supply and demand of the electricity. This discrepancy is automatically traded on the balancing market, which is a market controlled by a single instance: the Transmission System Operator (TSO). The TSO keeps track of any trades that are done on the balancing market in a given time period (the settlement period) and in hindsight determines the price of the electricity that was required to restore the balance across the grid.

The balancing process consists of three separate processes: (1) Collect the forecasted supply and demand from so-called *Balancing Responsible Parties*; (2) Contract different parties for reserve capacities; (3) Restore imbalance and settle the costs with the responsible parties.

Collection of Forecasted Supply and Demand

All of the parties that have an impactful influence on the electricity market, either through demand or supply have a responsibility to maintain the balance across the network, hence they are called *Balancing Responsible Parties* (BRPs). Each of these parties has to submit a forecasted supply (and/or demand) profile. Generally these profiles are given for a certain settlement length, which is typically 15 or 30 minutes (15 for the Netherlands). The forecasts need to be submitted before a deadline (the gate-closure time) which is a fixed period before the moment of delivery.

Contracting Reserve Parties

Market parties can offer reserve capacities to the TSO, to be used whenever there is an imbalance in the grid. Production sites that are of larger capacities are always obliged to offer a predetermined portion of their capacity as a reserve for the TSO. The availability of the reserve capacity is offered in the form of a bid. In the Netherlands, the capacities for available reserves need to be offered to the TSO one hour before the settlement period. Whenever an imbalance occurs, the TSO calls on these reserves in order of the bid prices to restore the imbalance in such a way that the imbalance costs are minimized.

Settlement of Imbalances

Whenever there is a discrepancy between the forecasted supply by suppliers and the actual demand of consumers, this discrepancy is immediately being traded on the balancing market. At the end of each settlement period, which typically is 15 minutes, the TSO makes up a balance of all deficits and surpluses that occurred during this period for each of the BRPs. Based on the net imbalance of all BRPs together an imbalance price is determined by the TSO, which is the price at which all trades with the imbalance market were made. Generally this imbalance price is substantially different from the price at which energy is traded at the spot market, which generally makes having an imbalance disadvantageous for suppliers, however it could be advantageous as well. Parties that contribute to the overall imbalance in the grid need to pay an additional (fixed) administration fee. Whenever the net imbalance yielded a surplus of electricity in the grid, most of the suppliers had to sell some of their supply to the imbalance market. As a result, the imbalance price will be lower than the price that suppliers payed on the spot market, and the suppliers who had a surplus of electricity will have to sell this surplus at a loss. However, when an individual supplier would have a deficit in this scenario, they implicitly had a positive impact on the balance of the grid. Any extra electricity that they required was purchased at a lower price from the TSO than it would have been bought at the spot market. On the other hand, when the net imbalance yielded a deficit, the imbalance price is higher than the spot price. In this situation individual suppliers who contributed to the imbalance due to their deficit had to purchase the extra electricity for a higher price, while suppliers who had a surplus of electricity could sell this surplus for a higher price than they initially purchased it. Table 1.1 shows an example for each of the four scenarios that can occur for suppliers with an imbalance. In this example each supplier bought a certain amount of electricity on the spot market for a price of 40 e/mWh. In the case of a net surplus the imbalance price is

Table 1.1: The settlement results for individual suppliers after imbalances have been resolved by the TSO. When suppliers contribute to an imbalance they make a loss, when they resolve an imbalance they gain from it. The prices here are given in euros per megawatt hours.

		net imbalance	
		surplus	deficit
individual imbalance	surplus	spot price: 40 e imb. price: 30 e result: −10 e	spot price: 40 e imb. price: 50 e result: +10 e
	deficit	spot price: 40 e imb. price: 30 e result: +10 e	spot price: 40 e imb. price: 50 e result: −10 e

set to 30 e/mWh and in the case of a deficit the imbalance price is set to 50 e/mWh.

1.3 Problem Description

As can be seen in table 1.1, assisting in resolving imbalances on the balancing market can be highly beneficial for aggregators due to the favorable electricity prices. However, while doing so the aggregator will have to take its own demand profile into account: there is no point in resolving, for example, a shortage by offering all electricity that is produced within the cluster, resulting in another shortage within the aggregators' own cluster in the near future. This might result in bigger losses for the aggregator than the gain that was obtained by resolving the initial imbalance and the contribution to the global balancing problem becomes negligble, if not negative.

The challenge for the aggregator thus becomes to offer its flexibility to a certain extent to the TSO, while trying to maintain its load profile as closely as possible. Or in terms of a max/min problem: the aggregator aims to optimize the flexibility it offers to the TSO, while minimizing its own imbalances.

For each settlement period the aggregator makes an offer to the TSO, containing the maximum amount of electricity that the aggregator is willing to provide as both ramp up and ramp down capacity. Throughout this thesis, ramp up capacity is considered to be additional supply offered to the TSO (i.e. that the cluster decreases its consumption) while ramp down capacity means lowering supply to the TSO (i.e. that the cluster demands more electricity). The combination of these offers will be referred to as the boundary conditions. During each settlement period, when the TSO is required to perform balancing tasks, it is able to trade either ramp up or down capacity with the aggregator up to the given boundary. This means that the aggregator does not know up front whether it will have to ramp up or ramp down (or neither), and neither does he know to what extent, except for its provided boundaries.

When the TSO requires the aggregator to ramp up or down, this influences the demand for the aggregator's cluster. When the TSO requires ramp up, the aggregator sells some of its electricity to the TSO and the consumption within the cluster should decrease. When the TSO requires ramp down capacity, the aggregator buys electricity from the TSO and consumption should increase. Hence, when the aggregator makes an offer in one direction, it will have to be able to deviate from its demand profile in the opposite direction in the case the TSO (partially) accepts to trade with it. Hence one of the main things to take into account when determining the boundary conditions is the load profile for the near future.

Another influential factor in the consumption and production within a cluster is the weather. Looking at weather forecasts might give a large amount of relevant information for determining the expected amount of electricity that will be produced and consumed, as well as the flexibility that will be available in the near future. For example, when a sunny and warm afternoon is forecasted, the aggregator can assume that a limited amount of electricity will be required for, for example, heating, while a lot will be produced by PV panels. As a result there will be a small amount of flexibility available for the aggregator at that point in time and the aggregator might even get in a state of imbalance due to the high amount of production and low amount of consumption. On May 8th in Germany the amount of electricity produced through RES was so high that consumers got paid to consume this electricity rather than having to pay for it[1]. Knowing this up front, the aggregator may wish to start offering ramp up capacity to the TSO, so that it decreases the amount of electricity that is used early on the day, so that its buffers are empty. In this way, the cluster might be in a state of overproduction during the hot afternoon. However, offering too much electricity too early might result in a shortage as well when all buffers are depleted too early or the weather is less hot or sunny than expected.

Reinforcement learning agents will be used to determine the boundaries within which an aggregator can safely offer its available flexibility to a TSO to aid in resolving any imbalances. This is done by looking at the demand profile and weather forecasts for the next 24 hours. Using these forecasts the agents should be able to learn and identify the moments at which offering either ramp up or ramp down capacity can be most beneficial to the aggregator, while minimizing any costs by deviating from its load profile. These agents are to be developed and tested using TNOs PowerMatcher

[1]As reported by the Daily Mail and The Independent

Technology [34], however these could be applied to any other Smart Grid technology as well.

The aim of this project is twofold: (1) to test the applicability of an existing online reinforcement learning algorithm, called the Continuous Actor Critic Learning Automaton (CACLA) [73, 71] and (2) to test a novel batch variant of CACLA, inspired on CACLA and an existing batch algorithm, the Neural Fitted Q iteration with Continuous Actions (NFQCA) [26, 27]. This novel algorithm is dubbed the Neural Fitted Continuous Actor Critic Learning Automaton, or in short: NFCACLA. Recently another neural fitted variant of CACLA was introduced, dubbed NFAC [43], which has been developed independently from the work in this thesis.

1.4 Research Questions

The main research question to be answered in this thesis is **can reinforcement learning be used to determine the boundaries within which a smart grid cluster can contribute to resolving imbalances on the reserve market?** This research question is divided in the subquestions that are given below. In these subquestions a distinction is made between internal and external imbalances. Internal imbalances are imbalances that occur within an aggregator's cluster. External imbalances are imbalances that occur outside of the aggregator's cluster. The aim of the aggregator is to assist in resolving these external imbalances, while minimizing the amount of the internal imbalances.

1. How well are the CACLA and NFCACLA algorithms able to utilize flexibility in smart grids to maintain and or improve stability in the electricity grid?
2. How suitable are weather predictions as indicators for offering flexibility?
3. How does NFCACLA perform in comparison to CACLA?
4. How does the number of epochs influence NFCACLA's learning process?

The algorithms are to be evaluated both in terms of the amount of imbalances that are resolved and caused, as well as in the amount of money that was spend or made by trading on electricity with the TSO. A comparison will be made in both a theoretical scenario, in which the agents trade all electricity offered to them, as well as in a real-world scenario in which he amount of electricity that is traded is based on historical imbalance data. Details on this can be found in the section on validation.

1.5 Outline

The remainder of this thesis is structured in the following manner. In the next chapter a more thorough introduction to Smart Grids is given, along with a more specific description of TNO's PowerMatcher. Additionally some of the recent developments and proposals that stimulate/enable smaller producers and or consumers to actively participate in resolving external imbalances are described.

In Chapter three a general introduction is given on the concept of reinforcement learning and Markov decision processes. This is followed by a more thorough description of the different algorithms that are used for this thesis. The chapter in concluded with a description of how the trade of flexibility can be modelled as a Markov decision process and solved through reinforcement learning.

The fourth chapter gives a more thorough description of the experiments that are performed for this thesis in order to validate the algorithms and how their performances are measured.

Chapter five shows the results of these experiments. Finally, in the sixth chapter the research questions will be answered and this thesis will be concluded.

Chapter 2

Smart Grids

In order to present the reader with some more background into the developments around the electricity grid, this chapter will give a general introduction into the field of Smart Grids. First off a definition and general description of Smart Grids will be given, along with an explanation on why their development is essential within the field of the electrical power industry. Secondly an overview will be given of some of the recent developments on the coordination on the demand side of the grid. This will be followed by an overview of some proposed real time markets, in which end-users can directly participate. Finally, some developments will be presented that explore the possibilities to enable market participation for smaller parties in the current electricity markets.

2.1 Smart Grids: What are they and why do we need them?

Everywhere across the world the development of Smart Grids is one of the biggest research interests in the field of the electrical power industry. Even though the research interests and aims of their development may vary, smart grids are a major interest across different regions in the world.

In China a large increase in electricity demand was projected [29]. Since the main source of electricity is mainly from fossil-fuel-based sources [80], China is developing Smart Grids both in order to cope with this large increase in demand and increase the integration of Renewable Energy Sources (RES) within the grid [80, 28]. India has one of the largest power grids that currently exists, however electrical demand is still rising. Additionally there are still rural areas that are not connected to the grid yet [63].

Finally, the Indian grid is coping with large losses (both technical and financial) [63, 28]. A roadmap was laid out by the Indian Ministry of Power [44] to use smart grid technologies to improve in these areas. In Europe and North America the current electrical grid is aging. Additionally there is a political incentive to more economical prices, an increase in more sustainable energy sources [28, 17], and to encourage the demand side of the grid to actively participate in the supply chain of electricity [16].

Due to involvements all over the world, a wide variety of definitions of what a smart grid is, have already been given by different parties such as the EPRI (Electric Power Research Institute) [1], IEC (International Electrotechnical Commission)[28], NETL (National Energy Technology Laboratory) [48], ETP (European Technology Platform) [16] and CEA (Canadian Electricity Association) [8]. Even though the number of different definitions is large, they generally agree on that the smart grid is "a dynamically interactive real-time infrastructure concept that encompasses the many visions of diverse power system stakeholders."[12]. However, in order to hold on to a more specific definition of the smart grid, we will adopt the definition given by the ETP [16]:

> A Smart Grid is an electricity network that can intelligently integrate the actions of all users connected to it - generators, consumers and those that do both - in order to efficiently deliver sustainable, economic and secure electricity supplies.

Now that a general definition of what a Smart Grid is, has been given, it is time to identify the reasons why they are essential to develop. In his PhD thesis [34] Koen Kok presents three main reasons why the development of Smart Grids is essential: (1) The electrification of everything; (2) The transition to sustainable energy resources; and (3) the decentralization of generation.

The Electrification of Everything refers to the fact that the consumption of electricity increases throughout the world. The number of devices that run on electricity ever increases (for example by the increased use of Electric Vehicles), while the current power grid starts to become more and more outdated. The traditional response to an increase in demand is to simply increase the capacity of the grid. The proposed *smart* solution would be to use the available flexibility in order to shave the peaks and shift consumption from peak moments to moments where demand is not that high.

In order to make the electricity grid more sustainable, the use of RES is increasing. In the traditional grid a large group of producers would adjust their supply to the demand of the consumers (i.e. supply follows demand). However, the supply of RES can not be fully controlled, which means that

[1] `www.smartgrid.epri.com`

it becomes harder for the supply of electricity to follow the demand. The smart solution would then be to let the demand follow the supply of power. This means that demand should increase whenever the supply of renewable sources is high and vice versa.

Finally a shift can be seen from the situation in which there is a small number of suppliers of large amounts of electricity, to a situation in which there is an increasing amount of smaller producers/suppliers, which is being referred to as *distributed generation* (DG). Since these smaller producers can not actively participate in the current wholesale market, their contribution to the supply of electricity can not be controlled and is simply viewed as negative demand. This approach is also called *fit and forget*. The development of smart grids provides us with methods to integrate these smaller suppliers within the grid.

2.2 Demand Side Management

Now that the concept of Smart Grids has been defined, and an overview has been given of the importance of their development, it is time to zoom in on some recent developments. This section will provide an overview of recent developments on *demand side management* (DSM) [25, 24], which focuses on controlling the demand side of the grid. The main goal of DSM is to shift the demand towards preferable moments, such as when supply of renewable sources is high, or moments at which demand is generally low. One of the biggest tools to perform DSM is *demand response* (DR): the ability of (groups of) devices to adjust their electrical demand to changes in the pricing of electricity. The PowerMatcher, amongst others, is one of the developments that aims to capitalize on this *dynamic pricing*.

2.2.1 PowerMatcher

The PowerMatcher [34] is a smart grid development by TNO which coordinates supply and demand through dynamic pricing. It is developed as a multi-agent based system, in which each appliance is represented as an agent that tries to operate in an economically optimal way. The PowerMatcher is designed as a scalable solution to perform near real-time coordination tasks in smart grids, which has been successfully applied in several field trials [35, 5]. Since the PowerMatcher will be used for this project it will be explained a bit more thoroughly than its alternatives, which will be discussed afterwards.

The PowerMatcher Technology uses a tree-based agent structure, with different types of agents. The different appliances within a cluster are represented

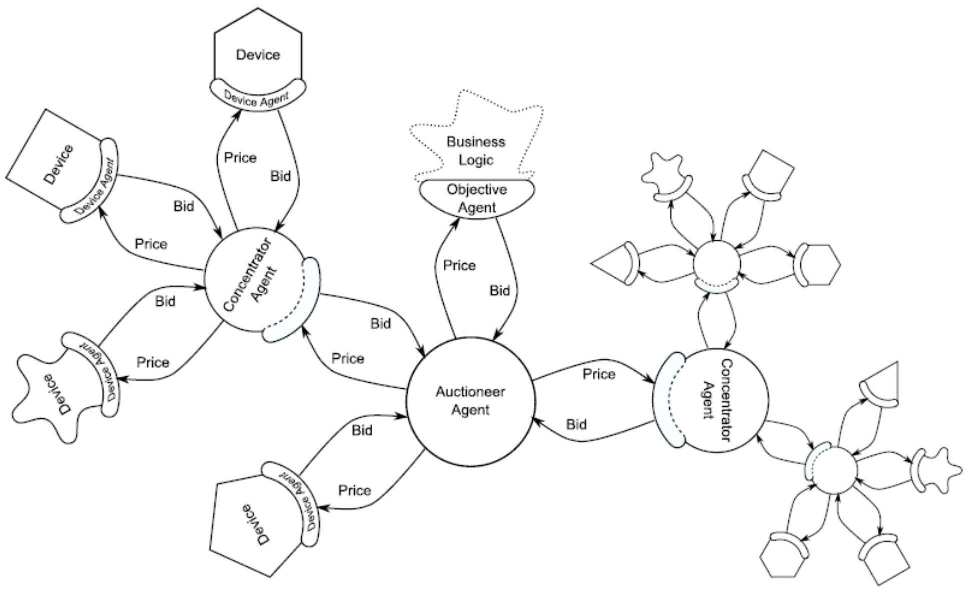


Figure 2.1: A typical PowerMatcher cluster containing each of the four agent types: device agents, concentrator agents, an auctioneer and an objective agent. Adopted from [34]

by *device agents*. Some of these device agents might be clustered together into a smaller cluster by a *concentrator agent*. At the root of the tree, there is an *auctioneer*, the agent that controls the prices of electricity within the cluster. Finally an *objective agent* might be added to the cluster, which steers either the prices (price steering) or the allocation (volume steering) in order to achieve some predefined goal set for the cluster. When an objective agents steers the price it aims to encourage or discourage devices to consume electricity by decreasing or increasing (resp.) its price at a certain rate. When the objective agent steers the allocation this means that the objective agent requires the cluster to consume (or produce) an additional (or smaller) amount of electricity, and steers the aggregated bid so that supply and demand meet each other at this amount of allocation.

Figure 2.1 provides a graphical overview of a typical PowerMatcher cluster. It can be seen that each of the device agents is connected to either the auctioneer or a concentrator agent. The concentrators in turn are connected to either the auctioneer or another concentrator agent as well. The objective agent is typically connected to the auctioneer.

As mentioned earlier, the goal of PowerMatcher is to coordinate supply and demand through dynamic pricing. It does so by collecting bids from all agents, describing their willingness to pay for electricity and the amount of electricity they demand. In these bids we speak of demand, but they include

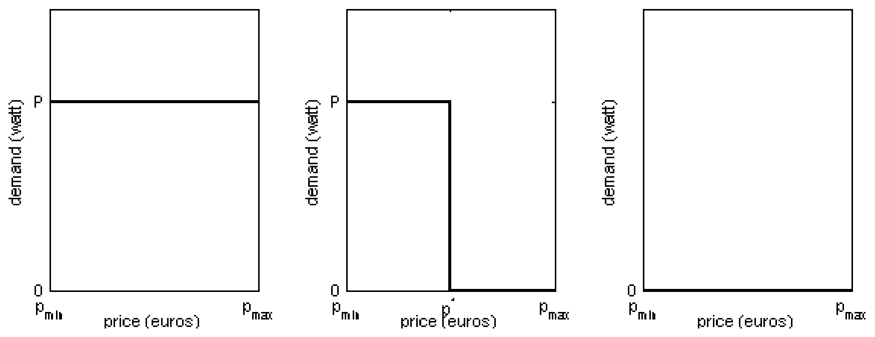


Figure 2.2: typical bid curves for device agents. The left figure shows the bid curve of a device that requires to run, using P watt. The middle figure shows the curve for a device that has some flexibility and only wishes to run if the price is below p^*. The right image shows the curve for a device that won't or can't run, thus requiring no power.

supply as well, which is simply treated as a negative demand. All the bids are sent to the auctioneer, which aggregates them and then determines the price at which the total demand is 0, i.e. when supply equals demand. This price is then communicated back to each of the devices, that in turn switch on or off based on the price. Note that device agents connected to a concentrator send their bids to the concentrator rather than the auctioneer. The concentrator than creates the aggregated bid for all the devices (or concentrators) that are connected to it and send it to the auctioneer (or another concentrator). Prices are communicated back through the same route (i.e. via the concentrators).

The bid that the agents send is a function of their electricity demand versus the price of electricity and is highly dependent on the amount of flexibility that is available to an agent at a given moment. Agents that have no flexibility will sent flat bids: i.e. they either require to run or they don't, regardless of the price of the electricity. However, agents that do have flexibility might only wish to run when the price is below a certain threshold. Figure 2.2 shows some of the common forms of the agent bids. The left image shows the bid of an agent without flexibility that is required to run, and needs P watt to do so. On the other hand, the right image shows an agent that can not (or is not supposed to) run, and hence demands 0 watt, regardless of the price. More interestingly, the *bid curve* shown in the middle picture describes the bid of agents that contain some flexibility. Due to this flexibility they can decide that they wish to run whenever the price is below a certain threshold, set at p^*. However, when the price passes this threshold, the agent no longer desires to run.

The bids that are sent by the agents are collected by their parent, being either a concentrator or the auctioneer. Once the parent has collected the bids from all of its children it aggregates them into an *aggregated bid.* This aggregated bid is then a step-function describing, again, the demand versus price. In the case the parent is a concentrator, it sends the aggregated bid to its parent in turn. When the parent is the auctioneer, it determines the price of the electricity and communicates it back to its children. Note that when an objective agent is present in the cluster, the aggregated bid is first sent to the objective agent so that the price or demand may be steered according to the objective agent's goals before it is sent back to the other agents.

Ideally the price is set in such a way that the demand of the agents equals the supply in the grid. Since supply can be viewed as negative demand and is included in the aggregated bid as such, this price is given by the price at which the demand in the aggregated bid is zero. At this point the market has reached an equilibrium, and as such this price is called the *equilibrium price.* However, such a point might not always exist, in which case there is an imbalance in the cluster and any additional demand or supply that is required is to be resolved with the imbalance market. In these scenarios the price wil be set to either a predefined minimum (in the case of a surplus) or maximum price (in the case of a deficit). Figure 2.3 shows some example aggregated bids for the different balancing scenarios. The middle figure shows the scenario in which the balance is maintained. The aggregated bid contains a price p^* at which there is an equilibrium, at which the price will then be set. The left figure shows a scenario in which there is a deficit of electricity: the demand is higher than the supply. In this scenario the price of electricity will be set to p_{max}. The left figure describes a scenario in which there is a surplus of electricity: the supply is higher than the demand. In this scenario the price is set to p_{min}.

Once the set price is communicated back to all agents by their respective parents they either run or not, depending on the bid that they sent earlier.

2.2.2 Alternative Solutions

Another coordination mechanism (also developed in the Netherlands) is Triana [3]. Triana is a control methodology that coordinates electrical demand in three steps. During these steps a distinction is made between two different levels: a local level, which represents individual buildings, and a global level. The first step of Triana occurs on a local level: a local controller, which has learned the behavior of electrical consumption and potential external influences, forecasts the demand profile, including some possibilities to diverge from this profile (the *scheduled freedom*), of a building. During the second step, a global planner uses the scheduled freedom of these forecasts in order to optimize the forecasts with respect to some global objective. In order to

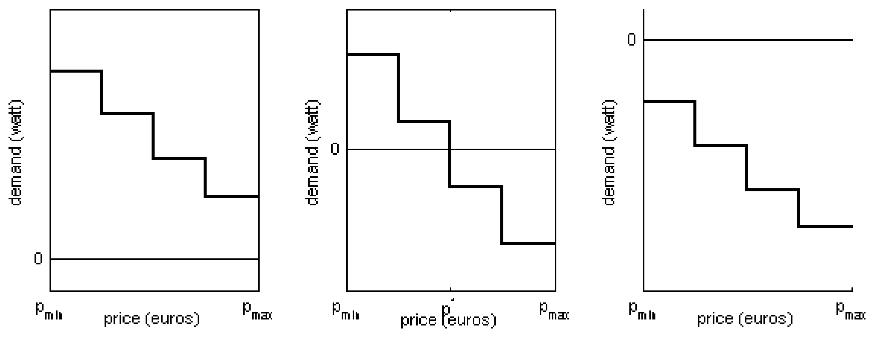


Figure 2.3: Example aggregated bid curves in different balance situations for a PowerMatcher cluster. The right image shows an aggregated bid in a state where there is a deficit of electricity. The middle image shows the aggregated bid in a state where the cluster is balanced. The right image shows the aggregated bid in a state where there is a surplus in the cluster.

achieve this objective the planner then sends steering signals to the local controllers to achieve the global goal. During the third step the local controller uses its knowledge on the devices within the building, together with the steering signals that were sent by the planner, to determine at which times certain devices should be turned on or off.

The PowerMatcher and Triana are examples of *virtual power plants* (VPPs). As mentioned in the introduction, VPPs function as an aggregator, to aggregate the demand and/or supply from different smaller parties. A number of other solutions to perform DSM using VPPs have been proposed as well. For example, Faria, Vale and Baptista [22] explored the possibility for VPPs to schedule devices consumption, through consumption reduction and shifting, using supply from distributed resources within its cluster and from the supply side of the grid. Sonnenschein et al. [67] proposed the *Dynamic VPP* (DVPP), which dynamically aggregates groups of suppliers over coalitions, corresponding to different product types (e.g. active or reactive power). For each coalition, bidding and trading is done by a single representative agent on behalf of a coalition. After the moment of delivery for a given time frame, coalitions dissolve again and new coalitions can be made. Pudjanto et al. [55] propose a virtual power plant that is split into two VPPs: the *commercial* VPP (CVPP) and the *technical* VPP (TVPP). The CVPP represents the combined demand profile of the devices connected to the VPP, and is thus used for trading on the wholesale market. The TVPP then performs system management, ensuring system balance, and enabling the VPP to make trades on the imbalance market.

The development of smart grids also provides a lot of new challenges and opportunities to the fields of Artificial Intelligence [74, 78]. A number of successful methods have been proposed, using genetic algorithms [65, 66], reinforcement learning [13, 51] and dynamic programming [38, 23], both on local (residential) and global (VPP) levels. Additionally several methods have been proposed for charging electric vehicles (EVs) in a smart way [47, 41]. Furthermore, a wide variety of solutions use multi-agent systems [10, 31, 41, 77] to represent the grid and/or its connected devices.

Finally a number of mathematical control methodologies has been used to schedule the use of different devices. The scheduling problem can, for example, be expressed as a non-linear minimization problem [7], minimizing the cost of electricity supply. By formulating the scheduling problem on a graph, graphical inference models, based on statistical inference can be used to find a global optimal solution for the distribution of electricity over the grid [33]. Finally a number of game theory-based solutions have been proposed as well [62, 2, 45, 50].

2.3 Real Time Markets

Real Time Markets (RTMs) are markets that function on a near-real time basis and allow small consumers, producers and prosumers to participate in the market. Since the demand and supply of the devices can be managed on a much shorter timescale than the traditional participants in the other power markets, the participants of the market can adapt their consumption (or production) pattern more easily than, for example, large-scale generators [76]. In this way, the distributed energy resources (DER) can be used for balancing services in short time intervals (e.g. 5 minutes, versus the 15-20 minutes when using traditional generators).

2.3.1 Real Time Market Types

In a survey Wang et al. [76] describe three types of real time markets (RTMs): (1) nodal price RTMs, (2) zonal price RTMs and (3) zonal price balancing markets. These three types differ from each other in terms of geographical pricing schemes and clearing interval. For the geographical pricing scheme a distinction is made between zonal, i.e. pricing for a specific zone, and nodal prices, i.e. specific pricing for each connection point (bus).

The first type of RTM, the nodal RTM, can mainly be found in the Northern U.S. markets. In these markets the prices are set for five-minute intervals for each connection to the grid. Whenever there is no imbalance, the price will theoretically be the same for each connection. However, when there is an imbalance, the price will be adjusted accordingly. The real time prices can be either announced at the start or at the end of each interval (this differs per market). Settlement of imbalances generally occurs every hour, but might be done at other intervals as well.

The second type of RTM is the zonal price RTM, which is applied in Australia. The country is divided into five zones and each zone maintains its own price. Just like in the Northern American RTMs, prices are set for five-minute price intervals. The markets are operated by the Australian Energy Market Operator (AEMO), which sets the price for each region. Balancing is performed on eight different markets, also operated by the AEMO.

The third type is represented by most European markets, and considers zonal-price balancing markets (BMs). A more specific description of balancing markets has already been given in the introduction, so this will not be discussed in depth.

2.3.2 EcoGrid EU

The EcoGrid EU project [30] proposes a real-time market concept, in addition to the existing balancing market. Through a large-scale demonstration of 2000 residential participants on the island of Bornholm in Denmark, the project aims to apply a smart grid solution in a real-world situation with a supply that is generated for 50% by renewable energy sources.

The proposed market is operated by a so-called real-time market operator (RTMO), which might be the TSO. The market prices are cleared at five-minute intervals and the prices well be made known to the market participants after each interval. These prices are based on the day-ahead market price and reflect the need for up or down regulation due to any imbalances within the grid. As a result, if there is no imbalance, the real-time market price (RTM price) will be equal to the day-ahead price. In order to perform some planning tasks a projected RTM price is sent to the end-users as a price forecast. Using this forecast the end-users can schedule their planned tasks in an efficient manner. The market can then steer this demand by adjusting this price, in accordance with the imbalance in the grid, thus utilizing the flexibility that is available within the different devices in the grid. For more technical details about the implementation of the market concept, see [11].

With the demonstration, it was shown that a peak load reduction of $670kW$ (1.2%) could be achieved, of which 87% was achieved through the response of households to the real time pricing [2]. Furthermore, there was in increase in the integration of wind energy by 8.6%, and the need for using reserves from the imbalance market was decreased by 5.5% [53].

[2]`www.eu-ecogrid.net`

2.4 Participation in the Balancing Market

As already becomes clear from the previous section, the increasing amount of flexibility in the demand side of the grid provides ample opportunities for increasing the balance across the grid. Apart from introducing a new real-time market type, various propositions have been made to utilize the availble flexibility in the current reserve market. Most of this research has been done with a focus on the perspective of the market operator, finding optimal clearing prizes. However, some limited focus has been put on the perspective of the consumers as well. This section will give a short overview of a number of models that have been proposed to enable consumers to participate and benefit from the participation in the balancing market.

Several methods for finding optimal clearing prices have been proposed, using mixed integer linear programming (MILP), such as [75, 1]. A relevant aspect of the participation in the balancing market is that adjusting demand in one direction, should be resolved eventually by adjusting demand in the opposite direction in the future (i.e. the system needs to recover). Karangelos [32] proposes a method that takes into account the future recovery consequences that will follow from balancing contributions from the demand side of the market. Mathieu et al. [42] proposed an agent-based model that participates in the day-ahead market, in which agents use price forecasts to optimize their expected profit, taking the required recoveries into account.

Liu and Tomsovic [39] proposed a full demand response model, able to participate in different reserve markets. However, the model was proposed in combination with a co-optimized reserve market model, rather than currently existing balancing markets. Using MILP they were able to reduce the amount of capacity that had to be committed by regular generators and the overall system operating costs.

Peters et al. [52] introduce a new class of SARSA reinforcement learners referred to as SELF, Smart Electricity market Learners with Function approximation. It is used to simulate a retailer, selling electricity. The objective is then to determine the tariff that the retailer should ask for its power. Given a set price, the actions that the agent can then perform is to either keep the price, or change it. This set of actions is discretized into a number of economically meaningful actions. The actions are not necessarily absolute, but can be relative to other retailers as well: for example an action could be 'ask a high/average/low price'. The states are composed of different economical factors, representing the current (economic) environment. The algorithm was tested in a newly developed simulation environment called SEMS (Smart Electricity Market Simulation), which is based on wholesale prices from the real world. The authors suggest that further development and research is necessary to apply SELF to more "sophisticated Smart Electricity Markets".

While former methods assume a direct interaction with the balancing market, the method proposed in this thesis does not. Rather it determines the boundaries within which the cluster can respond to external imbalances and offers this reserve to the TSO than trading it directly on the balancing market. This approach is more similar to the current real-world scenario in which the TSO contracts different parties for their reserves. How reinforcement learning is used to determine these boundaries will be discussed in the next chapter.

Chapter 3

Reinforcement Learning

Reinforcement learning [68, 79] is a learning method that aims to learn what action should be taken in an environment in order to achieve a certain goal. It does so in a trial-and-error process in which semi-randomly selected actions are performed. Each action is evaluated in terms of its added value to achieve this goal.

Reinforcement learning is generally used to find a solution to Markovian Decision Processes (MDPs) [57, 56]. In order to achieve this, a wide variety of different implementations of RL methods has been proposed. In this chapter a formal description of MDPs will be given, followed by an explanation of the different algorithms that will be used for the aims of this project. For a more detailed introduction and complete overview of the field of reinforcement learning, please look at [68, 79].

This section will start of with a general description of Markov Decision Processes, followed by a description of the methods that are used for this thesis to solve the decision problem for MDPs. Finally an explanation will be given on how the trade of flexibility can be modeled as an MDP.

3.1 Markov Decision Processes

Markov Decision Processes (MDPs) describe sequential decision making problems. In such a process an agent observes a state and decides to perform an action on the basis of this state. The result of this action causes the agent to reach a (new) following state. In this section a formal definition of MDPs will be provided.

MDPs can be described in the form of a tuple $< S, A, T, R >$ [79]. In this tuple S and A are finite sets, representing the states that can be observed and the actions that can be performed by an agent respectively. T defines a state-transition function $T : S \times A \times S \rightarrow [0,1]$ and R defines a reward (or cost) function as $R : S \times A \times S \rightarrow \mathbb{R}$. This tuple could be extended with a a variable γ, which represents a discount factor, $0 \leq \gamma \leq 1$ [73, 71]. This discount factor is generally regarded as an element of a given algorithm rather than an element of MDPs in general, and as such it will be regarded in this chapter. Each of these elements will now be examined more thoroughly. The definitions that are given are mainly acquired from [79, 73].

States The set of observable states is defined as $S = \{s^1, \ldots, s^N\}$, where $N \in \mathbb{N}$ represents the number of states that can be observed by an agent. Please note that every state $s \in S$ is a unique element, s.t. $\forall i, j$ where $0 \leq i < j \leq N$ it holds that $s^i \neq s^j$. Finally the state that is observed by an agent at timestep t is given by $s_t \in S$.

Actions The set of actions that can be performed by an agent is defined as $A = \{A^1, \ldots, A^M\}$, where $M \in \mathbb{N}$ is the number of total actions that can be performed. Each state $s \in S$ has a particular subset of actions that can be performed. This subset of actions is given by $A(s) \subseteq A$ (however, it generally holds that $A(s) = A$). Each action $a \in A$ is unique in an analogous way as states are. The action that is performed at timestep t is given by $a_t \in A$.

Transition function During each timestep t the agent observes state $s_t \in S$ and performs action $a_t \in A$. After performing this action the agent will be able to observe state $s_{t+1} \in S$ (note that it is possible that $s_t = s_{t+1}$). The transition function T defines the probability that after performing action a_t in state s_t, the agent reaches state s_{t+1}, for all $s \in S$ and $a \in A$. More formally this is defined as $T : S \times A \times S \rightarrow [0,1]$. The probability that state s_{t+1} is reached after performing action a_t in state s_t is then defined as $T(s_t, a_t, s_{t+1})$, where for all states and actions it holds that $0 \leq T(s, a, s') \leq 1$. Since it is certain that after performing an action in a certain state a transition is made to a next state, it is given that for all $s \in S$ and $a \in A$ it holds that $\Sigma_{s_{t+1} \in S} \quad T(s_t, a_t, s_{t+1}) = 1$. Note that when an action a can't be performed in a given state, this means that $\forall_{s_{t+1} \in S} \quad T(s_t, a, s_{t+1}) = 0$.

Reward function After an action is performed, the outcome of this action is to be evaluated. The reward function has the form of $R : S \times A \times S \rightarrow \mathbb{R}$ and assigns a reward to the transition from state $s_t \in S$ to $s_{t+1} \in S$ by performing action $a_t \in A$. Generally, throughout this thesis, the reward function will be referred to with $R(s_t, a_t, s_{t+1})$ and $r_t \in R$ represents the reward that is given at timestep t.

Note that the reward function does not necessarily give positive feedback and can yield negative feedback (punishments) as well. As a result the reward function sometimes is referred to as the *cost function* as well.

3.1.1 Policies and Optimization Criteria

In order to determine what action an agent should perform given an observed state, a policy function is defined. A distinction can be made between two types of policies: *deterministic* policies and *stochastic* policies [79]. A deterministic policy function maps each state directly to a single action, while a stochastic policy defines a set of actions, each with its own probability of being performed in a given state. More formally, a stochastic policy function π is defined as $\pi : S \times A \rightarrow [0, 1]$, such that $\forall_{s \in S, a \in A}\ \pi(s, a) \geq 0$ and $\Sigma_{a \in A}\ \pi(s, a) = 1$ [79].

The aim for an agent is then to learn a policy that maximizes the total sum of rewards gained throughout its course (the cumulative reward). An agent tries to learn this policy by gathering rewards according to some optimization criterion [79] for the actions that it performs. When the agent does not care about future influences that its current action might have, only the immediate reward has to be taken into account and the optimization criterion can be defined as $E[r_t]$. However, it might be beneficial to take the future into account as well. This is generally done in two different ways. The first way is to take a discounted amount of future rewards into account, yielding the optimization criterion as defined in equation 3.1.

$$E\left[\sum_{t=0}^{\infty} \gamma^t r_t\right] \tag{3.1}$$

In this equation γ is the discount factor that was mentioned earlier with $0 \leq \gamma \leq 1$, which defines the extent to which future rewards are deemed relevant. The second way is to take the average reward over all future rewards, yielding the optimization criterion as defined in equation 3.2.

$$\lim_{h \to \infty} E\left[\frac{1}{h}\sum_{t=0}^{h} r_t\right] \tag{3.2}$$

For these optimization criteria the number of timesteps is assumed to be infinite (or unknown). However, when the number of remaining timesteps is known it is possible to simply maximize the sum of rewards over all these timesteps. In other words, the maximization criterion is yielded by $E\sum_{t=0}^{T} r_t$, where $T \in \mathbb{N}$ is the number of (remaining) timesteps.

3.1.2 V-values and Q-values

Now that it has been defined what the optimization criteria are, we are able to measure the values of the states that can be observed in terms of the expected reward an agent could receive when it has reached a given state. This is done with a so-called *value function.* The value function $V^\pi(s)$ is defined as the expected reward when policy π is followed from state s [79]. Alternatively a *state-action* value function $Q^\pi(s,a)$ can be defined, which yields the expected reward when action a is performed in state s, followed by policy π. The latter is generally used when the transition and/or reward function(s) are unknown. First a formal definition of how state values are determined will be given. Afterwards, an analogous definition will be provided for state-action values. The mathematical equations used in this section are adopted from [79].

Since the value function is simply the reward (as defined in the previous section) that is gained by following the policy π, the state value for state s can be expressed as:

$$V^\pi(s) = E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k} | s = s_t \right\} \tag{3.3}$$

The recursive nature of $V^\pi(s)$ makes it so that it can be rewritten to a so-called Bellman equation [4], as shown in [79] with equation 3.4

$$\begin{aligned} V^\pi(s) &= E_\pi \left\{ r_t + \gamma V^\pi(s_{t+1}) | s = s_t \right\} \\ &= \sum_{s'} T(s, \pi(s), s') \big(R(s,a,s') + \gamma V^\pi(s') \big) \end{aligned} \tag{3.4}$$

$V^\pi(s)$ describes the expected reward for any policy π, ran through from state s. Now, the goal of any MDP is to find the sequence of actions that yields the maximum expected reward. This means that it is the goal to find a policy π^* for which $V^{\pi^*}(s) \geq V^\pi(s)$ for all $s \in S$ and all policies π. Then let $V^*(s)$ describe the value function that follows π^* (i.e. $V^*(s) = V^{\pi^*}(s)$). Inserting $V^*(s)$ in equation 3.4 yields equation 3.5.

$$V^*(s) = \sum_{s'} T(s, \pi^*(s), s') \big(R(s, \pi^*(s), s') + \gamma V^*(s') \big) \tag{3.5}$$

Since $\pi^*(s)$ describes the policy in which the action is performed that maximizes the expected reward, equation 3.5 can be rewritten to equation 3.6.

$$V^*(s) = \max_{a \in A} \sum_{s'} T(s,a,s') \left(R(s,a,s') + \gamma V^*(s') \right) \tag{3.6}$$

And equivalently we wish to find the action that belongs to the optimal policy π^* in state s, which is simply the action that maximizes $V^*(s)$. This is found according to equation 3.7

$$\pi^*(s) = \arg\max_{a \in A} \sum_{s'} T(s,a,s') \left(R(s,a,s') + \gamma V^*(s') \right) \tag{3.7}$$

While V-functions map each state to an expected cumulative reward, Q-functions make a mapping $Q : S \times A \rightarrow \mathbb{R}$ that maps each state-action pair to an expected cumulative reward. Analogous to equations 3.4 and 3.5, the Q-function $Q^\pi(s,a)$ and its optimal value, $Q^*(s,a)$, can be defined as:

$$\begin{aligned} Q^\pi(s,a) &= E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k} | s = s_t, a = a_t \right\} && (3.8) \\ Q^*(s,a) &= \sum_{s'} T(s,a,s') \left(R(s,a,s') + \gamma \max_{a'} Q^*(s',a') \right) && (3.9) \end{aligned}$$

Since a Q-value already is a mapping of both a state and an action to the expected cumulative reward, it is simple to compute the optimal action from a given state. Hence the difference between V-values and Q-values is that when working with V-values, $V^*(s)$ requires to find the action a that maximizes the expected reward, while when looking at $Q^*(s,a)$ we are simply looking at the expected (maximum) cumulative reward, provided that action a is selected. This yields the following mathematical relation between V^* and Q^*:

$$\begin{aligned} Q^*(s,a) &= \sum_{s'} T(s,a,s') \left(R(s,a,s') + \gamma V^*(s') \right) && (3.10) \\ V^*(s) &= \max_{a \in A} Q^*(s,a) && (3.11) \\ \pi^*(s) &= \arg\max_{a \in A} Q^*(s,a) && (3.12) \end{aligned}$$

3.1.3 Learning an Optimal Policy

In the previous section it was explained that the aim of an MDP is finding an optimal policy. Hence, solving an MDP means computing what this optimal policy looks like. When computing this optimal policy, a distinction can be made between model-based and model-free solutions [79].

Model-based methods use a model that is known a priori in order to compute the value-functions. Once the (state-)value functions are known, they can be applied to the model, yielding an optimal action in a given state.

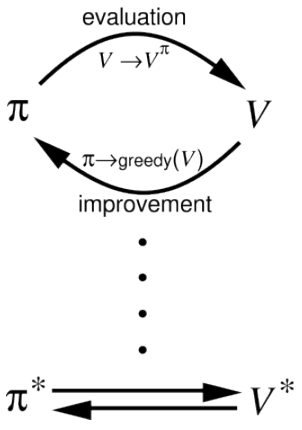
Model-free methods do not relate to a known model, but rather interact with the environment to learn how it is influenced by the selected actions. A model-free method will hence first explore the environment by performing exploratory actions. A (state-)value function is then obtained through the evaluation of these actions. In this way an agent using the model-free approach creates a model of its own, which is then used in the same way as in the model-based approach to compute what actions are optimal for an observed state.

The key problem with both methods is the interaction between the value function and policy selection. This interaction is governed by two processes: *policy evaluation* and *policy improvement* [68]. The former consists of keeping the value function up-to-date with respect to the current policy, while the latter tries to improve the policy (or, make it greedy) with respect to the current value function. The interaction between these two processes is also referred to as *Generalized policy iteration* (GPI). However, in model-free approaches it is common practice that either the value function or the policy is implicitly represented and computed on the fly, when necessary [79].

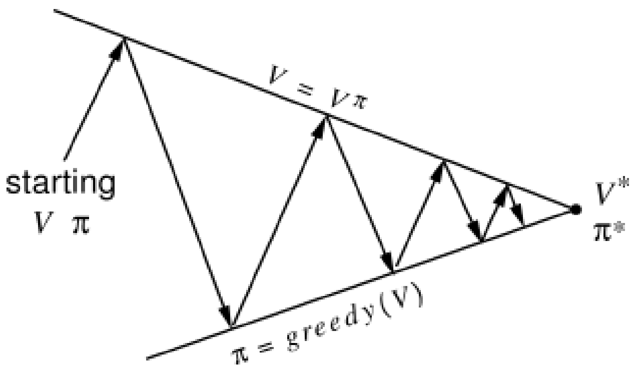
The overall scheme of GPI is depicted in figure 3.1 (adopted from [79]), which depicts a repeating cycle (figure 3.1a) in which in turn a greedy policy is selected on the basis of a current value function and a value function is updated on the basis of a selected policy. This cycle continues until the point where the policy and value function have converged and no longer change, at which point π^* and V^* are found (figure 3.1b).

3.1.4 Discrete versus Continuous States and Actions

Due to their computational complexity, MDPs are generally defined with a small and finite set of discrete state and action spaces. Due to this characteristic, the V- or Q-values for a given policy can generally be stored in a so-called lookup table [68, 79]. However, often (especially in real-world scenarios) the problem space might not be well-defined in discrete partitions or



(a) The interaction between the *policy improvement* and *policy evaluation* steps



(b) The convergence of *policy improvement* and *policy evaluation* yields π^* and V^*

Figure 3.1: an overview of the Generalized Policy Iteration principle

be infinitely large. In such cases lookup tables can not be used, and alternative methods should be used when dealing with non-finite or continuous action spaces.

One of the ways through which the extension of discrete to continuous state (or action) spaces can be achieved is through the use of function approximators (FAs) [6]: rather than storing a value for each possible state, the value of a state is measured through some evaluation function. Generally this evaluation function is built using regression-based supervised learning methods.

There are three main methodologies that can be used to approximate continuity within an MDP [71]: (1) Model Approximation, (2) Value Approximation and (3) Policy Approximation. Applying the Model Approximation method entails that the Transition and Reward functions are to be approximated. The Value and Policy Approximation methods use FAs in order to approximate V^* (or Q^*) and π^* respectively. These methodologies can each be applied individually or combined in any permutation in an MDP to introduce continuity. In this thesis there will be a focus on approaches that use FAs for both Value and Policy Approximation within the class of actor critic models [54, 64]. More on this will follow in the next section.

3.2 Solving the Decision Problem

In order to solve the decision problem for this thesis both an existing and a new algorithm will be applied. The existing algorithm is CACLA [73, 71], introduced in 2007 by van Hasselt and Wiering. The novel algorithm is a batch variant of CACLA, dubbed NFCACLA. Inspired on an alternative method (NFQCA [26]).

Both CACLA and NFCACLA belong to the class of actor-critic models [54, 64]. This set of RL methods are characterized by the explicit distinction that is being made between the value and policy functions. These methods contain two separate instances, of which one, the actor, explicitly determines a policy independent from the value function. The other instance, the critic, then evaluates the actions that are performed by the actor. Please note that the critic can represent either the V- or the Q-function.

The first method to be discussed is the Continuous Actor Critic Learning Automaton (CACLA) [73, 71], in which the critic represents the V-function and an online learning mechanism is applied. Secondly the Neural Fitted Q-iteration for Continuous Actions (NFQCA) [26] will be discussed, which is an extension to the class of Neural Fitted algortihms, which ordinarily are unable to cope with continuous action spaces. In this method the critic represents a Q-function, and the training is performed in batches. Even though this method is not tested in this thesis, it has greatly influenced some of the design choices of the NFCACLA algorithm. Finally NFCACLA will be discussed, in which the critic represents the Q-function as well and the learning process is performed in batches.

3.2.1 Continuous Actor Critic Learning Automaton

In order to cope with continuous state and action spaces, the Continuous Actor Critic Learning Automaton (CACLA) method was introduced [73, 71]. This method uses function approximators to approximate both the V-function and policy function and has been shown to be efficient and effective, even when applied to problems in a non-continuous space [72].

The V-function and policy function are approximated by the weights of two artificial neural networks: the critic and the actor. The critic evaluates a given observed state s, determining the maximally expected gained reward (or minimal costs), while the actor greedily selects actions that yield an optimal evaluation by the critic. Once an action has been performed, the following state is observed. When this next state yields a higher V-value than the previous state the actor will be reinforced in its previous action and the critic will be updated with respect to the improvement correspondingly.

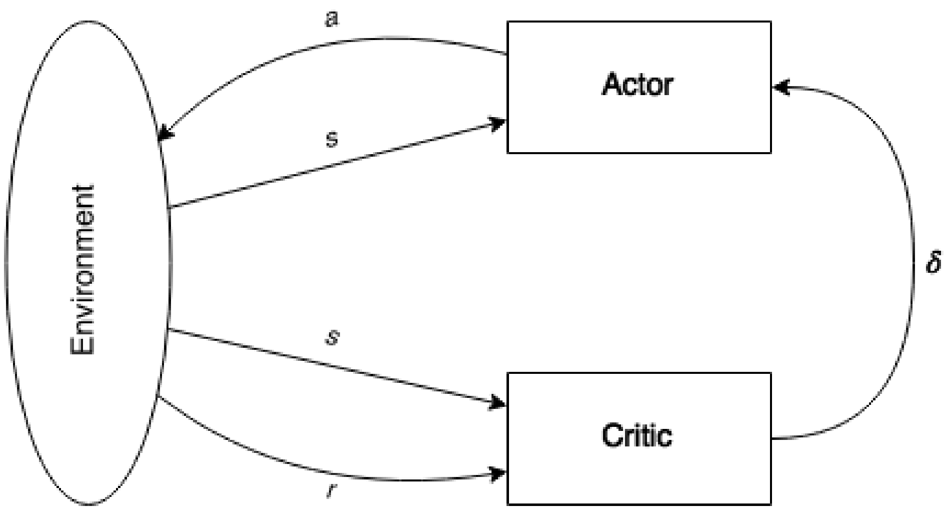


Figure 3.2: The interactions between the critic and actor modules and their environment for the CACLA algorithm. Straight lines represent input relations, while curved lines represent feedback relations. The critic and actor both observe a state s from the environment. The actor performs an action, which influences the state. The critic receives feedback from the environment in terms of a reward, and in turn sends the TD-error, δ, to the actor.

A graphical overview of the interactions between the actor, critic and their environment can be seen in figure 3.2. The mechanics behind the functioning of both the critic and actor will now be explained more thoroughly.

The Critic

Recall from the previous section that the V-function measures the reward that an agent expects to gain in the future, given its currently observed state s when following a policy π. The critic is a neural network that approximates the V-function for the current policy function. Now let the weights of this network be given by the vector θ^V, which can be referred to as the parameter vector.

It has been shown that within a discrete state space the V-function can be trained through Temporal Difference (TD) learning, according to the update rule provided in equation 3.13, in which α represents a learning rate such that $0 \leq \alpha \leq 1$. The factor δ_t is generally referred to as the TD-error, which yields the weighted change in the expected value with respect to the discount factor.

$$\begin{aligned} V_{t+1}(s_t) &= V_t(s_t) + \alpha\delta_t \quad \text{with} \\ \delta_t &= r_t + \gamma V_t(s_{t+1}) - V_t(s_t) \end{aligned} \tag{3.13}$$

Now, in a continuous state space, V is approximated by θ^V, and the update rule 3.13 needs to be updated accordingly. This is done by backpropagating the change in outcome of the critic back to its weights, yielding the update rule as given by equation 3.14. In this update step θ_i^V, t represents the i-th weight of the vector θ^V at timestep t, while $V_t(s)$ represents the output of the critic at timestep t, given the observed state s.

$$\theta_{i,t+1}^V = \theta_{i,t}^V + \alpha\delta_t \frac{\delta V_t(s_t)}{\delta\theta_{i,t}^V} \tag{3.14}$$

The Actor

Like the critic, the actor is defined as a neural network, this time with the weight vector θ^π as a function approximator for the policy function. Now let the action space $\mathcal{A}$ be given by equation 3.15, such that $a_i \in \mathbf{a}$ represents the i-th action from the action vector $\mathbf{a}$, with $1 \leq i \leq m$. The aim of the actor is to improve θ^π so that it approximates the optimal policy π^* as closely as possible.

$$\mathcal{A} = \{\mathbf{a} = \begin{pmatrix} a_1 \\ \dots \\ a_m \end{pmatrix} \in \mathbb{R}^m]\} \tag{3.15}$$

In order to improve the approximation the actor explores actions that are similar to the optimal action according to its current approximation. One of the ways it can do so is through Gaussian exploration: an action is selected from a Gaussian distribution $\mathcal{N}(\Pi_t(s_t), \Sigma)$, where $\Pi_t(s_t)$ denotes the output of the actor at timestep t, given observed state s_t, and Σ is the covariance matrix, which is a diagonal matrix, containing a parameter σ that contains the amount of exploration for each a_i. Under these conditions, the probability of an action a_i being selected at timestep t is given by equation 3.16. The notation $\Pi_{i,t}(s_t)$ represents the i-th outcome at timestep t of the actor, given the observed state s_t.

$$\pi_t(s_t, a_i) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(a_i - \Pi_{i,t}(s_t))^2/(2\sigma^2)} \tag{3.16}$$

In most circumstances it is desirable to define a decay factor for sigma $\Delta\sigma$, such that $0 \leq \Delta\sigma \leq 1$, with which σ is multiplied after each exploration step. Using this decay factor the exploration becomes more focused around $\Pi_t(s_t)$ while it becomes a more accurate approximation. Generally $\Delta\sigma$ is chosen to be close to 1.

After an action has been performed, it is evaluated by the critic by determining the TD-error δ_t. The actor then uses the sign of the TD-error to determine the influences of the action that it has performed. If $\delta_t < 0$, the action had a worse outcome than the optimal action. In this scenario it might not be desirable to train the actor, since decreasing the chance of this action being performed might increase the chance of even worse actions being performed. However, if $\delta_t > 0$ the action was an improvement of the current approximation and θ^π should be adjusted accordingly. This adjustment is done by training the neural network with an error of $E_t = (\mathbf{a}_t - \Pi_t(s_t))^2$. Through this method, the parameters θ^π are updated according to the update rule as given by equation 3.17

$$\text{IF} \quad \delta_t > 0: \quad \theta^\pi_{t+1} = \theta^\pi_t + \alpha(\mathbf{a}_t - \Pi_t(s_t))\frac{\delta\Pi_t(s_t)}{\delta\theta^\pi_t} \tag{3.17}$$

The Training Procedure

Now that the mechanics of both the critic and the actor for the CACLA method have been described, a final overview of the training procedure can be found in algorithm 1.

In order to find the closest approximation of the optimal policy function it is necessary to execute the training procedure until convergence. However, sometimes the function approximators reach a point where they are a 'close-enough' representation of the optimal policy function. In this case the weights of the function approximators still change, but these changes have no significant influences on the impact on the environment anymore. Hence continuing the training process becomes pointless. When this appears to be the case it is an option to define a max_Δ, which is small, and repeat the training until the changes in the weights become smaller than max_Δ. Another option to prevent extremely long training times is to simply limit the number of training epochs and stop training once this limit has been reached.

Algorithm 1 The training procedure for CACLA

```
Ensure: $\theta^{\pi}$: the function approximator that approximates the optimal
  policy function most closely
  $\theta^{V} \Leftarrow$ a neural network with randomly initialized weights
  $\theta^{\pi} \Leftarrow$ a neural network with randomly initialized weights
  $t \Leftarrow 0$
  repeat
      $\mathbf{a}_t \Leftarrow selectAction(\Pi_t(s_t), \Sigma)$
      $\delta_t = r_t + \gamma V_t(s_{t+1}) - V_t(s_t)$
      $\theta^{V}_{i,t+1} = \theta^{V}_{i,t} + \alpha \delta_t \frac{\delta V_t(s_t)}{\delta \theta^{V}_{i,t}}$
      if $\delta_t > 0$ then
          $\theta^{\pi}_{t+1} = \theta^{\pi}_{t} + \alpha(\mathbf{a}_t - \Pi_t(s_t))\frac{\delta \Pi_t(s_t)}{\delta \theta^{\pi}_{t}}$
      end if
      $t \Leftarrow t + 1$
  until convergence
  return $\theta^{\pi}$
```

3.2.2 Neural fitted Q-iteration for Continuous Actions

One of the more popular methods to solve the decision problem faster is the Neural fitted Q-iteration method [58]. However, one of the main shortcomings of this algorithm is that it can only cope with problems in a discrete action space. As a solution to this problem, the Neural fitted Q-iteration for Continuous Action (NFQCA) was proposed [26, 27, 59].

Traditionally the NFQ algorithm consists of a single neural network, with a finite number of m output nodes. Each output node represents the Q-value for one of the m discrete actions for a given input state. However, this method no longer suffices when the number of actions becomes infinite or the action space is continuous. In order to solve this problem, the NFQCA is designed as an actor critic model (AC-model). This AC-model consists of two neural networks, $\Pi(s, w_\pi)$ and $Q(s, a, w_q)$ respectively representing the actor and critic. Within these networks w_π and w_q represent the weights for the actor and the critic respectively. These networks are linked together, in such a way that the output of the actor is a part of the input of the critic.

Given a state space $\mathcal{S}$ and action space $\mathcal{A}$, given by a set of rational numbers, the interaction between the actor and the critic is as follows. The actor is assumed to approximate the optimal policy with respect to the current Q-function. Given an observed state s, the actor greedily finds the optimal action a, which is passed on to the critic. The critic then evaluates the action, given the current state, and propagates its feedback back to the actor through the means of gradient descent. Please note that due to the

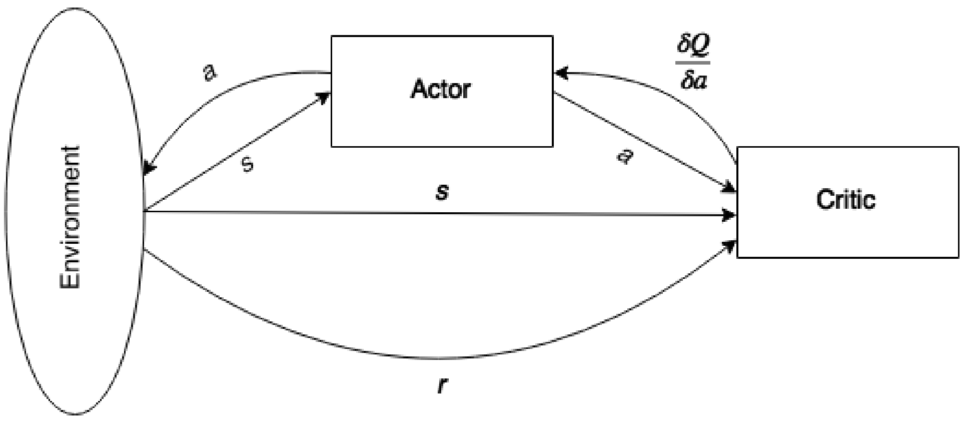


Figure 3.3: The interactions between the critic and actor modules and their environment for the NFQCA algorithm. Straight lines represent input relations, while curved lines represent feedback relations. The critic and actor both observe a state s from the environment. The actor performs an action, which is evaluated by the critic and influences the state. The critic receives feedback from the environment in terms of a reward, and in turn propagates this feedback back to the actor.

connection between the critic and the actor, the outputs of the actor are in the range $[-1, 1]$. If necessary these actions can be scaled to different ranges, however this is done outside of the scope of the neural networks. A graphical overview of the interaction between the critic and actor can be seen in figure 3.3. A more formal explanation of both the actor and critic will now be given.

The Actor

As mentioned above, the actor represents the policy function in the form of a neural network $\Pi(s, w_\pi)$ with weights w_π, assumed to yield the optimal action a given the current Q-function, observing state s. Hence at a given timestep t the output of the actor is assumed to be given by equation 3.18.

$$\Pi_t(s_t, w_\pi) = \arg\max_a Q_t(s_t, a, w_q) \tag{3.18}$$

The actor learns to approximate the optimal policy by maximizing the Q-function, under the assumption that the Q-function is unchanging. It does so by propagating its results, together with s_t forward through the critic. By propagating the results back through the critic to the actor, using gradient descent, the actor learns whether its action(s) yielded an improvement with respect to the current Q-function. Under the assumption that $\Pi_t(s_t, w_\pi)$ yields the action with the optimal expected value, at timestep t, it holds that the output of the Q-function is given by equation 3.19. Then, the output of the critic is propagated back to the actor, so we have a backpropagation step in which the number 1 is propagated back to the actor. According to the rules of gradient descent the partial derivatives of the Q-function with respect to the action inputs, as given by equation 3.20, are used to propagate the results of the actions back to the actor.

$$Q_t(s_t, \Pi_t(s_t, w_\pi), w_q) = \max_a Q_t(s_t, a, w_q) \tag{3.19}$$

$$\nabla Q_s(\Pi(s, w_\pi)) = \left(\frac{\delta Q(s, \Pi(s, w_\pi), w_q)}{\delta a_1}, \ldots, \frac{\delta Q(s, \Pi(s, w_\pi,), w_q)}{\delta a_m} \right) \tag{3.20}$$

Once the gradients have been determined for each of the action inputs of the critic, these gradients are propagated backward to the weights of the actor w_π. Hence, the update rule for the actor weights are given by equation 3.21. A complete overview of the training method for the actor is given by algorithm 2 (adapted from [26]).

$$\frac{\delta Q(s, \Pi(s, w_\pi), w_q)}{\delta w_\pi} = \frac{\delta Q(s, \Pi(s, w_\pi), w_q)}{\delta \Pi(s, w_\pi)} \cdot \frac{\delta \Pi(s, w_\pi)}{\delta w_\pi} \tag{3.21}$$

The Critic

The critic is the instance that provides feedback to the actor about the influences that its actions had on the world. Generally its purpose is to maximize an agent's utility gained (or minimize its costs). It is represented by a neural network $Q(s, a, w_q)$, where w_q represent the weights of the network.

The critic can be trained using any training strategy for neural networks, using a set of examples $\mathcal{P}$, containing the information of the expected utility for each action that has been performed by the actor. This expected utility is determined by equation 3.22.

$$\hat{Q} = (1 - \alpha) \cdot Q(s_t, a_t) + \alpha \cdot (r_t + \gamma Q(s'_t, \Pi(s'_t))) \tag{3.22}$$

Algorithm 2 Training procedure for the actor

```
Require: $Q$: a fixed representation of the Q-function, $\mathcal{D}$: a dataset on
  which the actor is to be trained
Ensure: $\Pi(s, w_\pi)$: the policy function, representing a greedy evaluation of
  the current Q-function,
  $\Pi_0 \Leftarrow$ a neural network with randomly initialized weight,
  $t \Leftarrow 0$
  repeat
    for $i = 0, \dots, \#\mathcal{D}$ do
      $a = propagate\_forward(\Pi_t, s^i)$
      $q = propagate\_forward(Q, s^i, a)$
      $\left(\frac{\delta q}{\delta s^i_1}, \dots, \frac{\delta q}{\delta s^i_n}, \frac{\delta q}{\delta a_1}, \dots, \frac{\delta q}{\delta a_m}\right) = propagate\_backward(Q, 1)$
      $\frac{\delta q}{\delta w_\pi} = \frac{\delta q}{\delta w_\pi} + propagate\_backward(\Pi, \left(\frac{\delta q}{\delta a_1}, \dots, \frac{\delta q}{\delta a_m}\right))$
    end for
    $\Pi_{t+1} \Leftarrow update\_weights(\Pi_t, \frac{\delta q}{\delta w_\pi})$
    $t \Leftarrow t + 1$
  until convergence
  return $\Pi_t$
```

The Training Procedure

The training procedure for the NFQCA consists of two different phases [27]: the observation phase and the learning phase. During these phases the actor and critic are viewed as two independent instances.

During the observation phase an agent simply follows its current action policy and adds its observed transitions, together with their rewards, to a database $\mathcal{D}$. The instances in this database are then used in the learning phase to train both the actor and the critic. This training is done by first determining the expected utility for each instance of $\mathcal{D}$, according to the rule given by equation 3.22. These expected utilities are stored in a new database $\mathcal{P}$, which is used to train the critic. An overview of the construction of $\mathcal{P}$ is given in algorithm 3 (adapted from [26]). Note that Q_0 means the Q-network at timestep 0, being the Critic as it was initialized, with out training and the critic as trained starting from its initial weights during each learning phase.

Once $\mathcal{P}$ has been obtained, the critic can be trained using any training method for neural networks, such as gradient descent or RPROP (which is used by [26, 27]). Finally the actor is trained on $\mathcal{D}$, with respect to the newly updated Q-function. An overview of the complete training procedure is given in algorithm 4 (adapted from [26]).

Algorithm 3 Construction of $\mathcal{P}$

```
Require: $Q$: the representation of the Q-function
Require: $\Pi$: the neural network, representing the greedy evaluation of $Q$.
Require: $\mathcal{D}$: a dataset containing all transitions from the observation phase
Ensure: $\mathcal{P}$: a dataset containing updated Q-values
  $\mathcal{P} \Leftarrow \emptyset$
  for all $d_i \in \mathcal{D}$ do
      with $d_i = \{s_i, a_i, s'_i, r_i\}$
      $\hat{Q} \Leftarrow (1-\alpha) \cdot Q(s_i, a_i) + \alpha \cdot (r_i + \gamma Q(s'_i, \Pi(s'_i)))$
      $\mathcal{P} \Leftarrow \mathcal{P} \cup ((s_i, a_i), \hat{Q})$
  end for
  return $\mathcal{P}$
```

Algorithm 4 an overview of the learning phase for NFQCA

```
Require: $Q_t$: the Q-network at timestep $t$
Require: $\Pi_t$: the policy network at timestep $t$.
Require: $\mathcal{D}$: a dataset containing all transitions from the observations
  phase
Ensure: $Q_{t+1}$: the updated representation of the Q-network
Ensure: $\Pi_{t+1}$: the updated representation of the policy network
  $\mathcal{P} \Leftarrow generateP(Q_t, \Pi_t, \mathcal{D})$
  $Q_{t+1} \Leftarrow train(Q_0, \mathcal{P})$
  $\Pi_{t+1} \Leftarrow trainActor(Q_{t+1}, \mathcal{D})$
  return $Q_{t+1}, \Pi_{t+1}$
```

In order to explore new actions, NFQCA is proposed to use an exploration method similar as in [73]. As a result, given the outcome of the actor $\Pi(s)$, the action that is performed is chosen to be $\pi(s)$ as given by equation 3.23.

$$\pi(s) = \Pi(s) + \mathcal{N}(0, \sigma) \tag{3.23}$$

3.2.3 Neural fitted Continuous Actor Critic Learning Automaton

In this section the Neural Fitted Continuous Actor Critic Learning Automaton (NFCACLA) is proposed. The NFCACLA variant aims to fit the simplicity of CACLA into the shape of a neural fitted batch algorithm. In this way its structure is highly similar to (and inspired on) the NFQCA algorithm, however it uses a more intuitive learning step for the actor, based on the learning method in CACLA, rather than using backpropagated feedback from the critic.

NFCACLA is built as an AC-model and uses Q-values to determine the expected result of each action (as in NFQCA) rather than V-values to determine the expected reward from a given state (as in CACLA). The model again consists of two neural networks $\Pi(s, \theta^{\pi})$ and $Q(s, a, \theta^{q})$ with the weights θ^{π} and θ^{q}, respectively representing the actor and critic. As in NFQCA, the networks are linked in such a way that the output of the actor is part of the input from the critic.

Given a state space $\mathcal{S}$ and action space $\mathcal{A}$, given by a set of rational numbers, the interaction between the actor and the critic is as follows. Given an observed state s, the actor determines the optimal action a according to its current policy. Afterwards it might, according to a predetermined exploration strategy, adjust this action to action to an action a' to explore alternative possible actions. The critic cetermines the expected reward of the selected action, given s. Whenever a' seems to yield a higher expected reward than a, the actor learns to perform a' rather than a, when in s. A graphical overview of the interaction between the actor and critic can be seen in figure 3.4. A more formal explanation of both the actor and critic will now be given.

The Critic

The critic in the NFCACLA algorithm functions the same as the critic in the NFQCA algorithm: it determines the expected discounted sum of rewards

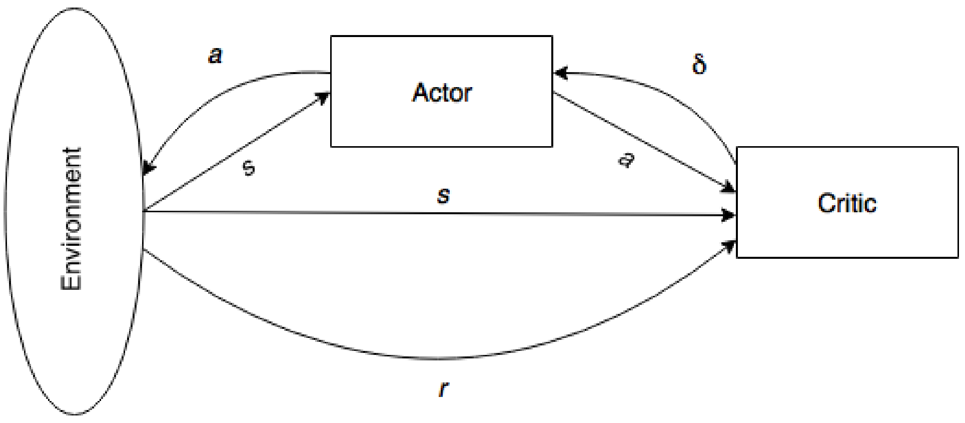


Figure 3.4: The interactions between the critic and actor modules and their environment for the NFCACLA algorithm. Straight lines represent input relations, while curved lines represent feedback relations. The critic and actor both observe a state s from the environment. The actor performs an action, which is evaluated by the critic and influences the state. The critic receives feedback from the environment in terms of a reward. Whenever an action improved the atcion that was previously thought to be optimal the actor adjusts its policy.

that a given action performed by the actor will yield. The critic is represented by a neural network $Q(s, a, \theta^q)$, where s, a and θ^q are the observed state, an action selected by the actor and the weights of the network, respectively.

The critic can be trained with any batch or online learning algorithm suitable for neural networks, given a dataset $\mathcal{P}$ containing information on the expected utility for a set of state-action pairs. This expected utility is determined in the same way as for the NFQCA algorithm, as given by equation 3.22.

The Actor

The actor is built as a neural network $\Pi(s, \theta^\pi)$, with weights θ^π that is assumed (or aims) to represent the optimal policy for a given Q-function (represented by the critic). Observing a state s, the actor yields an action a that is assumed to give the maximum amount of expected utility by $Q(s, a, \theta^q)$. Hence, the output of the actor at timestep t is given by equation 3.24

$$\Pi_t(s_t, \theta^\pi) = \arg\max_a Q_t(s_t, a, \theta^q) \tag{3.24}$$

The actor occasionally performs exploratory actions, deviating from the optimal policy. The frequency of exploratory actions can be determined with a parameter ϵ, such that $0 \leq \epsilon \leq 1$ and the variance of the exploratory actions can be determined with a parameter σ, such that an action selected by the actor is given by equation 3.25.

$$\pi_t = \begin{cases} \Pi_t(s_t, \theta^\pi) + \mathcal{N}(0, \sigma) & \text{if } \mathcal{U}(0,1) < \epsilon \\ \Pi_t(s_t, \theta^\pi) & \text{otherwise} \end{cases} \tag{3.25}$$

Whenever an exploratory action π_t yields an improvement in $Q(s, a, \theta^q)$ over $\Pi_t(s_t, \theta^\pi)$, the actor learns to prefer π_t over $\Pi_t(s_t, \theta^\pi)$. Whether an action is an improvement is determined in the same manner as in the CACLA algorithm: by determining whether $\delta_t > 0$. The update step that should be done to update the actor policy is given by equation 3.26 and the formal definition of the complete learning step is then given by equation 3.27.

$$\theta^\pi_{t+1} = \theta^\pi_t + \alpha(\pi_t - \Pi_t(s_t)) \frac{\delta \Pi_t(s_t)}{\delta \theta^\pi_t} \tag{3.26}$$

$$\begin{aligned} \text{IF} \quad & \delta_t > 0 : \\ \text{THEN} \quad & \theta^\pi_{t+1} = \theta^\pi_t + \alpha(\pi_t - \Pi_t(s_t)) \frac{\delta \Pi_t(s_t)}{\delta \theta^\pi_t} \end{aligned} \tag{3.27}$$

In contrast to NFQCA, NFCACLA is not trained until the point of convergence. Rather, the number of iterations is set to a relatively small number to prevent overfitting to a specific set of actions. This number of iterations is given by the parameter η and will be referred as the number of epochs or training iterations. Throughout this thesis the number of training iterations were kept the same for the critic and the actor. However, this is not a requirement.

A complete overview of the training procedure for the actor is given in algorithm 5.

The Training Procedure

The training procedure for NFCACLA is exactly the same as in the NFQCA algorithm. Again the procedure consists of an exploration phase and a training phase. During the exploration phase the current policy is followed, mixed with some exploratory actions. These actions are stored in a dataset $\mathcal{D}$, along with their transitions and rewards. During the training phase the expected utilities are determined in a database $\mathcal{P}$, as is shown by algorithm 3. After $\mathcal{P}$ has been constructed, the critic is updated with respect to $\mathcal{P}$ and

Algorithm 5 Training procedure for the actor in NFCACLA

```
Require: $Q$: a fixed representation of the Q-function,
  $\mathcal{D}$: a dataset on which the actor is to be trained
Ensure: $\theta^{\pi}_{i,t+1}$: the weights of the policy function, representing a greedy
  evaluation of the current Q-function,
  $\theta^{\pi}_{0}$ randomly initialized weights of the policy network, where $\theta^{\pi}_{i,t}$ represents
  the $i$-th weight at timestep $t$
  $t \Leftarrow 0$
  for $\eta$ iterations do
      for all $d_j \in \mathcal{D}$ do
          with $d_j = \{s_j, \pi_j, s'_j, r_j\}$
          $\pi = propagate\_forward(\Pi_t, s_j)$
          $\delta_t = r_j + \gamma Q(s'_j, \Pi_t(s'_j, \theta^{\pi}), \theta^{q}) - Q(s_j, \pi_j, \theta^{q})$
          if $\delta_t > 0$ then
              $\theta^{\pi}_{i,t+1} = \theta^{\pi}_{i,t} + \alpha(\pi_j - \pi)\frac{\delta \Pi_t(s_t)}{\delta \theta^{\pi}_{i,t}}$
          end if
      end for
      $t \Leftarrow t + 1$
  end for
  return $\theta^{\pi}_{i,t+1}$
```

the actor is then updated with respect to the newly updated Q-function. Since the procedure is the same as for the NFQCA algorithm, on overview can be found in algorithm 4.

3.3 Modeling the Trade of Flexibility

Now that a general descriptions of MDPs has been given, along with a number of methods of how to solve them, it is time to set up a framework to model the trade of flexibility. Recall that it's the goal of an agent to find a policy π^* that optimizes the cumulative reward that is obtained by the agent. In this section a definition will be given of the set of observable states (S), the set of actions (A) that can be observed and performed by an agent in order to do so. Additionally the reward function R will be defined that determines that reward that an agent will receive after performing an action. The transition function is unknown. Within the model each timestep is assumed to be on a moment at which a new imbalance price is made available. In the Netherlands this is done on a 15 minute interval, which is the interval that will be used in this thesis, so each timestep $t + 1$ occurs 15 minutes after timestep t.

3.3.1 Observed States

At each timestep t, the trading agent observes its current state. This state consists of three elements: (1) knowledge about the current state of the flexibility within the grid, in terms of demands of the devices that are connected to it; (2) weather forecasts for the next 24 hours, in order to take the production from renewable energy sources into account; and (3) the day-ahead schedule describing the load profile for the next 24 hours for the cluster.

The current state of the grid is provided by the aggregator (the auctioneer) in the form of an aggregated bid that stores the demand for all devices in the grid and the price they are willing to pay for their demand. These demands are expressed in watts.

The weather forecast contains information about the weather conditions that might influence the amount of production from RES. This forecast contains information on the expected temperature (expressed in 0.1°C), the amount of sunshine there will be (expressed in 0.1 hour per hour), and the strength (irradiation) of the sun (expressed in J/m^2). The forecasts are for the next 24 hours, on an hourly basis. When clusters would contain sources of wind electricity, additional information should be added.

The expected demand contains information about the complete demand profile for the cluster. This demand profile shows the amount of electricity that is expected to be consumed by the entire cluster and is also used by producers when trading on the different power markets. These values are, like the current demand, expressed in watts in an hourly resolutions.

In total this results in 196 features (100 for the current grid state and 24 for each of the other features, each value describing one of the next 24 hours). Each of these features will have to be normalized to a value in the range of $[-1, 1]$. Given that the observation of one of the above mentioned features f lies in the range $[f_{min}, f_{max}]$, an observed measurement $f_x \in [f_{min}, f_{max}]$ is normalized according to equation 3.28. The values for f_{min} and f_{max} will have to be defined for each of the features.

$$2\frac{f_x - f_{min}}{f_{max} - f_{min}} - 1 \tag{3.28}$$

For the day ahead schedule and the aggregated bid, the values for f_{min} and f_{max} depend on the size of the grid, since they are bound by the maximum amount of electricity that could be supplied and demanded by the cluster. For the aggregated bid we found that the extreme values could be normalized using $f_{max} = 1000h$ and $f_{min} = -1000h$, where h is the number of households that is being simulated. These findings are based on a number of simulations with a small number of households and connected devices.

Table 3.1: The minimum and maximum values as observed for each of the weather features, together with the values between which they are to be maximized

feature	**observed minimum**	**observed maximum**	$\mathbf{f_{min}}$	$\mathbf{f_{max}}$
temperature	-46	174	-50	175
solar exposure time	0	10	-1	11
solar irradiation	0	250	-1	251

In order to create the day ahead schedule a simulation was run in which devices were not steered by PowerMatcher controllers, but rather with traditional controllers that are not steered through dynamic pricing. The day ahead schedule was then given by the total allocation of all devices in the cluster. This yielded a demand profile with values ranging between -36565 and 61000 watts, which are being normalized using $f_{min} = -40000$ and $f_{max} = 65000$.

The values of f_{min} and f_{max} for the weather-related features can be found in table 3.1. The temperature is normalized with respect to a minimum value of $-5°$C and a maximum value of 17.5°C, which are slightly more (and less) than the extreme temperatures for the month of March, in which all simulation are run. The maximum solar irradiation value has been set at a little over the highest observed value, in order to achieve a maximum distinction, while maintaining a little room for fluctuations. In the simulations that were run for this project there was no supply from wind power. In simulations where this will be the case, wind velocity should be included as a feature as well.

3.3.2 Actions

When trading flexibility the actions that are performed by an agent can not be considered actions in a traditional sense: they are rather boundaries within which can be traded. The trades are made in response to the imbalance market, which makes it fair to assume that whatever trade the TSO wishes to make is at a price that is better than the price on the day ahead market. As a result an agent would always wish to trade as much as it can. However, the question remains: how much can the aggregator trade, without compromising the stability of his own cluster? The actions of the agent are thus the boundaries within which the agent can trade, without compromising this stability.

More formally, the actions that an agent performs at each timestep is determining the ramp up and ramp down boundaries, given the observed state.

The set of all actions is then given by $A = \{a_u, a_d\} \in \mathbb{R}^2$, where a_u represents the ramp up capability and a_d represents the ramp down capability. These boundaries are constrained to both an upper and lower value. The lower value for both a_u and a_d are given by $a_u, a_d > 0$. The upper boundaries however are not fixed but rather depend on the current grid state. As mentioned in the previous subsection, the grid state is represented by the aggregated bid. Given an aggregated bid B in a situation without imbalance, with minimum demand $b_{min} < 0$ and maximum demand $b_{max} > 0$ the upper boundaries of the actions are given by $a_u < -b_{min}$ and $a_d < b_{max}$. Whenever an agent wishes to select actions outside of these boundaries, the actions are restricted to the boundaries and the agent receives a penalty, as explained in the following section.

Since the actions are rather boundaries than actions, state transitions do not follow directly from the actions that are the output of an actor. They rather follow from the trade that is being made with the TSO, which is being constrained by the ramp up and down boundaries provided by the agent.

3.3.3 Rewards

The reward that an agent receives for trading flexibility consists of two factors: (1) the reward that it receives by trading electricity and (2) the penalty that it receives for offering too much of its flexibility. All terms are based on amounts of electricity, in watts.

The first component is simply given by the amount of watts that the agent traded with the TSO during a single settlement period and is referred to as W^+. The penalty that an agent receives is based on the amount of imbalance that occurs within the cluster during a settlement period due to deviations from the day-ahead schedule and is referred to as W^-. The height of both factors are known only after each settlement period.

As mentioned in the previous section, an additional penalty is given when an agent wishes to offer any flexibility outside of the boundaries of the grid state. This penalty is based on the amount of watts that the agent exceeds the boundary with, influenced by a parameter β that determines the severity of exceeding the boundaries. Let the amount of watts with which the agent exceeds its boundaries be C, then the total reward that an agent receives after a given settlement period is given by equation 3.29. Throughout this thesis β was set to 1, and will not be mentioned and used any further.

$$r_t = W_t^+ - W_t^- + \beta C_t \tag{3.29}$$

Chapter 4

Validation

In order to validate the different learning techniques, they were implemented as an objective agent (see chapter 2) for an internal TNO tool used for simulating (smart) electricity grids. The objective of the agents was to steer the allocation within the cluster (through volume steering) in such a way that devices would consume more or less electricity whenever a trade was to be made on the reserve market. The agents are validated using the PowerMatcher software, but can be applied to other smart grid technologies as well.

In this chapter a description will be given of how the different algorithms are validated. First an overview will be given on the different conditions that are being tested for each algorithm. All experiments are to be done in the same smart grid scenario, which will be described in the section thereafter. Finally a description is given on the criteria on which the algorithms are validated.

4.1 Experimental Setup

In order to fully test both algorithms four series of experiments were run. The first series is to determine a baseline demonstrating scenarios in which no flexibility is offered, or simple strategies are used to determine how much of the flexibility is offered. Two series of simulations (one using CACLA and one using NFCACLA) were run in order to see how robust the algorithms are to noise in weather predictions. Finally a series of simulations was run using NFCACLA to study the influence of the number of repetitions that were used to train the actor and critic. Each of these series will be explained shortly.

The algorithms were trained and tested in the month of March (from March

1st until March 30th). In order to train each algorithm, simulations were run over the period from March 1st until March 28th (27 days, 90%). The algorirthms were tested over the remaining 3 days (March 28th until March 30th, 10%). The baseline simulations were only run over the latter three days.

4.1.1 Baseline Scenarios

A number of cases is examined to determine a baseline to compare the algorithms with. The most basic case is a scenario in which no PowerMatcher agents are used. This scenario is referred to as the *business as usual* scenario. Furthermore a scenario is run in which the grid consists of devices using PowerMatcher controllers, but no flexibility is offered. Finally a number of situations is looked at in which flexibility is offered according to simple, straightforward strategies. The strategies to be tested are simply offering a fixed percentage of both ramp up and down flexibility, ranging from 10% to 100% in steps of 10%. In the remainder of this thesis the scenario in which the devices are equiped with PowerMatcher controllers, but don't offer any flexibility will be considered as the strategy of offering 0% of tis flexibility.

4.1.2 Variation Over Noise

In order to examine the robustness of the algorithms with respect to noisy (or faulty) weather forecasts a number of simulations were run for both the CACLA and NFCACLA algorithm, using four different levels of noise: (1) 0% (perfect forecasts), (2) 1%, (3) 5% and (4) 10%. Both algorithms were run with the parameters as shown in table 4.1. These parameters were kept constant throughout the different simulations per algorithm. Due to the large amount of time it takes to train the algorithms no parameter tuning was done for any of the algorithms, so performances for each of the algorithms might not be optimal. The critic and actor in the NFCACLA algorithm were trained for iterations per repetition.

4.1.3 Variation Over Iterations in NFCACLA

The final series of simulations run is to determine the effects in NFCACLA of training the actor and critic for a different number of iterations during each training step. Scenarios using four different numbers of iterations were examined: (1) 1 iteration, (2) 5 iterations, (3) 10 iterations and (4) 25 iterations. The algorithms were run using the same parameter settings as the previous experiments, as shown in table 4.1 and with perfect weather forecasts.

Table 4.1: parameters that were used for both actors and critics in the different reinforcement learning algorithms. Note that the CACLA algorithm always selects an action within a certain range, so no exploration chance needs to be defined.

	CACLA		**NFCACLA**	
Parameter	*Actor*	*Critic*	*Actor*	*Critic*
learning rate (α)	0.01	0.01	0.01	0.01
discount (γ)	0.98	0.98	0.98	0.98
exploration chance (ϵ)	-	-	0.5	0.5
exploration factor (σ)	0.05	0.05	0.05	0.05
# input nodes	196	196	196	198
# hidden nodes	50	50	50	50
# output nodes	2	1	2	1

4.2 Validation Scenario

All simulations are run in a scenario containing 100 households. Each household is equipped with a PV panel and either a heat pump (80 households) or a micro-CHP (20 households). Both the heat pumps and micro-CHPs have heat buffers of 100 liters for both space heating and tap water, allowing flexibility in their operations. The PV-panels don't add any flexibility to the grid but produce a certain amount of electricity, depending on the irradiation by the sun. In this way production and (in response) demand are dependent on weather conditions in the area.

In order to simulate the demand from the imbalance market in a realistic manner historical data for the year 2013, published by the Dutch TSO TenneT [69], was used. Given that the simulations that were run contain only a limited number of households, the data were scaled down from 8.6 million households to 100 households. Furthermore the data were scaled with another factor of 0.22 to account for the lack of industrial supply and demand in the simulation [40]. Weather conditions were simulated using historical data from the Royal Netherlands Meteorological Institute (KNMI) from the year 2011 [1]. These weather conditions were used as forecasts as well, with a predetermined amount of noise added to them. The demand profiles for the heat pumps and micro-CHPs were adopted from MacDougall, Warmer and Kok [40].

[1]`http://knmi.nl/`

4.3 Validation Criteria

All simulations are compared on the basis of two different criteria. The first criterion is the amount of electricity that the cluster traded with the TSO in order to restore imbalances. The second criterion is the amount of imbalance that occurred within the cluster. Both criteria are expressed in watt. The algorithms are to be compared on the basis of both criteria separately and combined. Note that in the 'business as usual' and 'PowerMatcher base' simulations no flexibility is offered to the TSO. These simulations merely function as a baseline for the amount of imbalance that would occur when no electricity would be traded.

The algorithms are tested under real-world circumstances. This means that the amount of ramp up or ramp down capacity that is requested by the imbalance market is determined by historical data of the imbalance in the Netherlands in 2011, published by the Dutch TSO Tennet [69].

In order to express the performances of the algorithms in terms of benefits for the end-users an additional evaluation will be made in terms of financial profits or costs that were made by trading with the TSO and the internal imbalances. However, these performances are not used to compare the algorithms since the imbalance prices are unknown to agents at the time that they offer their flexibility. In this case the agents would be evaluated positively when they would coincidentally trade too much energy when the prices are highly favorable, followed by a long term situation of imbalance while the imbalance prices are not too extreme. In this situation the agent itself could make a profit without actually improving (or possibly even worsening) the overall imbalance over the grid, which is undesirable. The benefits and costs will be based on historical prices form the APX day-ahead market [2] and imbalance prices set by the Dutch TSO [69] in March 2011 and expressed in euros.

[2]http://apxgroup.com/

Chapter 5

Results

In this chapter the outcomes of the different simulations will be shown. Initially the results of the baseline scenarios are shown. Secondly, the results of the different experimental scenarios are shown on an individual basis. Finally, some general observations over all scenarios will be discussed.

As explained in the Validation section, the main criteria that is to be evaluated is the amount of imbalance that is resolved and caused by the cluster. Whenever there is an imbalance, it's determined whether this imbalance is in the same direction as was required by the TSO. When the directions are the same, the imbalance had a positive effect on the overall imbalance, and is thus considered as resolving external imbalances. When the directions are different the imbalance is contributing to the net imbalance and this causes additional imbalances. The second validation criteria are the financial consequences of participation in the imbalance market. This is determined by the money that is paid or obtained by buying or selling electricity on the balancing market. Note that the costs for trading electricity on the other markets is not included, since this is not actively done by the agent. In the scenario in which the agents were tested the costs for trading on the spot market amounted to €32.96. Recall that the tests were performed in the period of March 28th until March 30th of the year 2011 in a scenario containing 100 household each equipped with a PV panel and either a heat pump or a micro-CHP.

For each of the simulations the available flexibility will be shown, together with the amount that is offered to the TSO and that is finally used to steer the cluster. Furthermore, the resulting allocation will be examined and compared with the load profile, together with the amount of caused and resolved imbalance for each settlement period. Additionally some results can be found in the appendices on the amount of imbalance that is resolved and caused over the course of the simulations, together with the allocation, and

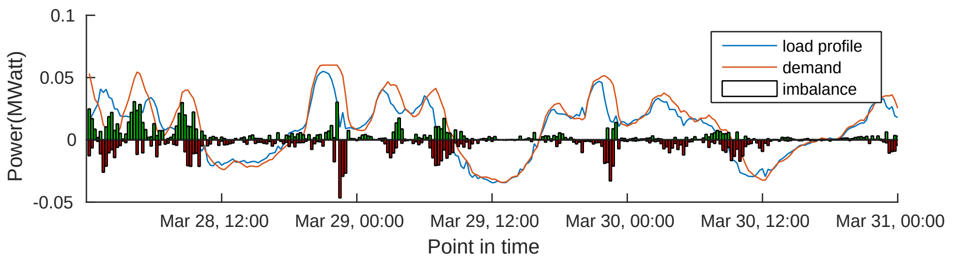


Figure 5.1: The allocation during the business as usual scenario versus the load profile

the amount of flexibility that is remaining. Note that any small discrepancies between mentioned values might be caused by rounding. An overview of all results together can be found in appendices A and B.

5.1 Baseline Simulations

5.1.1 Business as Usual

The business as usual scenario is used as a reference to the current real world situation: no smart grid controllers are used. This means that all devices either run or don't at each given moment and no flexibility is available. The supply and demand are not steered in any way and the allocation is directly compared to the load profile. Figure 5.1 shows the allocation versus the load profile for this scenario.

A total of 990.43 kW of imbalance is resolved during this simulation while an additional 1152.68 kW of imbalance is caused. This means that over three days a net imbalance is caused is 162.25 kW. The costs that have been made in order to resolve the internal imbalances are €32.14, while the gains of resolving imbalances are €7.19, yielding a net loss of €24.95.

5.1.2 Straightforward Strategies

Table 5.1 shows an overview of the electrical results (i.e. the resolved and caused imbalance together with the difference between the two). In the table can be seen that generally the amount of imbalance that is resolved increases when an increasing amount of flexibility is offered, up unto the point where around 50% of the available flexibility is offered, after which this amount remains at the same level. Surprisingly, however, the amount of imbalance that is caused remains fairly constant when the amount of

Table 5.1: The results in KWatt for offering no or fixed percentages of the available flexibility.

% offered	**imb. resolved**	**imb. caused**	**net difference**
0%	571.52	620.35	-58.85
10%	662.30	579.82	82.41
20%	681.71	555.32	115.66
30%	682.71	585.46	197.25
40%	741.33	501.98	239.34
50%	751.56	507.15	244.42
60%	753.58	509.67	243.92
70%	714.14	555.50	158.63
80%	723.55	522.54	201.01
90%	710.97	546.63	164.34
100%	708.50	526.46	182.04

offered flexibility increases. Due to a combination of a rather high amount of resolved imbalance but a relatively low amount of caused imbalance, the optimal amount of flexibility offered appears to be around 50% (ranging between 40% and 60%) of the available flexibility.

Table 5.2 shows an overview of the financial costs and benefits of offering fixed amounts of flexibility. In concurrence with the electrical results, the gains increase when the amount of offered flexibility increases, while the costs decrease. Again the optimal amount of offered flexibility appears to lie around 50% (and again ranging around 40% and 60%) of the offered flexibility. Noteworthy is that, while the financial results mainly follow the same pattern as the electrical results, less profit is made when offering 30% than when offering 20%, while the amount of net resolved imbalance is much higher.

Figures 5.2 to 5.5 show the available and offered flexibility and the amount of steering that is done by the objective agent, while figures 5.6 to 5.9 show the allocation versus the day ahead schedule, together with the amount of caused and resolved imbalances. The results are shown for the scenarios where 0%, 50%, 80% and 100% of flexibility is offered respectively. The flexibility plots for the scenarios in which other percentages are offered are similar and can be found in appendix C. Note that in order to highlight that the offered boundaries are the same as the capacity, the capacity is dashed in the scenario in which all flexibility is offered.

In the plots that concern a scenario in which any flexibility is offered the available boundaries are shown by thick lines, where the blue line represents the available flexibility that can be used to ramp down, while the red line

Table 5.2: The financial results (in euros) for offering no or fixed percentages of the available flexibility.

% offered	**gain**	**costs**	**net profit**
0%	€12.76	€9.07	€3.69
10%	€14.62	€9.65	€4.97
20%	€16.39	€17.33	€9.06
30%	€16.33	€8.54	€7.79
40%	€17.93	€6.11	€11.82
50%	€18.29	€5.97	€12.32
60%	€18.44	€6.14	€12.31
70%	€17.91	€7.67	€10.24
80%	€18.23	€6.94	€11.29
90%	€18.03	€6.93	€11.10
100%	€17.61	€6.78	€10.83

shows the available flexibility to ramp up. The boundaries that are offered by the aggregator are depicted by the thinner lines in the corresponding color. The green line shows the amount of steering that is performed by the agent. Since no boundaries are determined when no flexibility is offered, these lines have not been thickened even though they represent the available flexibility, rather than boundary conditions. Note that traditionally ramping up traditionally means increasing production (or decreasing demand) for the balancing market, while ramping down means lowering production (or increasing consumption). Since the bids done by the agent express the demand in the grid in order to be able to present this ramp up and ramp down flexibility, the ramp up values are negative (the demand is decreased) and the ramp down values are positive (the demand is increased).

The most notable observation from these figures is that the amount of electricity that the cluster is steered with is only a small amount of the flexibility that is offered. It can as well be seen that the resulting deviations from the day ahead schedule are only small. Additionally it can be observed that the available flexibility is hardly influenced due to the small amount of steering that is done.

5.2 Variations Over Noise in Weather Forecasts

5.2.1 CACLA

Tables 5.3 and 5.4 show the electrical and financial results respectively for the simulations that were done with CACLA with different levels of noise in

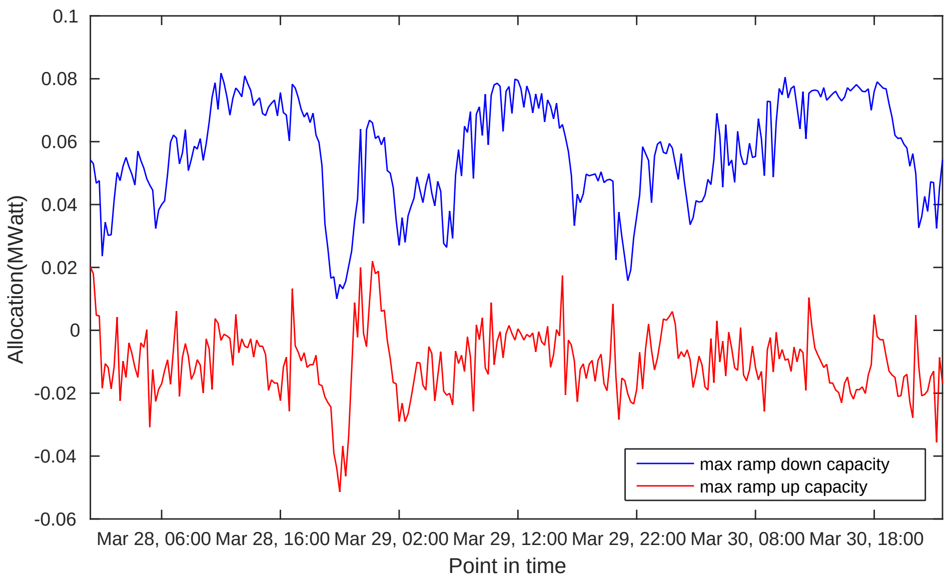


Figure 5.2: Available flexibility

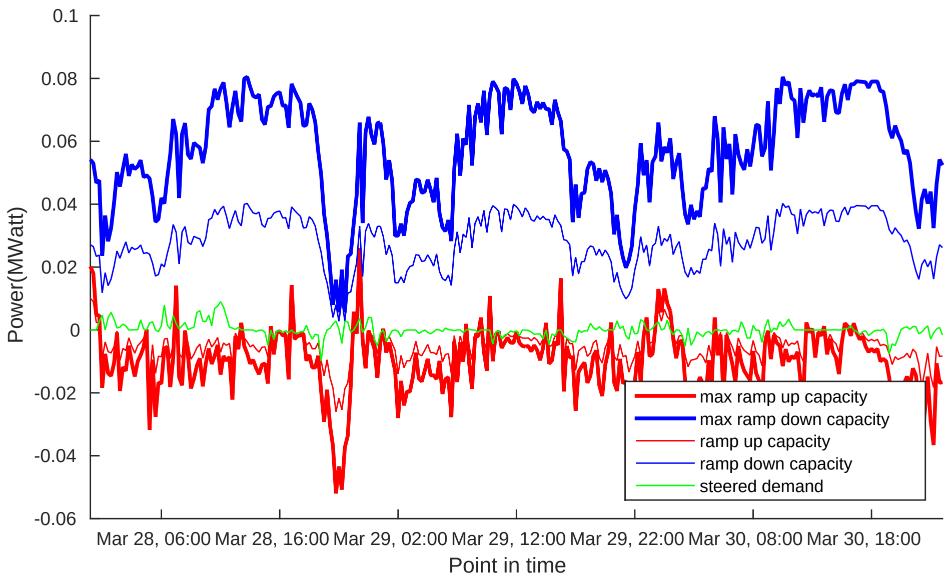


Figure 5.3: Available flexibility, offered flexibility and the adjustment made by the objective agent

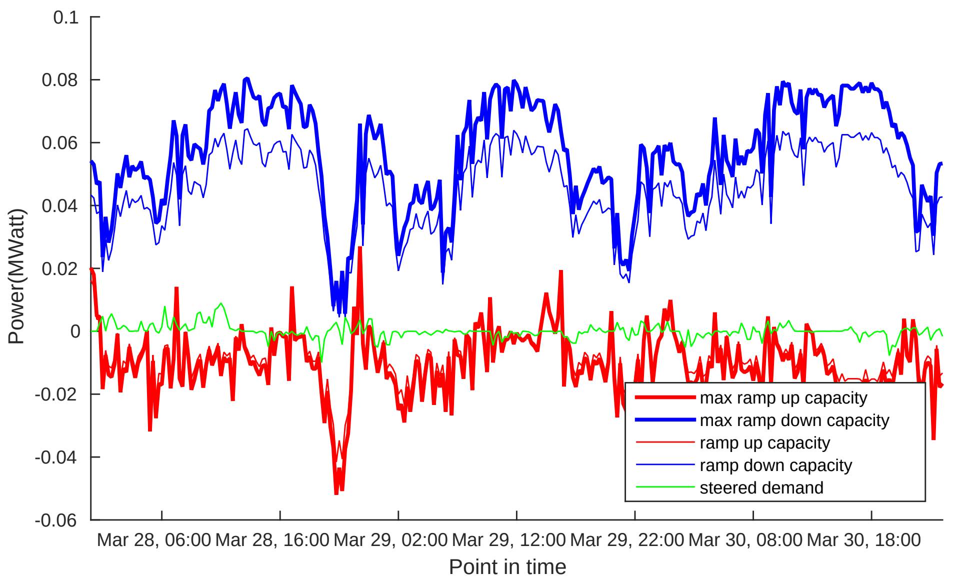


Figure 5.4: Available flexibility, offered flexibility and the adjustment made by the objective agent

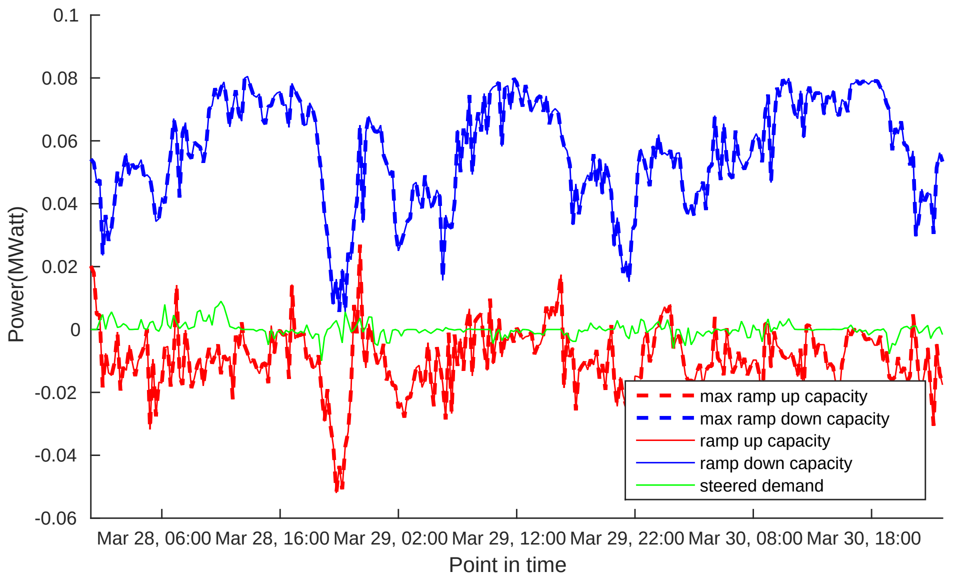


Figure 5.5: Available flexibility, offered flexibility and the adjustment made by the objective agent

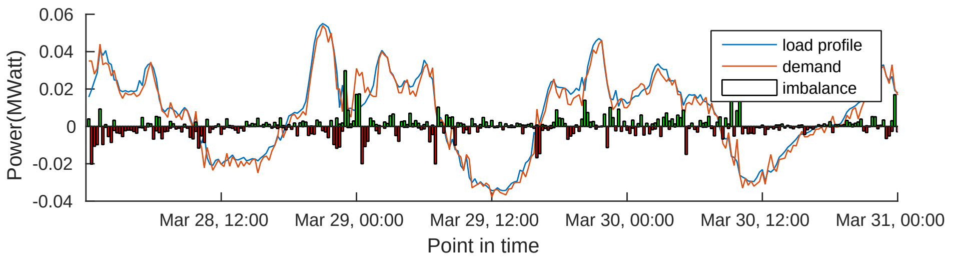


Figure 5.6: Behavior of a cluster using an objective agent that doesn't offer any flexibility to the TSO

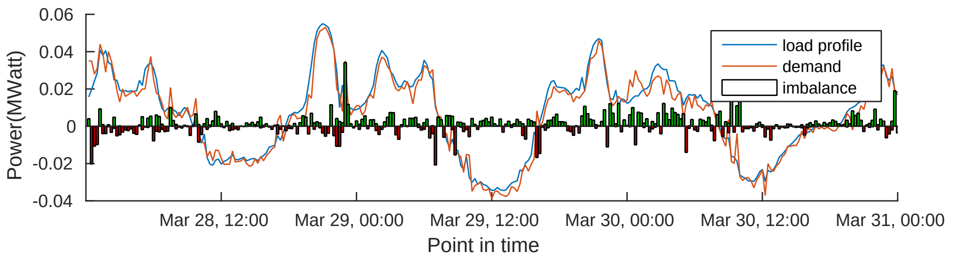


Figure 5.7: Behavior of a cluster using an objective agent that offers 50% of its available flexibility to the TSO

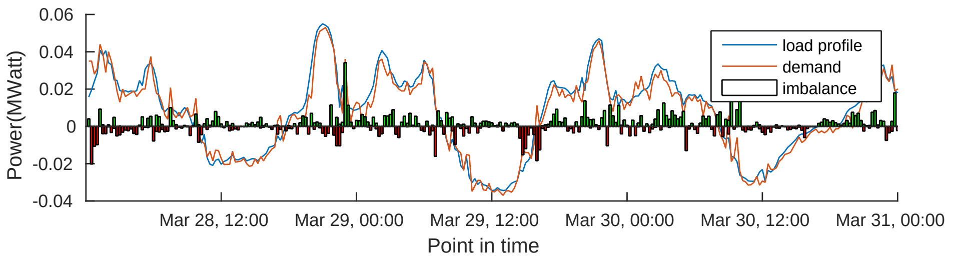


Figure 5.8: Behavior of a cluster using an objective agent that offers 80% of its available flexibility to the TSO

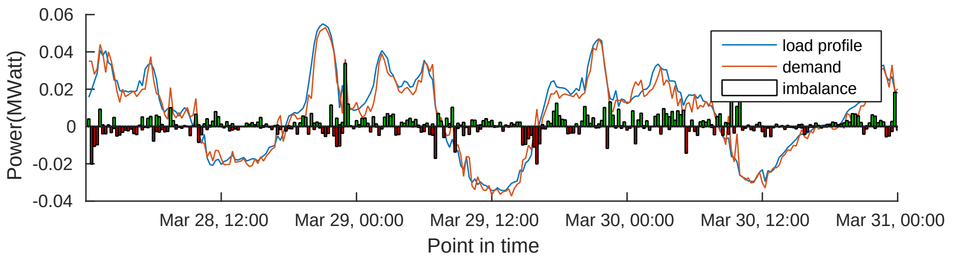


Figure 5.9: Behavior of a cluster using an objective agent that offers all of its available flexibility to the TSO

Table 5.3: Results for CACLA in terms of resolved external imbalances and caused internal imbalances and their net difference, expressed in kW.

noise	**imb. resolved**	**imb. caused**	**net difference**
0%	705.05	545.64	159.42
1%	657.00	558.06	98.94
5%	734.21	552.96	181.25
10%	664.12	620.11	44.01

Table 5.4: Results for CACLA in terms of money that is made and/or saved by offering flexibility and the costs that were made by additional deviations from the day ahead schedule, together with the net profit that is made.

noise	**gain**	**costs**	**net profit**
0%	€18.04	€7.35	€10.69
1%	€15.18	€6.85	€8.33
5%	€17.85	€7.41	€10.43
10%	€17.13	€8.73	€8.40

the weather forecasts that were used by the agents to determine the optimal amount of flexibility they were to offer. Surprisingly, when looking at the electrical results, no pattern can be seen with the increase in noise: in the scenarios with 0% and 5% noise the results lie between the baseline scenarios where 20% and 30% of the available flexibility is offered. On the other hand, the results for the cases with 1% and 10% noise performed as if no more than 10% of flexibility is offered. More details on the development of these results over the course of their respective simulations can be found in appendix A. However, when compared to the baseline results, the financial results of the CACLA run are slightly better: the profit made in the cases with 0% and 5% noise are comparable to the baseline scenarios where 70% and 100% of the available flexibility is offered, while the results of the other two cases lie somewhere between the baseline scenarios where 20% and 30% were offered.

In figure 5.10 the development of the cumulative reward over the training period is shown for the different noise levels in the different CACLA experiments. In all simulations the agent starts off with a huge penalty. Due to their initialization the agents they might initially try to offer flexibility outside of the specified boundaries, which is penalized heavily. Independent of the noise levels the agents quickly learn to correct these mistakes and start to make more efficient bids. This process seems to go slightly slower for the scenarios where the weather forecasts are more noisy, especially in the scenario with 10% noise. However, it does not appear that the CACLA agents are easily able to find an optimal solution and keeps on searching for improvements, yielding peaks and dips in the cumulative reward. It is

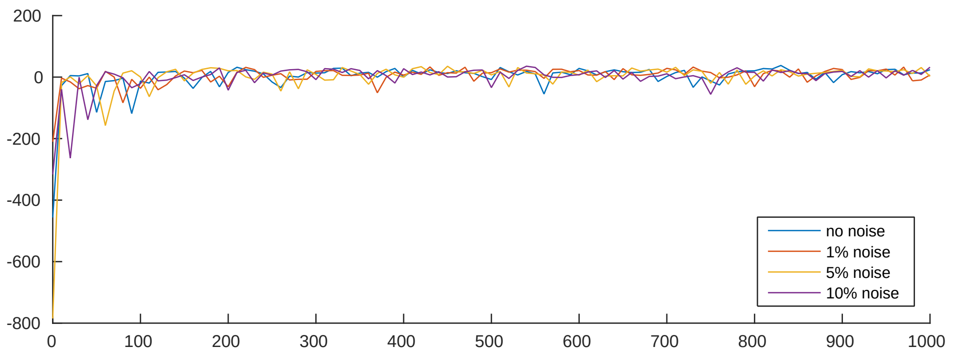


Figure 5.10: The development of the acquired cumulative reward over an entire simulation during the training progress for the CACLA algorithm with noise added to the weather forecasts.

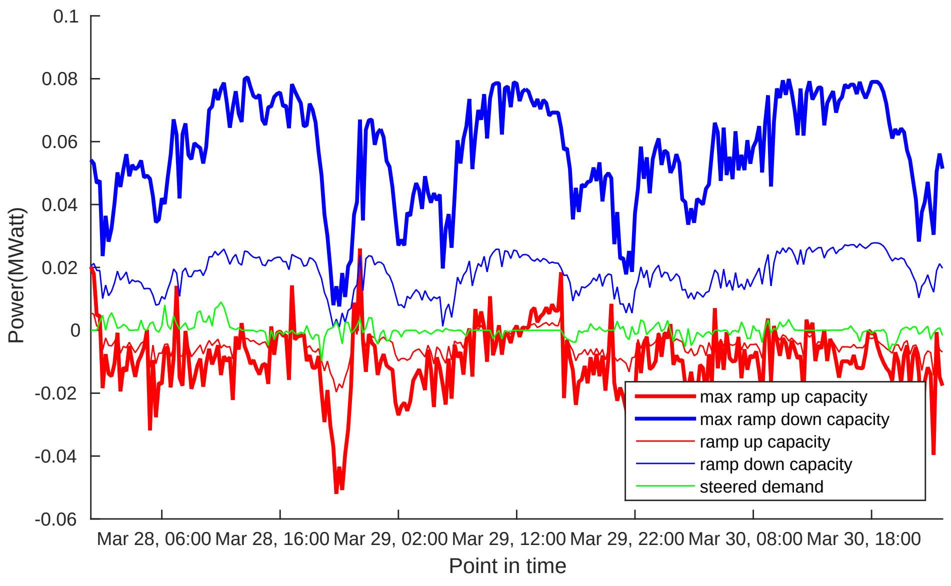
interesting to note that for the 1% scenario the cumulative reward seems to be in such a dip by the end of the training period. This might be the reason that the results of this scenario are less positive than the results of the other scenarios.

Figure 5.11 shows the behaviour of the agents for the different noise levels. Again the thick line represent the current available boundaries, while the thin (red and blue) lines show the corresponding boundary conditions that were given by the agent. The green line shows the amount of ramp up or down capacity that is required by the TSO in the end. In the scenarios with 0% and 5% noise the agents behave similarly, offering little more than 0.02 mW, while in the other two scenarios more flexibility is offered (in the scenario with 10% noise a little more than in the 1% scenario). This contrasts with the results of the baseline experiments, where offering a larger amount (at least until 50% of the available flexibility) appeared to be beneficial.

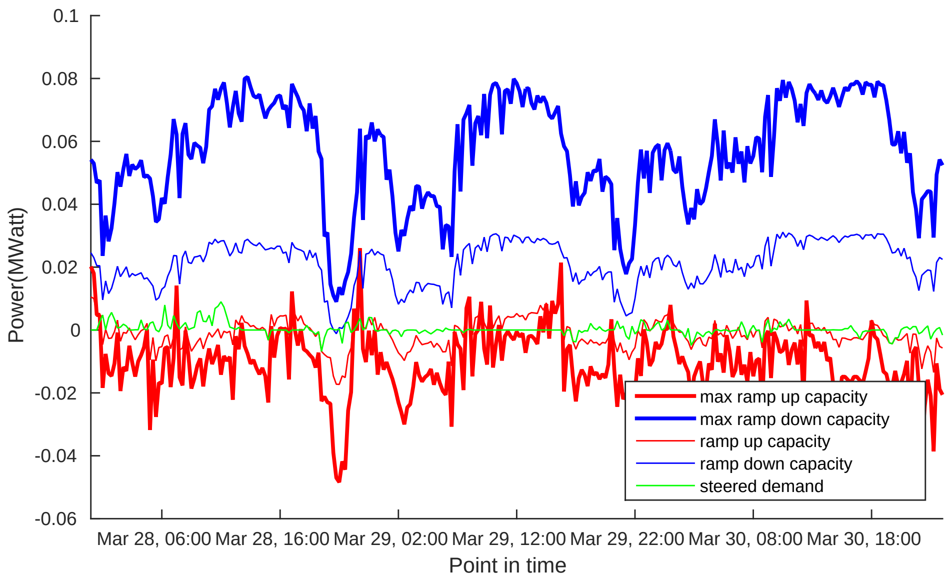
Finally, figure 5.12 shows the allocation for the CACLA agents with the different noise levels in comparison with the day ahead schedule. It can be seen that only small deviations are made at all points in time and that all deviations are similar for all experiments. Generally the deviations are the result of a lower demand than expected. A little after the 29th of March, 12.00h there is a moment with a large amount of imbalance. This period of imbalance seems to be more heavy in the 10% noise case than in the other cases.

5.2.2 NFCACLA

Tables 5.5 and 5.6 show the electrical and financial results respectively for the experiments with noise variations for the NFCACLA algorithm. Both

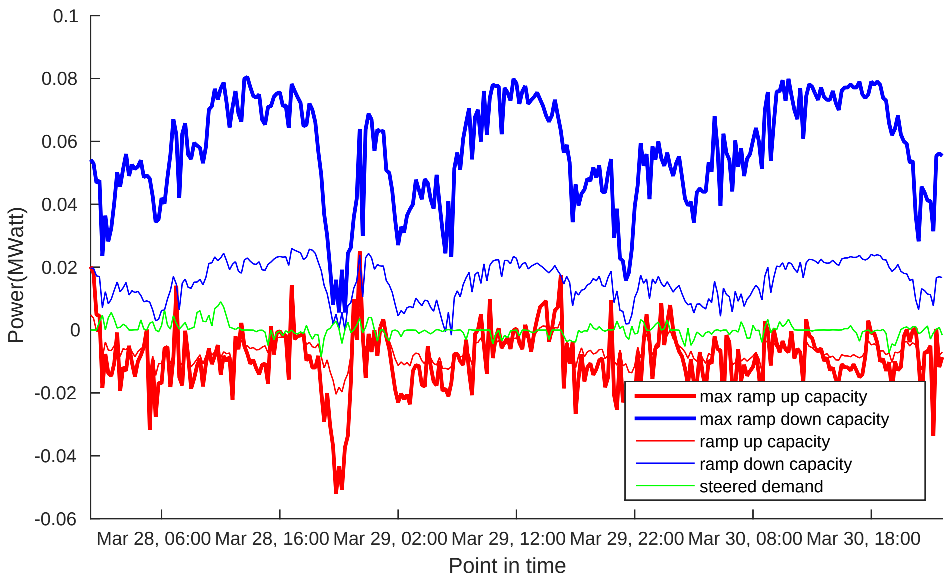


(a) no noise

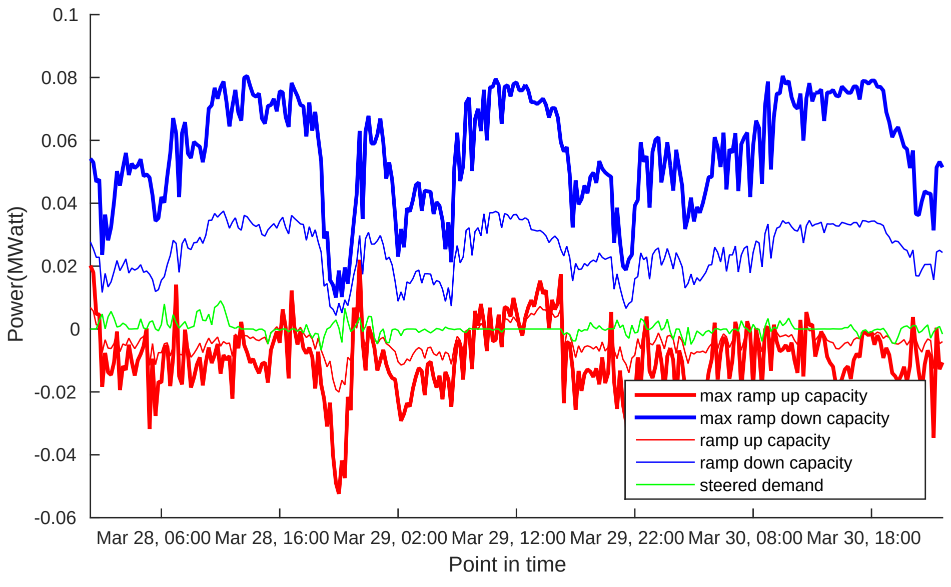


(b) 1% noise

Figure 5.11: Available flexibility, offered flexibility and the adjustment made by the objective agent using the CACLA algorithm with different levels of noise in the weather forecasts.

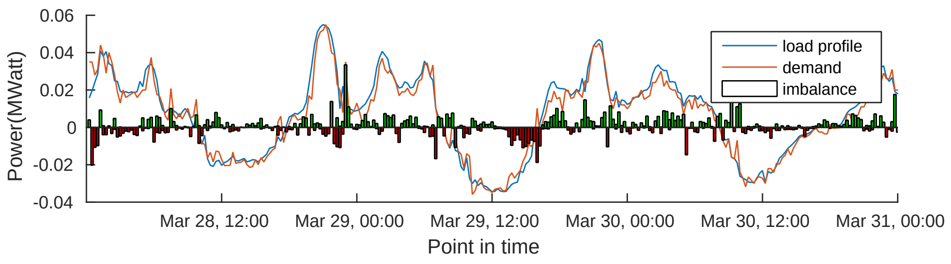


(c) 5% noise

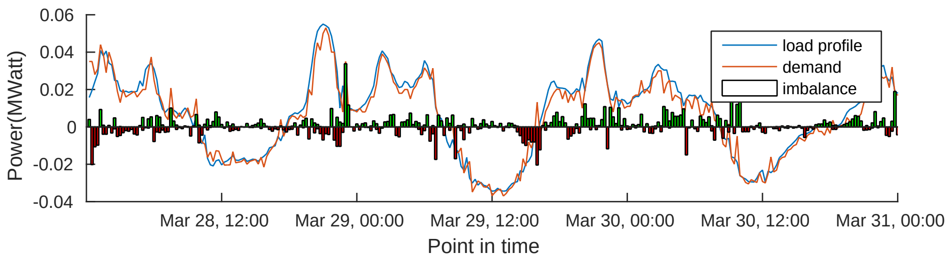


(d) 10% noise

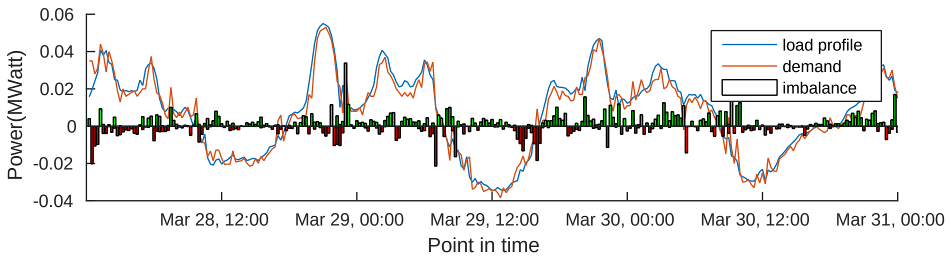
Figure 5.11: Available flexibility, offered flexibility and the adjustment made by the objective agent using the CACLA algorithm with different levels of noise in the weather forecasts.



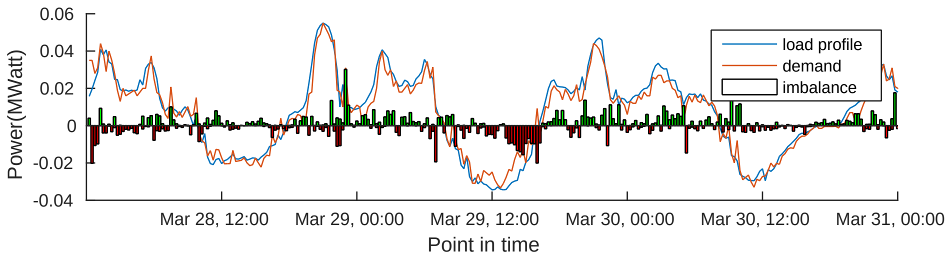
(a) no noise



(b) 1% noise



(c) 5% noise



(d) 10% noise

Figure 5.12: Allocation versus the day ahead schedule for the different CACLA experiments.

Table 5.5: Results for NFCACLA in terms of resolved external imbalances and caused internal imbalances and their net difference, expressed in kW.

noise	imb. resolved	imb. caused	net difference
0%	577.41	383.68	193.72
1%	572.64	419.57	153.07
5%	583.47	448.75	134.73
10%	580.57	446.52	134.02

Table 5.6: Results for NFCACLA in terms of money that is made and/or saved by offering flexibility and the costs that were made by additional deviations from the day ahead schedule, together with the net profit that is made.

noise	gain	costs	net profit
0%	€17.61	€4.71	€12.90
1%	€17.89	€5.23	€12.66
5%	€17.75	€6.40	€11.35
10%	€16.80	€6.42	€10.38

in the electrical and financial results, it can be seen that the results become worse when the amount of noise increases. When looking at the electrical results it's notable that both the amount of imbalance that is resolved and caused are lower than all of the baseline cases. However, the net difference of the case in which no noise was added to the weather forecasts are in line with the baseline scenarios in which more than 50% of the available flexibility is offered. However, in the cases where noise was added this difference decreases. The financial results however are better than all baseline scenarios for both the cases where 0% and 1% noise was added. When the added noise increased, the financial results dropped slightly, but were still in line with the baseline scenarios in which more than 50% of the available flexibility is offered.

Figure 5.13 shows the cumulative received reward for all NFCACLA agents during the training period. It can be seen that when there is no or little noise, the agent improves pretty fast, while this take may take a bit longer when the noise increases to five or ten percent. Interestingly, the difference in the amount of iterations it takes to converge doesn't differ much between the 5% and 10% noise cases. For any amount of noise the NFCACLA algorithm seems to find a stable solution pretty well.

Figure 5.14 shows the available and offered flexibility for the different cases, along with the amount of ramp up or down capacity that is required by the TSO. The amount of flexibility seems to range between approximately 50%

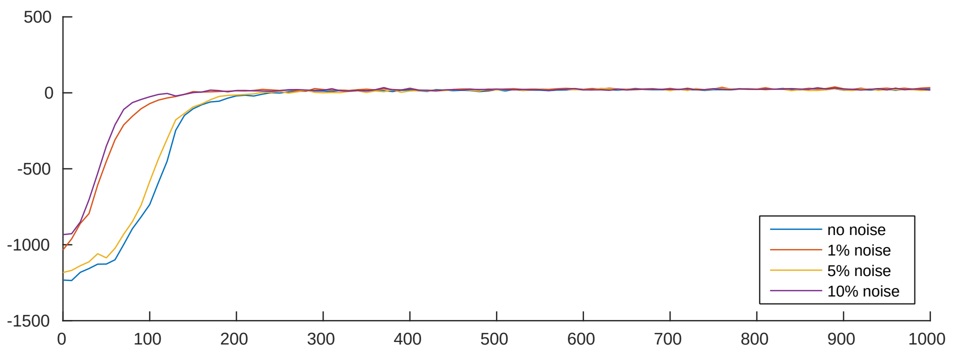


Figure 5.13: The development of the acquired cumulative reward over an entire simulation during the training progress for the NFCACLA algorithm with noise added to the weather forecasts

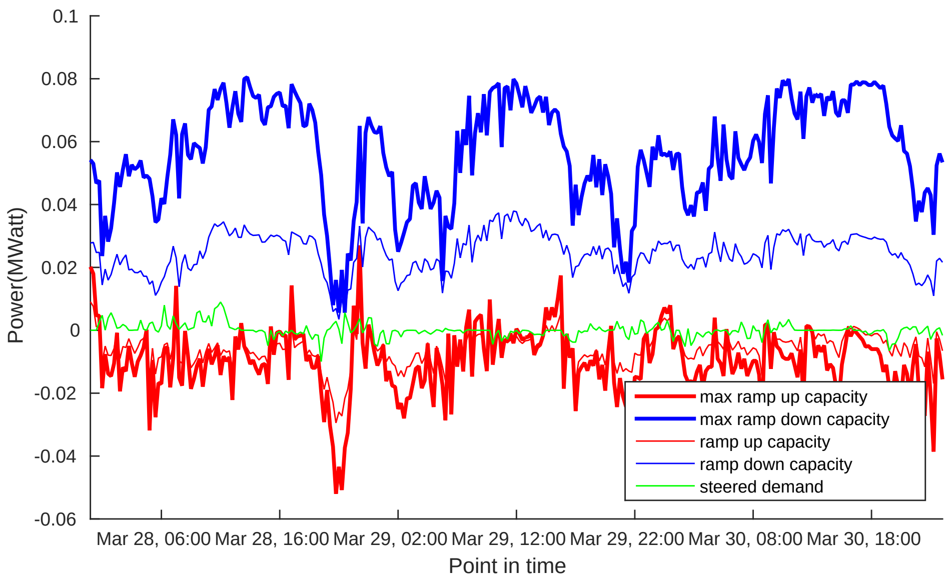
and 25% of the available flexibility. In the case with 1% noise the amount of offered flexibility seems to be a bit lower than in the other scenarios, while this seems to be a bit higher in the scenario with 5% noise.

Figure 5.15 shows the allocation over the entire simulation for each of the cases, plotted together with the day ahead schedule and the imbalances for each moment. When looking at the period slightly after March 29th, 12.00h again, it can be seen that the block of imbalance that is caused at that point in time increases when more noise is added to the weather forecasts (arguably with the exception of the 10% noise case).

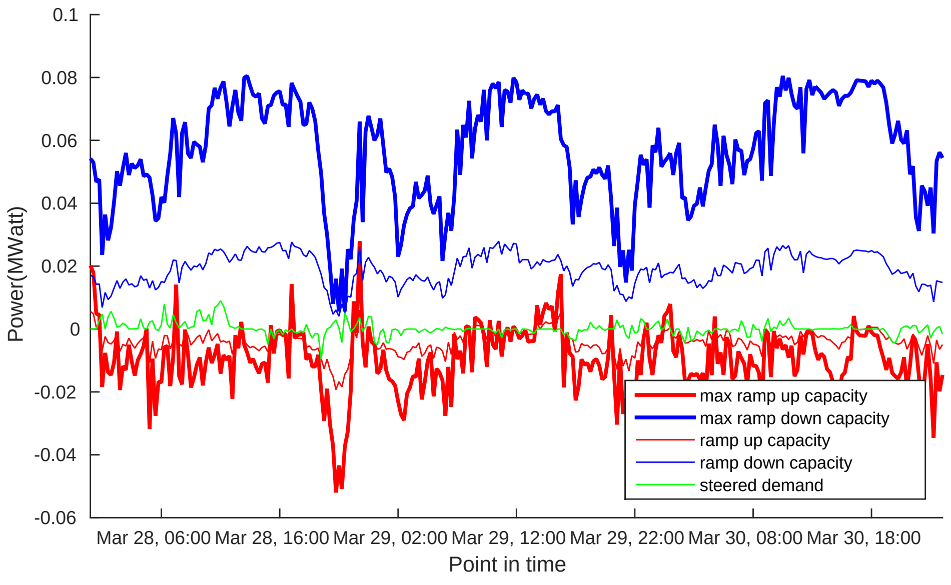
5.3 Different Training Iterations for NFCACLA

This set of experiments is only run for the NFCACLA algorithm using weather forecasts without any noise. Since the NFCACLA agents in the previous experiments were trained for 5 iterations, this means that the configuration of this case is the same as in the previous set of experiments with NFCACLA (without noise) and the same results are used for evaluation.

Tables 5.7 and 5.8 show the electrical and financial results for the simulations with different amounts of iterations for the NFCACLA algorithm. When looking at the electrical results, it can be seen that the scenario in which the networks were trained with five iterations is performing worse than the other scenarios, and along the line of offering between 20% to 30% of the available flexibility in comparison to the baseline results. The other scenarios seem to perform better, with the best scenario appearing to train the networks only for a single iteration. Surprisingly, all cases, with the exception of the case using five iterations yield better financial results than all of the baseline

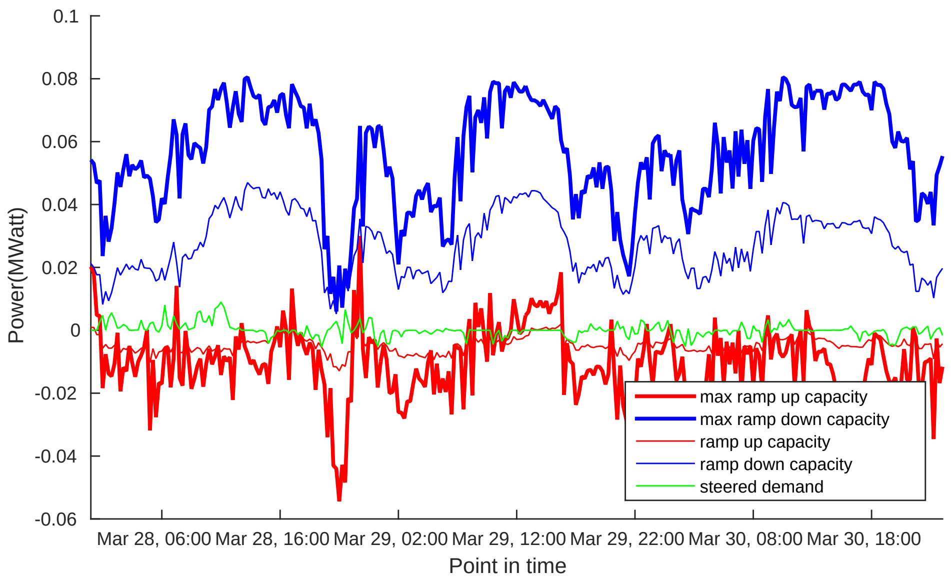


(a) no noise

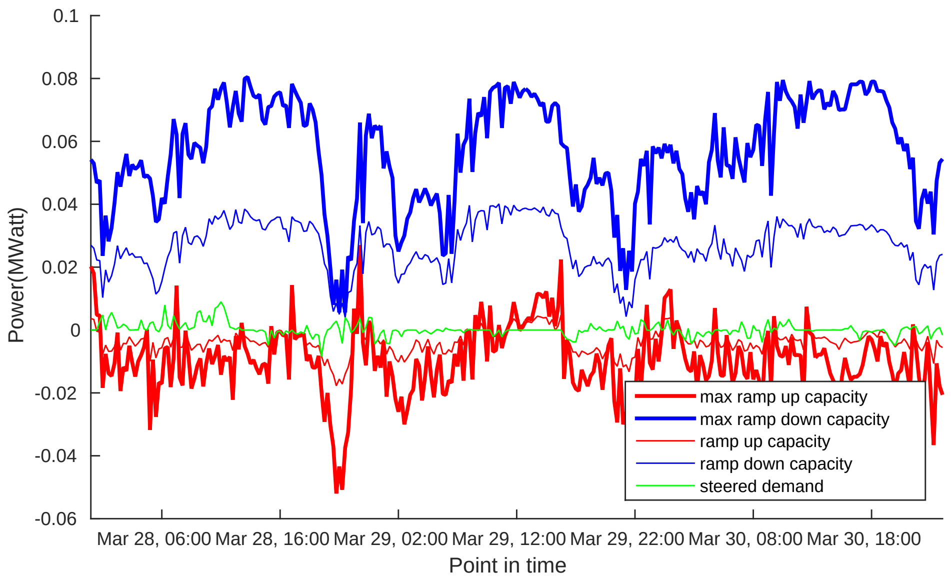


(b) 1% noise

Figure 5.14: Available flexibility, offered flexibility and the adjustment made by the objective agent using the NFCACLA algorithm with different levels of noise in the weather forecasts.

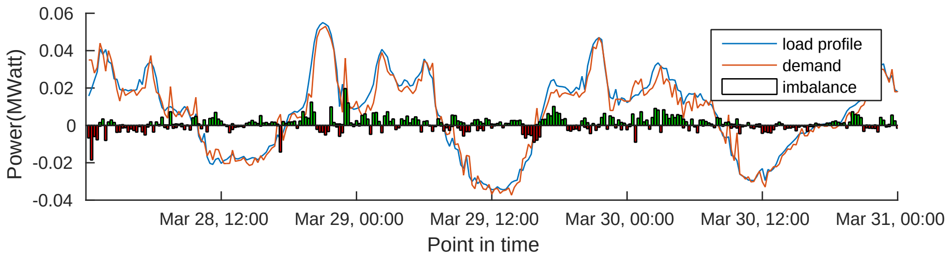


(c) 5% noise

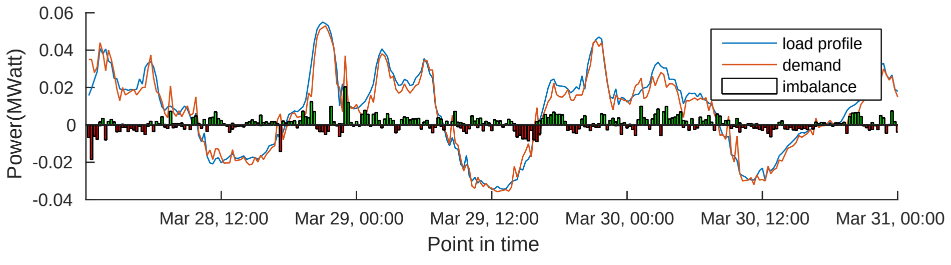


(d) 10% noise

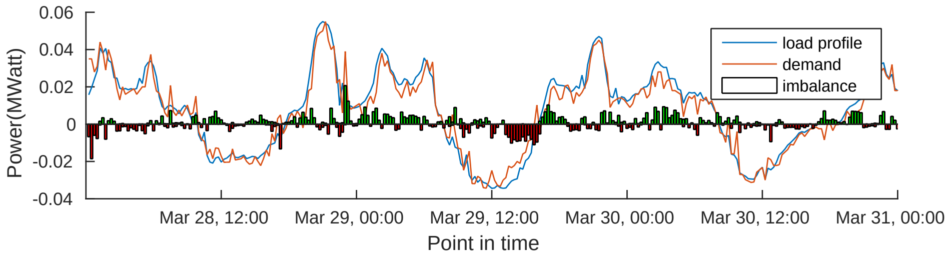
Figure 5.14: Available flexibility, offered flexibility and the adjustment made by the objective agent using the NFCACLA algorithm with different levels of noise in the weather forecasts.



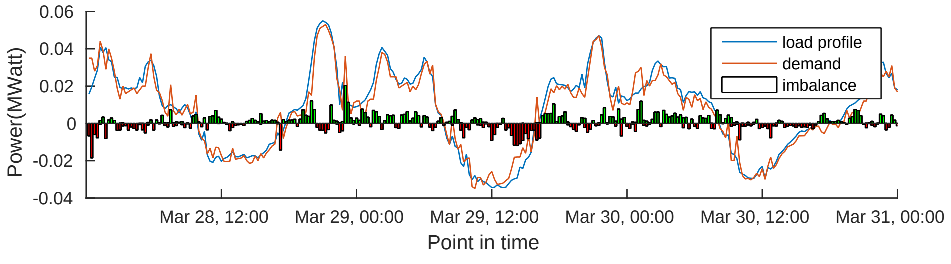
(a) no noise



(b) 1% noise



(c) 5% noise



(d) 10% noise

Figure 5.15: Allocation versus the day ahead schedule for the NFCACLA algorithm with different levels of noise in the weather forecasts.

Table 5.7: Results for NFCACLA in terms of resolved external imbalances and caused internal imbalances and their net difference, expressed in kW.

epochs	**imb. resolved**	**imb. caused**	**net difference**
1	583.95	382.21	201.74
5	583.47	448.75	134.73
10	556.22	374.50	181.72
25	594.48	413.41	181.06

Table 5.8: Results for NFCACLA in terms of money that is made and/or saved by offering flexibility and the costs that were made by additional deviations from the day ahead schedule, together with the net profit that is made.

epochs	**gain**	**costs**	**net profit**
1	€17.99	€4.74	€13.25
5	€17.75	€6.40	€11.35
10	€17.58	€4.96	€12.62
25	€18.25	€5.57	€12.69

scenarios, while the five iterations case still scores better than almost all of the baseline scenarios.

Figure 5.16 shows the progress of the obtained cumulative reward over the training period. When looking at the scenarios using one, five and ten iterations it can be seen that NFCACLA converges faster when the number of iterations increases. The exception is the case where 25 iterations are used: this scenario follows a similar progress as the simulation with five repetitions. It can be seen that the cumulative reward sometimes decreases when the current policy is still poor, however increases rapidly after the first (ca.) 50 iterations.

Figure 5.17 shows the available and offered flexibility for each simulation using a different number of iterations, along with the amount of ramp up or down capacity that is required by the TSO. It can be seen that in the simulation where the agent is trained for only one iteration around 75% of the available ramp down capacity is offered, while nearly all ramp up capacity is offered. In the simulation where the training step took five epochs the agent learned to offer slightly less than 50% of its available flexibility. In the simulation with ten epochs the agent offered slightly less of its ramp down capacity than in the scenarios using 1 and 25 training iterations (scenario using 25 iterations offered around 75% as well), while offering very little of its ramp up capacity. In the scenario where the agents were trained for 25 iterations the amount of ramp up capacity that is offered is around 50% of

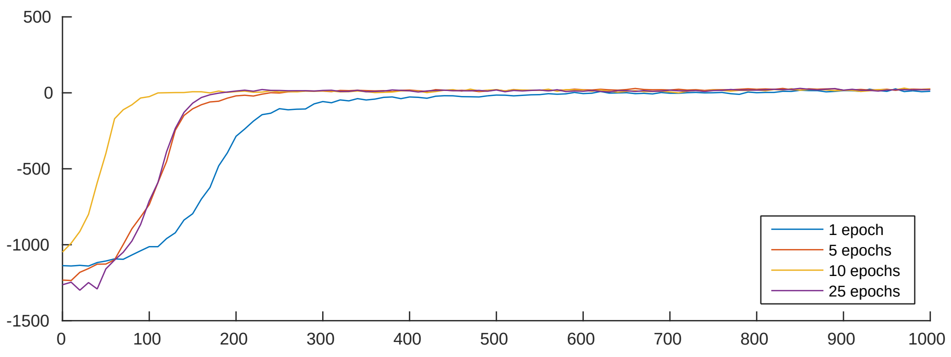


Figure 5.16: The development of the acquired cumulative reward over an entire simulation during the training progress for the NFCACLA algorithm with different numbers of training iterations.

the ramp up capacity.

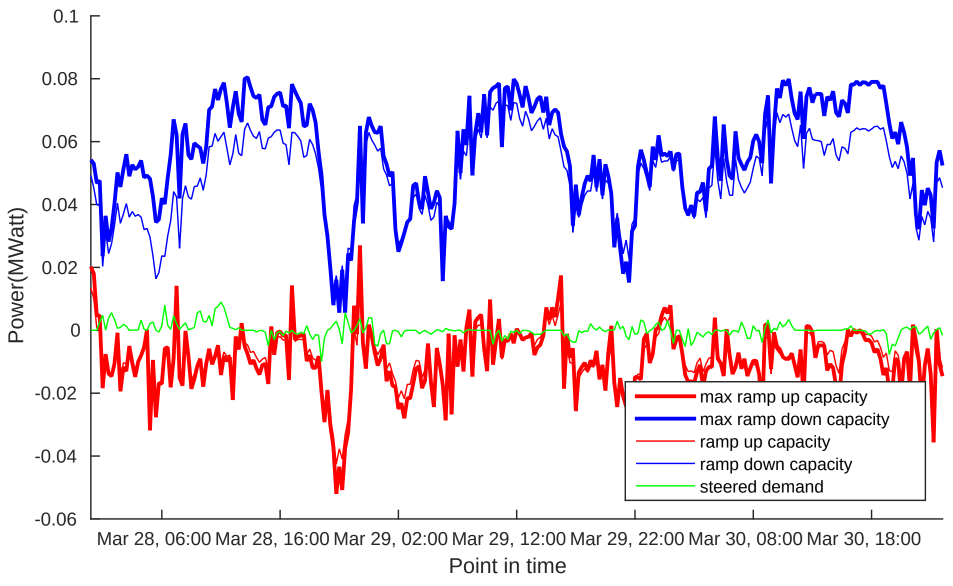
Figure 5.18 shows the allocation during each of the simulations plotted against the day ahead schedule. Since the agent offers more of its flexibility to the TSO, but the amount of steering is still limited by the requested ramp up or down capacity by the TSO it can be seen that the allocation is similar to the allocation in the previous simulations: the allocation only slightly deviates from the load profile.

5.4 General Observations

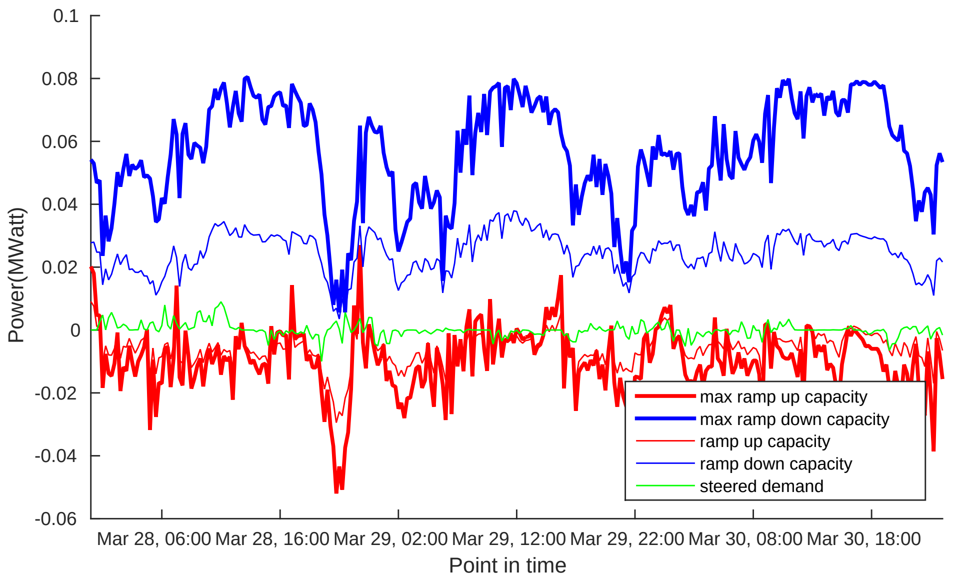
Through all experiments (both using reinforcement learning agents and basic strategies) a number of observations can be made that are common throughout all scenarios.

The first observation is that the demand from the TSO is very limited. The TSO generally demands only a small fraction from the offered flexibility, resulting in only small deviations from the day ahead schedule. Since the deviations are only small, they have little effect on the future demand within the grid. On the one hand this results in only a limited amount of imbalance to resolve, but on the other hand it also prevents the grid from causing too much internal imbalance.

The second observation is that the strategies reinforcement learning agents learned resulted in smaller imbalances (both resolved and caused), even when large portions of the available flexibility were offered. This can especially be seen in the experiments with variations over the number of training iterations for NFCACLA.

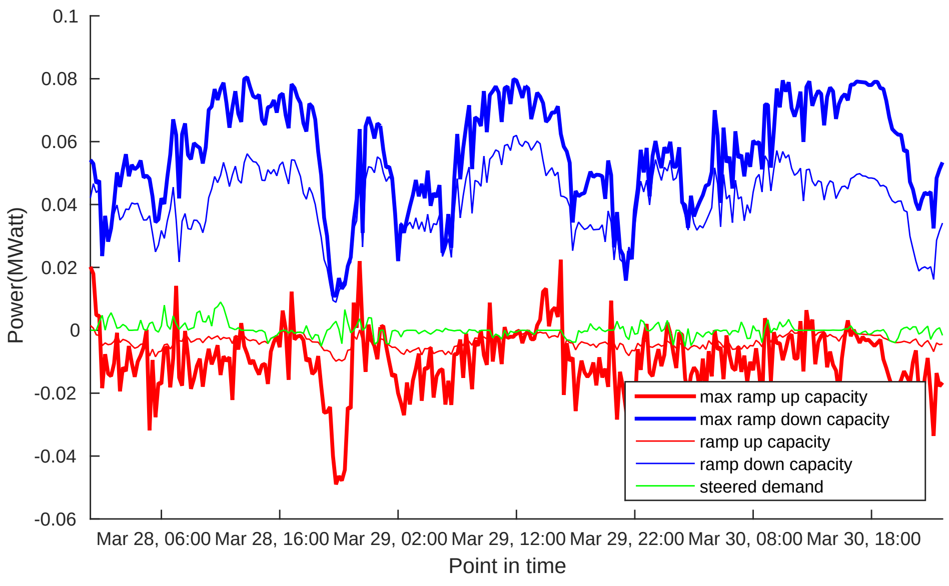


(a) 1 iteration

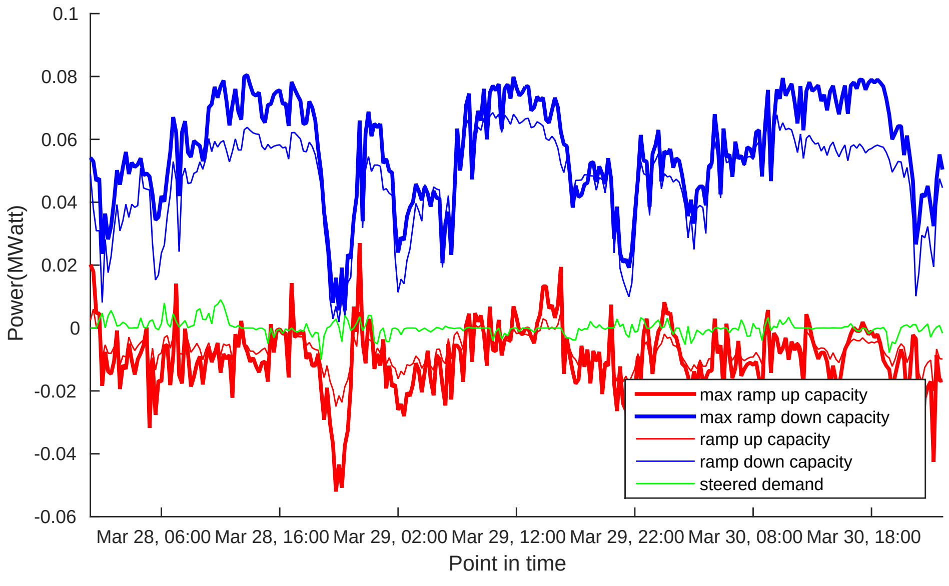


(b) 5 iterations

Figure 5.17: Available flexibility, offered flexibility and the adjustment made by the objective agent using the NFCACLA algorithm in which the number of training repetitions is varied.

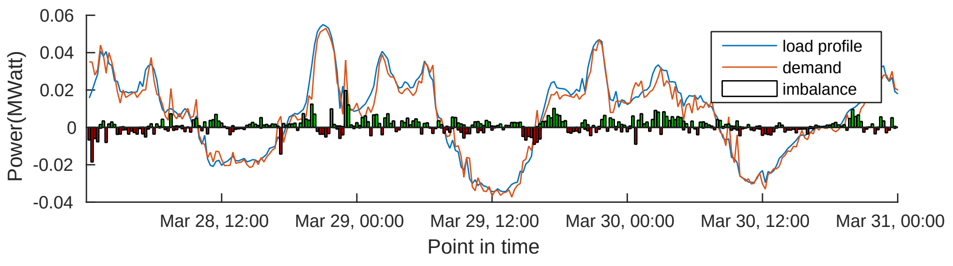


(c) 10 iterations

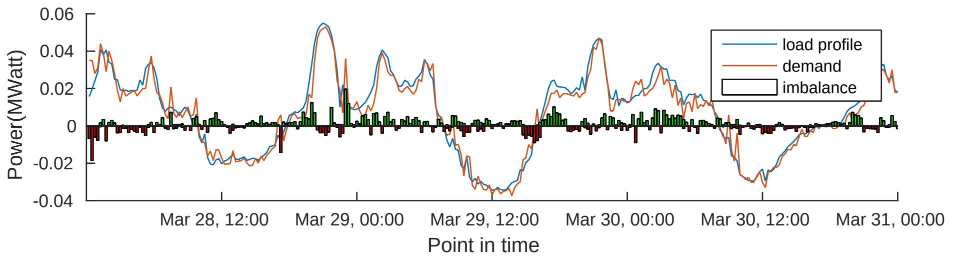


(d) 25 iterations

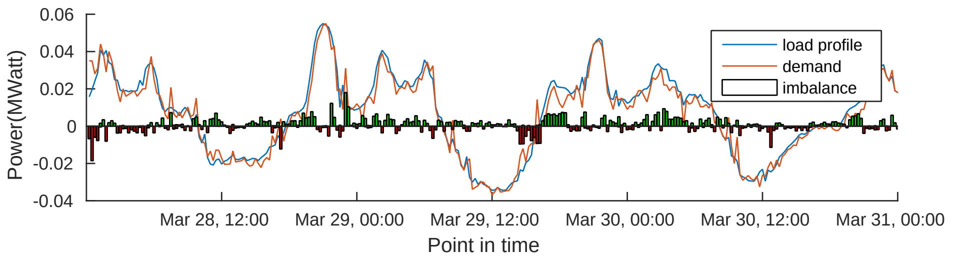
Figure 5.17: Available flexibility, offered flexibility and the adjustment made by the objective agent using the NFCACLA algorithm in which the number of training repetitions is varied.



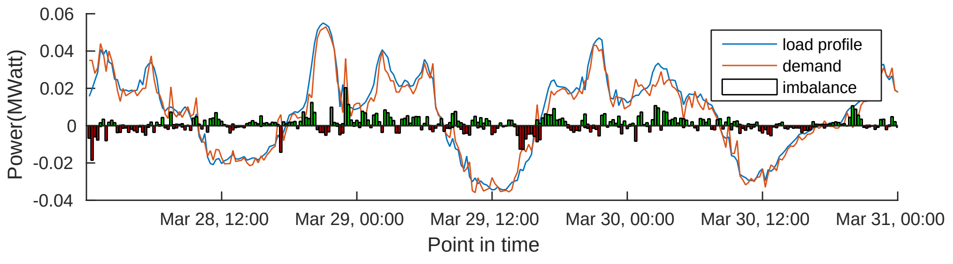
(a) 1 iteration



(b) 5 iterations



(c) 10 iterations



(d) 25 iterations

Figure 5.18: Allocation versus the day ahead schedule for the NFCACLA algorithm in which the number of training repetitions is varied.

The third observation is that, when comparing the baseline results to the results of the other experiments, the financial results are generally better than the electrical results. Especially when looking at the results for the two experiments done with the NFCACLA algorithm: in these experiments the financial results are often better than those of the baseline results (e.g. €12.90 and €12.66 in the NFCACLA scenarios with 0% and 1% noise respectively, versus €12.32 in the baseline scenario where 50% of the available flexibility was offered, which was the largest amount of profit that was made in the baseline scenarios), while the electrical results are not (with a nett difference of 193.72 and 153.07 kW versus 244.42 kW for the same respective scenarios). This is despite the fact that the agents were rewarded on the basis of the imbalances that were caused and resolved rather than their financial consequences.

A final observation that can be made is that the different strategies in the scenarios are fairly similar, while the strategies vary more when the amount of training iterations is changed.

Chapter 6

Conclusion

This final chapter will conclude this thesis by answering the research questions that were proposed in the introduction, followed by a discussion on these answers. Finally a number of directions for future research will be proposed.

6.1 Answers to Research Questions

How well are the CACLA and NFCACLA algorithms able to utilize flexibility in smart grids to maintain and or improve stability in the electricity grid? According to the results of this thesis both CACLA and NFCACLA are well able to resolve external imbalances without causing too many additional internal imbalances. This means that both algorithms are well able to contribute to increasing stability maintenance in the electricity grid. However, when looking at simple strategies that were used as a baseline, it can be seen that some simpler solutions appear to provide better results. However, since the amount of requested ramp up or ramp down power is small no strong conclusions can be drawn on whether the boundaries that were determined by the reinforcement learning agents were indeed sufficient to prevent additional imbalances when more of the offered flexibility was requested by the TSO.

How suitable are weather predictions as indicators for offering flexibility? When adding an increasing amount of noise to the weather forecasts one might expect two resulting changes: (1) the amount of caused imbalances increases due to unexpected drops or increases in the amount of electricity production (or demand) and (2) due to uncertainty the agents would become more cautious when offering their flexibility due to unexpected weather influences. When looking at the results for the NFCACLA experiments, it can be seen that the performances indeed decrease. However, neither of the expected effects occur in the CACLA experiments. One of the reasons for this was that the same weather conditions were used as forecasts as those that were used during the simulations in which the load profile was determined. This means that the expected demand already included the expected demand and production that were related to any influences by the weather and the forecasts would then become less relevant (if not irrelevant) since they are already implicitly known to the aggregator.

In order to definitely conclude the irrelevance (or relevance) of weather forecasts when determining the boundary conditions, however, it would be recommended to perform more simulations in which the load profile is created independently of any weather conditions.

How does NFCACLA perform in comparison to CACLA? The performances of the NFCACLA algorithm are higher than those of CACLA. One of the reasons is that NFCACLA appears to find a more stable solution, while CACLA keeps swinging between different strategies. However, it might be that the CACLA algorithm would be able to find a more stable solution, given more time to train. In contrast to this, however, the CACLA algorithm takes less repetitions of the simulation to learn a semi-stable policy.

How does the number of epochs influence NFCACLA's learning process? As expected, the NFCACLA agents increase the cumulative rewards they gain faster when the number of epochs that they train upon their observations increases. There is no clear relation between the number of training iterations and the performances: the two smaller amounts of epochs achieve both the best (1 repetition) and worst (5 repetitions) performance. However, it appears that using a large number of repetitions (25) it might become harder to improve from suboptimal strategies, however more experiments should be done on this with a larger amount of repetitions.

6.2 Discussion

Reinforcement learning appears to be a suitable technique to determine the boundary conditions for offering flexibility to the TSO. However, it's arguable whether more simple solutions aren't more successful. However, one drawback from the data used in the current scenarios is that the amount of external imbalance was very small. As a result only small parts of the offered flexibility was used and the negative impact of resolving imbalances on the cluster are very limited.

Furthermore a number of observations in the Baseline experiments are very a-typical, which makes the applicability of the current scenario to real-world scenarios more questionable. For example, deviating from the load profile should generally yield high additional costs. However, when looking at the baseline scenario in which no flexibility is offered but PowerMatcher controllers are used to steer the cluster so that it follows the day ahead schedule as closely as possible, it can be seen that more imbalances occur in such a way that they resolve external imbalances rather than contributing to them. Furthermore, it would be expected that offering an increasing amount flexibility will always cause an increase of the amount of imbalance that is caused, if only because deviating from the day ahead schedule in one direction will probably result in a deviation in the other direction at a later moment in time. However, in the current scenarios this effect is not observed. In order to draw stronger conclusions the reinforcement learning algorithms would have to be tested on more realistic scenarios.

6.3 Future Work

To conclude this thesis a number of directions for future research will be proposed. These propositions consider three different topics that have been focused on in this thesis: (1) the applicability of reinforcement learning for improving grid stability in smart grids, (2) the use of weather forecasts to determine the boundary conditions of offered flexibility to the TSO and (3) the development of NFCACLA as a model free reinforcement learning algorithm.

In order to further test the applicability of reinforcement learning to the proposed problem a wide variety of experiments can be done. For example, scenarios should simulated that challenge aggregators by requesting a larger amount of flexibility, forcing the aggregator to ramp up or down to its provided boundaries and thus proactively seeking out the limits of the boundary conditions. Additionally scenarios should be tested that are a better reflection of real world scenarios and where ramping up and down would have a bigger impact on the aggregator's cluster. Finally it would be interesting to run simulations in scenarios that cover a larger period of time. The current experiments were conducted in March, which covers only a small part of the year with specific weather conditions. It would be interesting to see simulations run over a full year to see the adaptability of reinforcement learning through different seasons and their weather conditions throughout the year.

Apart from more experiments that follow the line of this thesis, the effect of a number of adaptations might be interesting to look at as well. Since the financial results appeared to be better than the electrical results, it seems interesting to train the agents based on the feedback that the agents receive on these results, rather than the electrical results. Another interesting experiment, which is rather a different topic than an adaptation, would be to primarily focus on minimizing imbalances while making resolving external imbalances only a secondary goal.

In order to dismiss the possibility of any congruence between the weather forecasts and the demand profile, simulations should be performed in which the load profile is developed independently of specific weather conditions. However, alternatively it would be interesting to compare the results found in this thesis to similar experiments in which no weather forecasts were used and the boundary conditions are found solely by looking at the grid state and the day ahead schedule. In this way it could be seen more clearly to what extent the weather forecasts made a contribution to determining the boundary conditions. In contrast however, it would be interesting to see simulations in which the heating demand of the households is dependent on the weather conditions, which was not the case in the scenarios used in this thesis.

Finally, in this thesis it was shown that the NFCACLA could be applied successfully as a reinforcement learning algorithm. However, the extent of its success could not be measured since no parameter (or other) optimizations have been performed. It would be interesting to see the potential of NFCACLA when applied to other problems and comparing its results to some state of the art reinforcement learning algorithms.

Appendices

Appendix A

Electrical Results

A.1 Overview

Table A.1: Results for all experiments in terms of resolved external imbalances and caused internal imbalances and their net difference.

series	**variation**	**imb. resolved**	**imb. caused**	**difference**
business as usual	-	990.43	1152.68	-162.25
baseline	0%	571.52	620.35	-58.85
	10%	662.30	579.82	82.41
	20%	681.71	555.32	115.66
	30%	682.71	585.46	197.25
	40%	741.33	501.98	239.34
	50%	751.56	507.15	244.42
	60%	753.58	509.67	243.92
	70%	714.14	555.50	158.63
	80%	723.55	522.54	201.01
	90%	710.97	546.63	164.34
	100%	708.50	526.46	182.04
CACLA	0%	705.05	545.64	159.42
	1%	657.00	558.06	98.94
	5%	734.21	552.96	181.25
	10%	664.12	620.11	44.01
NFCACLA (noise)	0%	577.41	383.68	193.72
	1%	572.64	419.57	153.07
	5%	583.47	448.75	134.73
	10%	580.57	446.52	134.02
NFCACLA (epochs)	1	583.95	382.21	201.74
	5	583.47	448.75	134.73
	10	556.22	374.50	181.72
	25	594.48	413.41	181.06

A.2 Baseline Scenarios

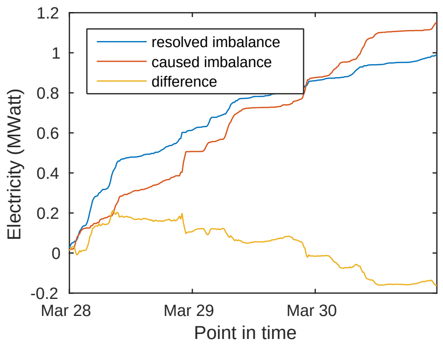


Figure A.1: business as usual

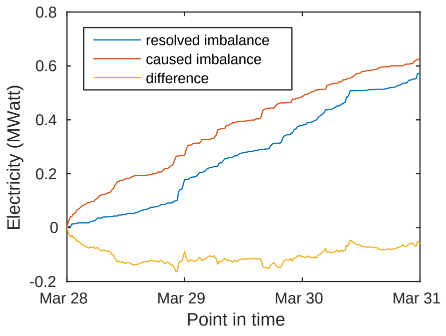


Figure A.2: offer no flexibility

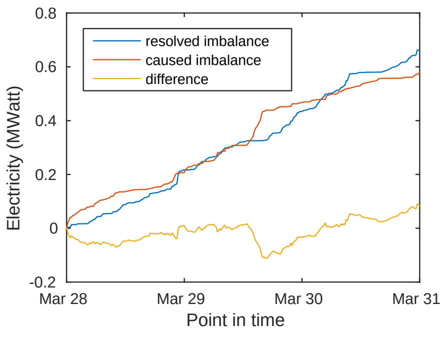


Figure A.3: offer 10% of flexibility

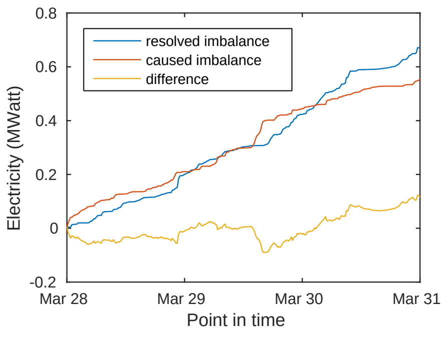


Figure A.4: offer 20% of flexibility

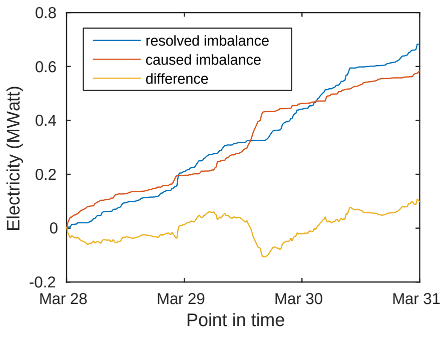


Figure A.5: offer 30% of flexibility

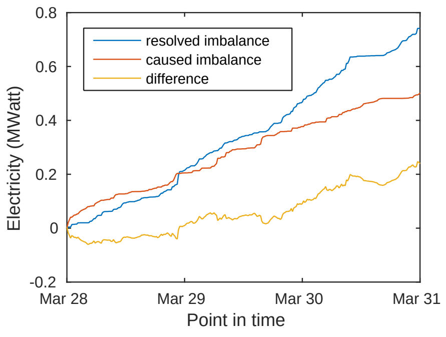


Figure A.6: offer 40% of flexibility

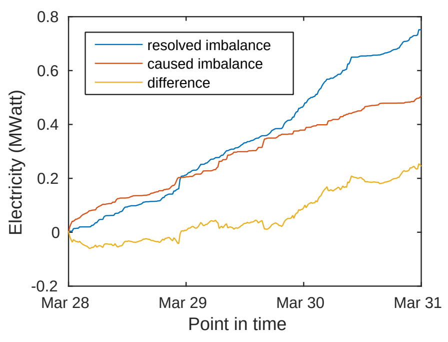


Figure A.7: offer 50% of flexibility

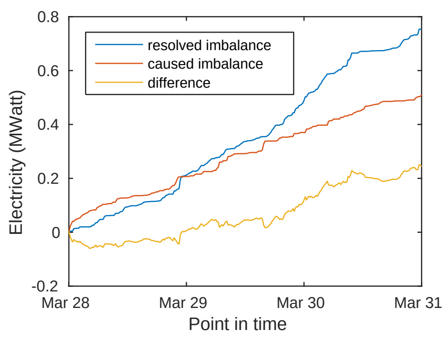


Figure A.8: offer 60% of flexibility

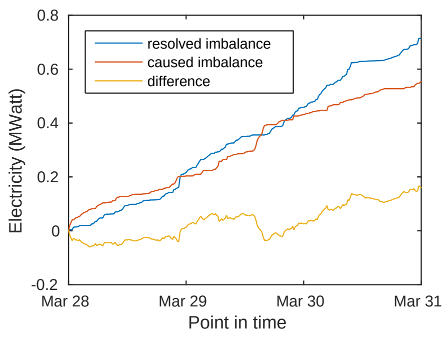


Figure A.9: offer 70% of flexibility

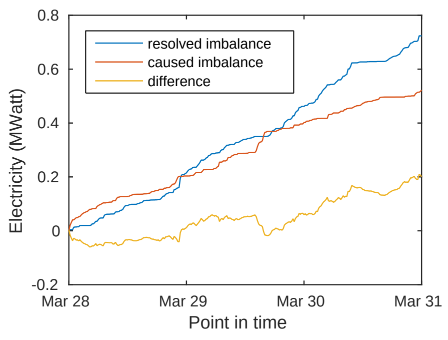


Figure A.10: offer 80% of flexibility

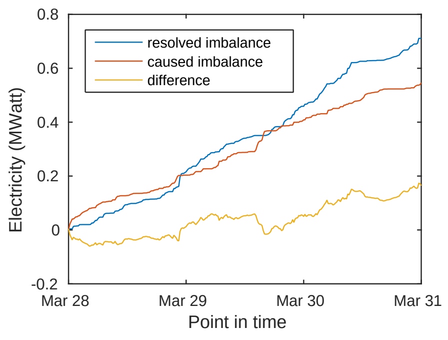


Figure A.11: offer 90% of flexibility

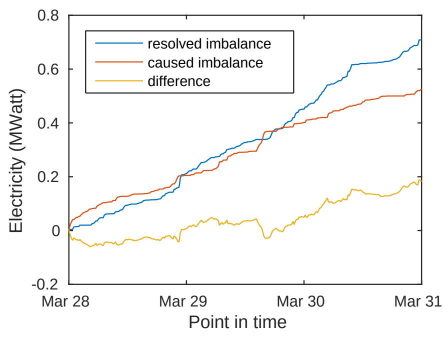


Figure A.12: offer 100% of flexibility

A.3 CACLA Experiments With Noise

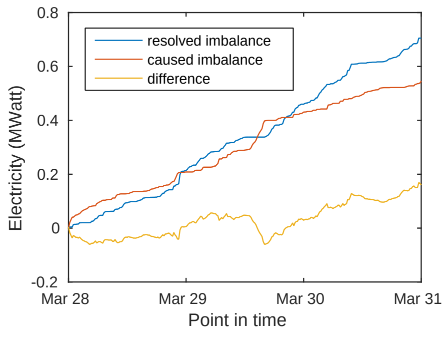


Figure A.13: no noise

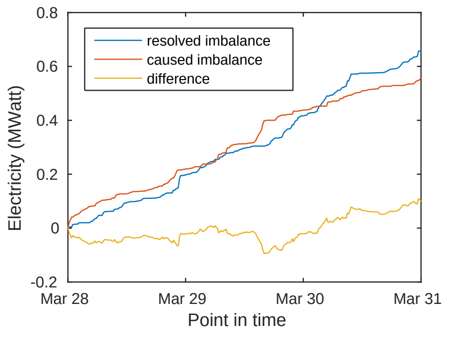


Figure A.14: 1% noise

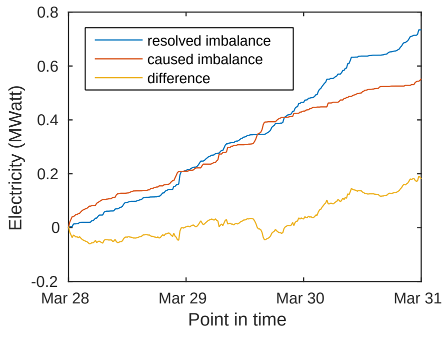


Figure A.15: 5% noise

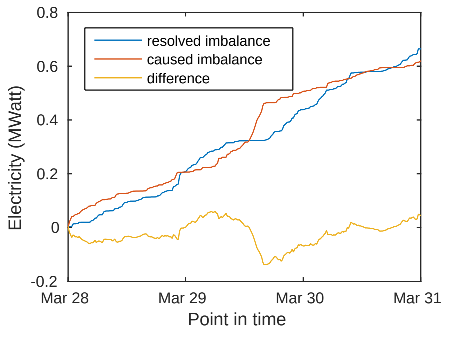


Figure A.16: 10% noise

A.4 NFCACLA Experiments With Noise

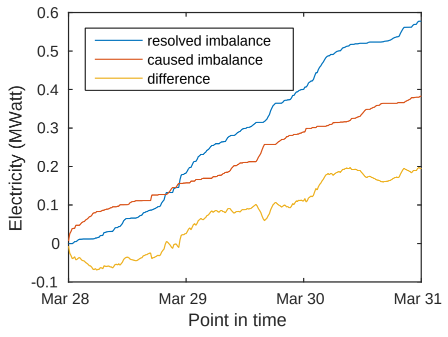


Figure A.17: no noise

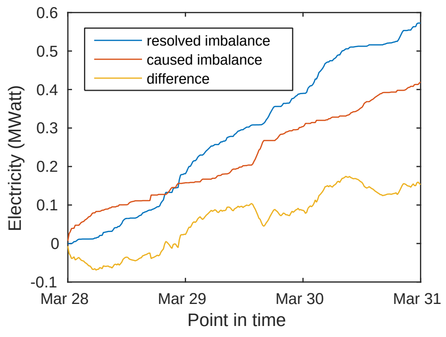


Figure A.18: 1% noise

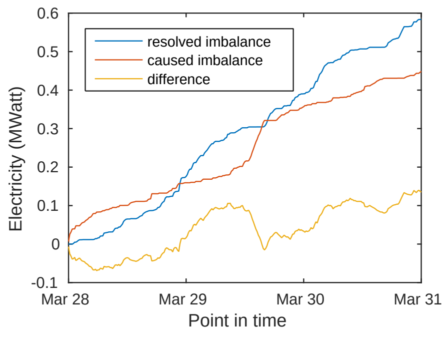


Figure A.19: 5% noise

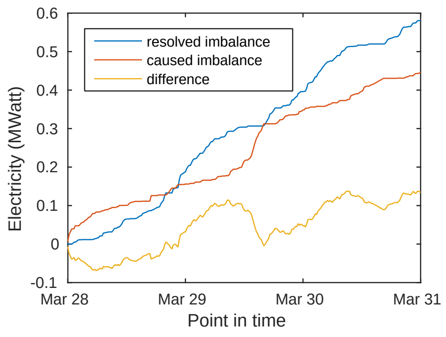


Figure A.20: 10% noise

A.5 NFCACLA Experiments With Training Iterations

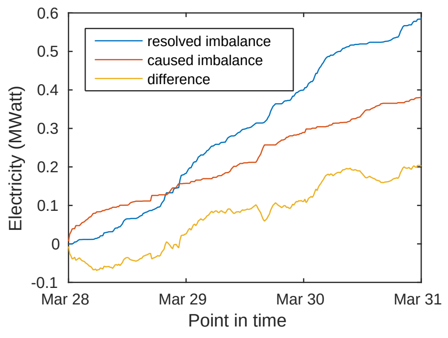


Figure A.21: 1 epoch

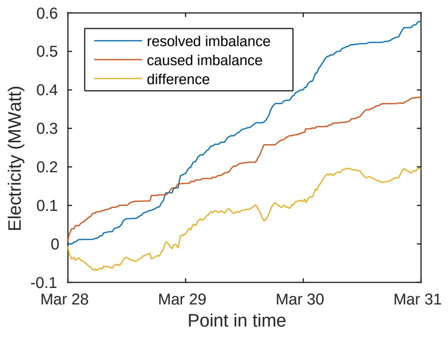


Figure A.22: 5 epochs

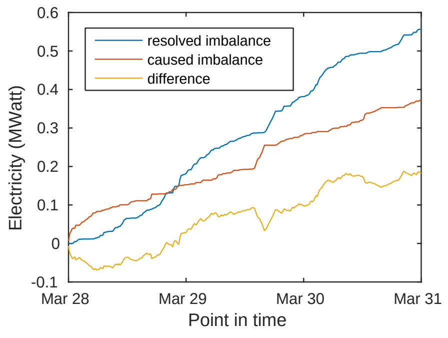


Figure A.23: 10 epochs

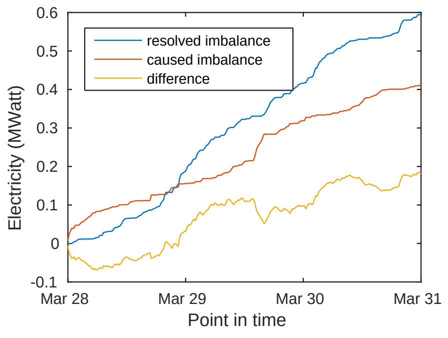


Figure A.24: 25 epochs

Appendix B

Financial Results

B.1 Overview

Table B.1: Results for all experiments in terms of resolved external imbalances and caused internal imbalances and their net difference.

series	**variation**	**gains**	**costs**	**profit**
business as usual	-	€7.19	€32.14	€-24.95
baseline	0%	€12.76	€9.07	€3.69
	10%	€14.62	€9.65	€4.97
	20%	€16.39	€17.33	€9.06
	30%	€16.33	€8.54	€7.79
	40%	€17.93	€6.11	€11.82
	50%	€18.29	€5.97	€12.32
	60%	€18.44	€6.14	€12.31
	70%	€17.91	€7.67	€10.24
	80%	€18.23	€6.94	€11.29
	90%	€18.03	€6.93	€11.10
	100%	€17.61	€6.78	€10.83
CACLA	0%	€18.04	€7.35	€10.69
	1%	€15.18	€6.85	€8.33
	5%	€17.85	€7.41	€10.43
	10%	€17.13	€8.73	€8.40
NFCACLA (noise)	0%	€17.61	€4.71	€12.90
	1%	€17.89	€5.23	€12.66
	5%	€17.75	€6.40	€11.35
	10%	€16.80	€6.42	€10.38
NFCACLA (epochs)	1	€17.99	€4.74	€13.25
	5	€17.75	€6.40	€11.35
	10	€17.58	€4.96	€12.62
	25	€18.25	€5.57	€12.69

B.2 Baseline Scenarios

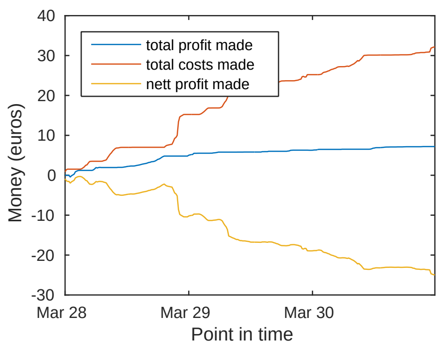


Figure B.1: business as usual

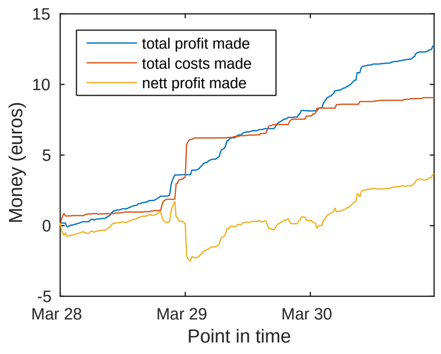


Figure B.2: offer no flexibility

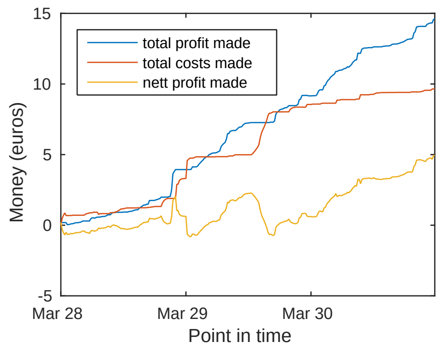


Figure B.3: offer 10% of flexibility

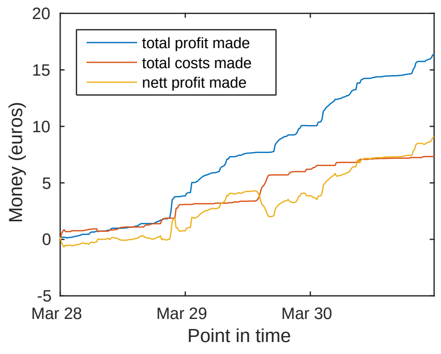


Figure B.4: offer 20% of flexibility

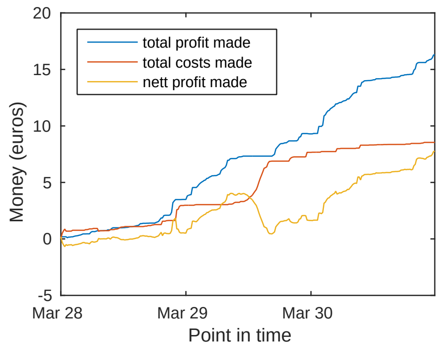


Figure B.5: offer 30% of flexibility

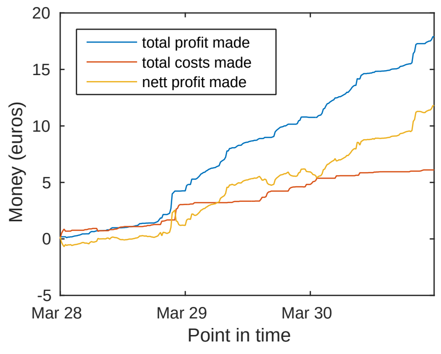


Figure B.6: offer 40% of flexibility

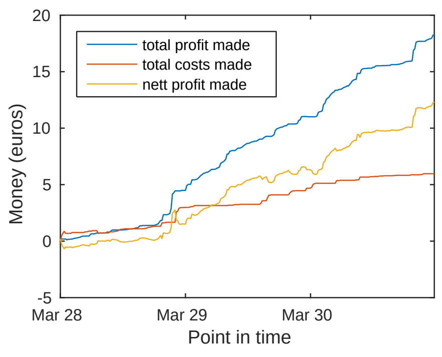


Figure B.7: offer 50% of flexibility

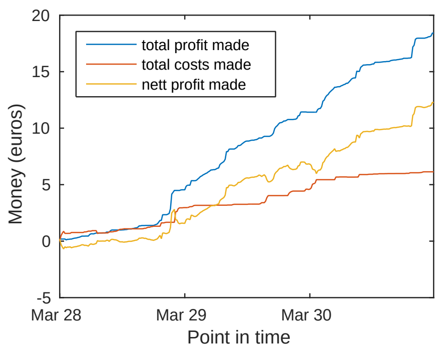


Figure B.8: offer 60% of flexibility

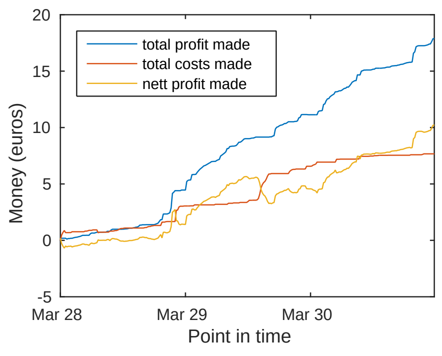


Figure B.9: offer 70% of flexibility

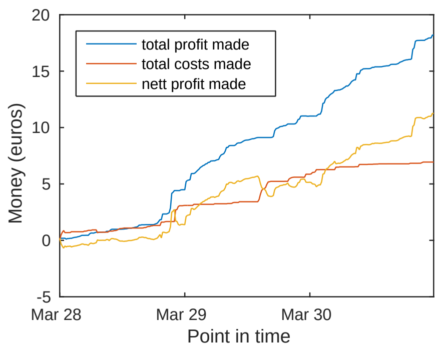


Figure B.10: offer 80% of flexibility

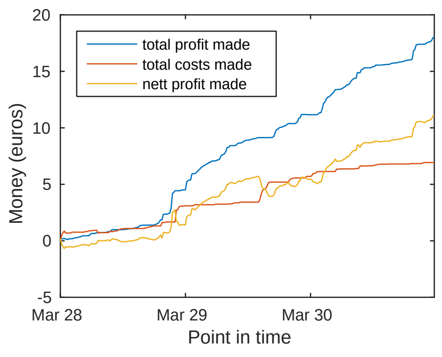


Figure B.11: offer 90% of flexibility

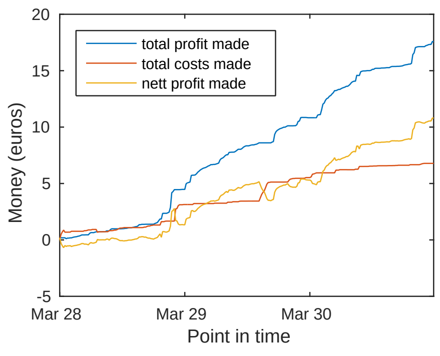


Figure B.12: offer 100% of flexibility

B.3 CACLA Experiments With Noise

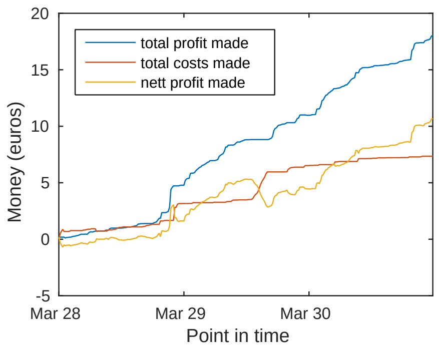


Figure B.13: no noise

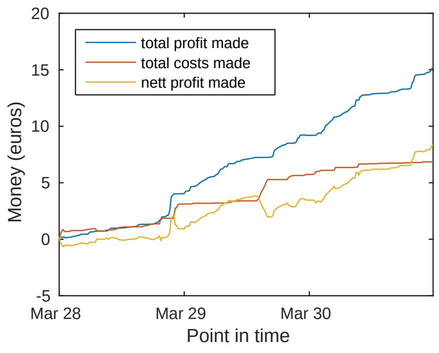


Figure B.14: 1% noise

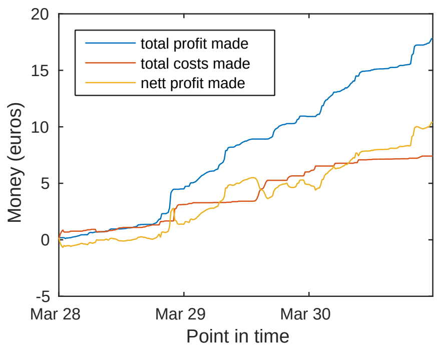


Figure B.15: 5% noise

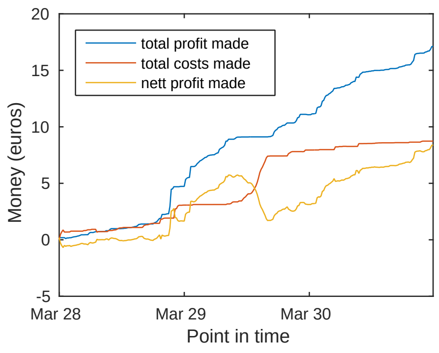


Figure B.16: 10% noise

B.4 NFCACLA Experiments With Noise

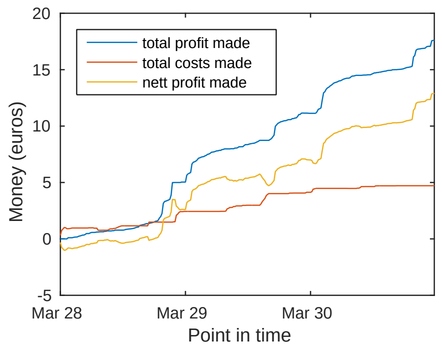


Figure B.17: no noise

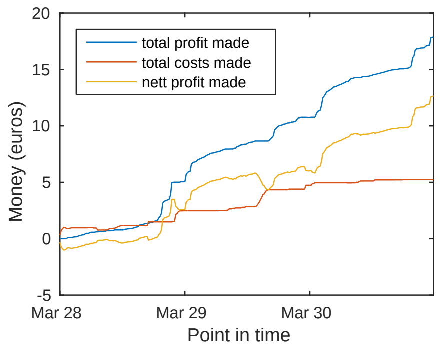


Figure B.18: 1% noise

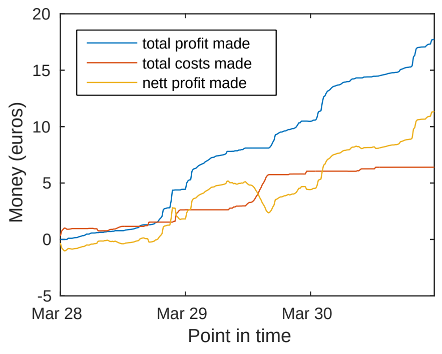


Figure B.19: 5% noise

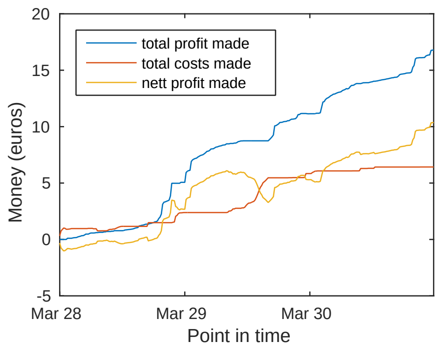


Figure B.20: 10% noise

B.5 NFCACLA Experiments With Training Iterations

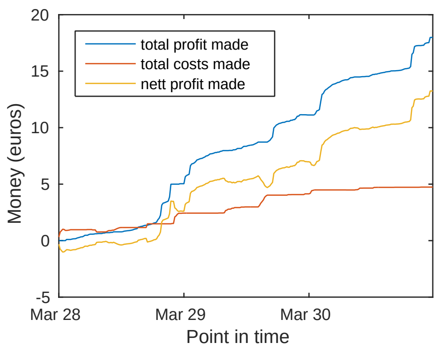


Figure B.21: 1 epoch

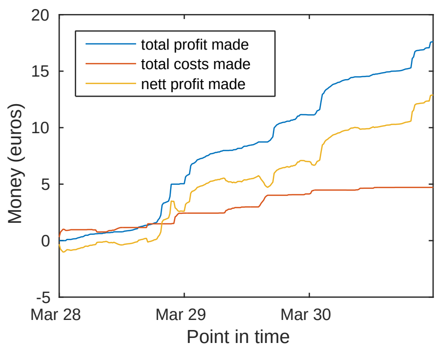


Figure B.22: 5 epochs

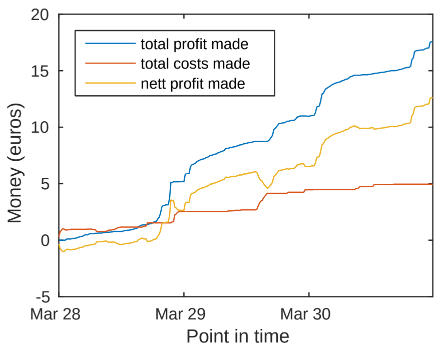


Figure B.23: 10 epochs

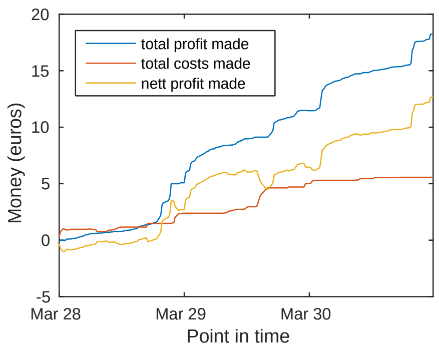


Figure B.24: 25 epochs

Appendix C

Available and Offered Flexibility for Baseline Results

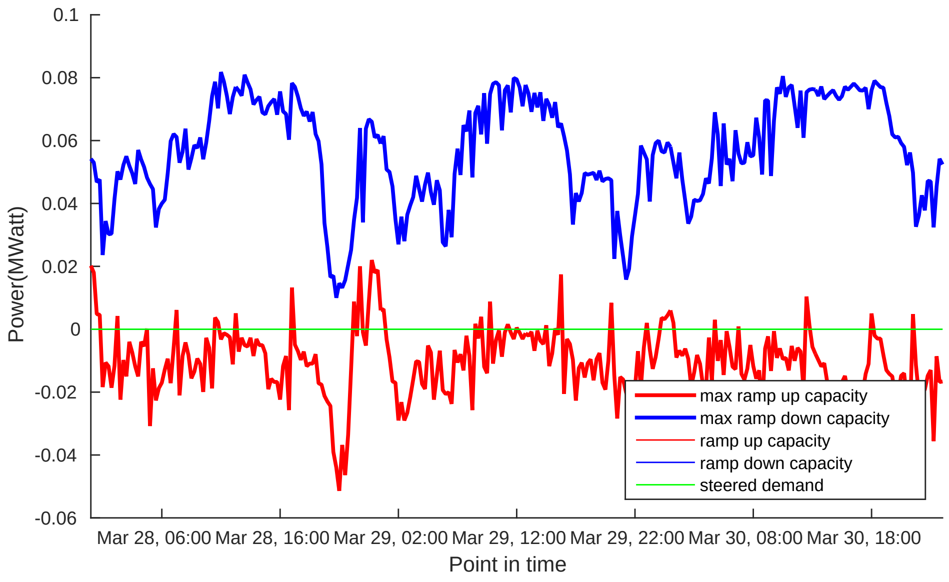


Figure C.1: offer no flexibility

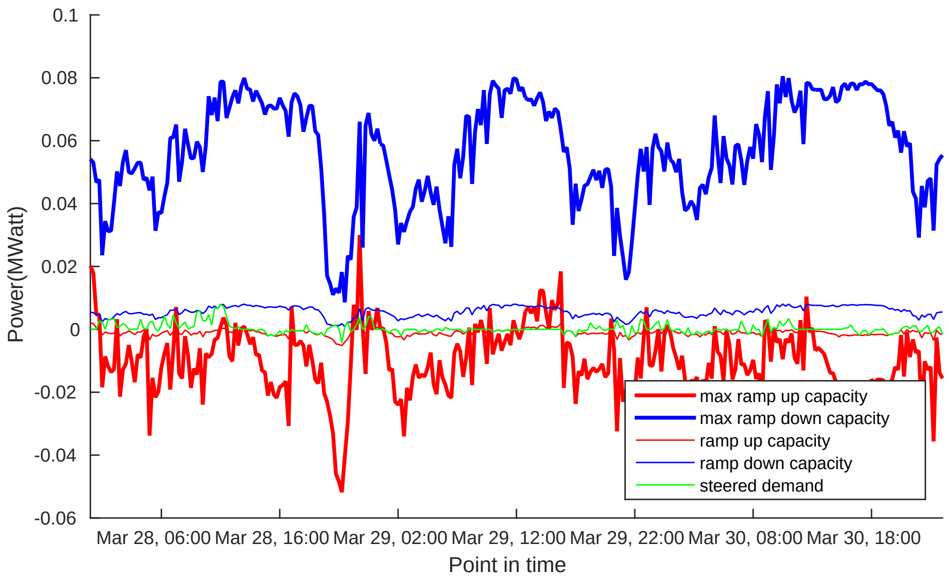


Figure C.2: offer 10% of flexibility

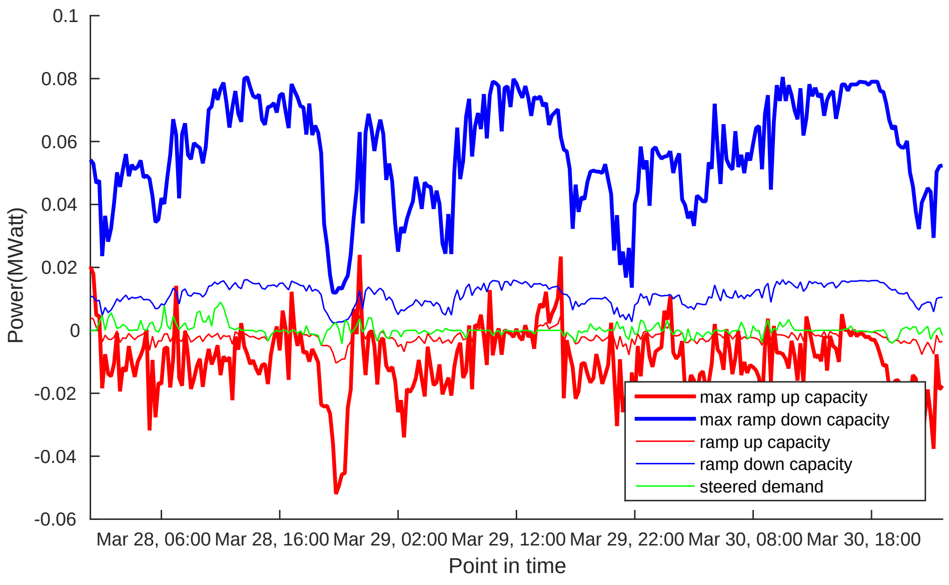


Figure C.3: offer 20% of flexibility

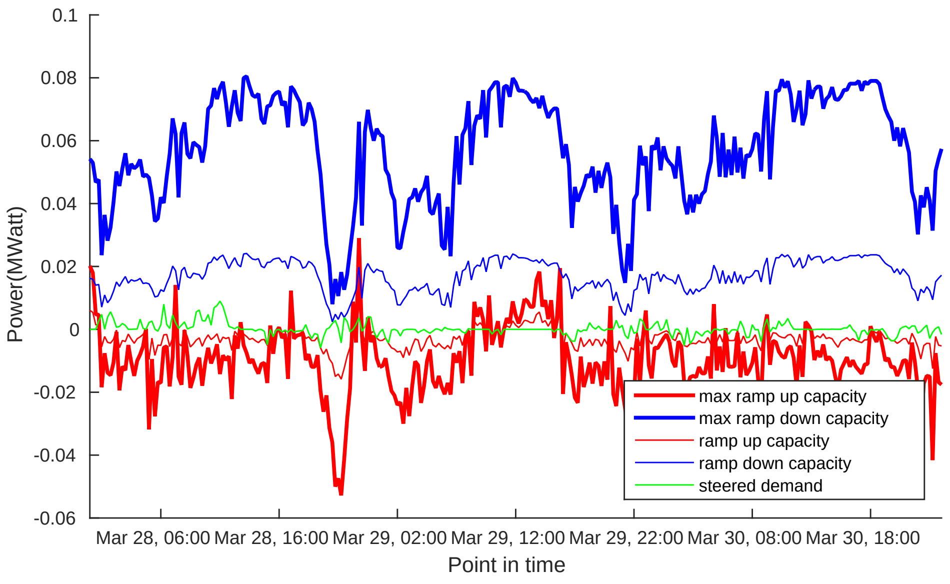


Figure C.4: offer 30% of flexibility

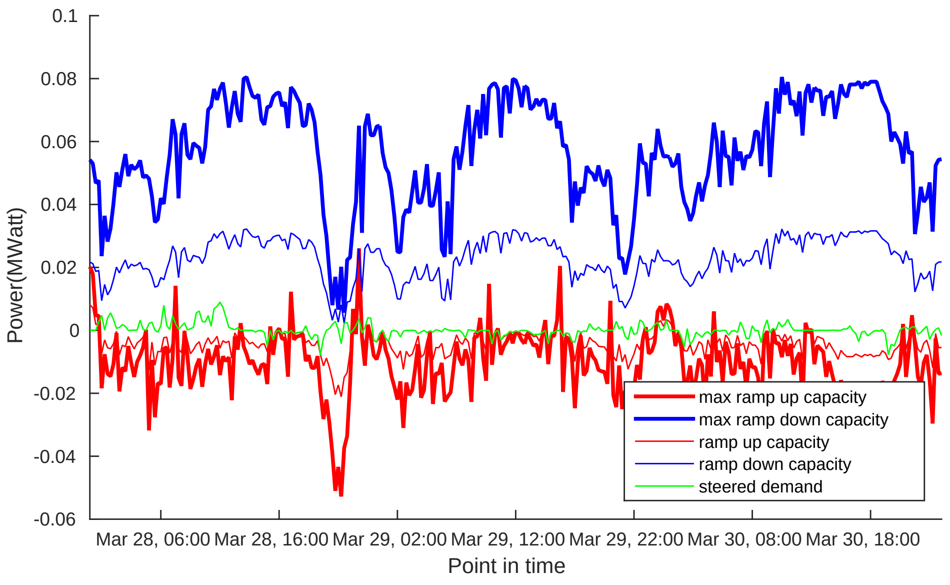


Figure C.5: offer 40% of flexibility

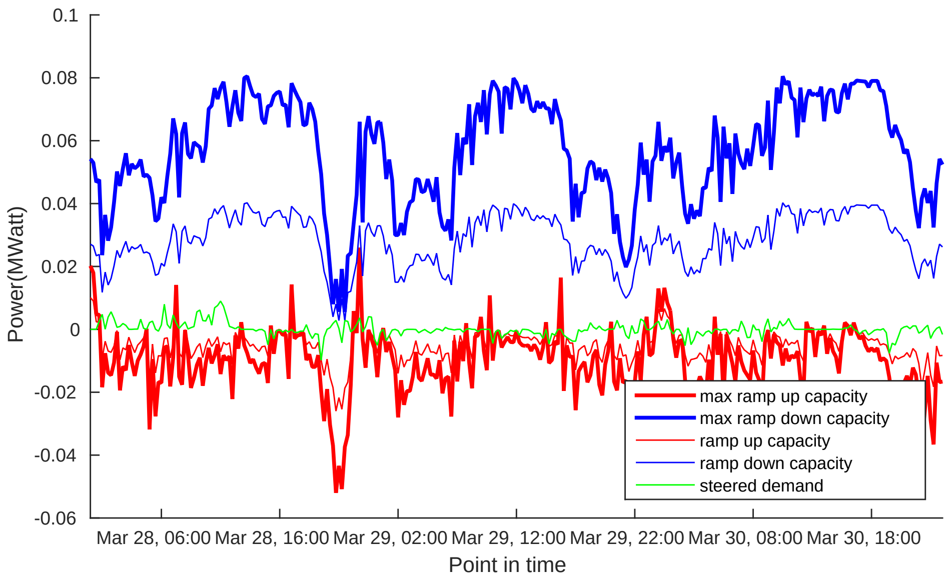


Figure C.6: offer 50% of flexibility

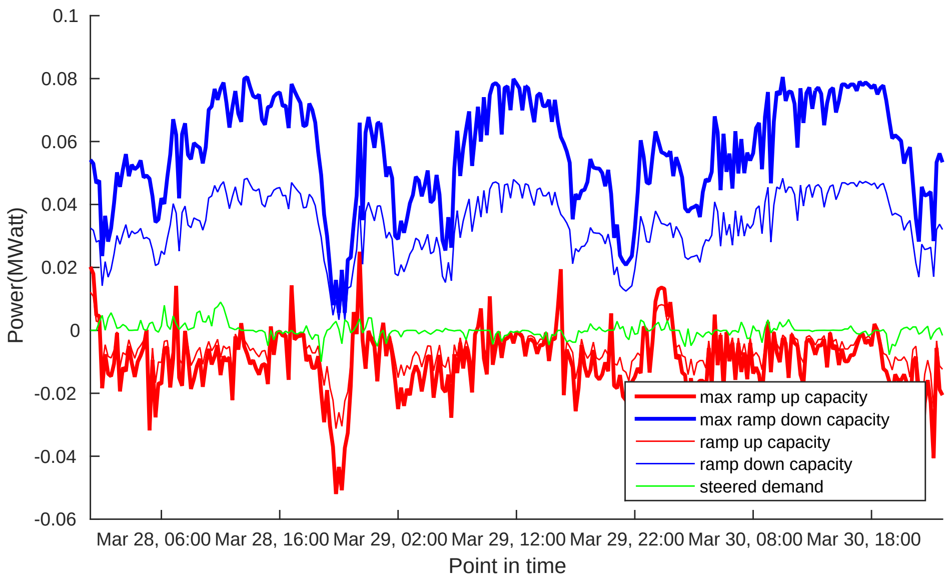


Figure C.7: offer 60% of flexibility

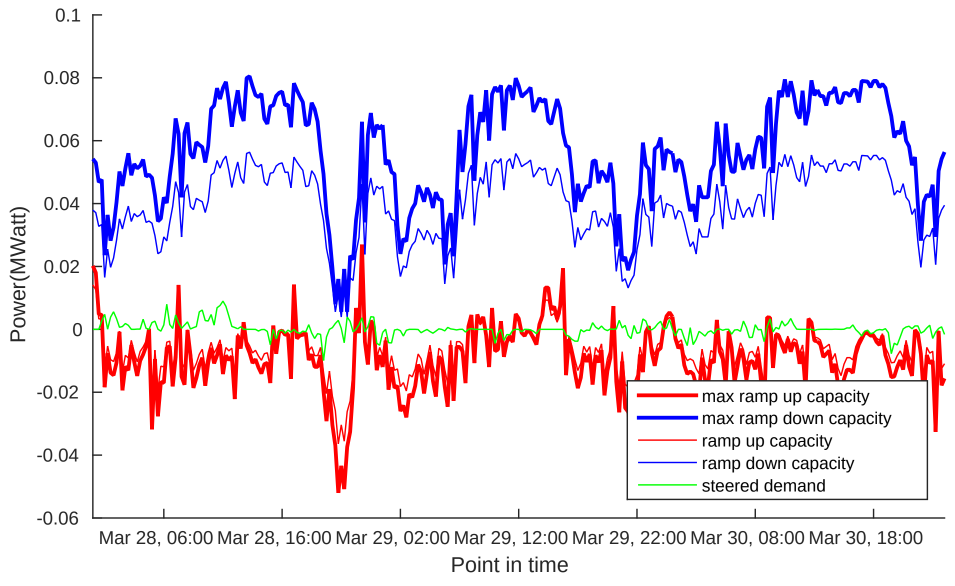


Figure C.8: offer 70% of flexibility

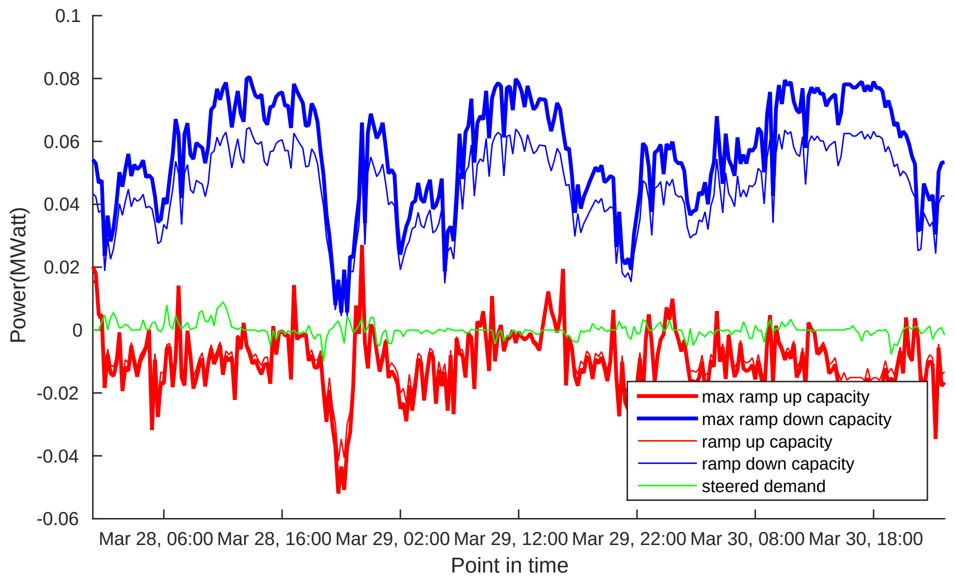


Figure C.9: offer 80% of flexibility

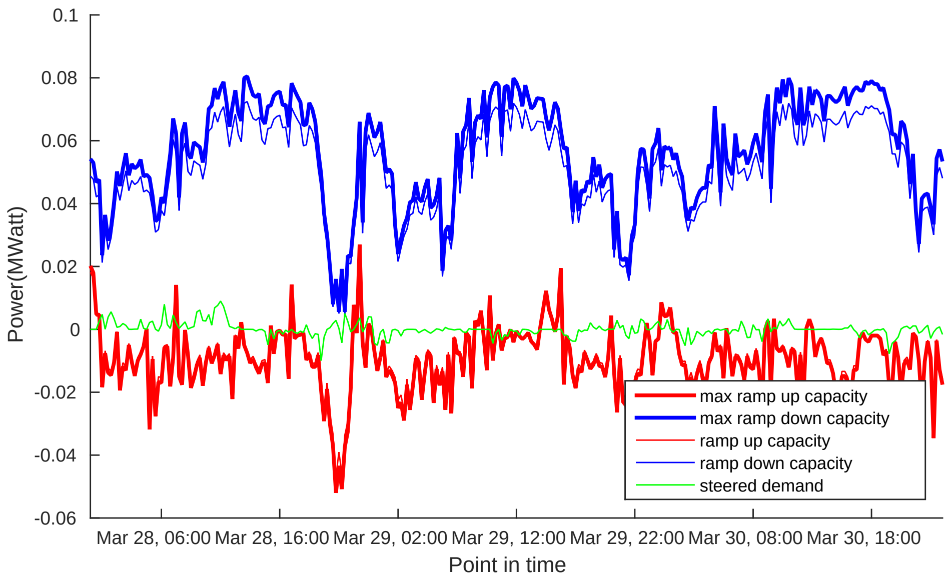


Figure C.10: offer 90% of flexibility

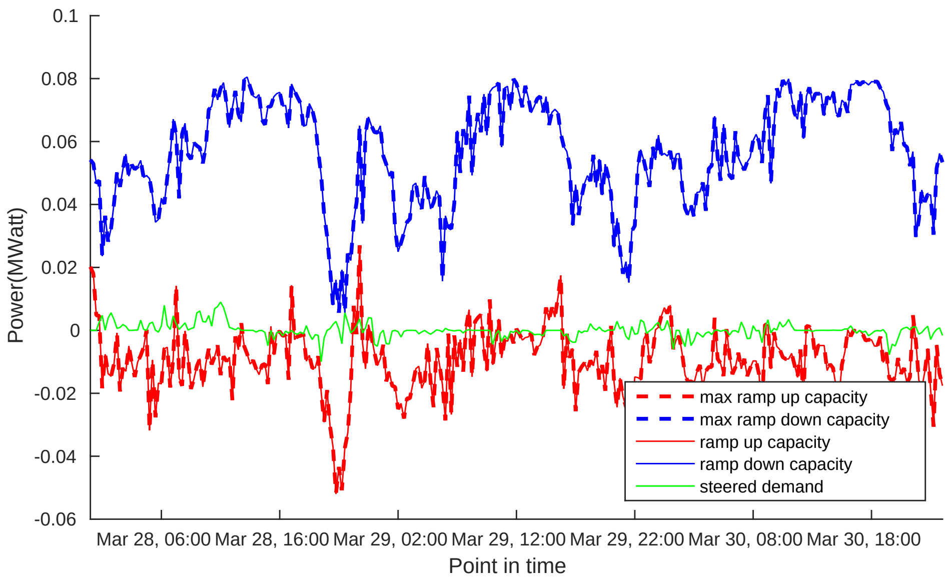


Figure C.11: offer all flexibility

Appendix D

Allocation and Flexibility

D.1 Baseline Scenarios

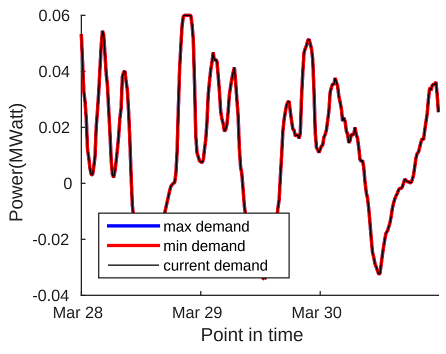


Figure D.1: business as usual

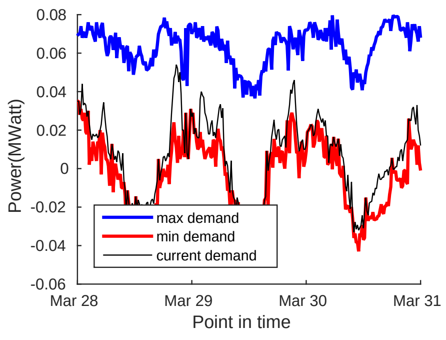


Figure D.2: offer no flexibility

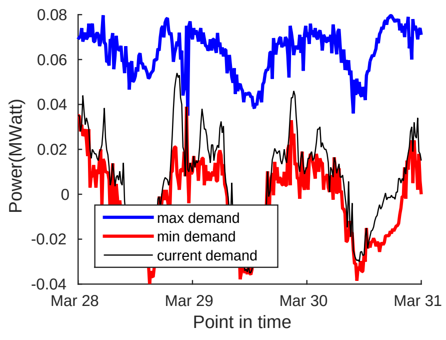


Figure D.3: offer 10% of flexibility

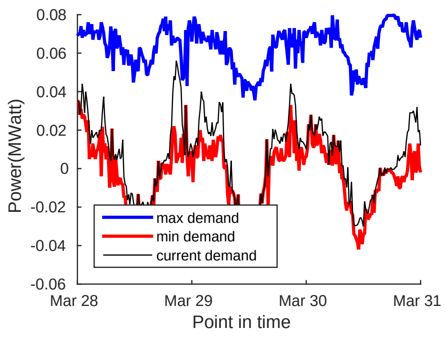


Figure D.4: offer 20% of flexibility

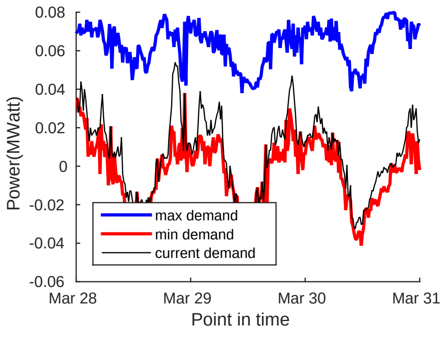


Figure D.5: offer 30% of flexibility

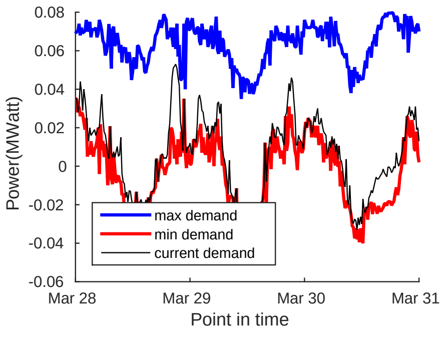


Figure D.6: offer 40% of flexibility

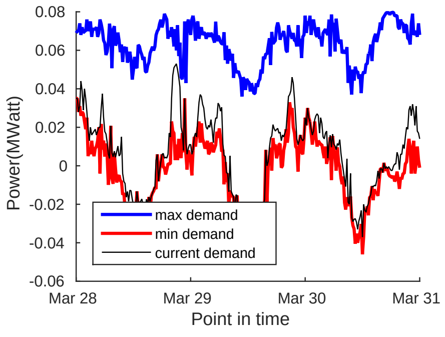


Figure D.7: offer 50% of flexibility

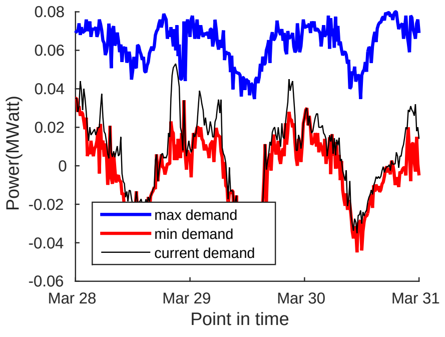


Figure D.8: offer 60% of flexibility

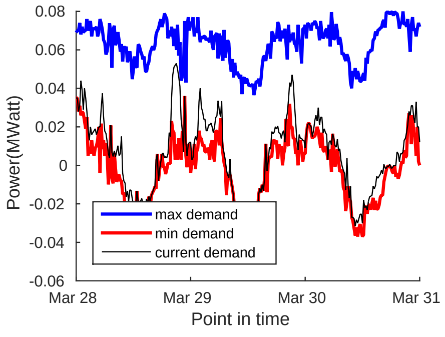


Figure D.9: offer 70% of flexibility

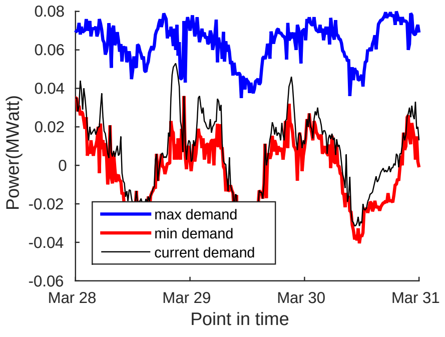


Figure D.10: offer 80% of flexibility

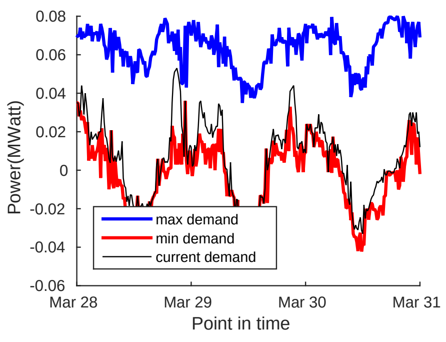


Figure D.11: offer 90% of flexibility

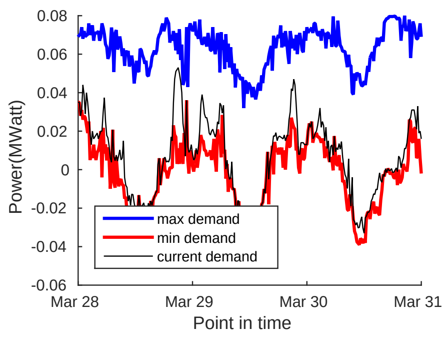


Figure D.12: offer 100% of flexibility

D.2 CACLA Experiments With Noise

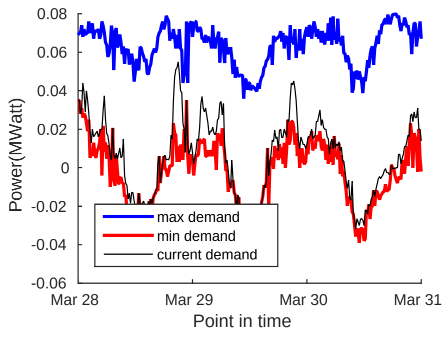


Figure D.13: no noise

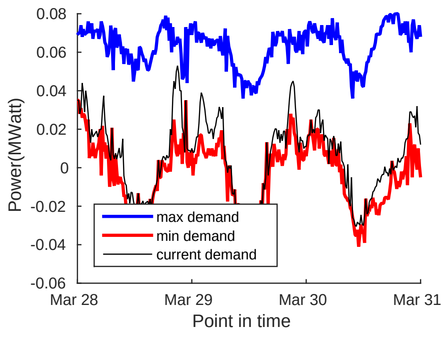


Figure D.14: 1% noise

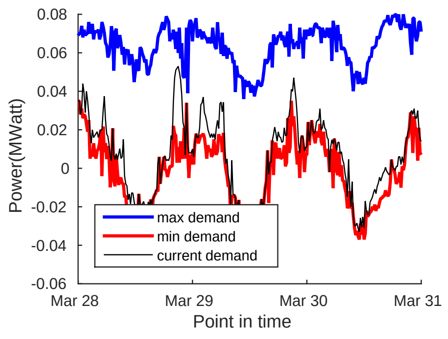


Figure D.15: 5% noise

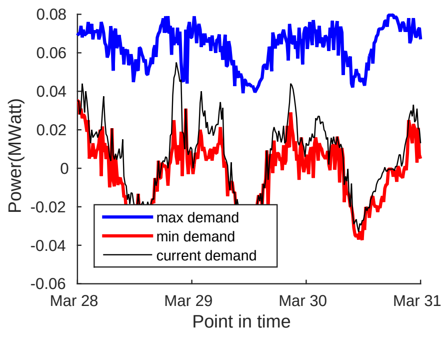


Figure D.16: 10% noise

D.3 NFCACLA Experiments With Noise

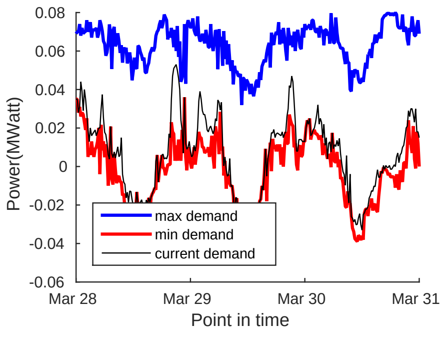


Figure D.17: no noise

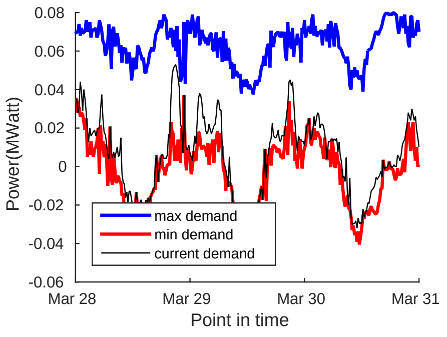


Figure D.18: 1% noise

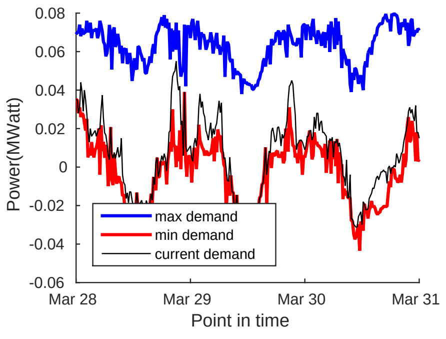


Figure D.19: 5% noise

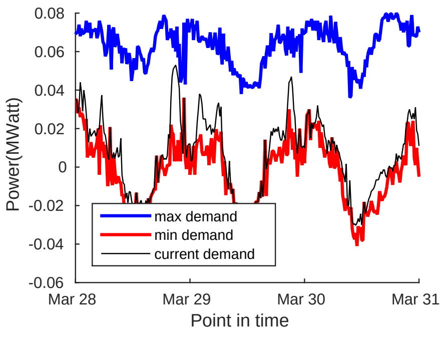


Figure D.20: 10% noise

D.4 NFCACLA Experiments With Training Iterations

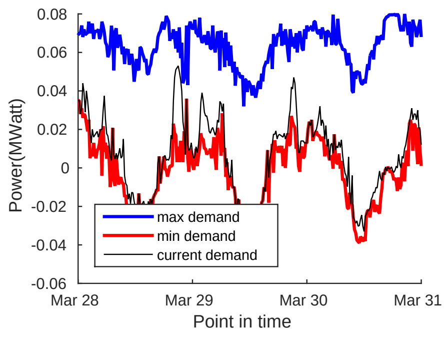


Figure D.21: 1 epoch

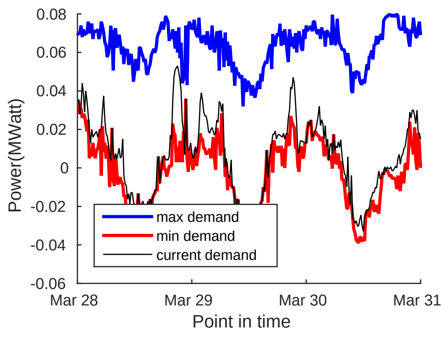


Figure D.22: 5 epochs

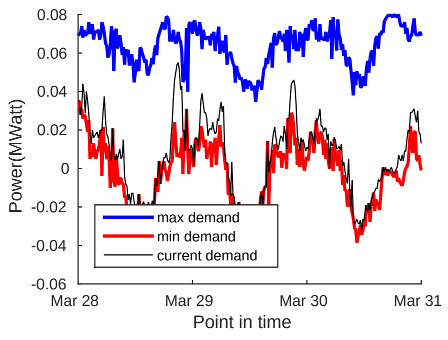


Figure D.23: 10 epochs

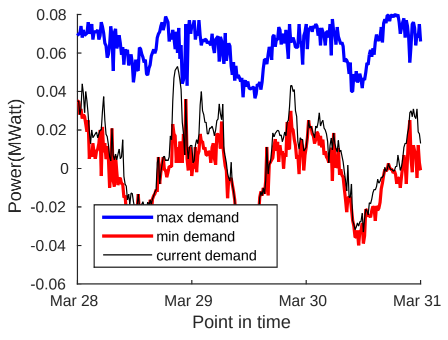


Figure D.24: 25 epochs

Bibliography

[1] G. Artac, D. Flynn, B. Kladnik, M. Hajdinjak, and A. F. Gubina. The flexible demand influence on the joint energy and reserve markets. In *Power and Energy Society General Meeting, 2012 IEEE*, pages 1–8, 2012.

[2] I. Atzeni, L. G. Ordóñez, G. Scutari, D. P. Palomar, and J. R. Fonollosa. Demand-side management via distributed energy generation and storage optimization. *Smart Grid, IEEE Transactions on*, 4(2):866–876, 2013.

[3] V. Bakker. *TRIANA: a control strategy for Smart Grids: forecasting, planning and real-time control.* University of Twente, 2012.

[4] R. Bellman. *Dynamic Programming.* Princeton University Press, Princeton, NJ, USA, 1 edition, 1957.

[5] F. Bliek, A. Van den Noort, B. Roossien, R. Kamphuis, J. De Wit, J. Van der Velde, and M. Eijgelaar. Powermatching city, a living lab smart grid demonstration. In *Innovative Smart Grid Technologies Conference Europe (ISGT Europe), 2010 IEEE PES*, pages 1–8, 2010.

[6] L. Busoniu, R. Babuska, B. De Schutter, and D. Ernst. *Reinforcement learning and dynamic programming using function approximators*, volume 39. CRC press, 2010.

[7] R. Caldon, A. R. Patria, and R. Turri. Optimisation algorithm for a virtual power plant operation. In *Universities Power Engineering Conference, 2004. UPEC 2004. 39th International*, volume 3, pages 1058–1062, 2004.

[8] Canadian Electricity Association. The smart grid: a pragmatic approach. `http://www.electricity.ca/media/SmartGrid/SmartGridpaperEN.pdf`, 2012. Accessed: 10-06-2016.

[9] K. Dielmann and A. Van Der Velden. Virtual power plants (vpp)-a new perspective for energy generation? In *Modern Techniques and Technologies, 2003. MTT 2003. Proceedings of the 9th International Scientific*

and Practical Conference of Students, Post-graduates and Young Scientists, pages 18–20, 2003.

[10] A. Dimeas and N. Hatziargyriou. Agent based control of virtual power plants. In *Intelligent Systems Applications to Power Systems, 2007. ISAP 2007. International Conference on*, pages 1–6, 2007.

[11] Y. Ding, S. Pineda, P. Nyeng, J. Ostergaard, E. M. Larsen, and Q. Wu. Real-time market concept architecture for ecogrid eua prototype for european smart grids. *Smart Grid, IEEE Transactions on*, 4(4):2006–2016, 2013.

[12] M. E. El-Hawary. The smart grid - state-of-the-art and future trends. *Electric Power Components and Systems*, 42(3-4):239–250, 2014.

[13] D. Ernst, M. Glavic, and L. Wehenkel. Power systems stability control: reinforcement learning framework. *Power Systems, IEEE Transactions on*, 19(1):427–435, 2004.

[14] European Commission. *Report from the Commission to the European Parliament, the Council, the European Economic and Social Committee and the Committee of the Regions on Renewable Energy Progress.* Publications Office, 2015.

[15] European Commission et al. Europe 2020 a strategy for smart, sustainable and inclusive growth. *COM (2010)*, 2020(3), 2010.

[16] European Technology Platform SmartGrids. Strategic deployment document for europe's electricity networks of the future, 2010.

[17] European Technology Platform SmartGrids. Strategic research agenda 2035, 2012.

[18] European Wind Energy Association et al. *Large Scale Integration of Wind Energy in the European Power Supply: Analysis, Issues and Recommendations: a Report.* European Wind Energy Association, 2005.

[19] European Wind Energy Association et al. *Wind energy-the facts: a guide to the technology, economics and future of wind power*. Routledge, 2012.

[20] European Wind Energy Association et al. Wind energy scenarios for 2020. Technical report, 2014.

[21] European Wind Energy Association et al. Wind energy scenarios for 2030. Technical report, 2015.

[22] P. Faria, Z. Vale, and J. Baptista. Constrained consumption shifting management in the distributed energy resources scheduling considering

demand response. *Energy Conversion and Management*, 93:309–320, 2015.

[23] V. François-Lavet, R. Fonteneau, and D. Ernst. Using approximate dynamic programming for estimating the revenues of a hydrogen-based high-capacity storage device. In *Adaptive Dynamic Programming and Reinforcement Learning (ADPRL), 2014 IEEE Symposium on*, pages 1–8, 2014.

[24] C. W. Gellings. The concept of demand-side management for electric utilities. *Proceedings of the IEEE*, 73(10):1468–1470, 1985.

[25] C. W. Gellings and J. H. Chamberlin. Demand-side management: concepts and methods. 1987.

[26] R. Hafner. *Dateneffiziente selbstlernende neuronale Regler*. PhD thesis, PhD thesis, University of Osnabrueck, 2009.

[27] R. Hafner and M. Riedmiller. Reinforcement learning in feedback control. *Machine learning*, 84(1-2):137–169, 2011.

[28] IEC. Iec smart grid standardization roadmap, 2010.

[29] International Energy Agency. *World energy outlook*. OECD/IEA, 2006.

[30] J. Jorgensen, S. Sorensen, K. Behnke, and P. B. Eriksen. Ecogrid eu - a prototype for european smart grids. In *Power and Energy Society General Meeting, 2011 IEEE*, pages 1–7, 2011.

[31] R. Kamphuis, P. MacDougall, W. van der Veen, E. Bakker, and J. van de Velde. Constrained capacity management and cost minimisation of ev-charging in a parking garage. In *PowerTech (POWERTECH), 2013 IEEE Grenoble*, pages 1–5, 2013.

[32] E. Karangelos and F. Bouffard. Towards full integration of demand-side resources in joint forward energy/reserve electricity markets. *Power Systems, IEEE Transactions on*, 27(1):280–289, 2012.

[33] E. Kellerer and F. Steinke. Scalable economic dispatch for smart distribution networks. *Power Systems, IEEE Transactions on*, PP(99):1–8, 2014.

[34] K. Kok. The powermatcher: Smart coordination for the smart electricity grid. *TNO, The Netherlands*, 2013.

[35] K. Kok, B. Roossien, P. MacDougall, O. van Pruissen, G. Venekamp, R. Kamphuis, J. Laarakkers, and C. Warmer. Dynamic pricing by scalable energy management systemsfield experiences and simulation results using powermatcher. In *Power and Energy Society General Meeting, 2012 IEEE*, pages 1–8, 2012.

[36] K. J. Kok, M. J. Scheepers, and I. G. Kamphuis. Intelligence in electricity networks for embedding renewables and distributed generation. In *Intelligent infrastructures*. Springer, Intelligent Systems, Control and Automation: Science and Engineering Series, 2009.

[37] P. Kotler. The prosumer movement: A new challenge for marketers. *Advances in Consumer Research*, 13(1):510–513, 1986.

[38] X. Lin, Y. Wang, and M. Pedram. Designing the optimal pricing policy for aggregators in the smart grid. In *Green Technologies Conference (GreenTech), 2014 Sixth Annual IEEE*, pages 75–80, 2014.

[39] G. Liu and K. Tomsovic. A full demand response model in co-optimized energy and reserve market. *Electric Power Systems Research*, 111:62–70, 2014.

[40] P. MacDougall, C. Warmer, and K. Kok. Mitigation of wind power fluctuations by intelligent response of demand and distributed generation. In *Innovative Smart Grid Technologies (ISGT Europe), 2011 2nd IEEE PES International Conference and Exhibition on*, pages 1–6, 2011.

[41] A. Marinescu, I. Dusparic, A. Taylor, V. Cahill, and S. Clarke. Decentralised multi-agent reinforcement learning for dynamic and uncertain environments. *arXiv preprint arXiv:1409.4561*, 2014.

[42] S. Mathieu, Q. Louveaux, D. Ernst, and B. Cornélusse. A quantitative analysis of the effect of flexible loads on reserve markets. In *Power Systems Computation Conference (PSCC), 2014*, pages 1–7, 2014.

[43] Y. B. Matthieu Zimmer and A. Dutech. Neural fitted actor critic. In *24th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, pages 671–676, 2016.

[44] Ministry of Power, Government of India. Smart grid vision and roadmap for india, 2013.

[45] A.-H. Mohsenian-Rad, V. W. Wong, J. Jatskevich, R. Schober, and A. Leon-Garcia. Autonomous demand-side management based on game-theoretic energy consumption scheduling for the future smart grid. *Smart Grid, IEEE Transactions on*, 1(3):320–331, 2010.

[46] J. M. Morales, A. J. Conejo, H. Madsen, P. Pinson, and M. Zugno. Virtual power plants. In *Integrating Renewables in Electricity Markets*, pages 243–287. Springer, 2014.

[47] C. Musardo, G. Rizzoni, Y. Guezennec, and B. Staccia. A-ECMS: An adaptive algorithm for hybrid electric vehicle energy management. *European Journal of Control*, 11(4):509–524, 2005.

[48] National Energy Technology Laboratory. A vision for smart grids, 2009.

[49] M. Negnevitsky, T. Nguyen, and M. de Groot. Novel business models for demand response exchange. In *Power and Energy Society General Meeting, 2010 IEEE*, pages 1–7, 2010.

[50] H. K. Nguyen, J. B. Song, and Z. Han. Demand side management to reduce peak-to-average ratio using game theory in smart grid. In *Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on*, pages 91–96, 2012.

[51] D. O'Neill, M. Levorato, A. Goldsmith, and U. Mitra. Residential demand response using reinforcement learning. In *Smart Grid Communications (SmartGridComm), 2010 First IEEE International Conference on*, pages 409–414, 2010.

[52] M. Peters, W. Ketter, M. Saar-Tsechansky, and J. Collins. A reinforcement learning approach to autonomous decision-making in smart electricity markets. *Machine learning*, 92(1):5–39, 2013.

[53] G. L. R. Pierre Pinson, Emil M. Larsen. Ecogrid eu evaluation: The concept and market place. EcoGrid EU Final Conference, DTU Lyngby, 2015.

[54] D. V. Prokhorov and Wunsch. Adaptive critic designs. *Neural Networks, IEEE Transactions on*, 8(5):997–1007, 1997.

[55] D. Pudjianto, C. Ramsay, and G. Strbac. Virtual power plant and system integration of distributed energy resources. *Renewable power generation, IET*, 1(1):10–16, 2007.

[56] M. L. Puterman. Markov decision processes. *Handbooks in operations research and management science*, 2:331–434, 1990.

[57] M. L. Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[58] M. Riedmiller. Neural fitted q iteration–first experiences with a data efficient neural reinforcement learning method. In *European conference on Machine Learning: ECML 2005*, pages 317–328. Springer, 2005.

[59] M. Riedmiller and R. Hafner. Learning (near) time optimal control for flexible robot joints. `http://ml.informatik.uni-freiburg.de/_media/publications/11_rss11_hafnerriedmiller.pdf`, 2015. Accessed: 10-06-2016.

[60] G. Ritzer and N. Jurgenson. Production, consumption, prosumption the nature of capitalism in the age of the digital ?prosumer? *Journal of consumer culture*, 10(1):13–36, 2010.

[61] F. Rumph, G. Huitema, and J. Verhoosel. Interoperability in power supply and demand coordination through conceptual modeling. IEEE 2013-

IFIP/IEEE International Symposium on Integrated Network Management. Ghent, 2013.

[62] W. Saad, Z. Han, H. V. Poor, and T. Başar. Game-theoretic methods for the smart grid: An overview of microgrid systems, demand-side management, and smart grid communications. *Signal Processing Magazine, IEEE*, 29(5):86–105, 2012.

[63] S. Samantaray. Letter to the editor: Smart grid initiatives in india. *Electric Power Components and Systems*, 42(3-4):262–266, 2014.

[64] J. Si, A. Barto, W. Powell, and D. Wunsch. *Handbook of learning and approximate dynamic programming*, volume 2. John Wiley & Sons, 2004.

[65] P. Siano, C. Cecati, H. Yu, and J. Kolbusz. Real time operation of smart grids via FCN networks and optimal power flow. *Industrial Informatics, IEEE Transactions on*, 8(4):944–952, 2012.

[66] M. Silva, H. Morais, and Z. Vale. An integrated approach for distributed energy resource short-term scheduling in smart grids considering realistic power system simulation. *Energy Conversion and Management*, 64:273–288, 2012.

[67] M. Sonnenschein, C. Hinrichs, A. Nieße, and U. Vogel. Supporting renewable power supply through distributed coordination of energy resources. In *ICT Innovations for Sustainability*, pages 387–404. Springer, 2015.

[68] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[69] Tennet.org. Tennet - export data, 2015.

[70] J. Twidell, T. Weir, et al. *Renewable energy resources.* Routledge, 2015.

[71] H. Van Hasselt. Reinforcement learning in continuous state and action spaces. In *Reinforcement Learning: state of the art*, pages 207–251. Springer, 2012.

[72] H. Van Hasselt and M. Wiering. Using continuous action spaces to solve discrete problems. In *Neural Networks, IJCNN. International Joint Conference on*, pages 1149–1156, 2009.

[73] H. Van Hasselt and M. A. Wiering. Reinforcement learning in continuous action spaces. In *Approximate Dynamic Programming and Reinforcement Learning, ADPRL. IEEE International Symposium on*, pages 272–279, 2007.

[74] G. K. Venayagamoorthy. Potentials and promises of computational intelligence for smart grids. In *Power & Energy Society General Meeting, 2009. PES'09. IEEE*, pages 1–6, 2009.

[75] J. Wang, N. E. Redondo, and F. D. Galiana. Demand-side reserve offers in joint energy/reserve electricity markets. *Power Systems, IEEE Transactions on*, 18(4):1300–1306, 2003.

[76] Q. Wang, C. Zhang, Y. Ding, G. Xydis, J. Wang, and J. Østergaard. Review of real-time electricity markets for integrating distributed energy resources and demand response. *Applied Energy*, 138:695–706, 2015.

[77] S. Weckx, R. D'hulst, B. Claessens, and J. Driesensam. Multiagent charging of electric vehicles respecting distribution transformer loading and voltage limits. *Smart Grid, IEEE Transactions on*, 5(6):2857–2867, 2014.

[78] P. J. Werbos. Computational intelligence for the smart grid-history, challenges, and opportunities. *Computational Intelligence Magazine, IEEE*, 6(3):14–21, 2011.

[79] M. Wiering and M. Van Otterlo. Reinforcement learning: state of the art. In *Adaptation, Learning, and Optimization*, volume 12. Springer, 2012.

[80] Z. Xu, Y. Xue, and K. P. Wong. Recent advancements on smart grids in China. *Electric Power Components and Systems*, 42(3-4):251–261, 2014.